JOINT PREDICTION OF WATCH RATIO AND SKIP BEHAVIOR IN RECOMMENDATION SYSTEM

A PREPRINT

Mahdi Rezapour

January 19, 2026

ABSTRACT

This study examines user engagement with online video content using a multi-task learning approach. In this study, we combine viewing histories, basic user attributes, and content datasets from several public sources to predict both the proportion of a video watched and whether a user skips a video. The two tasks are learned jointly, using a shared representation with separate outputs for regression and classification. Several common multi-task architectures are evaluated and compared under the same experimental setup. Techniques like Multi-Gate Mixture-of-Experts (MMoE), and Progressive Layered Extraction (PLE), and cross stick network were employed. Results of this study on a held-out test set show that watch ratio can be predicted with reasonable accuracy, while skip prediction remains challenging and only marginally better than random guessing. Differences between model architectures are small, suggesting that data size and label definition might have a stronger influence on performance than model choice. These findings highlight the difficulty of modeling discrete engagement outcomes from noisy behavioral data and point to the importance of careful label construction in future work. Especially, this study highlights the challenges of prediction of skip prediction due to likely reason of subjectively setting the threshold.

Keywords User engagement · watch ratio · skip behavior · multi-task learning · recommender systems · behavioral data · video consumption · user modeling

0.1 Introduction

Recommendations could employ various types of user feedback signals such as clicking, commenting, rating, etc. [11] [8]. It has been discussed that no single feedback could reflect accurately the users satisfactions [31]. For instance, over-concentration on clicking might aggravate the click-bait concern [18]. As a result, it makes sense to effectively estimate and learn multiple user behaviors at the same time by potential means of multiple-task learning as the technique addresses that challenge.

The term of multi-task learning (MTL) has used broadly [29] [26] to include various field such as selection [32] and transfer learning [22]. It has wide applications in different fields like natural language processing [20], computer vision [3] and genomics [7].

There have been an increasing trend to study users behaviors such as users' satisfaction and engagement through likelihood of clicking, finishing, sharing, which has been studies significantly in the industry studies [30] [27].

MTL shows learning efficiency by using a single model to learn multiple tasks. That efficiency has been proven to be achieved through information sharing between tasks [14]. However, in case that the recommendation systems being loosely correlated, the performance is expected to be deteriorated, negative transfer [5].

In this study, we used the four tasks of cross stitch, MultiGate Mixture-of-Experts (MMoE), and Progressive Layered Extraction (PLE), as we believe that the 4 tasks are closely related as the likelihood are closely correlated. These methods will be presented along with their methodological approaches in the order of their progression.

0.2 Related work

1 Cross-Stitch Networks

Multi-task learning has extensively employed in the domain of image processing like surface normal and edge labels [28], or action recognition [6].Similarly , Cross stich network in muti-task learning was initially used in the context of convolutional network in the field of image recognition. In that work, cross-stich units would combine the activations from multiple networks, learning optimal combination of shared and task-specific representations [10].

Sluice network [24]and cross-stich network [10] learns static linear combinations for information fusion of different tasks, which might not capture the sample dependence. In other words, the representations have been combined with the same static weights for samples in the models.

Sharing information across different categories through learning representation showed a gains in segmentations [4] and detection [2]. For instance, it was discussed that fault diagnostic suffer from the fact that they overlooked the estimation of fault size which might be equally important, and also they typically learn features by exploiting only either implicit or explicit learning without using their collaborative potential [14]. To address those shortcomings, that work employed implicit-explicit collaborative cross-stich network for investigating both fault diagnosis and size estimation. The results showed the promising practicality of the approach.

It was discussed that resource allocation has become a major research area, where model is trained for specific application [17]. In that work, multi-task learning approach was used to handle resource allocation problems.

Two related tasks in gas-liquid flow including void friction measurement (regression task) and voice fraction classification (classification task) with the help of cross-stich embedding with attention modules were used [13].

2 MultiGate Mixture-of-Experts (MMoE)

To tackle task differences, multi-gate mixture of experts (MMoE) could apply gating network to combine bottom experts. Multi-task learning helps us to learn multiple goals simultaneously. However, it is expected that the model performance to be sensitive to the relationship or correlation across the responses. The MMoE was proposed to explicitly learns to model task relationship from data [12]. In addition, gating will be employed to optimize sub models across each task.

Many recommendation system employed multi-task learning using deep neural network (DNN) model [25]. It has been discussed that multi-task learning is expected to improve model prediction on all tasks by utilizing regularization and transfer learning [1]. However, it should be noted that as mentioned the performance might be dependent on the relationship between responses and not necessarily multi-task learning not always outperform the single-task models on all tasks [15] [19].

Most MTL shares parameters across tasks at lower layers while keeping separate task towers at upper layers. Embedding-based Sparse Sharing Mechanism (ESSM) which shares embedding parameters across tasks such as click-through rate (CTR) and conversion rate (CVR) for enhancement of prediction performance of the CVR task [27]. However MoE still neglects the interaction and differentiation between experts which is expected to suffer from seesaw phenomenon [9].

A new MoE model was proposed to capture the task differences without requiring significantly more parameters, which was called Multi-gate Mixture-of-experts (MMoE) [12].

3 Progressive Layered Extraction (PLE)

Tasks in real-world recommender systems might be loosely correlated or even conflicted,
which could cause performance deterioration known as negative transfer [16]. Often the seesaw phenomenon, where enhancement in one task result in degradation in the other task, has been observed while applying technique like MMoE [9]. In addition, the shortcoming of MMOE is that it treats all experts equally, while PLE could explicitly separate task-common and task-specific exports [9]. To address those challenges, the Progressive layer extraction (PLE) could be employed, which separates shared and task-specific components, separating deeper semantic knowledge gradually, enhancing join representation learning [9].

3.1 Method

Now the following subsections highlight the mathematical equation of the above techniques in detail.

4 Cross-Stitch Method

Cross-stich tried to find the best shared representation in multi-task learning [10]. The method learns the shared representation using linear combination and learning the optimal combination of the employed set of tasks. For giving recommendations, the split-like architecture will be used for having both representation for both tasks and task specific representation. In terms of where split should be made, and at which layer, the choice is highly dependent on the task at hand and input dataset.

Suppose we have two tasks A and B with activation of X_A And X_B at a given layer. The model learns [17] parameters α_{AA}, α_{AB} , α_{BA} and α_{BB} , which control how much information is shared as:

$$\begin{bmatrix} \tilde{x}_A^{ij} \\ \tilde{x}_A^{ij} \end{bmatrix} = \begin{bmatrix} \alpha_{AA} & \alpha_{AB} \\ \alpha_{BA} & \alpha_{BB} \end{bmatrix} \begin{bmatrix} x_A^{ij} \\ x_B^{IJ} \end{bmatrix} \tag{1}$$

Based on the above equation, the process could make some layers task specific by setting α_{AB} and α_{BA} to zero or using a higher value for them for choosing a higher shared representation.

Two architects using cross stich network will be combined. The alpha values with 0s will be initialized. The cross-stich units help to come up with an optimal combination of shared and task-specific representation.

5 MMoE Method

In MMoE, several bottom layers following the input layers will be shared across all tasks and finally each task will have an individual tower of network on top of the bottom representation [12]. In the process, the gating network takes the input features, outputting SoftMax gates while assembling the experts with different weights, which allow different tasks to utilize experts differently. Thus, the gating networks for various tasks could learn different mixture patterns of experts and so it could capture the task relationship. MMoE could learn differences across various tasks which could enhance model quality and efficiency for tasks [23] Most multi-task learning (MTL) shares parameters across tasks at lower layers while keeping separate task towers at upper layers. It could be approximately classified into four categories.

The first category is hard parameter sharing where embedding sharing is the most common structure for sharing information.

The original formula for mixture of expert (MoE) could be written as [21]:

$$y = \sum_{i=1}^{n} g_i(x) f_i(x) \tag{2}$$

Where:

$f_i(x)$ is the output of the *i*th expert

$g_i(x)$ is the gating weights for expert *i,* produced by SoftMax over all experts where $\sum_{i=1}^{n} g_i\,(x) = 1$. The key idea is to modify the original MoE and add a separate gating network g^k for each task *k.* So, we will have

$$y_k = h_k(f_k(x)) \tag{3}$$

Where:

$f_k(x)$ is task specific representation for task *k*

$h_k(.)$ is task specific prediction head

y_k final output for task k

$f_k(x)$, on the other hand, could be written as

$$f_k\,(x) = \sum_{i=1}^{n} g_k(x)_i f_i(x) \tag{4}$$

Where

$f_i(x)$ output of expert *i*

$$g_k(x)_i isgatingweightforexpertifortaskk \tag{5}$$

n is number of experts

$g_k(x)$ could be written as

$$g_k\,(x) = softmax(W_k x) \tag{6}$$

Where

$g_k\,(x)$ is gating vector for task *k*

W_k task-specific gating weight matrix

x is shared-bottom representation

6 PLE Method

The benefits of PLE could be summarized as first, explicitly separating shared and task-specific experts to prevent harmful parameter interference, and second, multi-level experts and gating networks are introduced for fusing more abstract representation, finally progressive separation routing was made to model transferring between correlated tasks [9].

We stacked all expert output which will be used in the later equation below

$$S_j\,(x) = [h_{1,j}(x), \ldots, h_{E,j}(x)] \in R^{E\times H} \tag{7}$$

j: PLE layer index=*j*=1,. . .,*N*

E: Total number of experts at layer *j*:

$$E = E_s + E_r + E_c \tag{8}$$

E is the number of experts (shared and specific).E_s, E_r, E_c are number of shared, regression and classification experts,

The formulation of the gating network of task *k* in the *j*th extraction network for PLE could be written as:

$$d_{k,j}\,(x) = \sum_{i=1}^{E} w_{k,j,i}(x) h_{i,j}(x) \tag{9}$$

where

$$w_{k,j}\,(x) = softmax(W_{k,j} d_{k,j-1}\,(x) + b_{k,j}) \tag{10}$$

Where

$d_{k,j}\,(x)$ is task *k* representation at layer *j*.

$h_{i,j}(x)$ is the output of expert *i* at layer *j*, $w_{k,j}\,(x)$ is the SoftMax gate weight for expert *i* for task *k* at layer *j*. The summation will mix all experts' output using the gate weights.

The final PLE representation for task *k* could be written as

$\hat{y}_{reg} = w_{reg\_head} d_{r,N}\,(x) + b_{reg\_head}$	Regression	10
$\hat{y}_{cls} = w_{cls\_head} d_{c,N}\,(x) + b_{cls\_head}$	Classification	11

Where w_{reg_head} and w_{cls_head} are regression and classification head matrix.

Also, each expert, including reg+cls+shared experts, is a nonlinear transformation as

$$h_{i,j} = f_{i,j}(x) \tag{11}$$

Where $f_{i,j}(.)$ is implemented as a small feed-forward with ReLU activation.

Finally, the weighted sum of losses will be written as:

$$L\left(\theta_r, \ldots, \theta_c, \theta_s\right) = \omega_r L_r\left(\theta_s, \theta_s\right) + \omega_c L_c\left(\theta_s, \theta_s\right) \tag{12}$$

θ_s is shared experts parameter, and ω_r is task weights

7 Data and Hyper parameters

Three public sources of information were used including user watch histories, content metadata, and basic user attributes. These datasets were merged sequentially using shared columns of content_id and user_id so resultant data has a detailed info of what users watched and other details. A dummy variable of watch ratio was created by dividing the time a user spent watching a video by its total duration. This value served as the continuous response for one of the tasks. A second, simpler label was created by marking events as “skipped" when the watch ratio fell below 0.5. The value of 0.5 was set completely subjectively.

To prevent extremely sparse user histories, we kept only users with more than ten recorded watched movies. After filtering the data based on that condition, just over ten thousand users remained. The dataset covered a wide range of genres, with drama, comedy, and action appearing most frequently. However, still the data is considered not very large with so many variabilities.

In terms of feature preparation, we used a mixture of numerical and categorical variables. User age, content rating, and duration were fed as numerical inputs, while genre and gender were converted into onehot encoded vectors so they could be used by the model. As expected, these pieces were combined into a single feature matrix and converted into Torch tensors. The data was then split into training and testing sets using 30% for testing.

The predictive model was built based on various model architects and using a shared hidden layer and two separate output heads: one for predicting the watch ratio and another for classifying whether a user skipped the content. As expected, the regression output was trained using mean squared error, while the skip prediction used crossentropy loss. Both objectives were optimized jointly using the Adam optimizer with a learning rate of 0.0001. Training ran for 500 epochs on a GPUenabled environment in Kaggle.

Performance was evaluated on the heldout test set. The model achieved a mean squared error of 0.0856 for watchratio prediction, indicating reasonably accurate estimates. The skipclassification accuracy reached around 55%, reflecting the more challenging especially due subjectively setting value of 0.5 as threshold.

7.1 Results

The performance of the evaluated multi-task learning architectures on test dataset is presented in Table 1. Table 1 reports accuracy on test dataset for the skip classification task and means squared error (MSE) for the watch-ratio regression task.

As can be seen, all these three models achieved similar performance, with marginal differences across various architectures. For the regression task, mean squared error (MSE) values ranged from 0.083 to 0.085, indicating relatively stable and consistent predictions of watch ratio across models. Among them, MMoE achieved the lowest MSE (0.083), suggesting a slight advantage in modeling shared representations for continuous engagement prediction.

As for the classification task, accuracy remained modest, with results pointing at clustering around 51–52% for all models. The PLE model reached the highest accuracy (51.33%), followed by MMoE (51.32%) and Cross-Stitch (51.21%). These results indicate that predicting skip behavior is considerably more challenging than predicting watch ratio due to subjectively setting the threshold. Again, this difficulty is expected as a result of setting skips and the inherent noise and variability in user engagement behavior with a threshold of 0.5 used to.

Importantly, under the current data scale and feature set, the close performance gaps suggest that model architecture had limited impact. The relatively small dataset size, high behavioral variability, and coarse binary labeling likely constrained the achievable gains from more complex multi-task designs. Nonetheless, the results demonstrate that joint learning of regression and classification tasks is feasible and yields stable performance across different multi-task frameworks.

Model	Accuracy	MSE
Cross stich	51.21%	0.085
MMoE	51.32%	0.083
PLE	51.33%	0.084

Table 1: KPI over test data

In summary, while improvements in skip classification remain limited, the models show reasonable effectiveness in estimating watch ratio, supporting the idea of multi-task learning for engagement prediction under severely data-constrained settings.

7.2 Discussion

The presented results showed both the potential and the limitations of using multi-task learning for user engagement while having data-constrained and noisy environments. We used three models and across all considered architectures including cross-Stitch, MMoE, and PLE, the variation in performance was minimal. That indicates under the current dataset size and feature richness, model capacity and architectural sophistication might not be the primary concerns. Instead, data characteristics and label quality appear to be a more of dominant role.

The watch-ratio regression task achieved almost similar performance, with MSE values around 0.08–0.09 across all considered models. Due to the fact that watch ratio is between 0 and 1, it is inherently noisy due to diverse user behaviors and content types. As a result, this level of error might indicate that the models could capture meaningful engagement signals. Regression, on the other hand, benefits from preserving the continuous nature of user behavior, which provides richer supervision compared to binary labels.

On the other hand, the skip classification task showed significantly more challenging, with accuracy only slightly above random guessing of 0.50%. We assign a key contributing factor is the subjective threshold of 0.5 being used to define skip events. This hard cutoff most likely introduces label noise, particularly for watch ratios near the threshold, where user intent become unclear. As a result, the classification task might not exhibit a well-defined boundary, limiting the accuracy regardless of model architecture.

The limited gains from advanced multi-task designs such as MMoE and PLE further might point to the fact that task relatedness might be weaker than assumed in this study. While watch ratio and skip behavior are intuitively connected, the separation of one task from the other might reduce shared signal and potentially insert conflicting gradients during joint optimization. This effect is amplified when the dataset is relatively small and user histories remain sparse despite filtering.

From a data perspective, it should be noted that the modest dataset size and potentially high variability across users, genres, and content durations might constrain generalization. Another limitation is the reliance on basic user and content features, while practical, might not sufficiently take into consideration future engagement such as temporal context, sequential viewing patterns, or personalized preferences. A solution might be offered such as incorporating richer behavioral signals or sequence-aware models which might improve both tasks more effectively than further architectural tuning.

In summary, these findings indicate that future improvements are more likely to result from better data and clearer labels rather than more complex models. Increasing the size of the dataset, refining how skip behavior is defined, and including richer information about users and content could all help improve results. Especially trying to come up with a better threshold for skip item is expected to result in a better performance of the model.

References

[1] editor multitask learning: A knowledge-based source of inductive bias. *Machine learning: Proceedings of the tenth international conference*, 1993.

[2] editor fast r-cnn. *Proceedings of the IEEE international conference on computer vision*, 2015.

[3] Pentina A, Sharmanska V, and Lampert C. Curriculum learning of multiple tasks. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015.

[4] Hariharan B, Arbeláez P, Girshick R, and Malik J. *Simultaneous detection and segmentation. European conference on computer vision*, 2014.

[5] Olivas E, Guerrero Jdm, Martinez-Sober M, Benedito Magdalena, J, López Serrano, and A. *Handbook of research on machine learning applications and trends: Algorithms, methods, and techniques: Algorithms, methods, and techniques*, 2009.

[6] Gkioxari G, Hariharan B, Girshick R, and Malik J. R-cnns for pose estimation and action detection. 2014.

[7] Obozinski G, Taskar B, and Jordan M. Joint covariate selection and joint subspace selection for multiple classification problems. *Statistics and Computing*, 20:231–252, 2010.

[8] Hamidi H and Moradi R. Design of a dynamic and robust recommender system based on item context, trust, rating matrix and rating time using social networks analysis. *Journal of King Saud University-Computer and Information Sciences*, 36:101964, 2024.

[9] Tang H, Liu J, Zhao M, and Gong X. Progressive layered extraction (ple): A novel multi-task learning (mtl) model for personalized recommendations. *Proceedings of the 14th ACM conference on recommender systems*, 2020.

[10] Misra I, Shrivastava A, Gupta A, and Hebert M. Cross-stitch networks for multi-task learning. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[11] Liu J, Dolan P, and Pedersen E. Personalized news recommendation based on click behavior. *Proceedings of the 15th international conference on Intelligent user interfaces*, 2010.

[12] Ma J, Zhao Z, Yi X, Chen J, Hong L, and Chi E. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 2018.

[13] Zhang J, Ma C, Chen P, Li M, Wang R, and Gao Z. Co-attention-based cross-stitch network for parameter prediction of two-phase flow. *IEEE Transactions on Instrumentation and Measurement*, 72:1–12, 2023.

[14] Zhong K, Deng Y, Tian J, Lu Y, Liu X, and Wang P. Implicit-explicit collaborative crossstitch network with multitask disentanglement learning for fault diagnosis and size estimation. *IEEE Transactions on Instrumentation and Measurement*, 2025.

[15] Kaiser L, Gomez A, Shazeer N, Vaswani A, Parmar N, and Jones L. One model to learn them all.

[16] Torrey L and Shavlik J. Transfer learning. handbook of research on machine learning applications and trends: algorithms, methods, and techniques. pages 242–264, 2010.

[17] Jehanzeb M, Rasool A, Amin M, Zia H, Shaheen T, and Najaf M. Cross-stitch multi task feature learning for resource allocation in iot. *Spectrum of Engineering Sciences*, 2:269–289, 2024.

[18] Potthast M, Köpsel S, Stein B, and Hagen M. *Clickbait detection. European conference on information retrieval*, 2016.

[19] Luong M-T, Le Q, Sutskever I, Vinyals O, and Kaiser L. Multi-task sequence to sequence learning. 2015.

[20] Collobert R and Weston J. A unified architecture for natural language processing: Deep neural networks with multitask learning. *Proceedings of the 25th international conference on Machine learning*, 2008.

[21] Jacobs R, Jordan M, Nowlan S, and Hinton G. Adaptive mixtures of local experts. *Neural computation*, 3:79–87, 1991.

[22] Pan S and Yang Q. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22:1345–1359, 2009.

[23] Ruder S. An overview of multi-task learning in. *deep neural networks*.

[24] Ruder12 S, Bingel J, Augenstein I, and Søgaard A. Learning what to share between loosely related tasks. 2, 2017.

[25] Bansal T, Belanger D, and Mccallum A. *Ask the gru: Multi-task learning for deep text recommendations. proceedings of the 10th ACM Conference on Recommender Systems*, 2016.

[26] Evgeniou T and Pontil M. Regularized multi–task learning. *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2004.

[27] Ma X, Zhao L, Huang G, Wang Z, Hu Z, and Zhu X. Entire space multi-task model: An effective approach for estimating post-click conversion rate. *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 28:41–75, 1997.

[28] Wang X, Fouhey D, and Gupta A. Designing deep networks for surface normal estimation. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015.

[29] Amit Y, Fink M, Srebro N, and Ullman S. Uncovering shared structures in multiclass classification. *Proceedings of the 24th international conference on Machine learning*, 2007.

[30] Lu Y, Dong R, and Smyth B. Why i like it: multi-task learning for recommendation and explanation. *Proceedings of the 12th ACM Conference on Recommender Systems*, 2018.

[31] Ren Y, Du Y, Wang B, and Zhang S. *Deep Mutual Learning across Task Towers for Effective Multi-Task Recommender Learning*.

[32] Kang Z, Grauman K, and Sha F. *Learning with whom to share in multi-task feature learning. ICML*, 2011.