# Appendix: A Dynamic System Framework for Modeling Consumer Trajectories and Exploring Consumer Preferences in Recommender System

## Appendix I: Technical Details of DeepONet

The deep operator network (DeepONet) (Lu et. 2021) is a special type of Neural Network (NN) that could generate the numeric solution of ordinary differential equations with small generalization error. It is theoretically supported by the universal operator approximation theorem, which states that a NN with a single hidden layer can approximate accurately any nonlinear continuous functional (a mapping from a space of functions into real numbers) and nonlinear operator (a mapping from a space of functions into another space of functions). To illustrate the idea, we denote $G$ as a differential operator taking an input function $u$, with $G(u)$ being the corresponding output function as the solution to the differential equation. For any point $y$ in the domain of $G(u)$, the output $G(u)(y)$ is a real number that could be approximated by the neural network taking inputs of $u$ and $y$.

To obtain such neural network, we first focus on the approximation of the input function $u$, as it could take very complex functional forms in practice. Therefore, we represent the input functions discretely based on the function values at a sufficient but finite number of locations $\{x_1, x_2, \ldots, x_m\}$, so that neural network approximations can be applied. As discussed in (Lu et al. 2021), there are other alternative discrete function representation methods that lead to similar levels of performance. As a result, we do not need the input of the explicit function u, and we can feed its approximation $\{u(x_1), u(x_2), \ldots, u(x_m)\}$ into the model instead. Specifically, these values will be fed into the "branch" network that encodes the discrete input function space. To improve the generalizability of the DeepONet model, we formulate the branch network $b_i(.)$ as $p$ numbers of multiple stacked neural

networks, each representing a particular approximation of the input function. $p$ is the hyperparameter that balances between the complexity and accuracy of the model.

Then, for each specific input function approximation, we construct another special type of neural network $t_i(.)$, also named as the "trunk" network, to encode the domain of the output functions. It takes the input values $y$ and produces the coefficient value $t_i(y)$ for the corresponding input function approximation $b_i(u(x_1), u(x_2), \dots, u(x_m))$ in our final solution prediction task. By combining the branch networks and the trunk network together, we will be able to obtain the final solution as the aggregation of these outputs (the workflow of the DeepONet model is shown in Figure 1):

$$G(u)(\mathrm{y}) = \sum_{i=1}^{p} b_i(u(x_1), u(x_2), \dots, u(x_m)) * t_i(y)$$

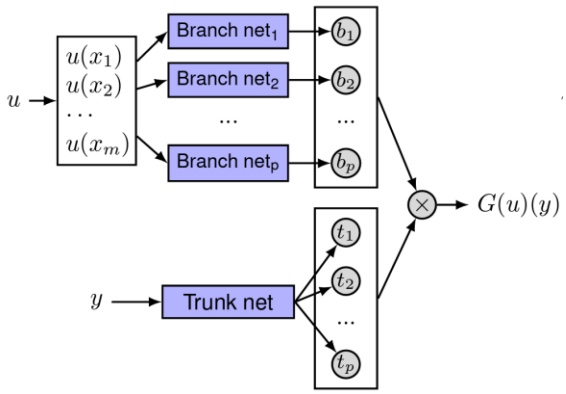


*Figure 1: Illustration of DeepONet for Solving the Dynamic System*

## Appendix II: Technical Details of Mixture Density Network

In order to obtain a complete description of the data, for the purposes of predicting the outputs corresponding to new inputs, it is important to model the conditional probability distribution of the given data (conditioned on the inputs). Note that in many cases, it would be very hard to describe the given data using explicit functional forms. Therefore, the Mixture Density Network (MDN) method has been proposed in the literature (Bishop

1994), which combines a conventional neural network with a mixture density model. The MDN model could in principle represent arbitrary conditional probability distributions in the same way that a conventional neural network can represent arbitrary functions.

The MDN model is motivated by the classical conventional least-squares technique, which is derived from maximum likelihood on the assumption of Gaussian distributed data. As we will be dealing with more complex data in real practices, we replace the Gaussian distribution in the classical method with a mixture network model, which has the ability to model completely general distribution functions. The probability density of the target data is then represented as a linear combination of kernel functions in the form:

$$p(t|x) = \sum_{i=1}^{m} \alpha_i(x)\varphi_i(t|x)$$

where $m$ is the number of components in the mixture. The parameters $\alpha_i(x)$ are called mixing coefficients, and can be regarded as prior probabilities (conditioned on $x$) of the target vector $t$ having been generated from the $i$-th component of the mixture. Note that the mixing coefficients are taken to be functions of the input vector $x$. The functions $\varphi_i(t|x)$ represent the conditional density of the target vector $t$ for the $i$-th kernel. While there are multiple choices for selecting the kernel functions, we focus on the Gaussian kernel functions in this paper to simplify the computation process.

The parameters of the mixture model, namely the mixing coefficients $\alpha_i(x)$, the means $\mu_i(x)$ and the variances $\sigma_i(x)$, are modeled using the outputs of a conventional neural network which takes $x$ as its input. The combined structure of a feed-forward network and a mixture model we refer to as a Mixture Density Network (MDN), and its basic structure is indicated in Figure 2. By choosing a mixture model with a sufficient number of kernel

functions, and a neural network with a sufficient number of hidden units, the MDN can approximate as closely as desired any conditional density $p(t|x)$.

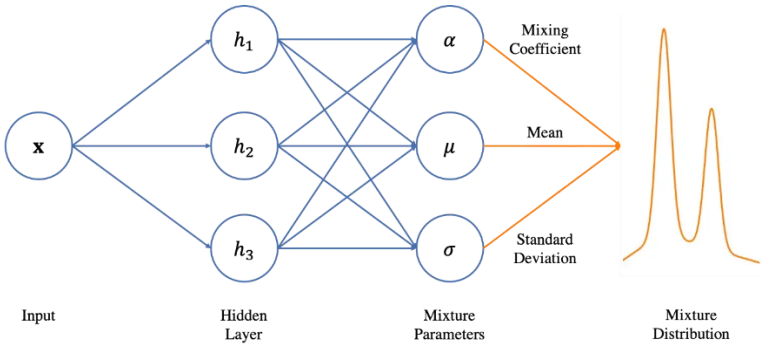


*Figure 2: Illustration of the Mixture Density Network for Computing the Expected Value*

## Appendix III: Robustness Check Results

To further validate the findings of our paper, we replicate our empirical analysis for the online experiment under these alternative settings: (more results available upon request)

(a) We include different combinations of consumer features, video features and time fixed effects in the regression models to evaluate the treatment effects. As shown in Table 1(a-c), our results are not affected by particular model specifications of features or fixed effects for the estimation process, as the estimated coefficients of all three variables remain largely the same across different columns of these tables.

| | VCTR | VCTR | VCTR | VCTR | VCTR |
|---|---|---|---|---|---|
| Treatment | 0.0617*** | 0.0622*** | 0.0623*** | 0.0613*** | 0.0620*** |
| | (0.0011) | (0.0011) | (0.0011) | (0.0011) | (0.0012) |
| User Features | Yes | No | Yes | Yes | No |
| Video Features | Yes | Yes | No | Yes | No |
| Time Fixed Effect | Yes | Yes | Yes | No | No |
| R-Squared | 0.0152 | 0.0078 | 0.0131 | 0.0178 | 0.0058 |
| Users | 4,007,354 | 4,007,354 | 4,007,354 | 4,007,354 | 4,007,354 |
| Observations | 87,984,376 | 87,984,376 | 87,984,376 | 87,984,376 | 87,984,376 |

*Table 1(a): Average Treatment Effect on VCTR with Different Combinations of User, Video Features and Time Fixed Effect. *p<0.1; **p<0.05; ***p<0.01*

| | VCR | VCR | VCR | VCR | VCR |
|---|---|---|---|---|---|
| Treatment | 0.0342*** | 0.0346*** | 0.0349*** | 0.0337*** | 0.0345*** |
| | (0.0004) | (0.0004) | (0.0005) | (0.0004) | (0.0006) |

| | | | | | |
|---|---|---|---|---|---|
| User Features | Yes | No | Yes | Yes | No |
| Video Features | Yes | Yes | No | Yes | No |
| Time Fixed Effect | Yes | Yes | Yes | No | No |
| R-Squared | 0.0046 | 0.0028 | 0.0037 | 0.0046 | 0.0021 |
| Users | 4,007,354 | 4,007,354 | 4,007,354 | 4,007,354 | 4,007,354 |
| Observations | 87,984,376 | 87,984,376 | 87,984,376 | 87,984,376 | 87,984,376 |

*Table 1(b): Average Treatment Effect on VCR with Different Combinations of User, Video Features and Time Fixed Effect.. *p<0.1; **p<0.05; ***p<0.01*

| | VPT | VPT | VPT | VPT | VPT |
|---|---|---|---|---|---|
| Treatment | 1.7638*** | 1.7738*** | 1.7798*** | 1.7561*** | 1.7701*** |
| | (0.0437) | (0.0611) | (0.0597) | (0.0540) | (0.0683) |
| User Features | Yes | No | Yes | Yes | No |
| Video Features | Yes | Yes | No | Yes | No |
| Time Fixed Effect | Yes | Yes | Yes | No | No |
| R-Squared | 0.0095 | 0.0062 | 0.0079 | 0.0092 | 0.0047 |
| Users | 4,007,354 | 4,007,354 | 4,007,354 | 4,007,354 | 4,007,354 |
| Observations | 87,984,376 | 87,984,376 | 87,984,376 | 87,984,376 | 87,984,376 |

*Table 1(c): Average Treatment Effect on VPT with Different Combinations of User, Video Features and Time Fixed Effect. *p<0.1; **p<0.05; ***p<0.01*

(b) We use discrete choice models of Logit and Probit, rather than LPM, to predict the coefficients of binary outcome variables VCTR and VCR, where the results in Table 2(a-b) show that the treatment effects of adopting our proposed approach are still significantly positive in these models. Note that the coefficients in Logit and Probit models are different from those reported in the LPM model, as they do not directly represent the treatment effect.

| | VCTR (LPM) | VCTR (Logit) | VCTR (Probit) |
|---|---|---|---|
| Treatment | 0.0617*** | 0.3329*** | 0.1016*** |
| | (0.0011) | (0.0036) | (0.0014) |
| User Features | Yes | Yes | Yes |
| Video Features | Yes | Yes | Yes |
| Time Fixed Effect | Yes | Yes | Yes |
| R-Squared | 0.0152 | 0.0179 | 0.0203 |
| Users | 4,007,354 | 4,007,354 | 4,007,354 |
| Observations | 87,984,376 | 87,984,376 | 87,984,376 |

*Table 2(a): Average Treatment Effect of Adopting Our Proposed Model on VCTR using Logit and OLS model. *p<0.1; **p<0.05; ***p<0.01*

| | VCR (LPM) | VCR (Logit) | VCR (Probit) |
|---|---|---|---|
| Treatment | 0.0342*** | 0.1907*** | 0.0883*** |
| | (0.0004) | (0.0011) | (0.0009) |
| User Features | Yes | Yes | Yes |
| Video Features | Yes | Yes | Yes |

| | | | |
|---|---|---|---|
| Time Fixed Effect | Yes | Yes | Yes |
| R-Squared | 0.0046 | 0.0069 | 0.0091 |
| Users | 4,007,354 | 4,007,354 | 4,007,354 |
| Observations | 87,984,376 | 87,984,376 | 87,984,376 |

*Table 2(b): Average Treatment Effect of Adopting Our Proposed Model on VCR using Logit and OLS model. *p<0.1; **p<0.05; ***p<0.01*

(c) We exclude the recommendation records of the premium video content in our analysis, as VIP consumers might be more willing to watch those videos due to their loyalty to the platform and the exclusivity of the content. As shown in Table 3, the estimation results are not significantly different from the original estimation.

| | VCTR | VCR | VPT |
|---|---|---|---|
| Treatment | 0.0618*** | 0.0342*** | 1.7642*** |
| | (0.0011) | (0.0004) | (0.0527) |
| User Features | Yes | Yes | Yes |
| Video Features | Yes | Yes | Yes |
| Time Fixed Effect | Yes | Yes | Yes |
| R-Squared | 0.0190 | 0.0059 | 0.0181 |
| Users | 3,997,803 | 3,997,803 | 3,997,803 |
| Observations | 85,889,886 | 85,889,886 | 85,889,886 |

*Table 3: Average Treatment Effect of Adopting Our Proposed Model Excluding Premium Video Content. The table shows robust standard errors in parentheses. *p<0.1; **p<0.05; ***p<0.01*

(d) To empirically demonstrate the ability to handle the cold-start problem, we analyze the treatment effect of our model on new consumers (registered on the platform less than one week before the experiment) and newly released videos (posted on the platform less than one week before the experiment). We observe from Table 4 significant performance improvements on new consumers and new videos over the latest production system.

| | VCTR | VCR | VPT |
|---|---|---|---|
| Treatment | 0.0603*** | 0.0338*** | 1.6833*** |
| | (0.0023) | (0.00014) | (0.1233) |
| User Features | Yes | Yes | Yes |
| Video Features | Yes | Yes | Yes |
| Time Fixed Effect | Yes | Yes | Yes |
| R-Squared | 0.0281 | 0.0096 | 0.0348 |
| Users | 103,227 | 103,227 | 103,227 |
| Observations | 827,654 | 827,654 | 827,654 |

*Table 4: Average Treatment Effect of Adopting Our Proposed Model on New Users and Videos. The table shows robust standard errors in parentheses. *p<0.1; **p<0.05; ***p<0.01*

(e) We drop the "cold-start" records from new consumer and videos that we discussed in (d) from the regression analysis, as new consumers might be more willing to click on whatever recommendations they receive due to the novelty effect, and newly released videos might be more appealing to the consumers due to their curiosity. The results in Table 5 show that our treatment effects remain robust even when we exclude those records.

| | VCTR | VCR | VPT |
|---|---|---|---|
| Treatment | 0.0619*** (0.0012) | 0.0343*** (0.0004) | 1.7710*** (0.0529) |
| User Features | Yes | Yes | Yes |
| Video Features | Yes | Yes | Yes |
| Time Fixed Effect | Yes | Yes | Yes |
| R-Squared | 0.0212 | 0.0064 | 0.0247 |
| Users | 3,904,127 | 3,904,127 | 3,904,127 |
| Observations | 87,156,722 | 87,156,722 | 87,156,722 |

*Table 5: Average Treatment Effect of Adopting Our Proposed Model Excluding New Users and Videos. The table shows robust standard errors in parentheses. *p<0.1; **p<0.05; ***p<0.01*

## Appendix IV: Randomization Check Results

We present the results of the Wilcoxon rank sum test of the consumer feature distribution between the control group and the treatment group in Table 6. We can observe from the table that the differences for each consume feature are all statistically insignificant (having p-value greater than 0.05 for all cases). Therefore, we further validate the randomized setting in our online experiment, which enables us to estimate the treatment effect directly.

| Consumer Features | | Mean | 25th percentile | Median | 75th percentile | Variance | p-value |
|---|---|---|---|---|---|---|---|
| Gender | Treatment | 0.003 | -1.000 | 1.000 | 1.000 | 0.990 | >0.05 |
| | Control | 0.004 | -1.000 | 1.000 | 1.000 | 0.990 | |
| Age | Treatment | 40.059 | 30.000 | 40.000 | 50.000 | 20.093 | >0.05 |
| | Control | 40.038 | 30.000 | 40.000 | 50.000 | 20.070 | |
| Province ID | Treatment | 8.238 | 0.000 | 6.000 | 13.000 | 8.930 | >0.05 |
| | Control | 8.154 | 0.000 | 6.000 | 13.000 | 8.880 | |
| City ID | Treatment | 2.036 | 1.000 | 2.000 | 3.000 | 1.550 | >0.05 |
| | Control | 2.029 | 1.000 | 2.000 | 3.000 | 1.550 | |
| Operating System | Treatment | 1.172 | 1.000 | 1.000 | 1.000 | 0.400 | >0.05 |
| | Control | 1.173 | 1.000 | 1.000 | 1.000 | 0.400 | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| VIP Status | Treatment | 0.361 | 0.000 | 0.000 | 1.000 | 0.480 | >0.05 |
| | Control | 0.365 | 0.000 | 0.000 | 1.000 | 0.480 | |
| Activity Days | Treatment | 20.635 | 8.000 | 19.000 | 27.000 | 8.510 | >0.05 |
| | Control | 20.447 | 8.000 | 19.000 | 27.000 | 8.470 | |

*Table 6: Consumer feature-level summary statistics for the control and treatment groups.*

## Appendix V: Additional Simulation Experiment Results

We replicate our simulation experiment presented in Section 5 with different consumer to product ratios (namely 10:1 and 1:10). Specifically, we present the performance of our proposed model and baseline models with 400 consumers, 40 products and 40 consumer, 400 products in Figure 3(a-e) and Figure 4(a-e) respectively. We can observe from these additional experiment results that our proposed model still significantly outperforms the baseline models in terms of both the exploration-based and exploitation-based metrics in these two different settings, further demonstrating the generalizability of our findings.

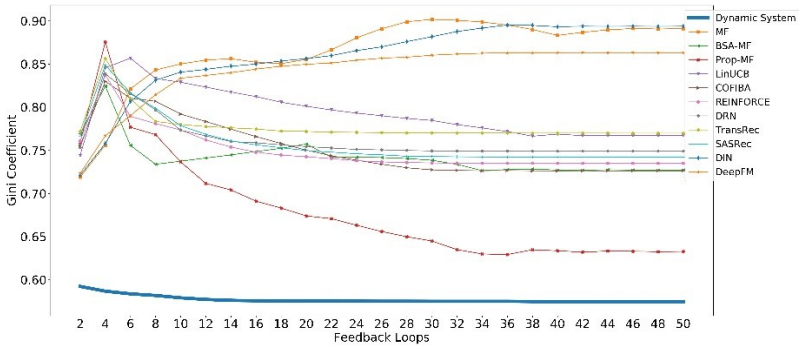


*Figure 3(a): Comparison of Gini Coefficients (40 Consumers, 400 Products)*

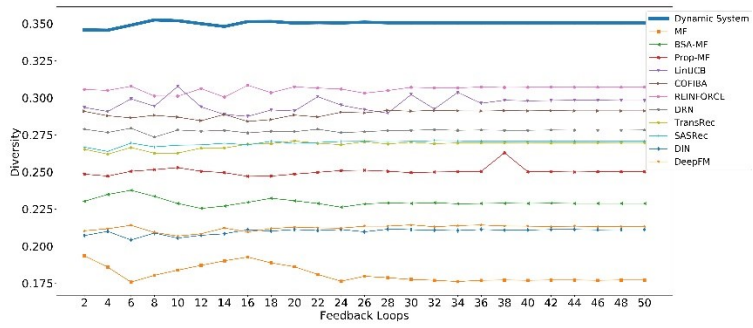


*Figure 3(b): Comparison of the Diversity Metric (40 Consumers, 400 Products)*

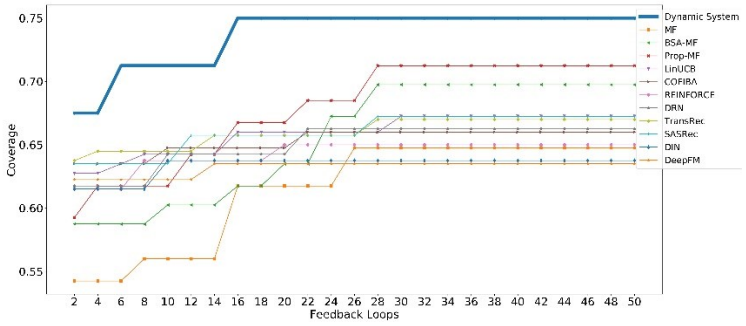


*Figure 3(c): Comparison of the Coverage Metric (40 Consumers, 400 Products)*

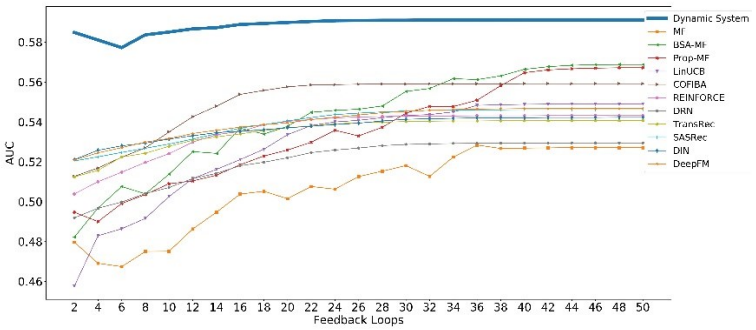


*Figure 3(d): Comparison of the AUC Metric (40 Consumers, 400 Products)*

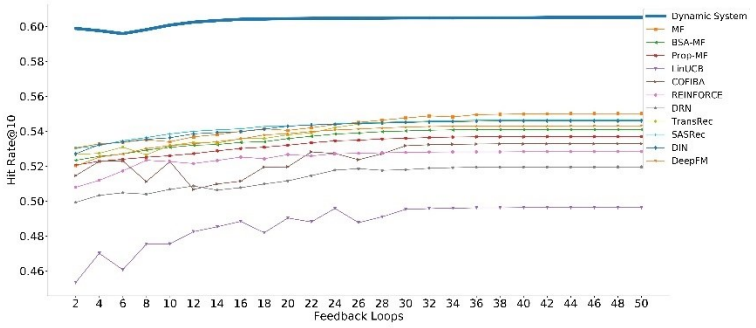


*Figure 3(e): Comparison of the Hit Rate@10 Metric (40 Consumers, 400 Products)*

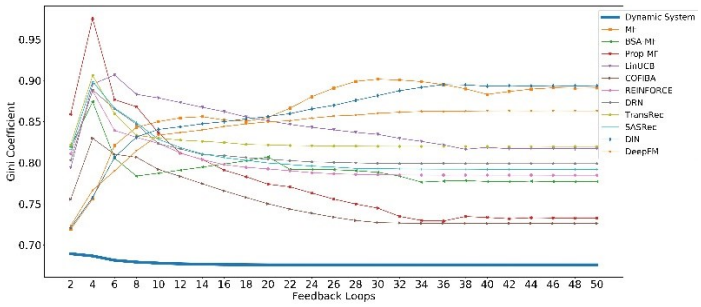


*Figure 4(a): Comparison of Gini Coefficients (400 Consumers, 40 Products)*

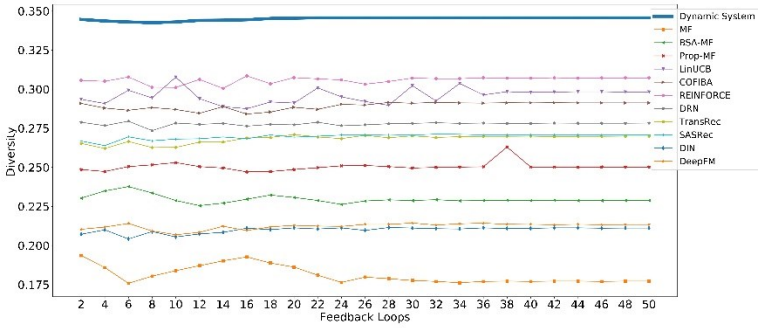


*Figure 4(b): Comparison of the Diversity Metric (400 Consumers, 40 Products)*

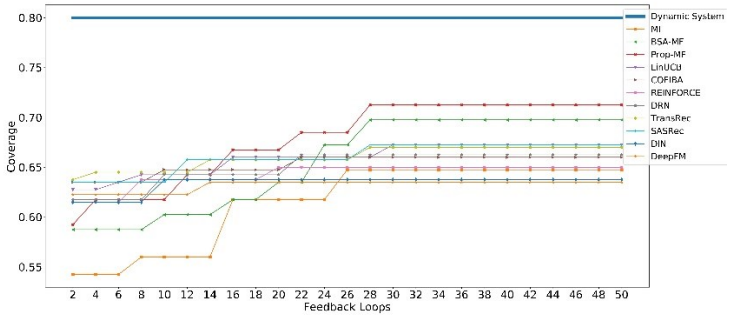


*Figure 4(c): Comparison of the Coverage Metric (400 Consumers, 40 Products)*

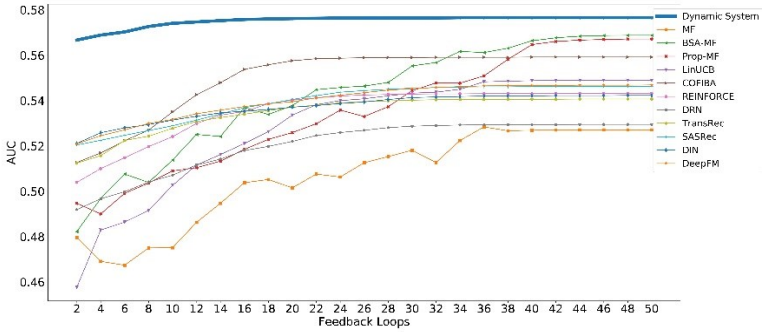


*Figure 4(d): Comparison of the AUC Metric (400 Consumers, 40 Products)*

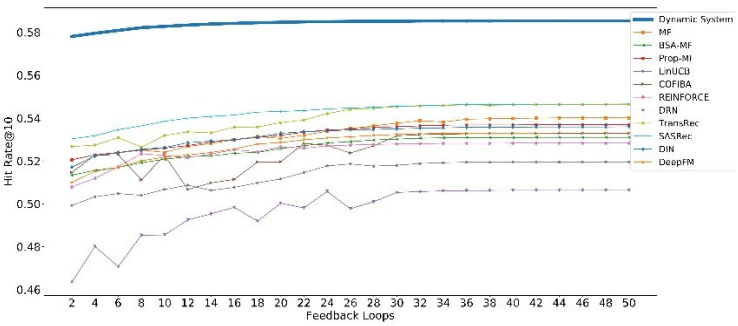


*Figure 4(e): Comparison of the Hit Rate@10 Metric (400 Consumers, 40 Products)*

## Appendix VI: Additional Scalability Analysis Results

We replicate the scalability analysis in Section 4.3 for all baseline models considered in our experiments. Similarly, we train each model using a personal laptop on multiple randomly selected subsets of the three offline datasets, with different data sizes ranging from 100 to 1,000,000 records respectively. We then record the time required for training 100 epochs on each subset, and plot the average training time across three datasets in figure 7. As Figure 5 demonstrates, our proposed dynamic system model requires significantly less training time compared to matrix factorization models (MF, BSA-MF, Prop-MF) and bandit-based models (LinUCB, COFIBA), and the same level of training time compared to other deep learning or deep reinforcement learning-based methods (REINFORCE, DRN, TransRec, SASRec, DIN, DeepFM). Therefore, we confirm that our proposed model achieves significantly better recommendation performance without adding additional computational burden to the training process.

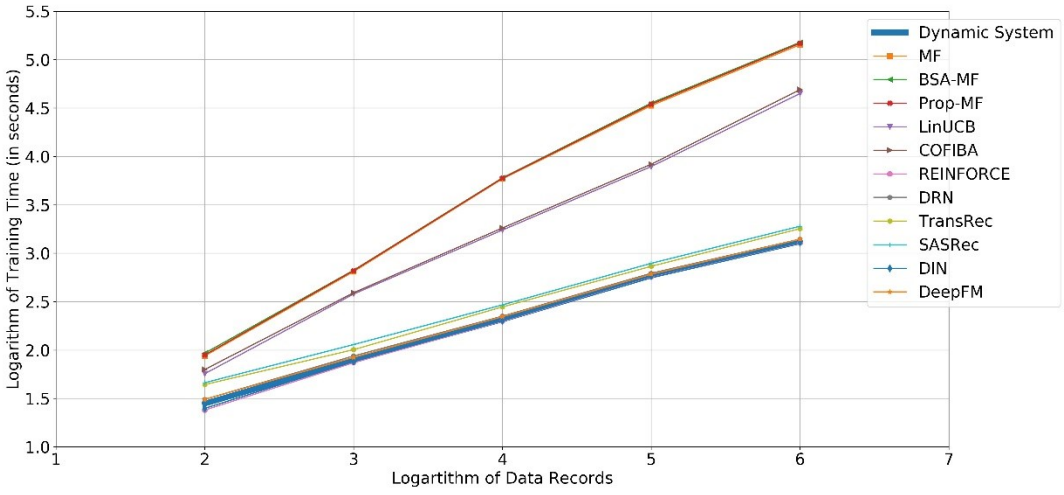


*Figure 5: Scalability Analysis on Three Offline Datasets with Baseline Models*