Then the parameters are updated by moving in the opposite direction of the gradient, yielding:

$$
\begin{aligned}
w_0 &\leftarrow w_0 - \eta \cdot 2(\hat{y}(\mathbf{x_i}) - y_i)\frac{\partial \hat{y}(\mathbf{x_i})}{\partial w_0} \\
w_j &\leftarrow w_j - \eta \cdot 2(\hat{y}(\mathbf{x_i}) - y_i)\frac{\partial \hat{y}(\mathbf{x_i})}{\partial w_j} \\
p_{j,l} &\leftarrow p_{j,l} - \eta \cdot \big(2(\hat{y}(\mathbf{x_i}) - y_i)\frac{\partial \hat{y}(\mathbf{x_i})}{\partial p_{j,l}} + 2\lambda p_{j,l}\big)
\end{aligned} \tag{8}
$$

where $\eta \in R^+$ is the learning rate for gradient descent. Given a training dataset, we iteratively update each parameter according to the gradient until convergence or the maximum number of iterations is reached. Then we can obtain the training model parameters $\Theta = \{w_0, w_1, ..., w_d, p_{1,1}, ..., p_{d,k}\}$.

Navigation Suggestion. Based on users' historical data, we use stochastic gradient descent (SGD) to train a factorization machine model and then employ the trained model to make suggestions when a user trigger a potential suggestion (clicking the cursor on the progress bar of a video). We use Eq. 5 to calculate the suggestion score for each segment based on the trained model and then rank all segments according to the scores. Finally, the top ranked segments are suggested to the user.

6. EXPERIMENTS

In this section, we present various experiments to evaluate the effectiveness of the method based on the observations in Section 4. All of the dataset and codes will be publicly available.

6.1 Experimental Setup

Setting. We split the dataset into training and test sets by time. The training set is comprised of data collected from the first 80% period of time of each course, and the test set is comprised of the data collected from the last 20% period of time of each course. For example, the course FAD opens on 2015.10.9 and ends on 2016.2.1. Then we obtain training data ranged from 2015.10.9 to 2016.1.8, and obtain test data ranged from 2016.1.8 to 2016.2.1. So do the other courses.

Generating Negative Samples. In our experiment, a positive sample corresponds to a complete-jump that truly occurs, denoted as (u, v, p_s, p_e). For each positive sample, we generate several negative samples by choosing different p_es. Given a p_s, there is a list of p_es that users have jumped back to, say, $[p_{e,1}, p_{e,2}, \ldots, p_{e,n}]$. While for a specific complete-jump, there is only one target $p_{e,t}$ ($1 \leq t \leq n$) that the user truly jumps back to. In the remaining list of p_es, we randomly select m p_es to generate negative samples and m is a tunable parameter in the experiment.

6.2 Performance Evaluation

The experiment is conducted in two stages: predicting stage and ranking stage. In predicting stage, we predict the probability of each end position from the same start position. In ranking stage, we rank the end positions by their probabilities predicted in the first stage. The result of ranking stage has two usages: (1) evaluate our method by the measurement hits@n, i.e., the proportion of true end position ranked in the top n. (2) suggest users the top 3 end positions for navigation. First, we conduct the predicting experiments with the following models:

Logistic Regression Classifier (LRC): a logistic regression model is trained and used to predict the probability of each end position.

Table 2: Prediction performance of different method on the dataset

Course	Model	AUC	Recall	Precision	F1-score
Science	LRC	72.46	64.28	25.95	37.37
	SVM	71.92	64.06	25.45	36.42
	FM	**74.02**	**68.36**	**27.61**	**39.28**
Non-science	LRC	72.59	72.96	**69.23**	70.69
	SVM	73.52	79.03	68.39	**73.28**
	FM	**73.57**	**79.82**	67.56	72.88

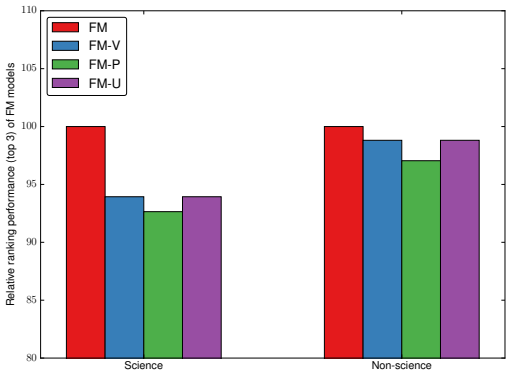

Figure 8: Relative Feature contribution analysis.

SVM: it trains a SVM model to predict predict the probability of each end position.

Factorization Machines (FM): the same features are used to train a FM model and we use it to predict the truly end position of a complete-jump.

We evaluate the performance of different approaches in terms of Area Under Curve(AUC), Recall(Rec.), Precision(Prec.), and F1-Measure(F1) [3]. The result is shown in Table 2.

We can see that the FM model has better performance than other methods. This may be because that there exists correlations among features from videos, start positions and users. The strength of FM just lies in capturing this kind of correlations.

Second, we use the predicting result of FM to compare our automatic suggestion method based on machine learning model with the baseline method based on frequency through a ranking experiment. In predicting experiment, we get the probability of each end position for a given complete-jump. Ranking these end positions by their probabilities produced in the prediction stage, we get an ordered list of them. The true end position of this complete-jump is ranked by a certain order in the list. We use hits@n to measure the suggestion of the true end positions of all complete-jumps. In the method based on frequency, we rank end positions by their frequency for a given complete-jump, and also use hits@n to measure the suggestion of the true end positions of all complete-jumps. Table 3 shows the result of the ranking experiment. We can see that our method based on machine learning model clearly outperforms the method based on frequency both in science courses and non-science courses.

Feature Contribution Analysis. Here we examine the contribution of different categories of features: users' general performances on videos (V), users' general performances at a specific position (P) and the preferences of users (U). First, we use all three categories of features to train a FM model. Then we respectively remove one of the three categories of features, denoted as FM-V, FM-P and FM-