5 An Adaptive Implicit Feedback Model

Collaborative filtering systems may use different types of user feedbacks, being explicit or implicit. However, the explicit valued user feedback according his preference is not suited for every scenario [55]. For breaking news short clips, for example, an user cannot rate a content according to its interest or based on the relevance of the content for him, but based on how good or bad are the news. For example, an user can rate with a low value a tragedy news video even if the content is relevant and interesting for him. In those cases, the explicit valued feedback becomes very subjective.

To tackle this subjective evaluation, it is possible to use implicit feedback to infer the real relevance of a video to an user. However, to improve quality of recommendations, this inference must exactly reflect the actual user interest. The model used to address this challenge, and described in the following of this chapter, is also detailed in the ACM Recommender Systems workshop paper by Pereira et. al. [71].

There are different ways to implicitly infer user's preferences in a specific video. The most basic model is the binary one, where the basic play action means that an user shown an interest by a video. However, this model doesn't represent the actual user interest. If the user stops or leaves the video a few seconds after play (probably because the content is in fact not interesting for him/her), the recommendation system made a wrong assumption assigning to it a positive feedback.

An alternative approach could be to consider a positive feedback only when the user watches the video completely. However, this could also generate a wrong feedback. For example, in some types of videos, such as movies or series, their last minutes usually exhibit the credits, and generally user leaves the content before the video end.

An even better approach would be if a multi-value feedback approach was used, and not a binary one. With this approach one can use different feedback values, such as star ratings, and could for example, increase the feedback value according to how long the user was actually watching the video. For example, it is possible to use a 10 level rating system that gives feedback 1 for users who watched 10% of video, 2 for users who watched 20%, and so on, linearly associated with playing duration. This model describes with more accuracy the actual user interest.

However, for some types of video content, it could also leverage wrong assumptions. For example, in a sports short clip video (a touchdown in football), the interest spot may be concentrated in the middle of the video, which means that if users left the video after 70%, or 80% or even 90%, all of them should have almost the same rating.

To make it more clear, lets analyze the user behavior when watching different types of content. Figure 15 illustrates a real user behavior data from Globo.com users, based on more than 500 Million video views, from more than 4 million different videos including movies, full episodes, and news, sports and entertainment short clips.

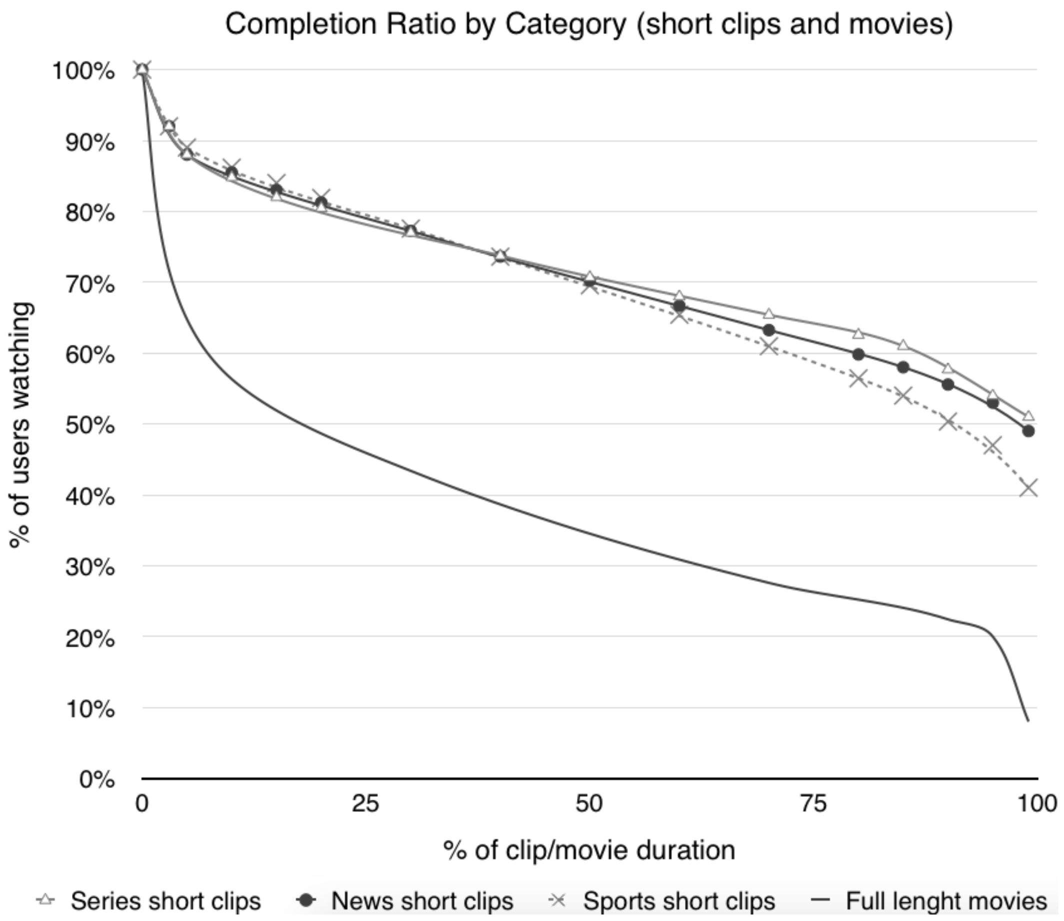

Figure 15 - Completion ratio by category of short videos and clips

In this figure, the initial abandon rate for movies is greater than for short videos, which is an expected behavior. However, when comparing short videos of different categories, one can see a difference in behavior especially in abandon rate in the last 50% of the video. In short clips of soap operas and series, a greater number of users remains playing until their ends, as in sports, the abandon rate increases in the end, after the video delivers the main information desired by the user.

However, we can not generalize this behavioral difference by content category. Different programs in the same category may have very different consumption patterns, as well as, ultimately, their own videos with each other. Figure 16 illustrates this difference, showing how the consumption pattern of short videos of the same program (National Journal - Jornal Nacional, the TV news with the largest audience in Brazil) when compared the the average of its news category.

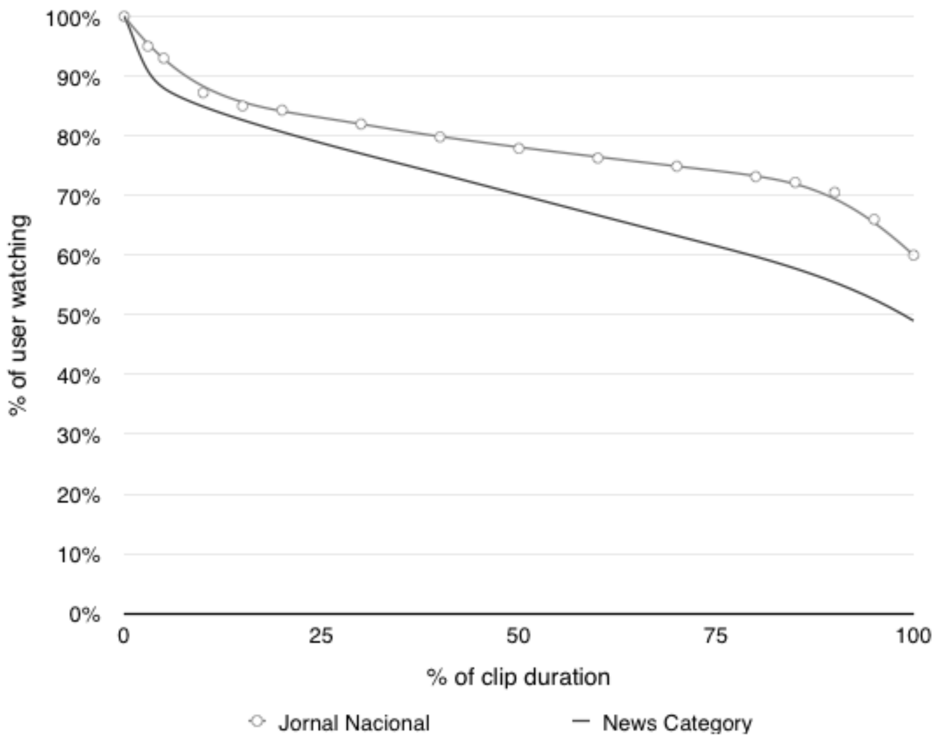

Figure 16 - Consumption pattern of short videos of the National Journal compared the the average of its category

This consumption behavior observed in Figure 15 and Figure 16 confirms the idea that the use of an implicit binary feedback on the video start is not the best approach to infer the interest of users. Moreover, to assign a scale linearly associated with the watched percentage also is not a good strategy, because the consumption profile is nonlinear and it varies greatly depending on the category of content, duration and other parameters which directly influence the user's interest.

So this thesis also proposes an implicit feedback model that takes into account the different patterns of consumption in the inference of interest, and that is able to dynamically adapt to changes in consumption patterns for each video.

To create a model that considers the different consumption patterns, one can use a feedback score scale individualized by video, that assigns increasing scores as the user keep watching it. The purpose of this feedback score scale be associated and individualized by video is inferring user interest in a particular content, based on the behavior of other users consuming exactly that same content.

As discussed above, use the same feedback score scale for all kinds of content can result in an inaccurate inference, and that is exactly what the proposal to have an individualized scale is tackling. However, to use only an individualized scale is not enough. As one can see in Figure 15 and Figure 16, the abandon rate is higher at the beginning of the video, especially in the first 10% of the content. This behavior is caused by users who eventually accidentally clicked on the video or immediately lost interest for the content, or thwarted in some way. Thus, the inferred feedback from these users should be zero or negative and should not be considered a score of interest.

To cope with this behavior, the proposed model only associates a score when an user watches at least 10% of the video. Figure 17 and Figure 18 illustrate the proposed model, where two separate videos, video A and video B have different consumption curves and hence different feedback scales, initializing assigning scores from only 10%. Note that for the video A, the user who watches 50% of the content will receive the score 2, while for the B video the assigned score would be 3.

Once the proposed model is individualized by video, where, the scale of feedback scores of a video varies according to the behavior of users who watch that particular video, a new problem arises. The behavior of users is not constant and unchangeable, which means that consumption curves varies over time. Thus, the model must be constantly adapted, so that the feedback scale varies constantly adapting to the pattern of consumption.

The proposed model is associated with the consumption pattern, which means that it is flexible so that the feedback scale varies depending on the variation of behavior. This is a fundamental point that should be considered in implementation.

Another important point is the cold start problem, that is, immediately after a video is published there is not enough information to determine an individualized scale, since there isn't an established pattern of consumption. Thus, for these cases, one can use a scale based on the consumption pattern of the same program, or ultimately the same category.

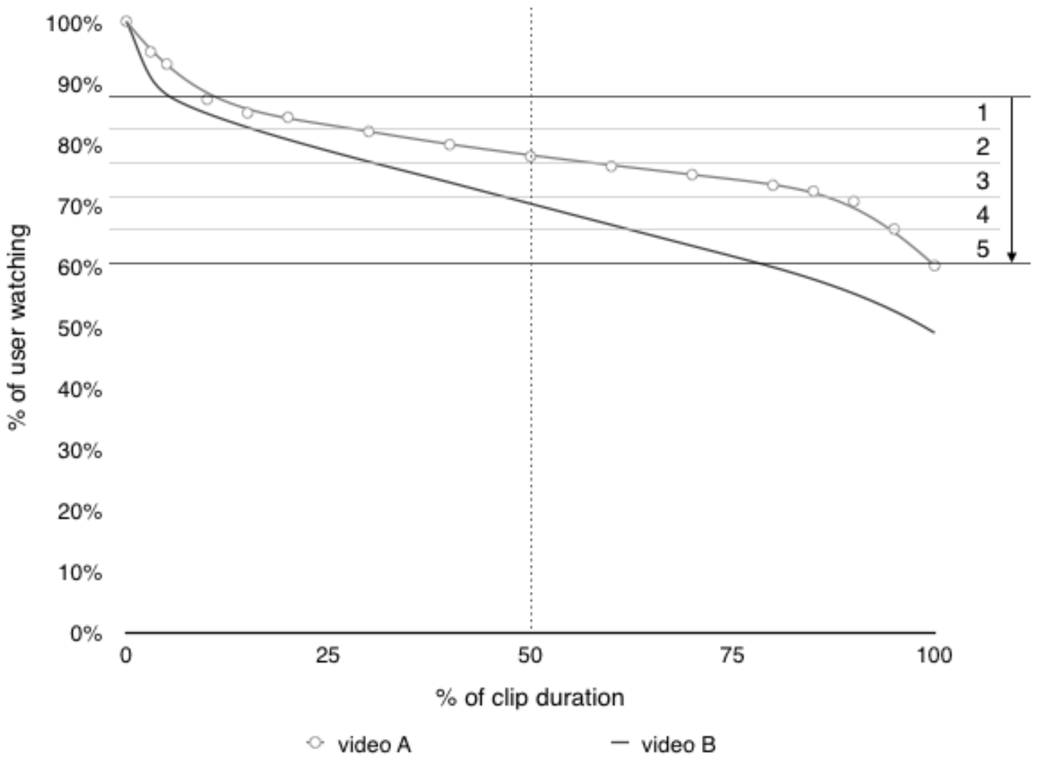

Figure 17 - Video A and video B have different consumption curves and hence different feedback scales

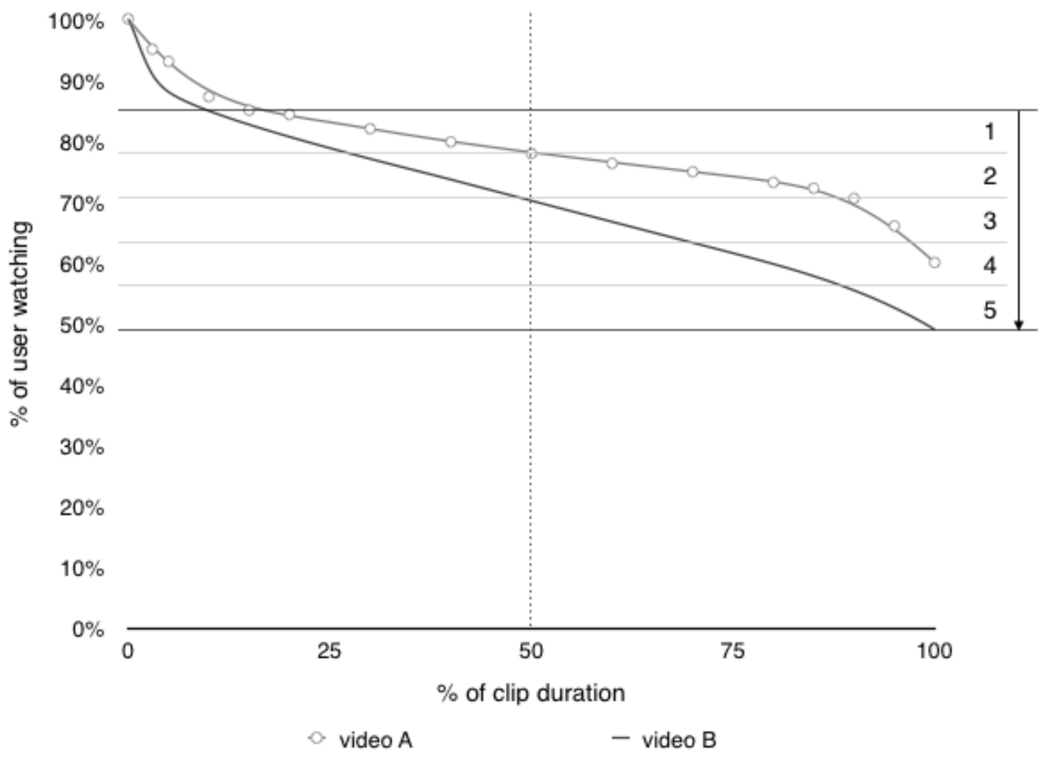

Figure 18 - Adapted feedback scales for videos A and B

To evaluate the proposed feedback model, it was necessary to change the binary feedback architecture presented in the previous chapter because of the following reasons: 1) consider a range of multi-valued feedbacks; 2) to store the consumption patterns and consider them in the feedback scores.

A key characteristic that enables the calculation of recommendations in real time proposed in the architecture presented in previous chapter is the use of SETs for storing feedback and the use of the intersection of SETs to calculate the cosine similarity. This is only possible because the architecture considers only binary feedback. To work with multi-valued feedbacks, would be necessary to store the feedback values in a matrix, which consequently generates numerous scalability issues when there are dozens of millions of items and users. These problems are also described previously.

So, to maintain all the architectural advantages such as scalability to dozens of millions of items, and the ability to generate real-time recommendations through the stream feedback processing, and the flexibility to run on a cloud platform allowing for the adaptation to changes in demand, whether in processing feedbacks or the volume of data, the adopted approach was to instantiate multiple architectural replicas, one for each level of the feedback scale. For example, assuming a scale of five levels of feedbacks, as seen in Figure 17, it would be necessary to create five instances, each instance would continue to process separately, and considering only binary feedback. This means that a multi-valued feedback must first be decomposed into multiple binary feedbacks that will be used as input to each of the instances. For example, a feedback value of a 4, will be broken down into four feedbacks value of 1, and will be sent to the instances 1 to 4 of architecture. In this case instance 5 would receive a 0 feedback.

In this approach, it was necessary to create an extra logic on the Layer I to transform a multi-valued feedback on a set of binary feedbacks, and change the service that provides recommendations to be able to consume the similarity values between two items from multiple instances.

It was also necessary to change the way that the player reports playback information. In the binary feedback described in previous chapter, the player was sending only the information that an user started to watch a video, through an API call like this: *http://:host:port/item/:UUID/user/:UUID.* However, this single call does not allow the reporting of playback progress needed to construct the playback behavior. So, in order to have this information, the player logic was changed to send multiple calls to report playback progress updates for every 5 seconds. In the server side, this calls were used to build the playback curves for each video and this information was used to obtain the valued feedback after playback stops, and to decompose this valued feedback in multiple binary feedbacks.

At this point, to solve the problem of having different similarities between items for each of the instances, weights were used referring to the scale values corresponding to each instance. For example, considering that we have five levels of feedback, we will have 5 instances processing independently and working as if all feedbacks were binary.

The end result is that for a given pair item-item, we will have a value of similarity for each instance. In the case, we will have 5 values. Thus, to get a final similarity, weights were used for each similarity value, i.e., for the first instance the weight of similarity is 1, for instance 2 the weight is 2 and so on. At the end, the result for similarity between a pair of items is the sum of each output of the instances multiplied by their respective weights.

Formally, the similarity between two items I_1 and I_2 considering L levels of feedbacks could be expressed as the sum of the similarity between I_1 and I_2 for each level, multiplied by the weight of the corresponding level, according to the equation below:

$$sim(I_1, I_2) = \sum_{l=1}^{L} l \left(\frac{\#\left(Set(I_{1,l}) \cap Set(I_{2,l})\right)}{\sqrt{\#\left(Set(I_{1,l})\right)} \cdot \sqrt{\#\left(Set(I_{2,l})\right)}} \right)$$

where # represents the cardinality, and $I_{1,l}$ and $I_{2,l}$ represents the item vectors in each respective level l of feedback.

As shown, this approach creates a new similarity metric that uses the same basic similarity computing process as described in previous chapter when considering only a single binary feedback, however, repeatedly for each feedback level and with a consolidation to obtain the final similarity.

To validate the proposed model, a real production scenario used for video recommendations was implemented. The next chapter describes such implementation and a comparison between the proposed model with the basic binary feedback.