For a Better Experience in Podcasting : Adaptive Prefetching Based on User Access Patterns

W. H. Rankothge and G. Dias

Abstract—Podcasting, including video podcasting, has become remarkably popular in recent years. However, it is still common for a user to experience pauses when watching video or listening to audio online. These are mainly due to insufficient bandwidth to stream the content at full speed. Delays introduced by buffering before and during video or audio playback can be annoying and can discourage users from watching video and listening to audio on-line. Most studies relevant to multimedia with limited bandwidth have concentrated on technologies to transmit multimedia efficiently in a limited bandwidth network. Few of them address the problem of viewing multimedia content in a limited bandwidth network from the perspective of a consumer.

This research introduces a method to prefetch video and audio content to eliminate or reduce pauses before and during video playback. Prefetching can be done overnight or when the network usage is low and can be stored locally. Videos may be prefetched fully or partially. Partial prefetching reduces the network traffic for prefetch misses, but the prefetched portion in the buffer can compensate for any insufficiency of bandwidth and absorb network delay, resulting in a smooth play out of the video. We use the Naive Bayesian Classification method to predict videos of the feeds selected by the user, he is likely to watch. The prediction is based on the title and description of the videos.

Index Terms—Podcasting, Prefetching, Naive Bayesian Classification, Limited Bandwidth Networks.

1 Introduction

It has been years since the Internet took its first steps towards becoming a mass medium in the world, and nothing epitomizes modern life better than the Internet technology. For better or worse, Internet technologies have infiltrated every aspect of our society and have extended their influence into our everyday life. The changes driven by the Internet have an enormous impact on the conduct of every aspect of our society business, government, education, and private life.

Internet with its many rapid developed applications is mainly used as a worldwide source of information which helps people to access information, knowledge and news quickly and easily from any location. There are lots of free podcast sites in the internet which share information, knowledge and news with friends, family, customers and thousands of people around the world. They are reliable, simple and quick to access, so they have become more popular in the internet community.

A podcast is no different than a webcast, a show that is broadcast over the web and is broken up into parts or episodes. Most podcasts are similar to news radio programs and deliver information on a regular basis, while some podcasts are comedy shows, special music broadcasts or even gospel. Podcast sites publish podcasts that people can consume on the web and subscribe to in order to get automatic updates. With the Internet, user's subscription to a podcast is instant, and the delivery of new podcast occurs as soon as it is available.

Besides the remarkable popularity of podcast sites, user experience with watching video from these sites can vary extensively. It is common that a user experiences a pause when watching a video or listening to an audio on line. Delays introduced by buffering before and during a video or audio can be quite annoying and can potentially discourage users from watching video. These interruptions during video playback are mainly due to the limitations of network capacity at the point of reception.

In this research, podcasts are prefetched in order to eliminate the potential of pauses during video playback and decrease the service delay. Prefetching can be done overnight, or when the network usage is low and can be stored locally. If a video has been even partly prefetched, the prefetched portion in the buffer can compensate for any insufficient bandwidth and absorb network delay, resulting in a smooth play out of the video.

Prefetching technique works by predicting a set of videos that are likely to be watched by the user in the near future based on his previous activities and avoids downloading videos the user is not interested in.

2 Related Works

Because of the high bandwidth requirements in a multimedia, watching a video or listening to an audio online in a limited bandwidth network has become a challenging task. Interruptions during video or audio playback can results user frustration and it will have a negative impact on the flourishing popularity of online multimedia files. With the rapid development and sky-scraping attraction of podcast sites and news sites, exploring an efficient as well as an economical technology to view multimedia content in a limited bandwidth network has become an emergent research area.

Most of the studies relevant to multimedia with limited bandwidth have concentrated on the technologies which can

• *W.H. Rankothge is currently pursuing PhD in Pompeu Fabra University Spain. E-mail: windhya.rankothge@upf.edu*
• *G. Dias is a professor at University of Moratuwa, Sri Lanka, E-mail: gihan@uom.lk*

be used to transmit multimedia efficiently in a limited bandwidth network. Only few of them were addressing the problem of viewing multimedia content in a limited bandwidth network, with the perspective of consumer. There are three most common techniques used for data prefetching in VoD systems: random prefetching, popularity aware prefetching and data mining based prefetching. In the research "Supporting VCR functions in P2P VoD services using ring-assisted overlays" [1], random prefetching has been used to prefetch the data in local cache before a seek operation is carried out. The popularity aware prefetching used in "Distributed Prefetching Scheme for Random Seek Support in P2P Streaming Applications" [2] research uses segments access probability for prefetching the content. In this technique logs are maintained by a management server (also called tracker) regarding users access pattern. The statistics gathered on user requests are used to determine the optimal number and placement of replicates for each individual video file. The research "VOVO: VCR-Oriented Video-on-Demand in Large-Scale P2P Networks" [3], used State maintenance and data mining as another technique for prefetching more desirable segments of video.

When moving along with above mentioned three prefetching techniques, research "Architecture for Cooperative Prefetching in P2PVideo-on- Demand System" [4] observed a trade-off between user behaviour and overhead in different aforementioned prefetching techniques. Moreover either the management server or each peer has to perform necessary computation which increases computational overhead. To overcome the above mentioned problem, they proposed a new prefetching technique called "cooperative prefetching technique". In this technique, each peer maintains the record of playback segments by other peers in the same session. This information is obtained through gossiping.

On the opposite of Video-On-Demand applications, in News-On-Demand applications, data have more complex structures and their lengths are relatively short; typically less than 5 minutes. "Semantic based Prefetching in News-On-Demand Video Servers" [5], research have developed a powerful prefetching technique which uses run time information as well as semantic information about the structure of the requested streams. This was the first research which addresses the potential benefits of prefetching in the context of short overlapping videos for the stratification technique.

"Watching user generated videos with prefetching" [6] is a research that addresses the need for prefetching by showing that video playbacks of videos of YouTube is often unsatisfactory. Researches have introduced a series of prefetching schemes: conventional caching scheme, search result-based prefetching scheme, and recommendation aware prefetching scheme. Proposed prefetching scheme conserves bandwidth by prefetching only a prefix of a video, since a video clip can playback smoothly if a sufficiently large prefix of the video is prefetched.

PREP (PREdict and Prefetch), a prediction-based prefetching scheme[7]; supports VCR-like operations which can be used in any practical gossip-based P2P VoD systems. A gossip protocol [8] is a style of computer-to-computer communication protocol inspired by the form of gossip seen in social networks. In the design of PREP, researchers have modeled user interactive behavior based on the large volumes of user viewing logs collected on the tracker via state abstraction and reinforcement learning (RL).

3 Methodology

Despite the tremendous popularity of podcasting sites, user experience with watching videos from these sites can vary significantly, it is common that a user experiences a pause when watching a video online. To further demonstrate the need for prefetching, this research performed an experiment to evaluate user experience in watching videos from "Ted Talks" podcast site. In particular, measurements of how likely it is that a user experiences pauses during video playback and how long the pauses are, were taken.

Volunteers were asked to use Wireshark network protocol analyzer on their computers to capture the traffic. The process of detecting pauses in video playbacks was automated to make the process easy for the volunteers and as precise as possible. The volunteers only had to start the capturing before clicking on the link to a video and stop the capturing after the video playback ends. The traced data from Wireshark was used to estimate whether pauses in a video playback occurred. The user experience on watching podcast contents online, varied across different locations. Results of this experiment lead to the conclusion that podcast subscribers indeed experience disruptive playbacks on podcast sites, especially when they watch videos with higher quality. Although a user can choose to wait for a video to buffer before she starts watching, it is undesirable. The results of this experiment motivated this research to devise a video prefetching approach that has the potential to reduce or even eliminate pauses during video playback.

The main principle of prefetching is to retrieve content from the source before it is requested by a user and store it in a location that can be accessed by a user conveniently and fast. The prefetching scheme proposed in this research consists of two components: Prefetching Agent (PA) and downloader.

The module that performs prefetching is called the PA and it has storage to store prefetched prefixes of videos. PA determines the videos to be prefetched, retrieves their prefixes from relevant podcast site, and stores them. In addition, the PA can perform caching.

In order to perform prefetching, the PA needs to determine the set of videos to be prefetched. Given subscribed podcast sites for a client, the PA needs to predict a set of podcast videos that are likely to be requested in the future.

For this research, description of each video will be taken into account to decide whether user going to like a video or not. In any podcasting site a description of each video will be given and it includes details of the contents of the video, posted date and people who are appearing in the video.

Since the description of a video, which is a text, is used to select videos, the main approaches of selecting videos for prefetching will be filtering based on statistical study of user access patterns. In fact it is a binary classification and all kinds of text classification methods can be used to select videos for prefetching, such as KNN Algorithm, SVM, Decision Tree and Bayesian Algorithm. For this research Bayesian method was used as the algorithm to select videos for prefetching.

There were many solid reasons for this selection; Bayesian method takes the whole text into account – It recognises keywords that identify videos that are going to be disliked by the user, but it also recognises words that denote videos that are going to be liked by the user [9]. It considers the most interesting words (as defined by their deviation from the mean) and comes up with a probability that a videos which is going to be disliked by the user.

A Bayesian filter is constantly self-adapting by learning from new videos liked by the user and new videos not liked by the user, the Bayesian filter evolves and adapts to new video selection techniques [9].

Bayesian technique is sensitive to the user and it learns the video selections of each user and understands that, for example, the word 'Microsoft' might indicate a video that is not going to be watched by the user, if the user's running the filter is, computer, whereas it would not indicate it as a video that is not going to be watched by the user if the user is a person who is always watching technological videos [10].

3.1 Training the Naïve Bayesian network for video selection

The first step to construct Bayesian classification system is to give the program a working data set that has already been established as either positive or negative. Traditionally this process is done using a pre-defined data set called a corpus [11]. In the context of video selection, a corpus is simply a text collection of either "liked videos" or "not liked videos" that have been collected accurately from each user based on his access patterns.

To get the initial interests of the user, the initial data system was developed and it noted down the different interests of a user. Based on the user's interest, user can answer the "Do you like to watch this video" question and the given answer will be used to build the initial "liked videos" and "not liked videos" corpora.

Bayesian system will scan the entire text of a given video contained in the corpus. This includes the details of the contents of the video, posted date and details of people who are appearing in the video. Scanning the video details requires dividing the context into individual segments of text, often referred to as tokens [12]. A token constitutes of all standard alphanumeric characters, dashes, and apostrophes. Any character that does not fit into this category is considered a token separator. Each token is stored into a hash table, where the key of the hash would represent the token and the value that the key resolves to represent the frequency of the given token [13]. Because there are two corpora ("liked videos" and "not liked videos"), two hash tables of this nature must be constructed.

After frequencies from each corpus have been collected, another hash table needs to be constructed. This hash table will contain all the tokens collected, and the values that resolve from the keys of this hash table (like the others, the keys of this hash table are the tokens collected) will be the ratio of the frequency of a not liked videos occurrence to the frequency of a liked videos occurrence. This number is referred to as the "interest". The "interest" is the factor that the classification process uses in order to run the Bayesian chain rule to determine whether or not a video detail belongs to the class liked videos or not liked videos.

3.2 Usage of the Naïve Bayesian network for video selection

The Naive Bayesian video selection can be conceptualized into the model presented in Fig. 1.

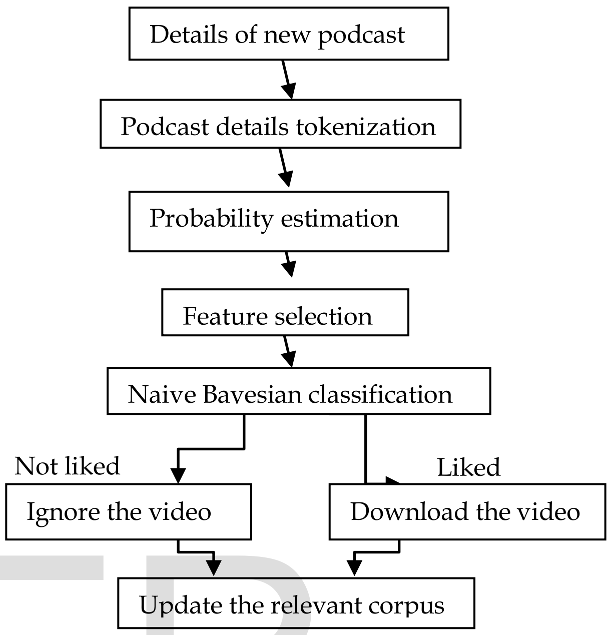

Fig. 1 Naive Bayesian video selection model

3.2.1 Details of a new podcast video

In order to prefetch and download podcasts, first of all users should provide the list of podcast sites that they are subscribed to. This list is dynamically maintained and user has the capability of editing it at any time. PA will read this list periodically and visit the relavant RSS sites. Title of newly published podcasts, details of the contents of the podcast, posted date and download link will be shown in these RSS. These details will be read from the XML of the relevant RSS by the PA.

3.2.2 Podcast details tokenization

This is the process of tokenizing details of each podcast video using carefully selected delimiter. In this research, Bayesian filters work with simple word-by-word tokenizing [14], where the text is separated into words and the words have their own values in the token-dictionary. In his case, one word will be one token.

Let the details of the new podcast be "Violence between competing unions at South Africa's mines is threatening to weaken Africa's largest economy".

During the tokenization process this text will be separated into words and each word will be considered as a token. Most commonly used words such as "a", "of", "to", "for" will be ignored. So the list of tokens generated from the above podcast details would be [Violence, Between, Competing, Unions, South, Africa, Mines, Threatening, Weaken, Largest, economy]. The corpus should be updated with all the new tokens from the processed video details and after the tokenization; the tokens are handled separately, considering the order of the tokens.

3.2.3 Probability

The most straightforward approach for estimating the probability of each token is to divide its count by the total counts. This estimate approaches the theoretical probability as the sample space increases. If the entire population is available an exact probability can be obtained, but for most cases when the sample space is finite the Maximum Likelihood Estimate (MLE) will be in the neighbourhood of the probability.

$$P_{MLE}\ (x_i) = \frac{x_i}{N}$$

where xi is the item and N is the total number of training instances.

The problem regarding this estimator is that it estimates the probability of unseen items to be zero. This is sometimes referred to as the zero-frequency-problem. In the case of a Naive Bayesian classifier where estimates are combined from two corpuses a single zero estimate of a token could result in an overall “not liked video” probability of 0. The same problem remains for rare words, those with low frequencies. For example a probability estimated from a token occurring once in the good corpus and ten times in the “not liked video” corpus is less reliable than a token with 10 occurrences in the “liked video” and 100 in the “not liked video” corpus. Even though the combined probability is the same the latter is more reliable.

Schemes do exist to solve the problems concerning Maximum Likelihood Estimation for cases with sparse data. It is usually said that these schemes perform smoothing. Smoothing is the method of moving probability mass from seen to unseen items.

First of all, the list of generated tokens is taken and in this token dictionary two values are recorded for every entry (every token). Let them call “N1, N2”. All are natural numbers (counters). Note that the numbering must be started from one to avoid dividing by zero in the future calculations.

N1 = No. of “not liked video” details, where the word has existed
N2 = No. of “liked video” details, where the word has existed

When a new podcast details is available, the token dictionary is searched for all of the words, included in the details of the podcast. (Note: in this research algorithm follows word-by-word tokenization.) Two sets of words are handled: one is the set of words matching the dictionary (an update will be necessary at value N1 or N2), other is the set of words not included in the details of the podcast (no update will be necessary).

Furthermore besides the token dictionary, another two numbers must be stored: the number of liked video and the number of all not liked video.
Let them call “LIKE and NOT LIKE”.

The final result of the filtering has the follow form:

Furthermore besides the token dictionary two numbers must be stored, the number of liked video and the number of all not liked video.

Let them call “LIKE and NOT LIKE”.

The final result of the filtering has the follow form:

$$Odd = Ratio\ of\ \frac{P}{1-P}$$

This formula is a standard idealization of to express odds. In this research the calculation involves the steps listed below:

$$\text{ALL (No. of all videos)} = \text{LIKE} + \text{NOT LIKE} \qquad (1)$$

(Number of liked videos, added to the number of not liked videos)

Let us call a word “matching word”, if the word has existed both in the details of the video and in the token dictionary.

$$P(\text{"matching words"} \mid \text{"video is a not liked video"}) = \Omega \text{ for all matched words } \frac{\text{N1 value of the current word}}{\text{NOT LIKE}} \qquad (2)$$

$$P(\text{"matching words"} \mid \text{"video is a liked video"}) = \Omega \text{ for all matched word } \frac{\text{N2 value of the current word}}{\text{LIKE}} \qquad (3)$$

$$P(\text{"video is a not liked video"}) = \frac{\text{NOT LIKE}}{\text{ALL}} \qquad (4)$$

$$P(\text{"video is a liked video"}) = \frac{\text{LIKE}}{\text{ALL}} \qquad (5)$$

$$P(\text{"video is a not liked video"} \mid \text{"matching words"}) = P(\text{"video is a not liked video"}) * P(\text{"matching words"} \mid \text{"video is a not liked video"}) \qquad (6)$$

$$P(\text{"video is a liked video"} \mid \text{"matching words"}) = P(\text{"video is a liked video"}) * P(\text{"matching words"} \mid \text{"video is a liked video"}) \qquad (7)$$

$$Final\ results = \frac{P(\text{"video is a not liked video"} \mid \text{"matching words"})}{P(\text{"video is a liked video"} \mid \text{"matching words"})} \qquad (8)$$

3.2.4 Naïve Bayesian Classification

Since probability value is a number it can be turned into a binary result and define a boundary value “x”. If the result is over “x”, the probability of not liked video is higher than the liked video, with the meaning of that the video is will not be liked by the user. If the result is between 0 and “x” the liked video classification has higher probability, so the video should be categorised as liked video. So if the If the result is over “x”, PA will decide to ignore this video while if the result is between 0 and “x”, PA will decide to download the video.

3.2.4 Downloading the Podcast

According to the bandwidth usage of the relevant network user has the capability of configuring the bandwidth thresholds to consider when starting the download of podcasts. PA will monitor the network bandwidth usage in real time and whenever network bandwidth usage reach the threshold given by the user, PA will be automatically activated. PA will start to prefetch podcasts based on the user interests and prefetched podcasts will be downloaded to the local machine. If the network bandwidth usage increased suddenly, PA will pauses its download temporarily and keep monitoring network bandwidth usage until it reaches the threshold given by the user again. All the downloaded podcasts will be saved in a local storage of user’s machine and PA will be responsible for podcasts saving process. Since PA has been configured to perform caching, before downloading each selected podcast, PA will check if the prefix of the podcast exists in its storage. If so, it serves the client with the prefix of the podcast that has been already downloaded to the machine.

4 Results and Discussions

As described in the methodology chapter, to further demonstrate the need for prefetching, this research performed three experiments to evaluate user experience in watching videos from “Ted Talks” podcast site. 15 volunteers were asked to capture video download traces from various environments representing different locations and network access technologies as shown in Table 1.

Table 1: Number of users in different environments

Environment	Location	Network Technology	No. of Users
E1	University 1	Campus LAN	3
E2	University 2	Wi-Fi	5
E3	Home 1	Wireless Network	3
E4	Home 2	ADSL	2
E5	Company	Company LAN	2

The process of detecting pauses in video playbacks was derived automatically by analysing video download traces to make the process easy for the volunteers and as precise as possible.

First experiment was carried out to determine how likely it is that a user experiences pauses during watching the video from various and network access technologies.

Fig. 2 shows the number of smooth playbacks and disruptive playbacks for each environment.

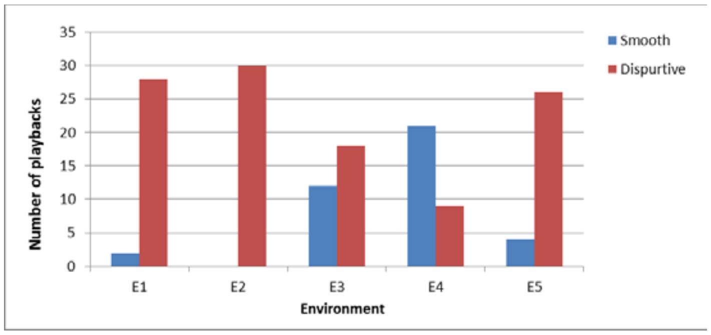

Fig. 2 Podcast video playback quality

The result of the first experiment shows that all most all the environments with different technologies contained playbacks with pauses. To be precise 111 out of 150 playbacks experienced playbacks and its percentage is 74%. Unfortunately university environments with campus LAN and Wi-Fi technologies showed the highest number of playbacks with pauses and this is mainly because of the limited bandwidth of the campus network. Since ADSL technology gives you a guaranteed bandwidth, home environment with ADSL technology showed the least number of playbacks with pauses.

Second experiment’s goal was to calculate the number of pauses in the interrupted video playbacks and Fig. 3 shows the results of the experiment. There were 111 video playbacks which experienced pauses and for each playback, number of pauses varies from 1 to 30.

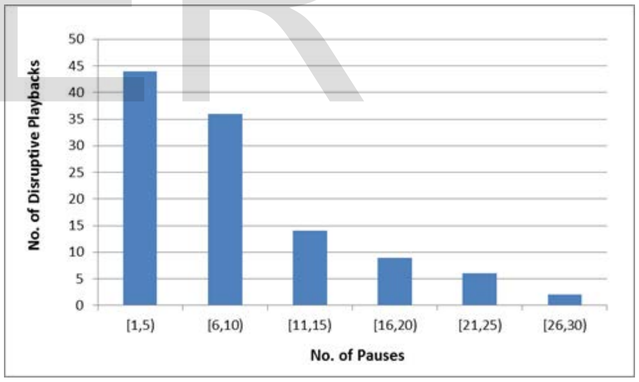

Fig. 3 Number of pauses in disruptive playbacks

According to the results, 31 out of the 111 disruptive playbacks experienced more than 10 pauses and as a percentage it is 28%. Considering that the duration of the videos in this experiment’ datasets ranges between 3 to 10 minutes, pausing as much as 10 times or more could have resulted in extremely unpleasant experiences to users. 72% of the disruptive videos had pauses less than 10 and even this number would have frustrated users because they were watching videos with small durations.

Finally, for the third experiment, the time that a user had to spend waiting for the videos was computed. In Fig. 4, the results of the experiment is shown as the ratio between accumulated pause time and accumulated video length from all the playbacks in each environment.

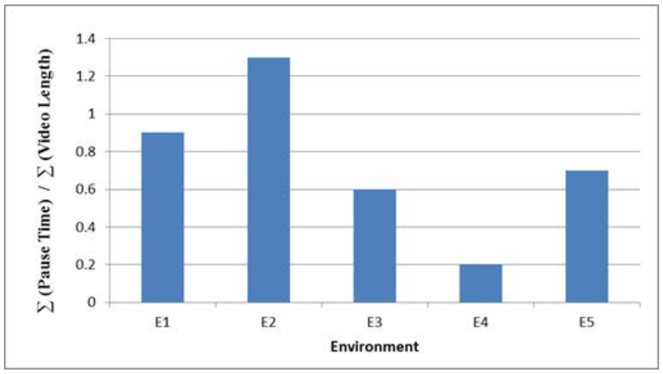

Fig. 4 Fraction of time spent in waiting for the videos

The user experiences of waiting for the videos were varies across different locations and the technology used. In three locations: E1, E3 and E5, the time spent waiting for the videos when the videos were paused, was longer than 40% of the total duration of the videos. In E2, which showed the worst case scenario, the time spent waiting was even longer than the total video length and this would have definitely frustrated users very badly.

Analysis of the above explained results lead to the conclusion that podcast subscribed users indeed experience disruptive playbacks on watching podcasts online, especially when they watch videos with higher quality. Although a user can choose to wait for a video to buffer before he starts watching, in the user's point of view it is undesirable and it leads to user frustration.

This research was motivated by the results of the above described experiments to devise a video prefetching approach that has the potential to reduce or even eliminate pauses during video playback. For this research, description of each video has been taken into account to decide whether user going to like a video or not. Since the description of a video, which is a text, is used to select videos, the main approaches of selecting videos for prefetching was filtering based on statistical study of user access patterns. In fact it is a binary classification and one of the text classification methods: Naïve Bayesian Algorithm has been used as the filtering approach.

Performances of filtering based on Naïve Bayesian Algorithm can be evaluated using two metrics.

The first metric is the "hit ratio", defined as a fraction of the number of videos that was watched by the user which was served from the prefetching storage. A higher hit ratio means we can serve more requests from the PA's storage, resulting in better user experience.

Even though user started to watch the videos downloaded by PA, user may have not liked these videos and he may have stopped watching them in few seconds. In that case prefetching algorithm has not been successful and its accuracy is low. A video that was watched for more than half of its full length was considered as a video that was actually liked by the user and in this research it was called as a hit video. So to evaluate the accuracy of the video selection algorithm, the second metric is used and it is called the "Completely viewed video ratio", which reflects the accuracy of the video selection algorithm. The Completely viewed video ratio is defined as a number of hit videos over the total number of watched videos:

To evaluate the hit ratio and the completely hit ratio of the prefetching algorithm, 15 volunteers were asked to use the PA system for podcast watching and they used the system for 65 days. These volunteers had various interests which represented different podcast video categories and podcast sites that they have been subscribed to. Table 2 shows the interested categories of the 15 users and the three main podcast sites that they have been subscribed to.

Table 2 : Interested categories of users

User	Interested Categories
1	Technology, Science, Business
2	Technology, Entertainment, Politics
3	Medical, Science
4	Religious, Medical, Environment
5	Technology, Business, Sports
6	Technology, Science, Business
7	Environment, Business
8	Environment, Business
9	Religious, Medical
10	Technology, Medical
11	Sports, Entertainment
12	Technology, Entertainment, Politics
13	Technology, Entertainment, Sports
14	Technology, Environment, Business
15	Sports, Entertainment

To retrieve the information about hit videos, a movie playback was developed and users were asked to use that movie playback to watch the videos. The number of seconds that user watched a particular video and the full length of the particular video were derived automatically by the movie playback.

Since the prefetching is done according to the user access patterns, videos watched by each user was different and PA had to make different predictions for each user. After using the PA system for 65 days, Fig. 5 shows the analysis of the results for the prefetching algorithm for its hit ratio which reflects the efficiency of the prefetching algorithm.

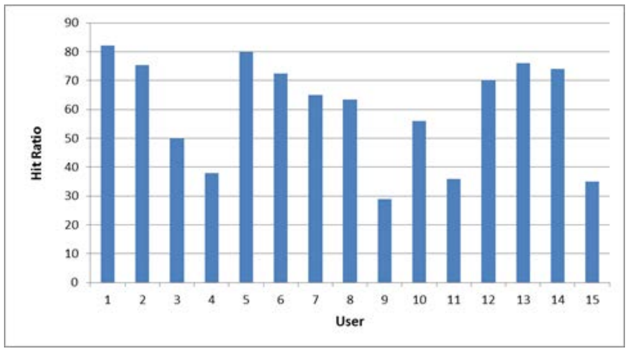

Fig. 5 Hit ratio for different users

For user 1, hit ratio was more than 80% and it reflects that user watched more than 80% of videos which were prefetched by the PA. Since user 1's preferred categories are Technology, Science and Business, there are a lot of podcasts available and because of that PA had enough set of data to predict videos that should be prefetched.

For user 9, hit ratio was just 29% and it reflects that user only watched 29% of videos which were prefetched by the PA. Since user 9's preferred categories are Religious and Medical, there are only few podcasts available and because of that PA had limited set of data to predict videos that should be prefetched.

When user's preferred categories are popular categories, such as technology, business and entertainment, they tend to have more podcasts available and this gives PA a fairly large and enough set of data to predict videos that should be prefetched. This situation resulted in a high hit ratio, which reflected the high efficiency of the prefetching algorithm.

Even though user started to watch the videos downloaded by PA, user may have not liked these videos and he may have stopped watching them in few seconds. So to evaluate the accuracy of the video selection algorithm, the second metric called the "Completely viewed video ratio" was used. After using the prefetching agent system for 65 days, Fig. 6 shows the analysis of the results for the prefetching algorithm for its completely viewed video ratio.

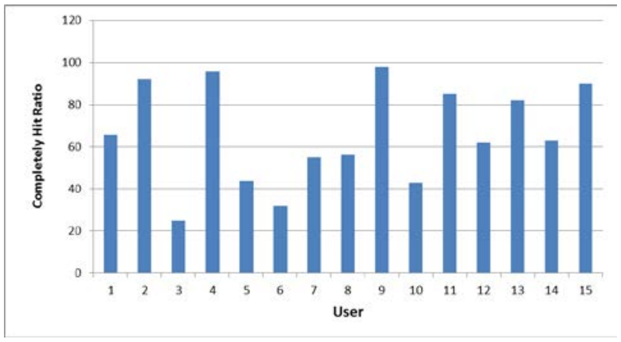

Fig. 6 Completely viewed video ratio for different users

For user 1, completely viewed video ratio is around 65% and it emphasise that even though user started to watch some of the videos downloaded by PA, user did not like those videos and he stopped watching them in few seconds.

For user 9, hit ratio was just 29% and it reflected that user only watched 29% of videos which were prefetched by the PA. But, for user 9, completely viewed video ratio is more than 90% and it emphasise that even though user only watched 29% of videos which were prefetched by the PA, he watched most of them completely, without stop watching them in few seconds.

Naïve Bayesian classification system training process was done using a corpus, a collection of either "liked videos" or "not liked videos" that have been collected from each user based on his access patterns.

To get the initial interests of the user, the initial data system was developed and at the beginning PA had limited set of data in the corpus, to refer to predict the videos that should be prefetched. When going along with the time corpus was updated according to the user access patterns and with the large set of data in the corpus, PA had enough set of data to predict videos that should be prefetched. Fig. 7 shows how the hit ratios for four different users, were increased as the time progressed and the result has been analysed after using the prefetching system for 65 days.

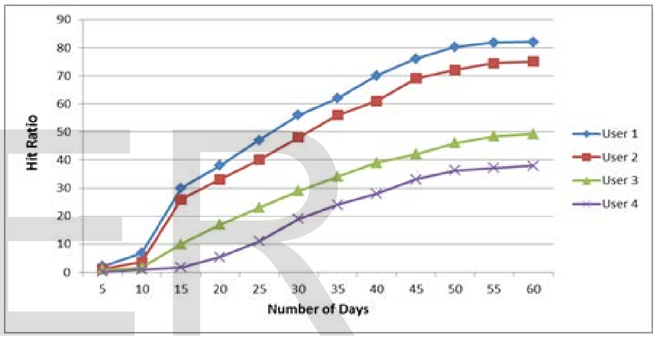

Fig. 7 Hit ratio vs. Time period

As the Fig. 7 clearly shows, during the first five days, with the initial data about user interests, hit ratios of the PA is very low. This is because; at the initial stage PA has only limited set of data in the corpus, to refer to predict the videos that should be prefetched. When going along with the time, corpus has been updated according to the user access patterns and with the large set of data in the corpus, PA have enough set of data to predict videos that should be prefetched. This has increased the hit ratios of the prefetching algorithm and PA becomes very successful with his predictions.

Naturally different people have different interests and they may like to watch different podcasts belongs to various categories. For this research, nine different categories have been considered that users may be interested in and they are as follows: Technology, Entertainment, Business, Religious, Environment, Sports, Politics, Medical and Science.

After users have used the PA system for 65 days, Fig. 8 shows the analysis of the results for the prefetching algorithm for its hit ratio with different podcast categories.

Since Technology, Entertainment, Business, Sports and Science are popular podcast categories, there are a lot of podcasts sites and podcasts available for these categories. Bcause of that PA had fairly large and enough set of data to predict videos that should be prefetched. This resulted in high hit ratio, which reflected the high efficiency of the prefetching algorithm for these categories.

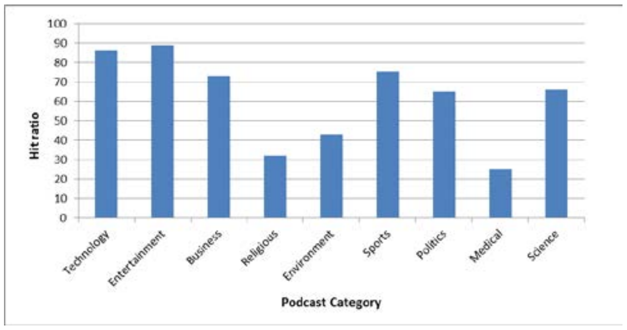

Fig. 8 Hit ratio for different podcast categories

Since Technology, Entertainment, Business, Sports and Science are popular podcast categories, there are a lot of podcasts sites and podcasts available for these categories. Bcause of that PA had fairly large and enough set of data to predict videos that should be prefetched. This resulted in high hit ratio, which reflected the high efficiency of the prefetching algorithm for these categories.

On the other hand for categories like Religious, Environment and Medical, which are not that much popular, there are only few podcasts sites and podcasts, are available. Because of that PA had less and limited set of data to predict videos that should be prefetched. Unfortunately this situation resulted in low a hit ratio which reflected a low efficiency of the prefetching algorithm for these categories.

For feature selection, most interesting "n" number of tokens were selected and "n" was an important parameter of this research which was needed to be decided carefully. Accuracy of the PA was heavily depending on this factor and therefore, several experiments were carried out to decide the "n", number of most interesting tokens that we have to consider.

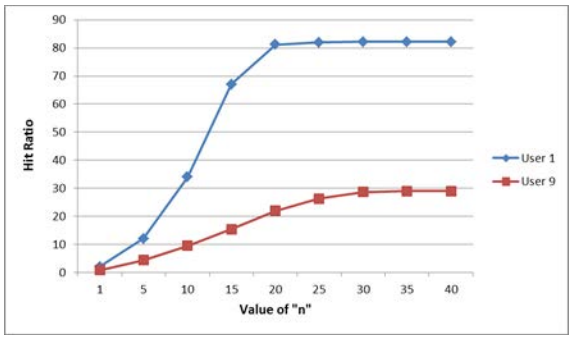

Fig. 9 Hit ratio varying with "n" value

Fig. 9 shows the results of hit ratios, for two users with different values for "n" and these values have been analyzed after using the prefetching system for 65 days.

User 1 and User 9 was selected for this evaluation because they were the users who had the highest hit ratio (82.2%) and the lowest hit ratio (29%) with the proposed prefetching approach.

When "n", number of most interesting tokens is very low (between 1 and 5), for both use 1 and user 9, hit ratio was remarkably very low. This is because, when the number of most interesting tokens is less than 5, PA has only few words to look at and its prediction is not that much effective. When it was made to increase the number of tokens, hit ratio for both user 1 and user 9, increased gradually and it showed a clear increase in efficiency of the prefetching approach. In this situation PA has enough words to look at and that made the prediction accurate and reliable. The most interesting result is, even though the number of tokens was made to increase continuously, after n is 30, there was no significant difference or increase in hit ratio. This was same for both user 1 and user 9 and therefore 30 was considered as a fair value that can be taken as the number of tokens.

So the prefetching approach was modified to use 30 as the "n", and PA will select most interesting 30 tokens for feature selection.

Because prefetching algorithm's efficiency was affected heavily by the boundary value "x", deciding a value for "x" also was an important decision in this research. If the probability result that is calculated is over "x", then that video can be considered as a video that is going to be not liked by the user and on the other hand if the result is between 0 and "x" the video can be considered as a video that is going to be liked by the user.

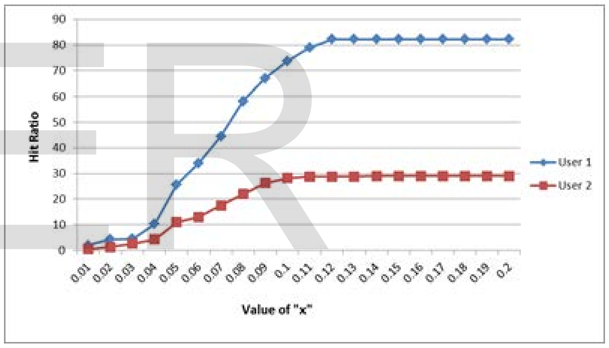

Fig. 10 Hit ratio varying with "x" value

When "x", the boundary value is very low (between 0 and 0.05), for both use 1 and user 9, hit ratio was remarkably very low. When it was made to increase the value of "x", hit ratio for both user 1 and user 9, increased gradually and it showed a clear increase in efficiency of the prefetching approach. The most interesting result is, even though the boundary value "x" was made to increase continuously, after x is 0.12, there was no significant difference or increase in hit ratio. This was same for both user 1 and user 9 and therefore 0.12 was considered as a fair value that can be taken as the boundary value.

5 Conclusion

Besides the remarkable popularity of podcast sites, user experience with watching video from these sites can vary extensively. It is common that a user experiences a pause when watching a video on line. Delays introduced by buffering before and during a video or audio can be quite annoying and can potentially discourage users from watching video. These interruptions during video playback are mainly due to the limitations of network capacity at the point of reception.

To demonstrate the need for prefetching, this research performed three experiments to evaluate user experience in watching videos from "Ted Talks" podcast site. In particular, measurements of how likely it is that a user experiences pauses during video playback and how long the pauses are, were taken. The result of the experiment shows that in all most all the environments with different technologies user experiences playbacks with pauses.

Motivated by the results of those experiments, in this research, podcasts are prefetched in order to eliminate the potential of pauses during video playback and decrease the service delay. Since the description of a video, which is a text, is used to select videos, the main approaches of selecting videos for prefetching is the filtering based on statistical study of user access patterns. The text classification method: Naïve Bayesian has been used as the algorithm to select videos for prefetching. PA determines the videos to be prefetched and it will retrieve their prefixes from relavant podcast site.

Performances of filtering based on Naïve Bayesian Algorithm were evaluated using two metrics: "hit ratio", defined as a fraction of the number of videos that was watched by the user which was served from the prefetching storage. And "Completely viewed video ratio", defined as a number of prefetched videos that are actually liked by users (called the hit videos) over the total number of watched videos.

Retrieving the prefixes of selected podcasts was done when the network usage was low and this minimized the effect of downloading videos in a busy peak time.
Because of the video has been prefetched, the prefetched portion in the buffer can compensate for any insufficient bandwidth and absorb network delay, resulting in a smooth play out of the video. This avoided delays due to buffering and encouraged users to watch podcast according to their preferences.

REFERENCES

[1] B. Cheng, H. Jin, and X. Liao, "Supporting VCR functions in P2P VoD services using ring-assisted overlays," in *Proc.* IEEE International Conference on Communications., pp. 1698 - 1703, June 24-28,2007.

[2] C. Zheng, G. Shen and S. Li, "Distributed Prefetching Scheme for Random Seek Support in P2P Streaming Applications," in *Proc.* ACM workshop on Advances in peer-to-peer Multimedia Streaming., pp. 29-38, Jan. 21 - 23,2005.

[3] Y. He and Y. Liu, "VOVO: VCR-Oriented Video-on-Demand in Large-Scale P2P Networks," IEEE Trans. Parallel and Distributed Systems, vol. 20, pp. 528 - 539, April 2009.

[4] Ubaid Abbasi and Toufik Ahmed, "Architecture for Cooperative Prefetching in P2P Video-on- Demand System," International Journal of Computer Networks & Communications, vol. 20, pp. 126-138, May 2010.

[5] Ahmed Mostefaoui, Harald Kosch and Lionel Brunie, "Semantic Based Prefetching in News-On-Demand Video Servers," Multimedia Tools and Applications, vol. 18, pp. 159-179, November 2002.

[6] Samamon Khemmarat , M. A. Amherst, Renjie Zhou and LixinGao, "Watching User Generated Videos with Prefetching," in *Proc.* Second ACM Conference on Multimedia Systems, pp. 187-198, Nov. 25-27, 2011.

[7] Tianyin Xu, Weiwe Wang, Baoliu Ye, Wenzhong Li, Sanglu Lu and Yang Gao, "Prediction-based Prefetching to Support VCR-like Operations in Gossip-based P2P VoD Systems," in *Proc.* International Conference on Parallel and Distributed Systems., pp. 1 - 8, Dec. 8-11, 2009.

[8] João Carlos Antunes Leitão, "Gossip-based broadcast protocols," M.S. thesis. Dept. Computer Science, University of Lisbon, Portuga, 2007.

[9] Yishan Gong and Qiang Chen, "Research of spam filtering based on Bayesian algorithm." in *Proc.* International Conference on Computer Application and System Modeling (ICCASM), 2010.

[10] V. P. Deshpande, R. F. Erbacher and C. Harris, "An Evaluation of Naïve Bayesian Anti-Spam Filtering Techniques." in *Proc.* IEEE SMC Information Assurance and Security Workshop, 2007.

[11] Yang Li, Binxing Fang, Li Guo and Shen Wang, "Research of a Novel Anti-Spam Technique Based on Users Feedback and Improved Naive Bayesian Approach." in *Proc.* International conference on Networking and Services (ICNS), 2006.

[12] K. Manjusha and R. Kumar, "Spam Mail Classification Using Combined Approach of Bayesian and Neural Network." in *Proc.* International Conference on Computational Intelligence and Communication Networks (CICN), 2010.

[13] Yishan Gong and Qiang Chen, "Research of spam filtering based on Extended Naïve Bayesian algorithm." in *Proc.* International Conference on Computer Application and System Modeling (ICCASM), 2010.

[14] Enrico Blanzieri and Anton Bryl, "A survey of learning-based techniques of email spam filtering," Springer Artificial Intelligence Review, vol. 29, pp. 63-92, March 2008.