MINLP Formulation for YouTube Content Curator

Agnish Upadhyay
Chemical Engineering
Indian Institute of Technology
Guwahati, Assam, India - 781039
agnish@iitg.ac.in

Rishabh Lal
Chemical Engineering
Indian Institute of Technology
Guwahati, Assam, India - 781039
l.rishabh@iitg.ac.in

***Abstract*—The paper proposes a novel mathematical formulation for a video recommendation system that is based on two primary parameters: similarity and likability. Specifically, the system recommends videos based on their degree of similarity to a target video and their overall likability. To develop and validate the system, we simulated various existing recommendation platforms and analyzed their behavior to identify key features and patterns. Using this information, we formulated an objective function that captured the essential elements of the simulated systems while simplifying their overall behavior.**
This approach allowed us to create a recommendation system that was both effective and efficient, providing accurate and relevant video suggestions.

***Index Terms*—optimisation, mathematical modelling, non-linear programming, video recommendation**

I. Introduction

Digital entertainment is a rapidly growing form of entertainment. Among digital entertainment, videos have been gaining traction in recent times. Human brains are interact better with storytelling, compared to other forms. And video is a great way to tell a story and engage. There's a direct relation between being a good storyteller and a good communicator.

With the COVID-19 pandemic in the recent times, video content has become intertwined with our daily lives. An internet user faces a large amount of data as such which would have made their daily dose of internet a boring one had it not been for Video Recommendation Algorithms. These algorithms use data analysis techniques to analyze user behavior and preferences and give personalized content recommendations.

The benefits of recommendation algorithms are not limited to viewers. Platforms and content providers can also benefit from these algorithms by improving user engagement and retention. By suggesting relevant content to users, platforms can keep users engaged and prevent them from becoming overwhelmed or disengaged. This can lead to increased loyalty and ultimately drive revenue for the platform or content provider.

In conclusion, recommendation algorithms play a crucial role in the video entertainment industry by making it easier for viewers to find the content they want to watch and by increasing engagement and retention for platforms and content providers. As the amount of content available continues to grow, the importance of recommendation algorithms is only likely to increase.

II. Discussion

Let us say we are designing an algorithm which optimises the videos that are to be shown next. One option would have been to study the patterns of the specific user and then use machine learning to predict the posts which are likely enjoyed by the user.

However, this most definitely would take a lot of time on analysing the data, building a model and then processing the same data once more. It is essential to note that machine learning undisputably has its own place in recommendation systems.

In a real world scenario, we need to consider the fact that a new user has entered or for any reason, our machine learning model doesn't have data available to it or is unable to perform well under a certain set of circumstances due to underfitting, overfitting or concept drift.

Hence, we come up with a mathematical model for the suggestion algorithm which is a fail-safe. Since the videos are ever-changing in relevance, the model must capture that. We will see ahead how the video only behaves as data to our model and changing any of the parameters does not affect our algorithm's running.

We referenced many algorithms to the basics and incorporated various functions depending on compatibility with the datatypes we have in our dataset. The beauty of the model lies in the fact that it can be used atop the existing machine learning models and provide another level of robustness and further improve the previous suggestion.

III. Problem Description

A. Problem Statement

The task at hand is to create a recommendation system based on an array of videos and their associated tags. The system is designed to suggest a set of 'n' videos to users based on their first clicked video, or target video. The interest level of each video is determined by two primary factors: the degree to which it is similar to the target video, and the overall likability of the video. By considering these parameters, the recommendation system can identify the most interesting videos to suggest to users. The solution is only considered valid when it also satisfies the constraints of time and internet.

B. Problem parameters

We have constructed a dataset with the following parameters:

- Tags- Signifies the presence or absence of a tag for a particular video.
- Likes- Tabulated number of likes received by a particular video by friends as well as from other people on the platform.
- Server- Denotes the presence of the video on one or more of the servers.
- Quality- Stores how much data is required for displaying particular resolutions of the video.
- Latency- For each server type, latency time depends on the processing speed of the server and is given for each.
- Watchtime- Duration of each video.
- Internet Quota- How much internet can be spent on watching short videos.
- Screentime- How much time can be spend on watching short videos.
- Count- How many videos are to be displayed from the dataset.

C. Expected Output

The problem is to be solved mathematically and then display the indices of the videos that are to be suggested. The resolution quality for each kind of video to also be shown.

IV. Nomenclature

Before proceeding to the next section, have a look at the nomenclature which shall be followed throughout afterwards. *Note- Some variables have been declared later on, kindly refer back to the tables if confused.

Variable	Type of Variable	Role of the variable
j	Integer	Ranges from 1 to total number of videos. Traverses the videos.
k	String	fr denotes total likes by friends; ot denotes total likes by non-friends
d	String	s1, s2, s3....Traverses the servers.
t	String	t1, t2, t3....Traverses the tags.
q	Character	Either l, m or h. Specifies the quality of video.
Like	Integer Table of size (k x j)	Tabulated number of likes received by a particular video by friends as well as from other people.
Server	Boolean Table of size (d x j)	Denotes the presence of the video on one or more of the servers.

Variable	Type of Variable	Role of the variable
Tags	Boolean Table of size (t x j)	Denotes whether a video is tagged with a particular tag or not.
Quality	Integer Table of size (q x j)	Denotes internet usage for each quality of a video (in millisecond)
Latency	Integer array of size d	Latency time to fetch a video from particular server.
Watchtime	Integer array of size j	Duration of each video (in sec)
InternetUsage	Integer	Available internet quota (in KiloBytes)
ScreenTime	Integer	Available time (in sec)
Count	Integer	Number of photos to display

Variable	Type of Variable	Role of the variable
target	Integer	First clicked video
Similarity	Float array of size j	Cosine similarity of each video WRT target video
Obj	Float	Objective function value
Z	Boolean array of size j	Indicates whether video was selected or not
Low	Boolean array of size j	Indicates whether a selected video's quality was low or not
Med	Boolean array of size j	Indicates whether a selected video's quality was medium or not
High	Boolean array of size j	Indicates whether a selected video's quality was high or not
F	Boolean array of size j	Indicates whether a video has been liked by at least 1 friend.
TotalLikes	Integer array of size j	Total likes on a video
Suggestion	Integer array of size j	Auxiliary variable for printing solution

V. Solving the problem

We have to create a mathematical model for the problem. Taking a step back, it's easy to notice that the problem can be divided into sub-problems, results of which can be combined to yield the final result.

The steps we took and reasoning behind each of them are given below:

1) Calculation of similarity score- From the problem statement, it is clear that the objective function directly depends on two parameters, one being similarity score.
2) Likability score- The second parameter is the overall likability score of the video.
3) Identification of constraints- This is an essential part for forming the equations. The constraint information was extracted from the problem parameters section.
4) Mathematical Modelling- After getting a good grasp of the problem statement, all that is left is to formulate the equations.
5) Implementation- We chose GAMS software to solve this problem.

A. Calculation of similarity score

Similarity score is calculated by using cosine similarity and scaling it up since the values range from 0 to 1.
Cosine similarity is one of the metrics to measure the text-similarity between two documents irrespective of their size in Natural Language Processing. A word is represented into a vector form. The text documents are represented in n-dimensional vector space. Mathematically, the Cosine similarity metric measures the cosine of the angle between two n-dimensional vectors projected in a multi-dimensional space. The Cosine similarity of two documents will range from 0 to 1. If the Cosine similarity score is 1, it means two vectors have the same orientation. The value closer to 0 indicates that the two documents have less similarity.
The mathematical equation of Cosine similarity between two non-zero vectors is:

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

Video No.	t1	t3	t5	t6	t7	t9	t10
vid_1	0	1	0	1	1	1	0
vid_2	1	0	1	0	1	0	1

In our data, the first video has tags : t3,t6,t7,t9
Again from at the data, second video has tags : t1,t5,t7,t10

$$\sum_{i=1}^{10} A_{\mathrm{i}}.B_{\mathrm{i}} = 0*1+1*0+0*1+1*0+1*1+0*1+0*1 = 1$$

$$\sum_{i=1}^{10} \sqrt{{A_{\mathrm{i}}}^2} = \sqrt{0^2+1^2+0^2+1^2+1^2+1^2+0^2} = \sqrt{4} = 2$$

$$\sum_{i=1}^{10} \sqrt{{B_{\mathrm{i}}}^2} = \sqrt{1^2+0^2+1^2+0^2+1^2+0^2+1^2} = \sqrt{4} = 2$$

On putting these values in the formula, we get
Similarity = ¼ = 0.25

This implies that the level of similarity between the first video and the second video is 25%.
In order to calculate the similarity score between videos, we similarly deployed the cosine similarity (which ranges between 0 and 1). Our approach involved calculating the similarity score of all other videos relative to a specific target video, with Video 1 serving as the current target. This similarity score was then integrated as a supplementary parameter in the GAMS model.
{Let N be total number of videos.}

$$TotalSimilarityScore = 10 * \sum_{j=1}^{N} Similarity_{\mathrm{j}} * Z_{\mathrm{j}}$$

B. Likability Score

We inferred from the problem that determining the ranking of suggested videos is primarily based on two factors: the degree of similarity with the target video and the number of likes received. While likes are the second most important factor in determining the ranking, it can be difficult to discern the exact relationship between the number of likes and the objective function. This issue is further compounded in real-world platforms, where popular posts among a user's followers/friends must also be taken into account. Our dataset incorporates this added layer of complexity to provide a more accurate representation of real-world scenarios.

Therefore, the likability part of the objective function depends on the number of likes received by friends as well as the number of total likes received. Among these two, friends are given priority.

- Total likes dependence- On studying existing patterns for social media and entertainment platforms, it has been found that for highly likes posts, the content saturation starts at a threshold and the quality does not increase a lot afterwards. The additional likes are superficial due to other factors such as previous popularity, bias, huge array

of content.

These factors are avoided in our model and the model is approximated as constant after the threshold is reached. This behaviour is captured by the logarithmic function. The behaviour of the (raw) logarithmic function is plotted below.

*Please note that negative values don't come into play since likes are natural numbers.

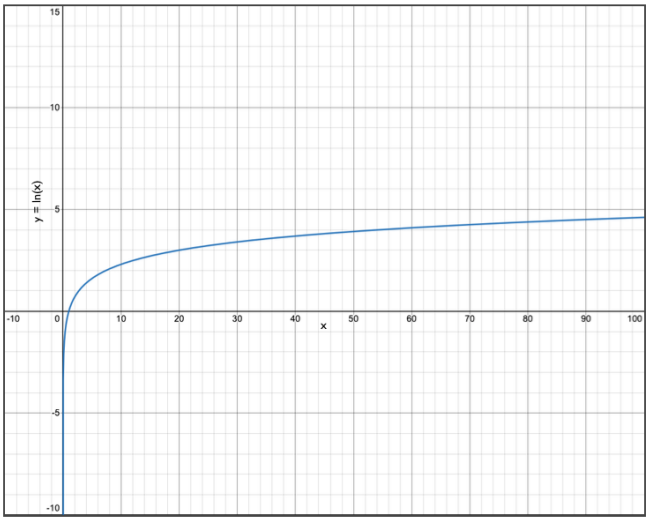

Fig. 1: Logarithmic Function

- Friend likes dependence- The mathematical function has been derived from the Minkowski distance. Minkowski distance is a mathematical concept used to measure the distance or dissimilarity between two points in a multi-dimensional space. The Minkowski distance metric is widely used in machine learning, particularly in clustering and nearest neighbor algorithms. The expression is given:

$$\left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{\frac{1}{p}}$$

However, we are not working with dissimilarity. We need a function that is low at very small values and then rapidly rises to an approximately constant slope function. By heuristic methods, we arrived at the following function.

$$f(x) = x^{(1-C/x)}$$

Here C is a tuning parameter which can be adjusted according to the dataset.

Given below is the plot with C = 10.

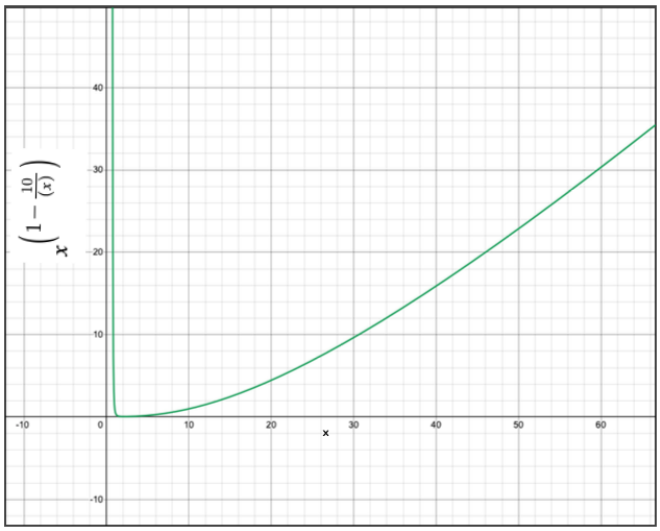

Fig. 2: Base function for friend likes

It is important to remember that in real world, total likes will play a role in the objective function even when a video is liked by friends. So the objective function must capture that as well. On the basis of scaling down the values by trial and error, the expression of likability factor is finalised as:
{Let T be total likes, F be friend likes}

$$Likability = \begin{cases} log(0.0001 + T/40) + \\ 0.066 * L^{(1-10/(x+0.001))}, & \text{if } L \neq 0 \\ log(0.0001 + T/10), & \text{if } L = 0 \end{cases}$$

C. Identification of Constraints

From the problem statement, it can be inferred that this problem is having some natural constraints which are listed below:

1) Internet availability- The total internet used for watching the videos at adjustable qualities must not exceed the amount of internet quota available to us.
2) Screentime constraint- In order to avoid addiction to mobiles, screentime management was added to electronic devices. Total watching time is restricted by the bound of available screentime.
3) Video count- In the problem, we have to display a fixed amount of videos. This forms another constraint.

These constraints indicate the need for optimisation since we need to generate the best possible outcomes which also satisfy the above and hence are feasible in real life too.

D. Mathematical Modelling

Equations used to create model are as follows:

The first equation tells the model which quality video must be chosen.Either one of the low,medium or high quality video has to be chosen for a particular index of the video.

$$Low_{\mathrm{j}} + Med_{\mathrm{j}} + High_{\mathrm{j}} = Z_{\mathrm{j}} \qquad \forall j$$

This equations tells that F(j) should be 1 if friends like is greater than 0,and if friends like is 0,then F(j)=0. For this purpose, signum function is used in the code implementation.

$$F_{\mathrm{j}} = \begin{cases} 1, & \text{if } Like_{\mathrm{fr\ ,\ j}} \neq 0 \\ 0, & \text{if } Like_{\mathrm{fr\ ,\ j}} = 0 \quad \forall j \end{cases}$$

This equation tells that if video is available in particular quality, then its file size must be added up. The binary variables(i.e., Low(j), Med(j), High(j) is multiplied with the video quality and the sum should be less than the internet quota available to us.

$$\sum_{j=1}^{N} Quality_{\mathrm{i,\ j}} * Type_{\mathrm{j}}) \leq InternetUsage$$

where i=l,m,h

For i=l , $\mathrm{Type_j} = \mathrm{Low_j}$
For i=m , $\mathrm{Type_j} = \mathrm{Med_j}$
For i=h , $\mathrm{Type_j} = \mathrm{High_j}$ $\forall j$

This creates an array of Total likes which is simply the sum of friend like and others like.This array will be used in our objective function.

$$TotalLikes_{\mathrm{j}} = Like_{\mathrm{fr,\ j}} + Like_{\mathrm{ot,\ j}} \qquad \forall j$$

This equation tells that the sum of all the videos that is shown to the user is equal to 'n',which is the number videos that we want to show.

$$\sum_{j=1}^{N} Z_{\mathrm{j}} = n \qquad \forall j$$

It tells the model that the number of medium quality videos must be equal to half the total videos that is shown to user.And accordingly, number of low and high quality videos are adjusted.This constraint is added in order to keep the user interested in watching the videos,because more no. of high quality videos might exhaust the internet available to us.And similarly,more number of low quality videos will cause the user to discontinue watching the available videos.

$$\sum_{j=1}^{N} med_{\mathrm{j}} = n/2 \qquad \forall j$$

This tells the model that the sum of the watchtime of all videos must be less than the recommended screentime, otherwise the user might get addicted to the social media platform.

$$\sum_{j=1}^{N} Z_{\mathrm{j}} * Watchtime_{\mathrm{j}} \leq ScreenTime \qquad \forall j$$

This constraint tells the model that the suggestion array(which contains the index of videos which is to be shown) should be equal to Z(j).Here Z(j) might be 0 or 1. '1' indicates that the video is suggested to be shown and '0' means that the video is not suggested.

$$Suggestion_{\mathrm{j}} = Z_{\mathrm{j}} \qquad \forall j$$

This constraint tells that a particular video may be available in more than one servers and at most in all servers.The mathematical model must choose which server it uses to extract a particular video minimising the latency at the same time.

$$Z_{\mathrm{j}} * (Server_{\mathrm{s1,\ j}} + Server_{\mathrm{s2,\ j}} + Server_{\mathrm{s3,\ j}} \leq 3 \qquad \forall j$$

It tells that if a video is extracted from a particular server than some latency time is spent.Suppose,we have been given Average latency,then total no. of videos multiplied by the former should be greater than the latency time taken to extract a particular video from a particular server.

$$\sum_{j=1,d=1}^{N,3} Server_{\mathrm{d,\ j}} * Latency_{\mathrm{d}} * Z_{\mathrm{j}}/1000) \leq AL * n \qquad \forall d, j$$

Our aim is to maximize Obj value.Above 10 constraints are used to solve the problem using Mixed Integer Non-Linear Programming (MINLP)

.

$$Obj = [max \sum_{j=1}^{N} Similarity_{\mathrm{j}} * Z_{\mathrm{j}} + Likefactor]$$

$$Likability = \begin{cases} log(0.0001 + T/40) \\ 0.066 * L^{(1-10/(x+0.001))}, & \text{if } L \neq 0 \\ log(0.0001 + T/10), & \text{if } L = 0 \end{cases}$$

where Likefactor = $\sum_{j=1}^{N} Z_{\mathrm{j}} * (1 - F_{\mathrm{j}} * Likability_1 + F_{\mathrm{j}} * (Likability_2 + Likability_3) \quad \forall j$

E. Implementation

Use of Python and GAMS

- Python is used to simply calculate the similarity score using 'Cosine Similarity' method.The tags associated with each video is useful to first create a vague idea of the video that are most similar to our target video.
- This similarity array is sent to GAMS.Since,we did not know how to import similarity array dynamically from Python to GAMS,therefore we created a parameter of similarity array and then used it accordingly in GAMS.
- Since,our problem has many non-linear constraints,therefore GAMS software is really helpful to sove the problems involving them.
- GAMS has a user-friendly interface that simplifies the process of creating, solving, and analyzing mathematical models.
- GAMS is known for its fast and efficient optimization algorithms, which allow users to solve large-scale optimization problems quickly and accurately.

  Theses are the reasons to use Python and GAMS together to create our model.

VI. Conclusion

Upon implementing the mathematical formulation suggested earlier in GAMS, we have observed that the model performs well on the dataset, and is scalable with parameters that can be adjusted to accommodate varying inputs without impacting the accuracy of the output.

The model can be used independently with fixed data or in conjunction with existing machine learning models to provide more accurate recommendations. By combining pre-existing data with mathematical objective function values, the hybrid model can deliver an overall better suggestion that considers both the data and mathematical insights.

Furthermore, the results demonstrate the significance of GAMS in solving a mixed integer non-linear programming problem like this, which is a testament to the potential of mathematical modeling in the field of recommendation systems. Going forward, we hope to see further advancements in this area that can expand the applications of these models.

Future Improvements

Like all things, there is a lot of scope for improvement in our project. Some of them are:

- Dynamic fetching of data from the backend python program for calculating similarity scores.
- After the first few clicks, the algorithm could then modify the results by more heavily inclining towards finding similarities between the previous videos.
- Space and time complexity for finding similarity score might be improved by using bitmasks.

Acknowledgment

We would like to express our gratitude to Prof. Prakash Kotecha and Prof. R. Anandalakshmi for their teachings in an interesting and memorable course on "Computer Aided Applied Optimisation Techniques".

Their insights, guidance, and expertise have inculcated a solid optimisation fundamental in us and we hope it was reflected in our work. We are deeply grateful for your generous support and guidance throughout this project. We also thank the teaching assistants, Rajani Kant Boro and Nikhil Anant Purao for actively helping and guiding us on obstacles we faced along the way.

References

[1] https://studymachinelearning.com/cosine-similarity-text-similarity-metric/

[2] How Youtube algorithm works https://blog.hootsuite.com/how-the-youtube-algorithm-works/

[3] $Problemreferencefile : ///D : /Downloads/hashcode_2017_qualification_round - 1.pdf/$