Proceedings of the Second International Conference on Electronics and Renewable Systems (ICEARS-2023).
IEEE Xplore Part Number: CFP23AV8-ART; ISBN: 979-8-3503-4664-0

YouTube and Movie Recommendation System Using Machine Learning

Suraj Suresh K
Electronics and Telecommunication Engineering
RV College of Engineering®
Bengaluru, Karnataka, India

Mehul Srivastava
Electronics and Telecommunication Engineering
RV College of Engineering®
Bengaluru, Karnataka, India

Mohana
Computer Science & Engineering (Cyber Security)
RV College of Engineering®
Bengaluru, Karnataka, India.

2023 Second International Conference on Electronics and Renewable Systems (ICEARS) | 979-8-3503-4664-0/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICEARS56392.2023.10084999

Abstract **- YouTube, Netflix and some other OTT applications follow a drastic and systematic approach to how they portray movie recommendations to their users in order to keep them attracted and interested in their websites and applications. The concepts of Machine Learning and Artificial Intelligence are being used in so many different ways ranging from smallest keyword suggestions while users' type, to the highly complex recommendation systems such as YouTube, that we solely use so often. In other words, recommendation system is used to suggest similar items to the user. The items can be movies, products, videos or anything depending on organizations or industries. Some modes of approach includes either using a candidate generation model or a ranking model to sort various forms of information which could vary from time spent while watching the videos to the age group of people watching the same videos. proposed work implemented using KNN machine learning model and obtained an accuracy of 98%.**

Keywords **- Youtube, Movie, KNN Algorithm, Artificial Intelligence, Machine Learning, Recommendation systems.**

I. INTRODUCTION

Over the past few decades, recommender systems and online services like YouTube, Amazon, Netflix, and many others have proliferated in our daily lives. In our daily online activities, recommender systems are more prevalent, whether in e-commerce (which proposes things to customers that they might find interesting) or online advertising (suggest to users the proper contents, matching their tastes). Basically, recommender systems use ML algorithms to try to present people with content that is relevant to them (items being movies to watch, text to read, products to buy or anything else depending on industries). Systems for making recommendations are highly prized by some businesses since they may be fairly profitable when they function effectively and help you differentiate yourself from the competition. Illustrated the importance of recommender systems using Netflix's most recent "Netflix reward" competition. Goal of the challenge was to develop a recommender system that outperformed Netflix's own algorithm. A $1 million prize was awarded to the winner. In this post, we'll look at a variety of recommender system topics. Given an example of how each one works, discusses its benefits and drawbacks, and explains the guiding principles. Recommender system uses many methods to recommend product to the user, among those one of the ways to narrow down the content is the relative filtering based on the product searched by similar types of users. This filtering comes under collaborative filtering methods [11]. This method is of great use for practical life-based recommender systems[13] [14] [15]. Figure 1 shows the finding of the relation among similar users.

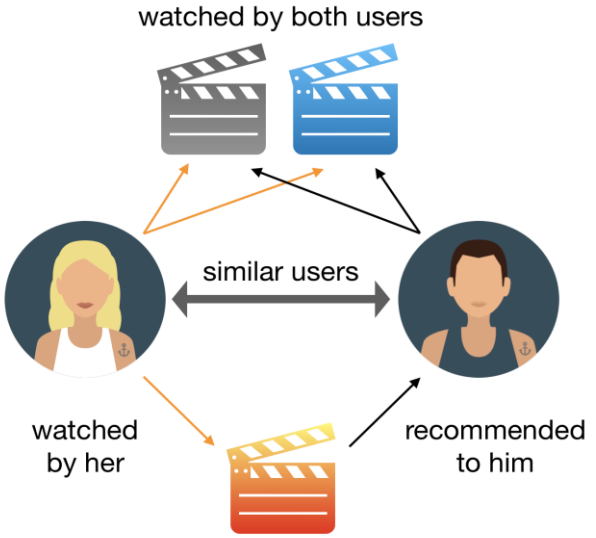

Fig.1. Relations between similar users [9].

II. LITERATURE SURVEY

Paul Covington *et al* [1] proposed two unique methods in order to recommend YouTube videos. One method involved the concept of candidate generation, and the other involved the concept of ranking. In candidate generation, huge ocean of videos is brought down to a set of a few hundred videos, thus making it more interesting to the user. In ranking model main aim is to calibrate predictions for the particular user interface. While working with the ranking part, many features that can describe the video and user's relationship. The video as only a narrow band of videos are being observed instead of thousands of videos in candidate generation. James Davidson *et al* [2] elucidated more on the two methods used, i.e., candidate generation and ranking models and elucidated more on mathematical part of the same by defining and introducing new formulas in order to analyze various aspects of the video uploaded and generate recommendation candidates.

Xavier Amatriain *et al* [3] elaborated recommender system has to improve offline metrics such as RMSE, it also has to consider other practical issues such as scalability or deployment. In this paper described practical issues that are as important as theory so as to build an industrial level recommender system that can be used in the real world. Joeran Beel *et al* [4] described the advancements as well as the imperfections of recommendation systems were inferred. Along with this a set of various approaches being implemented in terms of either Youtube as well as other recommendation systems such as Netflix, amazon etc were included. A general understanding gained from this paper is that almost 55% were content based filtering and almost 18% were collaborative filtering. Pradeep Kumar Singh *et al* [5] explained that in order to make a choice amongst multiple options keeping in mind the vast amount of information available online would be a very difficult and tedious task. Various Online recommendation systems have to thus be competitive and have higher accuracy. To achieve this, recommendation systems have to have efficient information retrieval systems and better filtering methodologies. In this paper a review of the general RS systems used such as content-aware recommendation, knowledge based, demographic filtering as well as context aware systems have been sighted. Various recommendation systems face a lot of problems in their functioning such as sparsity of right information, black box problem, limited content analysis, long tail problems etc were also described. Joeran Beel *et al* [6] Inference gained from this paper included some of the other aspects that play an important role in understanding the user requirement and filtering the vast amount of information. We were introduced to a new term called user modeling, which included recommending information based on the type of user requesting information. In order to achieve this the model uses information from items like social media tags, movies, emails, and news from media. However, they miss out on making mind maps in order to categorize users. This appears to be like a fault in the system because user modeling is assumed based on mind maps and its effectiveness is deemed equivalently important to a user system that has been defined on several other external items. Hence the main motive of this paper was to develop a user modeling approach on the basis of several mind maps.

Sophie Siebert *et al* [7] In the recent past the number of research and academic publications has been seeing a huge rise. As the number of publications and articles keeps increasing on the internet, it becomes more and more difficult to understand which article to refer to from the huge ocean of available information. It also poses a problem as researchers cannot find the right information effectively. In order to achieve this, the paper discusses using a different approach such as to use text similarity to categorize and provide relevant documents and in order to improve this system, scientometrics is to be used in the ranking system in order to bring in popularity into the system. This paper further dives into understanding the usage of scientometrics and usefulness of the system. Nitin Mishra *et al* [8] This reference was not a research paper but an article on recommendation systems and we further inferred from this article that with the large amounts of videos and information being uploaded to the internet the screen size is decreasing and this is one of the main issues being dealt with. Meaning to infer that screen sizes initially being 15 inches which is now coming down to 4 to 7 inches.

III. Design and Implementation

A. System Overview

Let us consider a system that contains two types of neural networks, mainly ranking and candidate generation. Candidate generation is applied to Youtube, we see that the model takes the user's particular Youtube history as input and receives a small set of videos from a huge number. These are called candidates and are meant to be in relevance to the user's requirements and interests. But how exactly does the model give the best recommendations? The ranking model helps in achieving this task by taking a set of videos and accomplishing the task by assigning a score to each particular video according to a desired objective function by using a set of rich features that shall help best describe the type of video being shown to the user. Ranked by their scores, the highest rated videos are presented to the user.

B. Candidate Generation

During candidate generation, the entire YouTube corpus is windowed down to a set of hundred videos that may be of importance to a particular user. The following pictorial diagram explains the entire process briefly. Figure 2 shows the block diagram of candidate generation model.

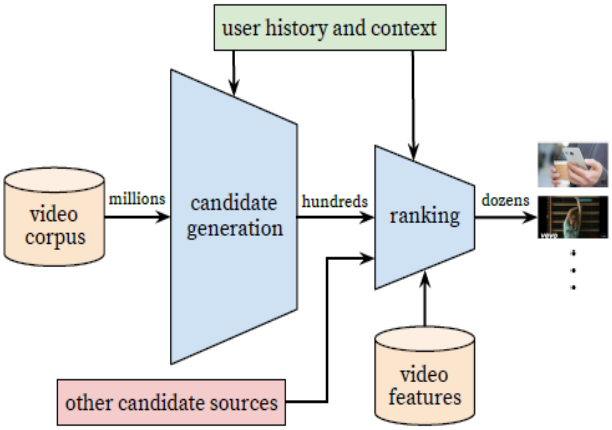

Fig.2. Block diagram of Candidate Generation Model [1].

C. Content based filtration

The content-based filtration method looks into taking more and more information from the user. This particular method recommends new items to users based on their previous actions or feedback. For example, if we consider the case of a movie recommender system that shall be discussed further in this paper certain additional information such as age, sex, the job and some other personal information can be used to recommend new movies that the user may like. In order to make this task less heavy to compute we see the usage of 'Utility Matrix' especially for content-based search methods. This utility matrix can help make the user's choices more significant for certain items. Thus, from the information and data that we gather from the users we can find a relation between the chosen items that are of liking to the user compared to the items that the user dislikes, for this particular function we utilize the utility matrix. In this method we assign to every user-item a particular value which is known as the degree of preference and thus a matrix of the user is then drawn and the chosen items to identify their preference relationship.

D. Ranking Model

In the second piece of the series on creating an enterprise-level recommender system, discussed matching algorithms and their definitions. In the first post, discussed how to create the system's overall design. This article will introduce the ranking algorithms used by recommender systems as well as the characteristics of online and offline training structures for ranking models. To better understand how the ranking module of a recommender system functions, when a user visits a platform, they will find a vast array of products. Therefore, we must get rid of anything that the user would find alluring. The matching module pre-filters the item list before passing it on to the ranking module.

In particular, the matching module first selects just a few products from a wide range of options that it believes the consumer will find appealing. For instance, 500 of the platform's 100,000 potential intriguing things are filtered out by the matching module. The rating module then assigns a rating to each item depending on the user's choices. To arrange the 500 items from the user's favorite to the least favorite, we need a ranking system that can provide a ranking of the objects based on user and item properties.

E. Model Working

The process opted for this model mainly involves the user and choices made by the user that are recorded by the system to help create a user profile. The user profile is constantly interacting with the recommender system while the user interacts with the webpage, hence we observe a parallel flow on information bi-directionally. The Recommendation system is also in contact with the webpage the user is using to constantly keep changing the information provided to the user. This helps make the information shown to the user more apt on a faster basis. The flowchart for the model is shown in figure 3.

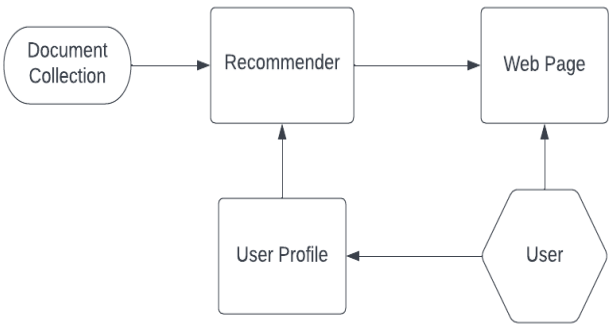

Fig.3. Flow chart of Model.

IV. RESULTS AND ANALYSIS

```
# code
import numpy as np
import pandas as pd
import sklearn
import matplotlib.pyplot as plt
import seaborn as sns

import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)

ratings = pd.read_csv("https://s3-us-west-2.amazonaws.com/recommender-tutorial/ratings.csv")
ratings.head()

movies = pd.read_csv("https://s3-us-west-2.amazonaws.com/recommender-tutorial/movies.csv")
movies.head()

n_ratings = len(ratings)
n_movies = len(ratings['movieId'].unique())
n_users = len(ratings['userId'].unique())

print(f"Number of ratings: {n_ratings}")
print(f"Number of unique movieId's: {n_movies}")
print(f"Number of unique users: {n_users}")
print(f"Average ratings per user: {round(n_ratings/n_users, 2)}")
print(f"Average ratings per movie: {round(n_ratings/n_movies, 2)}")

user_freq = ratings[['userId', 'movieId']].groupby('userId').count().reset_index()
user_freq.columns = ['userId', 'n_ratings']
user_freq.head()
```

Fig.4. Libraries used.

Figure 4 shows the necessary libraries that have been called such as the NumPy, pandas, Sklearn, matplotlib, seaboard libraries. Then created a list that reads the rating values from the csv file link. Similarly same for the movie names. Then store the length of number of ratings, movies and number of users into variables. Print to the users for necessary information of what has been obtained from the csv links. Further group the same using the 'groupie' function based on userId, movieId.

```
# Find Lowest and Highest rated movies:
mean_rating = ratings.groupby('movieId')[['rating']].mean()
# Lowest rated movies
lowest_rated = mean_rating['rating'].idxmin()
movies.loc[movies['movieId'] == lowest_rated]
# Highest rated movies
highest_rated = mean_rating['rating'].idxmax()
movies.loc[movies['movieId'] == highest_rated]
# show number of people who rated movies rated movie highest
ratings[ratings['movieId']==highest_rated]
# show number of people who rated movies rated movie lowest
ratings[ratings['movieId']==lowest_rated]

## the above movies has very low dataset. We will use bayesian average
movie_stats = ratings.groupby('movieId')[['rating']].agg(['count', 'mean'])
movie_stats.columns = movie_stats.columns.droplevel()

# Now, we create user-item matrix using scipy csr matrix
from scipy.sparse import csr_matrix

def create_matrix(df):

    N = len(df['userId'].unique())
    M = len(df['movieId'].unique())

    # Map Ids to indices
    user_mapper = dict(zip(np.unique(df["userId"]), list(range(N))))
    movie_mapper = dict(zip(np.unique(df["movieId"]), list(range(M))))
```

Fig. 5. Calculation of metrics for analysis.

Figure 5 shows the calculation of metrics to find mean, lowest and highest ratings from the group obtained. The Ids of lower and the highest rated movies are then stored in a particular location. Since the obtained values have a very low database, used Bayesian average. Thus, in a variable named 'Movie stats' we regroup the columns into movieId, rating and take the aggregate of the count of movies and the mean obtained previously. Create a user item matrix with the help of script car matrix function. Hence from the above library imported the csr_maric function. Created a new matrix were mapping the Ids to the indices and then further map the indices to IDs.

```
    # Map indices to IDs
    user_inv_mapper = dict(zip(list(range(N)), ns.unique(df["userId"])))
    movie_inv_mapper = dict(zip(list(range(M)), np.unique(df["movieId"])))

    user_index = [user_mapper[i] for i in df['userId']]
    movie_index = [movie_mapper[i] for i in df['movieId']]

    X = csr_matrix((df["rating"], (movie_index, user_index)), shape=(M, N))

    return X, user_mapper, movie_mapper, user_inv_mapper, movie_inv_mapper

X, user_mapper, movie_mapper, user_inv_mapper, movie_inv_mapper = create_matrix(ratings)

from sklearn.neighbors import NearestNeighbors
"""
Find similar movies using KNN
"""
def find_similar_movies(movie_id, X, k, metric='cosine', show_distance=False):

    neighbour_ids = []

    movie_ind = movie_mapper[movie_id]
    movie_vec = X[movie_ind]
    k+=1
    kNN = NearestNeighbors(n_neighbors=k, algorithm="brute", metric=metric)
    kNN.fit(X)
    movie_vec = movie_vec.reshape(1,-1)
    neighbour = kNN.kneighbors(movie_vec, return_distance=show_distance)
    for i in range(0,k):
```

Fig.6. Synthesis and calculations using KNN

Figure 6 shows the synthesis and calculations using KNN, store only the "rating", "movie index", "user index" in a separate variable 'X'From the library of sklearn.neighbours we import the function "Nearest Neighbours". Now using KNN start to

find similar movies. Define a variable ‘find_similar_movies’ that contains the following columns: movie_id, X, k, metric='cosine', show_distance=False which is used to compare the distance of the movie rating value as calculated from the mean value calculated. The shorter the distance/difference that particular is given, the higher priority in terms of order. The same movie name with id is amended in the list.

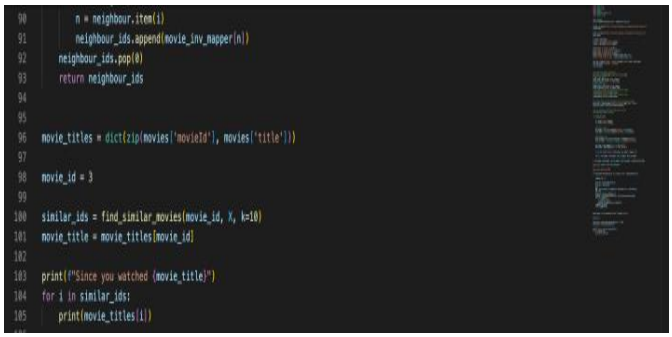

Fig.7. Storage of output.

Figure 7 shows the storage output. Created a variable named ‘movie_titles’ that contains the 'movieId' and 'title’. The same is then printed to the user and necessary outputs are obtained as shown in figure 8.

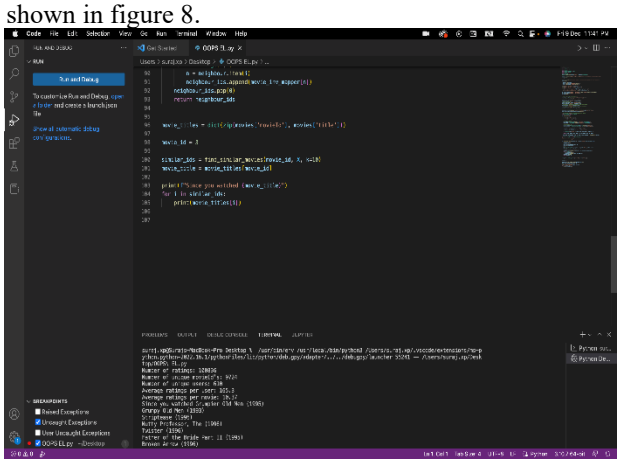

Fig.8. Displaying output-Model output

final output generated as follows:
Number of ratings: 100836
Number of unique movieId's: 9724
Number of unique users: 610
Average ratings per user: 165.3
Average ratings per movie: 10.37
Since you watched Grumpier Old Men (1995)
Grumpy Old Men (1993)
Twister (1996)
Father of the Bride Part II (1995)
Broken Arrow (1996)
Bio-Dome (1996)
Truth About Cats & Dogs, The (1996)
Birdcage, The (1996)

The data set for this model was extracted from an online csv file provided by amazon. Hence these are the closest movie recommendations that the model provided after going through the csv link to movie database chosen. This model achieves a good accuracy rate on successive runs.

ADVANTAGES

A YouTube and movie recommendation system that uses machine learning has several advantages. Firstly, it can personalize recommendations based on the user's viewing history and preferences, making it more likely for them to engage with the content and stay on the platform longer. Secondly, a machine learning-based recommendation system can continuously learn from user behavior and adapt to changing preferences, ensuring that recommendations remain relevant and interesting. Thirdly, a recommendation system can help content creators by promoting their content to users who are more likely to enjoy it, increasing engagement and revenue. Fourthly, machine learning algorithms can analyze large amounts of data quickly and accurately, identifying patterns and relationships that humans may miss. This allows you to receive more accurate and effective recommendations.

V. RESEARCH AND IMPLEMENTATION CHALLENGES

Data representation- The data collected must be presented in a meaningful way so that machine learning algorithms can understand it. There are various techniques such as feature engineering, embedding, and clustering that can be used to represent data.

The cold start problem- One of the problems with recommender systems is the cold start problem. The cold start problem means that it is difficult to recommend items to new users or new items without prior evaluation. To overcome this problem, techniques such as content-based filtering or hybrid models can be used.

Scalability- Recommender systems must be scalable to handle large numbers of users and items. As the number of users and items grows, so does the time and computing resources required to generate recommendations. Scalability can be improved by using techniques such as parallel processing, distributed computing, and caching.

Privacy and Bias- System-generated recommendations can be biased or discriminatory based on user data. User privacy is also a major concern when collecting and using user data. Techniques such as differential privacy and data anonymization can be used to mitigate these issues.

Evaluation- Evaluate the effectiveness of the recommender system to ensure it meets the needs of users. Metrics such as accuracy, completeness, and validity can be used to evaluate the effectiveness of a recommender system.

Applications

The application of candidate generation and ranking models in recommendation systems is to provide personalized recommendations to users based on their past behavior and preferences. These models play a crucial role in improving the user experience by suggesting relevant items and reducing the time and effort required for users to find items of interest.

For example, in a movie recommendation system, the candidate generation model may generate a list of movies that are likely to be of interest to a user based on their past viewing history and preferences. The ranking model can then be used to order the list of movies based on the user's likelihood of selecting each movie. The system may consider factors such as the user's viewing history, movie ratings, and movie genre preferences to generate the list of candidates and the order in which they are presented to the user. Similarly, in a music streaming service, the candidate generation model may generate a list of songs or playlists that are likely to be of interest to a user based on their past listening history and preferences. The ranking model can then be used to order the list of songs or playlists based on the user's likelihood of selecting each item. The system may consider

factors such as the user's listening history, song or artist ratings, and genre preferences to generate the list of candidates and the order in which they are presented to the user [10] [12].
Overall, the candidate generation and ranking models in recommendation systems help to improve the user experience by providing personalized recommendations that are relevant to the user's interests and preferences. This can lead to increased user engagement and satisfaction, as well as increased revenue for the platform.

VI. CONCLUSION

In this paper discussed different methods used for recommendation and ways to find related documents for academic purposes. Ranking and candidate generation are two widely used methods. Recommendation systems should be capable enough to address practical issues such as scalability or deployment. Content based filtering and collaborative filtering are prominently used. Online ranking system sorts the problem of choosing from a large amount of online data. Mind maps as a source of user modeling makes it more effective. Also, the proposed model achieves an accuracy of 98% using KNN machine learning model.

REFERENCES

[1] Paul Covington *et al.* "Deep Neural Networks for YouTube Recommendations" *10th ACM Conference on Recommender Systems, NY*, USA, pp.191-198, 2016.

[2] James Davidson *et al.* "The YouTube video recommendation system" *fourth ACM conference on Recommender systems*, NY, USA, pp. 293–296, 2010.

[3] Xavier Amatriain *et al.* "Building industrial-scale real-world recommender systems" *sixth ACM conference on Recommender systems,* NY, USA, pp.7–8, 2012.

[4] Beel J *et al* "Research-paper recommender systems: a literature survey" *Int J Digit Libr* 17, pp. 305–338, 2016.

[5] Pradeep Kumar Singh *et al* "Recommender Systems: An Overview, Research Trends, and Future Directions", *International Journal of Business and Systems Research,* Vol.15 No.1, pp.14 – 52, 2021.

[6] Beel Joeran *et al.* "Mind-Map Based User Modeling and Research Paper Recommender Systems."2014.

[7] Siebert Sophie *et al.* "Extending a Research-Paper Recommendation System with Bibliometric Measures" BIR@ECIR,2017.

[8] Nitin Mishra *et al.* "Research Problems in Recommender systems", Journal of Physics,2021.

[9] https://vitalflux.com/recommender-systems-in-machine-learning-examples/

[10] Biswas A *et al.* "Survey on Edge Computing–Key Technology in Retail Industry" *Lecture Notes on Data Engineering and Communications Technologies*, vol 58. Springer, 2021.

[11] A. Biswas *et al.* "Development of Product Recommendation Engine By Collaborative Filtering and Association Rule Mining Using Machine Learning Algorithms," *International Conference on Inventive Systems and Control (ICISC),2020*, pp. 272-277.

[12] C. V. Krishna *et al.* "A Review of Artificial Intelligence Methods for Data Science and Data Analytics: Applications and Research Challenges," *International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud),* 2018, pp. 591-594.

[13] Y. Li *et al.* "Reinforcement Learning based Path Exploration for Sequential Explainable Recommendation," *IEEE Transactions on Knowledge and Data Engineering,* 2023.

[14] P. K. Jain et al. "Unscrambling Customer Recommendations: A Novel LSTM Ensemble Approach in Airline Recommendation Prediction Using Online Reviews," *IEEE Transactions on Computational Social Systems,* pp. 1777-1784, Dec. 2022.

[15] Y. Tai *et al.* "Content-Based Recommendation Using Machine Learning," *International Workshop on Machine Learning for Signal Processing (MLSP),* 2021, pp. 1-4.