Movie Tag Prediction System Using Machine Learning (2021)

Tanya Singh, K. Tarun Kumar Reddy, V. Bhanu Prakash Reddy, Chirag Jain
Computer Science Engineering
Bennett University

***Abstract*—Movies can be tagged with various details such as genre, plot structure, soundtracks, and visual and emotional experiences etc. This information can be used to build automatic systems to extract similar movies, enhance user experience and improve the recommendations. We describe a way that allowed us to create a fine-tuned set of various tags that expose the varied characteristics of movie plots. How the correlate between tags is varying which we make a study on with different types of movies. By means of this project we will be solving a Multi-Label Classification problem. This system was built to study the inferring tags from plot synopses. We hope it will be useful in other tasks related to narrative analysis.**

I. Introduction

Various features of movies, such as genre, plot structure, soundtracks, and emotional responses, visuals and so on can be labelled Because of the massive volume of multimedia data generated these days, it' s difficult to imagine systems that can automatically analyse the content to determine the validity and classify them. Manual processing is important for extracting needed information from the data and assigning suitable tags. As a result, tag quality is based on a subjective criterion that differs from person to person. This generated metadata makes it difficult to gain complete insights into major elements of a movie and due to lack of precision, there are irregularities and hence data is less accurate which could impact user experience. This data may be utilised to create automated algorithms that extract related movies, dealing with the growing problem of information overload and improve the user experience and suggestions as well as provide users with a preview of what to expect from a movie. This tag prediction model has various applications like object detection, automatic subtitles generation and optimization of movie search engines and mainly in content censorship.

Only a machine learning based algorithm can efficiently complete such tasks. Here, we will address this problem by creating an automated engine that can extract tags from the plot of the movie which is a detailed description of a movie (synopsis of movie story line) or summary of a movie. A movie can have one or more tags this is when multi-label categorization comes into play where each sample is given a set of target labels. e. g., classifying a dataset which may be adventure or action, comedy, horror, flashback. Machine learning's remarkable breakthroughs have paved the road for discovering patterns in data with high accuracy, the completion of tasks with a machine learning based algorithm is that's why efficient. We describe a way that will allow us to create a fine-tuned set of various tags that expose the varied characteristics of movie plots. After this we check as to how these tags relate to different sorts of films. We will use this model to assess if we could somehow infer tags from plot synopsis.

Fig. 1. Movies

Fig. 2. Tag cloud

II. Related Work

We feel that there has been a low attention towards the tag prediction and categorization of movies in the literature. The relevant work in this sector is mainly focused on small-scale image, video, blog and other content-based tagging. For example, the redundancy across YouTube videos was used by Siersdorfer[1] et al to discover connections between videos and give tags to similar ones. While Chen et al[2] developed a video tagging approach in which a text-based representation of a video from various sources was generated on which a graph model was applied to find and score the important

keywords that would serve as tags. This method was reliant on a written description that has to be generated manually. Xin et al[3] investigated various free code information websites and suggested the "TagCombine" algorithm. The author took into account three factors: similarity-based ranking, multilabel ranking and Tag term ranking and according to Zhang et al[4], correlations between labels should be explored to gain more insights into multi-label learning. On other websites, Lipczak[5] et al applied collaborative tagging and used two different approaches: graph-based and content-based. In diverse areas such as music and images, automatic tag creation based on content-based analysis has received a lot of attention. For example, deep neural models have all been used to create tags for music tracks by Choi et al[6]. Lyrics were used to generate tags for music by van Zaanen[7] and Kanters[8]; and Eck et al[9], Dieleman[10] and Schrauwen[11] utilised acoustic features from the songs to generate various tags.

There have been some studies for predicting tags for web-content such as AutoTag by Mishne[12] described a model where, given a blog post, various tags are suggested that appear to be relevant; the blogger then explores the ideas and selects the ones which seem helpful and a similar model by Sood et al[13] called "TagAssist" that generates tag ideas for new blog articles based on previously tagged posts. To generate tags, most of these systems used content-based resources such as user metadata and tags given to similar resources.

There have been various similar works in the tag prediction domain using plot synopses like Kar et al[14] used plot analysis for tag prediction. For each movie, the algorithm forecasts a limited number of tags. However, because the system could only capture a tiny fraction of the multi-dimensional characteristics of movie plots, the tag space produced by the system for the test data only covers 73 per cent of the real set. On the other hand, Eric Makita[15] and Artem Lenskiy[16] present a Naive Bayes model for predicting movie genres based on ratings given by users. The notion is that users favour some genres more than others. Tag prediction has been studied by one more researcher Kuo[17] who used a co-occurrence approach to generate tags based on the words in the post and their relationship to tags. The model was created for next-word prediction in big datasets and then modified by limiting the predicted next word to just tags. This co-occurrence algorithm correctly predicts one tag per post with a classification accuracy of 47 per cent.

(Ka wing Ho,2011)[18] explored several techniques for categorising movie genres based on plot. Parametric-Mixture Model (PMM), One - Vs - All Support-Vector Machines (SVM), Multi-label K-nearest neighbour (KNN), and Neural Network are some which utilised frequency-inverse document frequency in the study of their approach of the words as features. This experiment was conducted on a limited dataset consisting of 16k titles of movies for both testing and training datasets. It predicted only some limited genres like adventure, comedy, crime, documentary, drama, family etc. The best F1 score obtained was 0.58

Alex Blackstock[19] and Matt Spitz[20] conducted an experiment on a limited dataset of 399 scripts, with the best F1 score being 0.56. They attempted to categorise movies with the help of the logistic regression approach by retrieving features from scripts like the ratio of descriptive to nominals words. The model assesses the likelihood that the movie belongs to each genre based on extracted characteristics and picks the k best scores as its predicted genres. Robert E Schapire[21] and Yoram Singer[22] provided two modifications namely, multi-class and multi-label text classification with as an extension to the AdaBoost algorithm(to solve classification,regression problems this was mainly developed.). The conversion of the multi-label problem into separate binary classification problems is done by 1st one and the ranking the labels in order for the correct one for achieving highest ranking is done by 2nd extension.

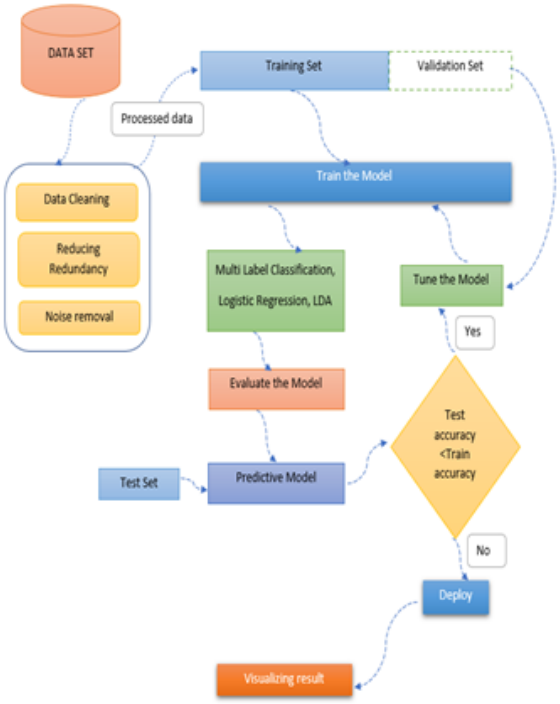


Fig. 3. Flow diagram 1

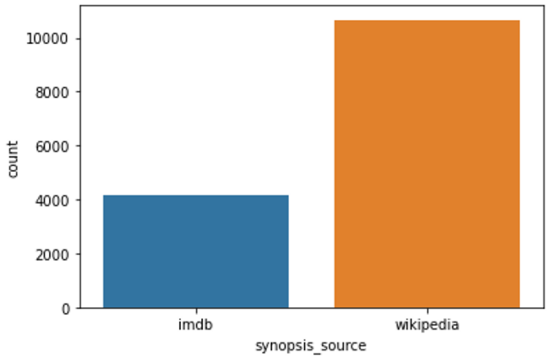
III. Proposed Methodology

Using the tags to create movie plot synopses corpus

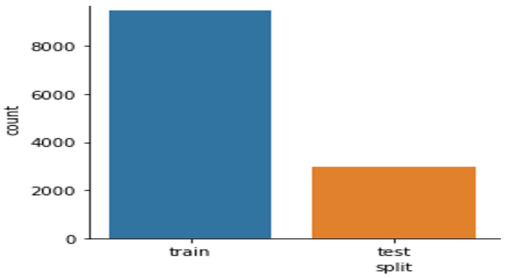
There are various datasets available that contain movies and their plots, but our main concern is to retrieve a dataset with certain expected attributes, such as tags should be closely related to the plot and not some metadata that is completely irrelevant to the plot, and redundancy in tags should be

avoided because we need to assign unique tags, so having tags that represent the same meaning would be ineffective. So, we have gathered data from various internrt sources that contains nearly 14,000 movies with unique tag set of 72 tags. here, each data point has six attributes including IMDB id to get the tag association information for respective movies in the dataset, Title of the movie, Plot of the movie, Tags associated with each movie, Split attribute indicating whether the data belongs to test, or train set and source which is either IMDB or Wikipedia.
Data Distribution:

SOURCE:
Most of the data is taken from Wekipedia and some from IMDB



SPLIT:
train and test split



TAG ANALYSIS:

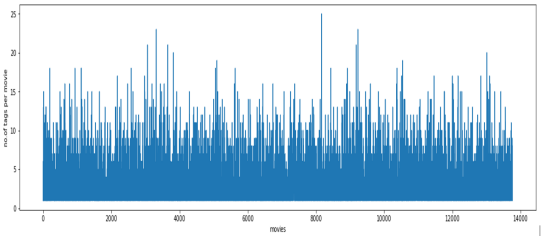


Fig. 4. movie vs no.of tags per movie

Moreover, Plot synopses should not contain any noise such as HTML tags or IMDB alerts and include sufficient

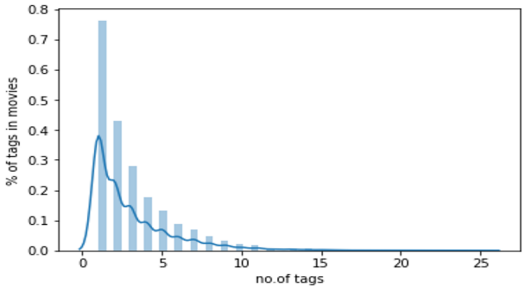


Fig. 5. percentage of tags in moves vs no.of tags

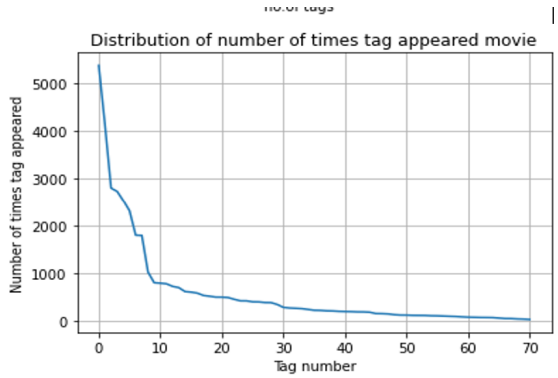


Fig. 6. Tag distribution

information because understanding stories from extremely short texts would be challenging for any machine learning system, each overview should include at least 10 sentences. Although the text is unstructured data, it is often created by individuals for the purpose of being understood by others. So, how can we handle a big volume of text and convert it into a representation that can be used to predict and classify using machine learning models? There are a variety of methods for cleaning and preparing textual data, and we used a few of them here.

- Converting every word to lowercase and removing HTML tags or other irrelevant elements present in the dataset.
- De-Contraction of words like can't to can not and removing any stop words if present such as "the", "a", "an", "in" as we would not want these terms to take important processing time or space in our database.
- Lemmatization of words, which typically refers to performing things correctly with the help of a vocabulary and returning a word to its root form, for instance, converting words that are in 3rd person to 1st person and future and past tense verbs to present tense.
- Stemming refers to reducing words to their word stem that affixes to prefixes and suffixes like "-ing", "-es", "-pre", etc.

Exploratory analysis for data of tags-data distribution and feature engineering.

we created a SQL database file of the given source CSV file and delete the duplicate entries and modify the same by adding a new custom attribute (6+1) tag-count which indicates the number of tags associated per movie so, we can find the exact count, how many movies are associated with how many tags. we also checked the number of unique tags present in the dataset using the BOW (bag of words) technique which is implemented using the count vectorizer method. We must transform it into vectors of numbers since machine learning algorithms do not accept the raw text as input data. Bag of words is the easiest way to represent Text Documents. In other words, it will determine how many times a specific word appears in a given document. This method yielded a tag cloud such as flashback, violence, murder, romantic and cult. This way we produced a more generic version of the common tags relevant to the plot of the movie. On the other hand, tags such as entertaining, and suspenseful are slightly less common got filtered out.

Machine Learning Approach for Predicting Tags using Plot Synopses

In this part, we'll go through some basic tests we've done using the corpus to predict movie tags. Using all tags:

• TFIDF Vectorizer Here, less frequent words are assigned comparatively more weight. TFIDF is the product of Term Frequency (the ratio of the number of times a word appears in a document to the total number of terms in the document) and Inverse Document Frequency (the log of the number of times a term appears in a document) (the total number of documents to documents with term present in it). We tested several approaches in the baseline model construction portion, including the multinomial Naive Bayes classifier, which is good for discrete feature classification, such as word counts for text categorization in a document. Integer feature counts are typically required for multinomial distributions, making them robust and simple to build, While Logistic Regression is used when the dependent variable (target) is categorical, and the question arises Why Linear Regression is not used for classification? Two things explain this. The first one is that classification problems mandate discrete values whereas linear regression only deals with continuous values which makes it not suitable for classification. The second thing is the threshold value for this considering a situation where we need to determine whether an email is spam or not. If we use linear regression here, we'll need to select a threshold by which we may classify the data. If the actual class is malignant with a predicted value of 0.45, and the threshold is 0.5, then it will be classified as non-malignant, which can result in serious consequences in real-time. So, it can be seen that linear regression is unbounded that's why we need logistic regression for instance, in a binary classification where we need to predict if a data point belongs to a particular class or not and the class with the highest probability is where the data point belongs. So, to fit a mathematical equation of such type we cannot use a straight line as all the values of output will be either 0 or 1. That's when sigmoid function comes into picture that transforms any real number input, to a number between 0 and 1.

Sigmoid equation:

$$S(x) = \frac{1}{1 + e^{-x}} \tag{1}$$

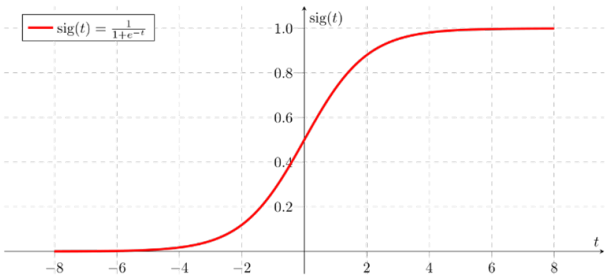


Fig. 7. Sigmoid function

$$Hypothesis => Z = WX + Bh(x) = sigmoid(Z) \tag{2}$$

Here if Z becomes close to infinity, Y will become 1 and if Z is close to negative infinity, Y will be predicted as 0.

Sigmoid equation for multiple features:

Here, we are computing output probabilities for all K-1 classes. For Kth class = 1 — Sum of all probabilities of k-1 classes.

So, we can say that multinomial logistic regression uses K-1 Logistic regression models to classify data points for K distinct classes.

Another method that we tried is SGD classifiers which is an optimization method, while Logistic Regression is a machine learning model that defines a loss function, and the optimization method minimizes/maximizes it.

In all the cases our goal is to maximize the micro averaged F1 score. We used micro-averaging as here, a movie might have more than two tags/labels associated with it. Micro averaged F1 score is the harmonic mean of micro-Recall and micro-Precision. Micro-precision is the sum of all true positives to the sum of all true positives and false positives.

$$Microprecision = \frac{TP1 + TP2 + TP3 + \dots}{(TP1 + TP2 + TP3 + \dots) + (FP1 + FP2 + FP3 + \dots)} \tag{3}$$

Micro-recall is calculated by first finding the sum of all true positives and false positives, over all the classes. Then we compute the recall for the sums.

$$micro - recall = \frac{TP1 + TP2 + TP3 + \dots}{(TP1 + TP2 + TP3 + \dots) + (FN1 + FN2 + FN3 + \dots)}$$

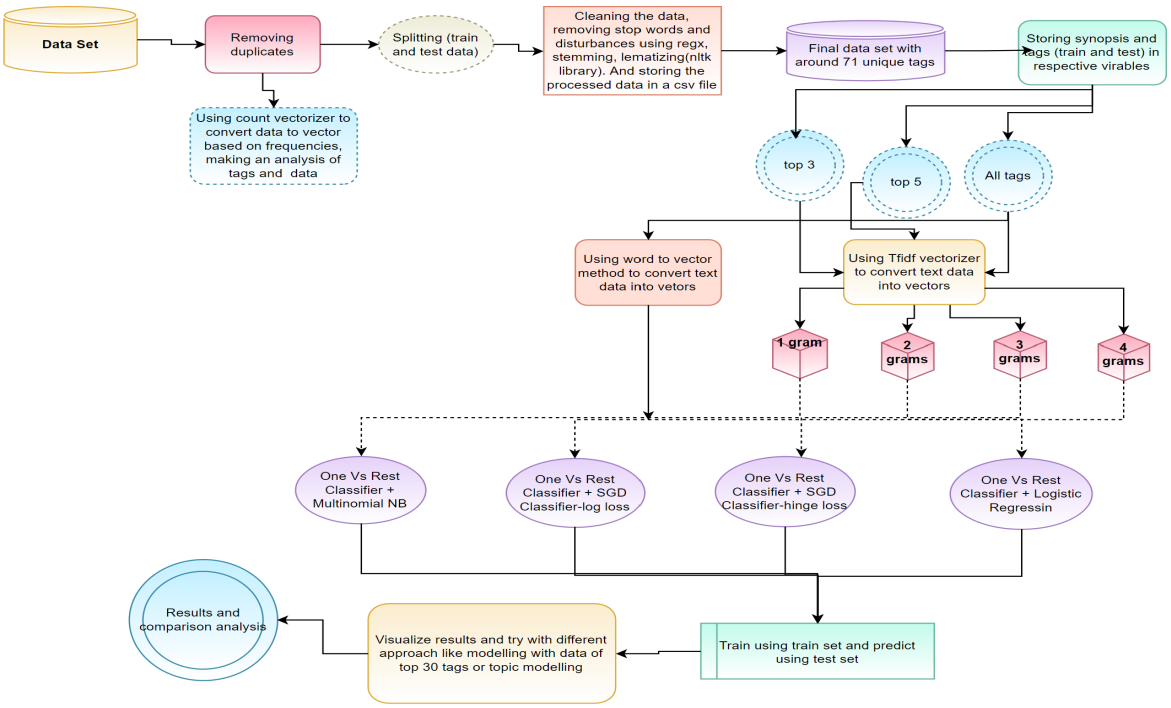


Fig. 8. Data-flow-diagram

(4)

Logistic Regression with Outliers was the model that provided us the highest micro averaged F1 score. One Vs Rest, sometimes known as one-vs-all, is a method that involves fitting a single classifier to each class. The class is fitted against all the other classes for each classifier. Despite the fact that this technique cannot handle multiple datasets, it trains fewer classifiers, making it a faster and more popular option.

AVGW2V

By capturing semantic information, word embeddings have been proved to be successful in text classification problems. As a result, we average the word vectors of each word in the plot to capture the semantic representation of the plots. Thus, we get a 1D vector of features corresponding to each document. By using this model also logistic regression gave the highest micro averaged F1 score.

According to the exploratory data analysis, a movie is typically associated with three tags. As a result, we attempted to create a model that could predict the top three tags. We utilised the same set of features this time, but the number of tags was set to three. By using TFIDF Vectorizer the highest F1 score was achieved by using SGD Classifier with log loss as well as accuracy score is also improving and by using AVGW2V (average word2vec) Logistic Regression model gave the highest F1 score. Similarly, for the next training of our model, we have set the tags to the top 5 and observed the F1 score for each model. We read the dataset and vectorize the tags using the BoW algorithm in the following training step to see which word appears how many times. Then, to construct a new data frame, we sorted these tags in decreasing order depending on how many times they appeared in the document. Out of the 71 distinct tags, we manually chose the top 30 tags based on their frequency. Then, except for the 30 tags present per movie, we eliminated all other tags along with their respective rows and repeated the procedure to train the model further.

For the next stage, we employed Python's Topic Modelling and Latent Dirichlet Allocation (LDA). We used Latent Dirichlet Allocation (LDA) to classify text in a document to a specific topic. Topic Modelling is a statistical model for discovering the abstract "topics" present in a collection of documents, and it is a commonly utilised for the discovery of hidden semantic meaningful structures in the body of text. It generates a topic per document and word per topic model based on the Dirichlet distribution. (LDA) is a common topic modelling approach with great Python implementations in the Gensim package. The LDA model above is made up of ten separate topics, each of which is made up of several keywords and given a specific amount of weight to the subject that represent the importance of a keyword to the particular topic. It determines the dominant subject of a particular text and is one of the practical applications of topic modelling.

Model	Vectorizer	precision	recall	F1-score	n-grams
Multinomial NB	Tfidf Vectorizer	0.467	0.686	0.556	1
		0.468	0.686	0.556	2
		0.468	0.686	0.556	3
		0.468	0.686	0.556	4
SGD Classifier Log-loss	Tfidf Vectorizer	0.502	0.68	0.578	1
		0.514	0.674	0.583	2
		0.521	0.662	0.583	3
		0.502	0.674	0.575	4
SGD Classifier Hinge-loss	Tfidf Vectorizer	0.492	0.7	0.578	1
		0.501	0.677	0.576	2
		0.499	0.684	0.577	3
		0.492	0.68	0.571	4
Logistic Regression	Tfidf Vectorizer	0.508	0.684	0.583	1
		0.508	0.685	0.583	2
		0.508	0.684	0.583	3
		0.508	0.685	0.583	4

Fig. 9. Top3-Tag-Prediction

Model	Vectorizer	precision	recall	F1-score	n-grams
Multinomial NB	Tfidf Vectorizer	0.116	0.646	0.197	1
		0.117	0.645	0.198	2
		0.117	0.645	0.198	3
		0.117	0.645	0.198	4
SGD Classifier Log-loss	Tfidf Vectorizer	0.157	0.597	0.249	1
		0.159	0.588	0.25	2
		0.16	0.589	0.252	3
		0.157	0.594	0.248	4
SGD Classifier Hinge-loss	Tfidf Vectorizer	0.153	0.584	0.243	1
		0.154	0.583	0.244	2
		0.152	0.586	0.241	3
		0.148	0.58	0.26	4
Logistic Regression	Tfidf Vectorizer	0.167	0.584	0.259	1
		0.167	0.584	0.26	2
		0.167	0.584	0.26	3
		0.167	0.584	0.26	4

Fig. 10. All-Tag-Prediction

To accomplish so, we search for the topic with the highest percentage contribution. Then we saved these dominating topics to a CSV file and concatenated it with our original data and used the same method to train the model even further to improve accuracy.

IV. Result and Analysis

To solve this Multi Label classification problem we used One VS Rest Classifier combining with different types of binary classification algorithms. First time we took data containing all tags (complete pre-processed data set). To make the model understand the plot synopsis we trained the model with different word counts like 1 grams, 2 grams, 3 grams etc. Here firstly Tfidf Vectorizer is used to convert the synopsis data into numeric data. After that we tried to train the model with One VS Rest classifier Multinomial NB, SGD Classifier-log loss, SGD Classifier-hinge loss and logistic regression with 1 grams (initially) and the same process is repeated with 2, 3 ,4 grams. Our aim here is to maximize the f1 score.

In case of Multinomial NB the score decreased from 1 to 2 grams and stayed same through 2, 3, 4 grams. And when it comes to SGD Classifier-log loss it decreased from 1 to 2 to 3 and increased in 4 grams (and this is the max). SGD Classifier-hinge loss increased gradually and highest is at 4 grams. And now comes the logistic regression it's score increased from 1 to 2 grams and remained unchanged later and logistic regression showed highest scores in individual models of 1, 2, 3, 4 grams and also in the overall case (and it is 0.26). And on a overview 4 grams model gave the better results when compared with the rest.Figure "fig 9"

Now to improve the model even further we try to implement this procedure with top3 and 5 tags as we saw in analysis of data, we take maximum features as 3 and 5 in these respective trainings. And follow the above that we have done with data with all tags.

Model	Vectorizer	precision	recall	F1-score	n-grams
Multinomial NB	Tfidf Vectorizer	0.411	0.676	0.511	1
		0.412	0.676	0.512	2
		0.412	0.676	0.512	3
		0.412	0.676	0.512	4
SGD Classifier Log-loss	Tfidf Vectorizer	0.448	0.676	0.539	1
		0.455	0.66	0.539	2
		0.444	0.665	0.532	3
		0.445	0.675	0.536	4
SGD Classifier Hinge-loss	Tfidf Vectorizer	0.435	0.674	0.529	1
		0.441	0.672	0.533	2
		0.434	0.668	0.526	3
		0.443	0.649	0.526	4
Logistic Regression	Tfidf Vectorizer	0.447	0.668	0.535	1
		0.447	0.668	0.535	2
		0.447	0.668	0.535	3
		0.447	0.668	0.535	4

Fig. 11. Top5-Tag-Prediction

Multinomial NB stays the same in all variants. The f1 of SGD Classifier-log loss increases from 1 to 2 grams and decreases from 3 to 4 grams (but still ¿ 1gram). SGD Classifier-hinge loss doesn't show much variation and logistic regression also stays the same in all cases. But this time SGD Classifier-log loss shows highest score (of all variants) in case of 2 grams, that is 0.568. Here 2 grams performance is better compared to the rest."fig 10"

Let's go a step further and try to predict top 5 tags "Fig . The f1 score of Multinomial NB it increases from 1 to 2 grams and remains unaltered. SDG Classifier-log loss f1 decreases in 3 and 4 grams where as equal in 1 and 2 grams. SDG Classifier-hinge loss increases the score gradually in all grams. Logistic Regression shows same scores with all 1 2 3 4 grams. But here also logistic regression is the highest f1 score achiever and the score is 0.535."fig 11"

Now let's compare among these models which try to predict all, top 5 and 3 tags. We can clearly see the significant increase in scores of f1 in top three and five tag prediction compared to all tags. And top 3 model is showing the best

Model	*Vectorizer*	*precision*	*recall*	*F1-score*
SGD Classifier Log-loss	AVG-W2G	0.12	0.632	0.201
SGD Classifier Hinge-loss	AVG-W2G	0.103	0.615	0.176
Logistic Regression	AVG-W2G	0.129	0.63	0.214

Fig. 12. All-Tag-Prediction-w2v

Model	*Vectorizer*	*precision*	*recall*	*F1-score*
SGD Classifier Log-loss	AVG-W2G	0.443	0.764	0.561
SGD Classifier Hinge-loss	AVG-W2G	0.395	0.723	0.511
Logistic Regression	AVG-W2G	0.477	0.688	0.563

Fig. 13. Top3-Tag-Prediction-w2v

results among these.

We also tried to use word to vector as it converts the data into more meaningful sense compared to Tfidf."fig 12", "fig 13", "fig 14"

And performed the operations just as above for all, 5 and three tag predictions, we also applied this for the data set containting top 30 tags and got the following f1 scores as follows.

All-tag prediction - Logistic regression (gave highest score that is 0.214)
Top 3 - Logistic Regression (gave highest score that is 0.563)
Top 5 - Logistic Regression (gave highest score that is 0.516)
Top 30 - data tags Logistic Regression (gave highest score that is 0.32)

As you can see this followed the same trend like when we used Tfidf. And Logistic Regression gave the best model outcomes in case of both vectorizers. Also top 3 prediction gave some what satisfying results in both. We also used Topic modelling (which finds the topics with in our synopsis) and LDA (latent dirichlet allocation) model to Implement this. Here also Logistic regression gave the highest result (0.366).

V. Comparative analysis

Here the approach that we have stated utilises One-Vs- Rest classifier to solve the problem as a multi-label (classification). A set of people [22] Khalid Haseeb, Najm us Sama, Miguel Á. MartÍnez-Del-Amor, Adnan Ahmed, Umair Ali Khan, Saleh M. Altowaijri, Naveed Islam and Atiq Ur Rehman who belongs to various fields and branches of Computer Science worked on the same problem (movie-tag prediction) with a different approach. The similar thing between us is we both felt that this field(predicting tags for movies) received less attention over years. They tried to predict the tags using segmentation of movies (movie-frames) and CNN (Convolutional-Neural- Network). We prepared a tag set of 71 tags(based on mostly used or occurring tags) and this team on the other hand prepared a tag set of 50 tags (dataset with combined information of each tag with 700 images and prepared a dataset with semantic information).
We tried combinations (two algorithms) of different classifying algorithms like Logistic regression, Multinomial-NB, GCD with One Vs rest classifier to train our model using N grams (to get a semantic sense) and Vectorizers (Count, TFIDF, Word - to - Vec), they used Inception V3 - pretrained model (CNN) and modified the final layers of classification in the sense with Soft-max classification for pridictions (transfer learning) .
In our case we tried to predict based on the plot synopsis and tried to gather the feel of plot and predicted tags accordingly. But this team used segmentation to achieve this. In this process they use videos of the movie and divide it into different frames and gather a single slide from the frame and try to predict 3 tags for this slide and they combine all the frames information and select top thre tags having higher weights or probabilities (dominant). They also designed a algorithm for boundary and key-frame detection and extraction. In our case we tried to classify our problem as a multilabel model and tried different combination of One-vs-rest classifier(which classifies the model considering one feature to rest all the features) and other multiclass classification algorithms and tried with n-gram approach and used different type of vectorizers to get meaningful numerical data.
Our highest f1-score is a decent 0.583 and they have a really efficient score of 0.88 as their highest. Both these are in top three tag predictions. So there is still a improvement scope when compared with other variant models.

Model	*Vectorizer*	*precision*	*recall*	*F1-score*
SGD Classifier Log-loss	AVG-W2G	0.41	0.68	0.512
SGD Classifier Hinge-loss	AVG-W2G	0.398	0.694	0.506
Logistic Regression	AVG-W2G	0.418	0.674	0.516

Fig. 14. Top5-Tag-Prediction-w2v

VI. Conclusion

We described a way that allowed us to create a fine-tuned set of various tags that expose the varied characteristics of movie plots. We took on the task of extracting tags relating to movie plots from noisy and repetitive tag spaces which needed to be pre-processed. We present an analysis where we tried to predict movie tags from plot synopsis. The highest micro averaged F1 score that we obtained from the entire project is 0.583. As, we observed in the EDA section that on an average a movie contains 3 tags. So, we tried to train our model multiple times and each time we set the tags to different values like top 3 and top 5 and we also selected the top 30 tags manually to analyse our model to a greater extent and we also used various semantics vectorizations such as LDA (Latent Dirichlet Allocation) and word2vec to improve our model. Although we had a small dataset, we still got a decent F1 score. *By means of this project we will be solving a Multi-Label Classification*

problem. We hope it will be useful in other tasks related to narrative analysis.

REFERENCES

[1] S. Siersdorfer, J. San Pedro, and M. Sanderson, Automatic video tagging using content redundancy, in Proc. 32nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr. (SIGIR), 2009, pp. 395–402.

[2] Z. Chen, J. Cao, Y. Song, J. Guo, Y. Zhang, and J. Li, "Context-oriented Web video tag recommendation," in Proc. 19th Int. Conf. World Wide Web (WWW), 2010, pp. 1079–1080.

[3] Xia, X., Lo, D., Wang, X., Zhou, B. (2013). Tag recommendation in software information sites. 2013 10th Working Conference on Mining Software Repositories (MSR).

[4] Zhang, M.-L., Zhang, K. (2010). Multi-label learning by exploiting label dependency. Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '10.

[5] Marek Lipczak. 2008. Tag recommendation for folksonomies oriented towards individual users. In In: Proc. of the ECML PKDD Discovery Challenge.

[6] K. Choi, G. Fazekas, M. Sandler, and K. Cho. 2017. Convolutional recurrent neural networks for music classification. In 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 2392–2396, March.

[7] Menno van Zaanen and [8] Pieter Kanters. 2010. Automatic mood classification using tf*idf based on lyrics. In Proceedings of the 11th International Society for Music Information Retrieval Conference, ISMIR 2010, Utrecht, Netherlands, August 9-13, 2010, pages 75–80.

[9] Douglas Eck, Paul Lamere, Thierry Bertin-mahieux, and Stephen Green. 2008. Automatic generation of social tags for music recommendation. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, Advances in Neural Information Processing Systems 20, pages 385–392. Curran Associates, Inc.

[10] Sander Dieleman and [11] Benjamin Schrauwen. 2013. Multiscale approaches to music audio feature learning. In Proceedings of the 14th International Society for Music Information Retrieval Conference, November 4-8. http://www.ppgia.pucpr.br/ismir2013/wp-content/uploads/2013/09/69 Paper.pdf.

[12] Gilad Mishne. 2006. Autotag: A collaborative approach to automated tag assignment for weblog posts. In Proceedings of the 15th International Conference on World Wide Web, WWW '06, pages 953–954, New York, NY, USA. ACM.

[13] Sanjay C. Sood, Kristian J. Hammond, Sara H. Owsley, and Larry Birnbaum, 2007. TagAssist: Automatic tag suggestion for blog posts.

[14] S. Kar, S. Maharjan, and T. Solorio, "Folksonomication: Predicting tags for movies from plot synopses using emotion flow encoded neural network," Aug. 2018, arXiv:1808.04943. [Online]. Available: https://arxiv.org/abs/1808.04943.

[15] Eric Makita and [16] Artem Lenskiy. A multinomial probabilistic model for movie genre predictions. arXiv preprint arXiv:1603.07849, 2016.

[17] Kuo, D. (2011, December). On Word Prediction Methods (Tech. Rep. No. UCB/EECS-2011147). EECS Department, University of California, Berkeley.

[18] Ka wing Ho. Movies genres classification by synopsis, 2011.

[19] Alex Blackstock and [20] Matt Spitz. Classifying movie scripts by genre with a memm using nlp-based features, 2008.

[21] Robert E Schapire and [22] Yoram Singer. Boostexter: A boosting-based system for text categorization. Machine learning, 39(2-3):135–168, 2000.

[22] U. A. Khan et al., "Movie Tags Prediction and Segmentation Using Deep Learning," in IEEE Access, vol. 8, pp. 6071-6086, 2020, doi: 10.1109/ACCESS.2019.2963535.