Predicting Video Memorability using Captions

Saumitra Das
Dublin City University
saumitra.das2@mail.dcu.ie

***Abstract*—This paper illustrates an approach to predict the memorability scores of short videos clips for the MediaEval Predicting Media Memorability Task. The method utilizes the caption features of the videos to develop models which aid in predicting the short-term and long-term memorability scores. The captions are processed using three different techniques to generate features which are fitted into three different models to predict video memorability. The results are compared, and weighted ensemble models are built using predictions of the best models. The ensemble models bettered the scores of the individual models.**

Keywords—Media Memorability, Captions, CountVectorizer, TfidfVectorizer, One-Hot Encoding, Ensemble Model

I. Introduction

The task of Predicting Media Memorability is part of the MediaEval Multimedia Assessment benchmarking initiative. The objective of this task is to automatically predict a memorability score for a video that will represent its probability of being remembered. An extensive dataset of 6000 short videos along with its pre-computed state-of-the-art features is provided for modelling purpose. The memorability of videos is assessed through recognition tests for both short-term and long-term memory which are provided as ground truth scores for training and testing the models.

In this paper, three main methods for text processing are explored for generating features from captions. *CountVectorizer*, *TfidfVectorizer* and *One-hot encoding* are the techniques that produced input for the prediction models. The features are fed into three different models to predict short-term and long-term memorability scores and evaluated using Spearman's correlation score as a standard measure. An optimized ensemble model is built using the best scores of two different models after analysing and comparing the models.

II. Related Work

The significant finding in recent works [1][2] states that the models based on video captions outperform models which are trained on other individual video features provided in the dataset. Tokenization is necessary before fitting the captions into the model as the input is in the form of text sequences [3]. *CountVectorizer, TfidfVectorizer* and *One-hot encoding* are the well-known and effective feature extraction methods of textual data [4][5]. Additionally, it is observed in many research papers that, building ensemble models using the predictions of best models can improve prediction accuracy significantly[1][6].

III. Approach

A. The Motivation of the Approach

The caption feature that produces the best individual results [2] is needed to be tokenized before feeding into a machine learning model. The primary motivation of this paper is to explore three powerful tokenization techniques for text sequence and compare their efficiency for this task. The secondary motivation is to check whether building weighted ensemble models from models trained with different features generated only from captions improves the prediction scores or not.

B. Data Pre-Processing and Feature Extraction

The following cleaning steps are performed on the caption data before tokenization:

- Replaced punctuations with space
- Converted all words to lower case
- Removal of stopwords
- Lemmatization with appropriate POS tag

Stemming was also applied but did not produce good results with the given data.

The cleaned data is used to create Bag-of-Words which are modelled using the following techniques to generate features for our models:

- *CountVectorizer*: This Bag-of-Words model converted the collection of video captions to a sparse matrix representation of token counts. It counted the number of times a word showed up in the document and used this value as its weight in the matrix [4].
- *TfidfVectorizer*: This vector space model converted the collection of captions to a matrix of TF-IDF features. TF-IDF is a statistical measure used to evaluate the importance of a word with respect to a document in a collection or corpus. This model used this value as a weight in the matrix representation [3][4].
- *One-hot encoding*: This technique is used to create one vector per document per input. The length of the vectors is the total size of the vocabulary. This model used binary values as weights based on whether or not each word is present in the document [5].

C. Modelling

The generated features are modelled using the following three model:

1. A Keras sequential Neural Network (NN) model is built with three layers for training the data. Early stopping monitor and a dropout layer are used to prevent overfitting.
2. Decision Tree Regression Model (DTR)
3. Random Forest Regression Model (RFR)

- Ensemble Model: The results of the above three models with different features are compared and analysed. Weighted ensemble models are built

using the two best predictions from two different models [6]. One ensemble model is set-up for predicting short-term memorability and another one for predicting long-term memorability. The weights that produce the lowest MSE (Mean Squared Error) are chosen as the optimal ones. For discovering the optimized weights, two methods, namely Sequential Least SQuares Programming (SLSQP) [7] and 'Nelder-Mead'[8][9] are used. The change in weights made by the Nelder-Mead method is found more efficient, and hence it is used to find the optimized weights for the ensemble models.

IV. RESULTS AND ANALYSIS

Table 1 presents the short-term and long-term memorability scores predicted by the models using various features. The scores are calculated using Spearman's rank correlation coefficient.

Table 1: Results of different features and models

Feature Extraction Model	Model	Short Term Score	Long Term Score
CountVectorizer	Sequential Neural Network	.433	.182
	Decision Tree Regression	.288	.131
	Random Forest Regression	**.400**	**.180**
TfidfVectorizer	**Sequential Neural Network**	**.436**	**.207**
	Decision Tree Regression	.249	.134
	Random Forest Regression	.379	.151
One-Hot Encoding	Sequential Neural Network	.413	.179
	Decision Tree Regression	.283	.130
	Random Forest Regression	.414	.169

It is observed from Table 1 that the sequential NN model performed consistently well with all the three inputted features and predicted the best results with the feature extracted using *TfidfVectorizer*. Decision Tree Regression model performed considerably poorly in predicting both short-term and long-term scores for all the three inputted features. Random Forest Regression model got decent scores and achieved the best results considering both short-term and long-term predictions with the input generated from *CountVectorizer*.

For building the weighted ensemble models, the predictions of sequential NN model using *TfidfVectorizer* and Random Forest Regression model using *CountVectorizer* are used. The optimized weights found using Nelder-Mead method are as follows:

- Ensemble 1 (for Short-Term Memorability):

  0.44 * RFG-CountVectorizer + 0.56 * NN-TfidfVectorizer

- Ensemble 2 (for Long-Term Memorability):

  0.24 * RFG-CountVectorizer + 0.76 * NN-TfidfVectorizer

Table 2: Results of weighted Ensemble Models

Model	Short-Term Score	Long-Term Score
Ensemble 1	.456	.219
Ensemble 2	.460	.223

It is evident from Table 2 that both the ensemble models improved prediction scores and produced better results than Table 1 models.

V. CONCLUSION AND FUTURE WORK

It is noticed from the results that caption features can produce decent scores individually. All the Bag-of-Words model generated good results but *TfidfVectorizer* produced better results for long term memorability prediction than other feature extraction models. The sequential Neural Network model produced better short-term and long-term scores than the other models with all the three inputted features. The ensemble models improved the prediction scores of the individual models by using the optimal weights produced by the Nelder-Mead method.

For future work, an unsupervised learning and pre-trained algorithm GloVe (Global Vectors for word representation) can be used for generating features from captions. The results from GloVe are pretty impressive as it produces word embeddings by aggregating global word-to-word co-occurrence matrix from a corpus [10]. Building ensemble models using the top predictions of more than two models and trying out different variations can yield better scores. Furthermore, Parallel grid search and Multi-Layer Perceptrons (MLP) weighted ensemble techniques can be used to improve the accuracy of weights of the models.

REFERENCES

[1] Gupta, R. and Motwani, K., 2018. Linear Models for Video Memorability Prediction Using Visual and Semantic Features. In MediaEval.

[2] Romain Cohendet, Karthik Yadati, Ngoc Q.K. Duong, and ClaireHélène Demarty. 2018. Annotating, Understanding, and Predicting Long-term Video Memorability. In Proceedings of the 2018 ACM.

[3] Vijayarani, S., Ilamathi, M.J. and Nithya, M., 2015. Preprocessing techniques for text mining-an overview. International Journal of Computer Science & Communication Networks, 5(1), pp.7-16.

[4] Tripathy, A., Agrawal, A. and Rath, S.K., 2015. Classification of Sentimental Reviews Using Machine Learning Techniques. Procedia Computer Science, 57, pp.821-829.

[5] Zhang, X. and LeCun, Y., 2017. Which encoding is the best for text classification in chinese, english, japanese and korean?. arXiv preprint arXiv:1708.02657.

[6] Sollich, P. and Krogh, A., 1996. Learning with ensembles: How overfitting can be useful. In Advances in neural information processing systems (pp. 190-196).

[7] Shahhosseini, M., Hu, G. and Pham, H., 2019. Optimizing ensemble weights and hyperparameters of machine learning models for regression problems. arXiv preprint arXiv:1908.05287.

[8] Nelder, J.A. and Mead, R., 1965. A simplex method for function minimization. The computer journal, 7(4), pp.308-313.

[9] Escobedo, F.A. and Martinez-Veracoechea, F.J., 2008. Optimization of expanded ensemble methods. The Journal of chemical physics, 129(15), p.154107.

[10] Makarenkov, V., Shapira, B. and Rokach, L., 2016. Language models with pre-trained (GloVe) word embeddings. arXiv preprint arXiv:1610.03759.