Predicting Short-term Media Memorability from Captions

An approach for maximising caption-based short-term memorability

Luke Scales
School of Computing
Dublin City University
Dublin, Ireland
luke.scales2@mail.dcu.ie

ABSTRACT

The leading edge of entries to the MediaEval Predicting Media Memorability Task of 2019 utilised models with embedded captions to succeed [1][2]. Captions alone have been proven to be useful in achieving high scores, particularly in the short-term memorability category. The ensembled approach of the winning entry in the 2019 task relied upon captions embedded using the GloVe embeddings model, however many other embeddings models are readily available [1]. This paper presents a competitive model for predicting short-term memorability from captions which was developed via text preprocessing, model design, and analysing a variety of text embeddings methods.

1 Introduction and Related Work

A plethora of word embedding models and Natural Language Processing (NLP) techniques are currently available for use in text-based machine learning (ML) tasks [3]. The results of the 2019 MediaEval Media Memorability task competitors, such as those of DCU Insight and MeMAD, have shown that captions alone can provide promising results in the short-term memory category. When word embeddings are employed these results can be improved significantly [1].

This work was heavily influenced by the MediaEval 2019 entry of DCU. DCU achieved the best-in-class results in the 2019 competition through ensembling results from various models. These models were trained on embedded captions, the output of a pre-trained ResNet152 model, as well as models trained on the DCU team's own pre-computed emotions and aesthetics features [4].

While the GloVe embeddings used by the DCU team have been very successfully applied here and in other tasks, there are many competitive embedding models available [3]. The DCU team also achieved remarkable results without preprocessing or cleaning the captions. This paper aims to compare some other embedding methods to find a potential alternative to the GloVe embeddings, while exploring text pre-processing techniques, in an attempt to maximise the accuracy of the model being used.

2 Approach

The captions were first preprocessed to correct any spelling mistakes (e.g. when checking the least common words of the raw captions, there were erroneous entries such as "`longof`", "`windowsk`" etc.). While some tokenizers and word embeddings will account for these mistakes by assigning appropriately insignificant indexes or by assigning them to out of vocabulary (OOV) buckets, they still affect the model. Potentially significant words that are misspelled, such as "windows" in the example errors shown, will not be correctly accounted for by the model.

To correct the spelling mistakes the python autocorrect library was used as a spell checker [5]. From analysing entries corrected by the spell checker many more errors in the captions were noticed which required manual correction. A list of words to be ignored by the spell checker were added, as it tried to unnecessarily correct some nouns to the closest common word (e.g. it replaced "ipad" with "pad" and corrected the names of some places), as well as a series of manual spelling corrections.

A variety of embeddings with OOV buckets were tested from TensorFlow Hub. The first embeddings tested were two variations of Google's GNews "Swivel" embeddings [6][7], an embeddings model trained on the Google News 130GB corpus using Google's Submatrix-wise Vector Embedding Learner (Swivel) method [8]. Both of these embeddings map text to 20-dimensions: the first uses 1 OOV bucket for unknown tokens while the second (annotated as "with-OOV") sends around 2.5% of the least frequent tokens and embeddings to hash buckets.

The other two models tested also send a small percentage of the least frequent tokens and embeddings to hash buckets but are based on a different word embeddings model [9][10]. The models (named "nnlm-en-dim50" and "nnlm-en-dim128") are based on a feed-forward Neural-Net Language Model [11] with pre-built OOV. The first is trained on the English Google News 7B corpus (with 7 Billion words) and maps inputted text to 50 dimensions. The second is trained on the equivalent 200B corpus and maps text to 128 dimensions

An initial model was used to experiment with train/validation splits and trainable/non-trainable embedding layers. The model used was taken from the core of the model used in the DCU entry (without the Recurrent Unit). It was developed as a Keras Sequential model with 8 layers (an input embedding layer, 1024, 512, and 256 relu activated Dense nodes separated by 0.25 Dropout layers, followed by a single Dense node with a sigmoid activation function). With results from the preprocessing and embeddings experiments the final model parameters were found through hyperparameter tuning.

Google Colab was used to train these models online and explore the dataset however the limitations of using this platform with a (free) hosted environment impeded training and analysis. The use of GPUs and TPUs is limited when running in Google's free hosted environments. To circumvent this a virtual machine (VM) was created on Google Cloud Platform (GCP) to train and run the experiment scripts. Port forwarding was used to connect colab to a local environment which utilised the processing power of the VM instance created.

3 Results and Analysis

By analysing the corrections made by the spell checker three main issues were noticed. The first is that many errors were present in the data that could not be corrected by the spell checker (and were not corrected manually), such as two words being concatenated into one in the captions (a common example is "`steadicamof`"). For these issues the spell checker would assume it to be one word and this would result in lost data.

The second issue noticed is that the spell checker made corrections on words that did not need to be corrected, which, as mentioned in the "Approach" section, were mostly attempts to replace nouns that were unrecognised by the spell checker. It is likely that a number of corrections are not accounted for in the test set.

The third (potential) issue that became evident was that some captions repeated in the provided devset. From checking the captions dataframe using the pandas nunique() function it showed that only 5199 out of 6000 captions were unique. This could potentially be from videos being flipped or rotated to create new data points, with the assumption that a video's memorability is unaffected by direction of the video, however there is no mention of this on the task website.

3.1 Preprocessing Methods Comparison

Experiments on the preprocessing techniques showed that accuracy increased for most models on a short 20 epoch training cycle when spelling errors were corrected. The results of this comparison (Table 1) indicated that training the 128 dimension model using spell checked captions could provide the most accurate results. The NLP techniques did not yield consistent improvements across all models however.

Table 1

	Spearman correlation coefficient			
NLP Methods	**Swivel 20dim**	**Swivel 20dim OOV**	**NNLM 50dim**	**NNLM 128dim**
Raw captions	0.328,	0.312	0.346	0.392
Spell checker	0.344	0.290	0.361	0.422
Stopwords	0.339	0.314	0.382	0.379
Lemmatized	0.327	0.321	0.357	0.392

A second round of testing was performed when the VM was deployed (Table 2), with the 1st run at 20 epochs and the 2nd at 50 epochs. This emphasised that the 128 dimension NNLM embeddings model was the most promising of the cohort.

Table 2

	Spearman correlation coefficient			
NLP Methods	**Swivel 20dim**	**Swivel 20dim OOV**	**NNLM 50dim**	**NNLM 128dim**
Raw captions	1: 0.319 2: 0.301	1: 0.317 2: 0.342	1: 0.372 2: 0.393	1: 0.411 2: 0.420
Spell checker	1: 0.323 2: 0.328	1: 0.302 2: 0.328	1: 0.365 2: 0.398	1: 0.414 2: 0.408
Stopwords	1: 0.359 2: 0.351	1: 0.325 2: 0.305	1: 0.397 2: 0.419	1: 0.379 2: 0.418
Lemmatized	1: 0.332 2: 0.329	1: 0.321 2: 0.332	1: 0.383 2: 0.389	1: 0.377 2: 0.384

3.2 Embeddings Comparison

The various embedding methods were compared using the initial model to isolate the most promising method. The model was trained using one-hot encoding as a control for the experiment while each of the embeddings methods were trained with both a trainable and non-trainable embedding layer. As per Table 3 it was found that the NNLM 128 dimension model using a non-trainable embedding layer resulted in the best performance for the initial 20 epoch experiment.

Table 3

	Spearman correlation coefficient	
Embedding Method	**Trainable Embeddings**	**Non-Trainable Embeddings**
one-hot encoding	N/A	0.377
GNews Swivel 20D	0.316	0.333
GNews Swivel 20D OOV	0.334	0.318
NNLM 50D OOV	0.359	0.383
NNLM 128D OOV	0.383	0.389

3.3 Final Model and Hyperparameter Tuning

While some initial work was done in a hosted environment, the final model was entirely trained in GCP. The initial results showed the 128 dimension NNLM model to be the most promising so this was used in the embedding layer. L2 kernel regularizers were applied to the 1024 node and 512 node Dense layers of the network to prevent overfitting of the model. With captions alone the results suggest a competitively accurate model with over 0.45 Spearman correlation coefficient in the short-term category when evaluating the model against the validation set. This score was achieved with a Dropout of 0.25 initially, before hyperparameter tuning.

Hyperparameter tuning was performed on GCP to test a range of Dropout values. Some issues and time constraints prevented the tuning of more parameters. From the results (which can be viewed using the notebook tensorboard extension) it was found that a Dropout of 0.2 produced the best results of the tested values. With hyperparameter tuning performed at 500 epochs, final models were trained taking the best Mean Squared Error of each run of 1000 epochs using Keras ModelCheckpoint.

This work focused on short-term memorability but the model was applied to the long-term data also for completeness. The model was applied as-is following tuning for the short-term category, producing reasonable long-term scores (Table 4). All models were trained with a 90/10 split of 5400 training to 600 validation points. As can be seen in the results this produced effective short-term memorability models, particularly in the case where the spell-checker and stopword removal was combined.

Table 4

	NNLM 128dim - Spearman Coefficient	
NLP Methods	**Short-term**	**Long-term**
Raw captions	0.475	0.176
Spell checker	0.466	0.196
Stopwords	0.481	0.203
Lemmatized	0.474	0.233

4 Discussion and Outlook

This work highlights the erroneous nature of the captions in the dataset, yet provides evidence that this does not adversely affect the predictability of the video memorability. The assumption held by the writer at the beginning of the experiments was that model accuracy would increase from correcting spelling mistakes and reducing the data to the most pertinent points (by removing stopwords and through lemmatization). While this work proves that reducing the data using these NLP techniques did produce more accurate results, it did so on top of captions with corrected spellings. Making spelling corrections alone did not reliably increase the accuracy of *this* model, however it could be beneficial with other embeddings such as GloVe.

It is plausible that the incorrect spellings in the raw captions are of benefit to this model: due to OOV buckets, the repeatedly misspelled words such as "`steadicamof`", may direct the development of weights within the model in a beneficial manner as they are assigned unique embedding value. It raises the possibility for improvement if an NNLM model is trained on the dataset directly to develop a set of embeddings specific to this task. It is also plausible that further corrections to the captions could possibly result in further improvements to the accuracy of the model in unison with the other NLP techniques.

Such improvements in the captions could be achieved through the use of enhanced spell checking, such as from the use of different spell checkers or from cross-referencing multiple spell checkers. An ML solution could be applied as a spell checking model also, even simply to detect bigrams being misrecorded as one word (like the examples “`longof`” and “`steadicamof`” mentioned above). More manual intervention in the spell checking process (similar to that used in this project) may produce more accurate results also.

While the work was successful in developing a competitive caption-based model through NLP techniques, these were stacked on top of the effects of the spell checking. This approach could yield positive results through applying the NLP techniques directly to the raw captions rather than with the use of a spell checker as shown here. By removing the stopwords from the spell-checked captions the evaluation on the validation set saw a noticeable improvement over raw captions and spell-checked captions in the short-term category. It would be insightful to show the accuracy of a model trained on raw captions without stopwords. The results suggest that this has the potential to be an effective tactic but further processing and analyses will be needed to confirm this.

Further work should aim to increase the accuracy of this model with the ultimate goal of ensembling it with other models. The use of Recurrent Neural Network units in this model would be an easily applied addition that could greatly increase the accuracy of the model. For example, the DCU entry utilised a Gated Recurrent Unit (GRU) [12] alongside their embeddings layer to great effect. The use of a GRU or a Bidirectional Long Short-Term Memory (LSTM) cell with this model could be explored in further work. Bi-directional LSTMs have been used successfully in models trained on sentences for various tasks. Similarly other embeddings models that are trained on sentences, such as ELMo or InferSent, could be tested here, as well as the use of a scheduled decaying learning rate.

Some ML models are available which produce captions for videos and images, such as those used in the Microsoft Common Objects in Context (MS COCO) Challenges [13]. Whether these models would be effective with this task or not is unknown, but the caption outputs of these models could be inputted to the model used here. Lastly, with more time and/or computing resources, more detailed hyperparameter tuning could be performed to maximise the accuracy of this model. In particular, further work could be performed to improve the accuracy in the long-term category. The captions with spell checking, stopword removal, and lemmatization produced optimistic results without being optimised for this category. One can reasonably assume that these results will be further enhanced with hyperparameter tuning specific to this category.

ACKNOWLEDGMENTS

The writer thanks ICT Skillnet for part-funding the degree that this work was performed for, as well as Tomas Ward and Eoin Brophy for their teachings and guidance throughout this project. Acknowledgements are also due to the Google Cloud Platform Support Team and Leonard from the Google Compute Team for expediting the GPU quota increase request.

REFERENCES

[1] Azcona, D., Moreu, E., Hu, F., Ward, T. E., and Smeaton, A. F. 2019. *Predicting Media Memorability Using Ensemble Models*. The Predicting Media Memorability Task at Mediaeval 2019. (2019).

[2] Laaksonen, J., Troncy, R., and Reboud, A. *MeMAD Project, Media Memorability.* 2019. The Predicting Media Memorability Task at Mediaeval 2019.(2019). https://github.com/MeMAD-project

[3] Perone, C. S., R. Silveira, and T. S. Paula. Evaluation of Sentence Embeddings in Downstream and Linguistic Probing Tasks. arXiv preprint arXiv:1806.06259, 2018.

[4] Azcona, D. 2019. *Insight@DCU in the Memorability Challenge at MediaEval2019* https://github.com/dazcona/memorability.

[5] Sondej, F., 2019. *Autocorrect: Spelling corrector in python* https://github.com/fsondej/autocorrect.

[6] Google. tf2-preview/gnews-swivel-20dim. *Token based text embedding trained on English Google News 130GB corpus*. TFHub. https://tfhub.dev/google/tf2-preview/gnews-swivel-20dim/1 .

[7] Google. tf2-preview/gnews-swivel-20dim-with-oov. *Token based text embedding trained on English Google News 130GB corpus*. TFHub. https://tfhub.dev/google/tf2-preview/gnews-swivel-20dim-with-oov/1.

[8] Shazeer, N., R. Doherty, C. Evans, and C. Waterson. *Swivel: Improving Embeddings by Noticing What's Missing*. arXiv preprint arXiv:1602.02215, 2016

[9] Google. tf2-Preview/nnlm-En-dim50. *Token based text embedding trained on English Google News 7B corpus.* TFHub. https://tfhub.dev/google/tf2-preview/nnlm-en-dim50/1.

[10] Google. tf2-Preview/nnlm-En-dim128. *Token based text embedding trained on English Google News 200B corpus.* TFHub. https://tfhub.dev/google/tf2-preview/nnlm-en-dim128/1.

[11] Bengio, Y., R. Ducharme, P. Vincent, and C. Jauvin. *A Neural Probabilistic Language Model*. Journal of machine learning research, Vol. 3, No. Feb, 2003, pp. 1137–1155.

[12] Cho, K.; van Merrienboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. (2014). "*Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation*". arXiv:1406.1078

[13] Laina, I., C. Rupprecht, and N. Navab. Towards Unsupervised Image Captioning with Shared Multimodal Embeddings. 2019.