Generating titles from movie posters

Elin Hagman
Project Report for Artificial Intelligence:
Cognitive Science
University of Gothenburg
gusihaliel@student.gu.se

Abstract

Automatic image description makes it possible to generate textual descriptions from images. Generating a textual description from an image is a challenging artificial intelligence problem, but recently deep learning methods have achieved good results in this task. This project's approach is inspired by this problem and the goal is to implement a language model that can generate movie titles from poster images.

1 Introduction

In recent years, the field of natural language processing has seen an increased interest in problems that require a combination of linguistic and visual information, such as understanding images and analysing and generating text. For example, the task of automatic image description involves analysing the visual content of an image and generating a textual description that verbalizes the most relevant features of the image (Sun and Ren, 2017). This project investigates the possibility of creating a language model that could generate interesting movie titles from posters, similar to how an image captioning model generates textual description from an image.

1.1 Related work

Janelle Shane created a character-level language model using a multi-layer Recurrent Neural Network to generate story titles from plot summaries (Story titles, invented by neural network, 2018). The dataset used was the WikiPlots corpus which has a collection of 112,936 story plots extracted from English language Wikipedia (wik). The dataset searches language articles that contains a sub-header with words like ”plot” or ”plot summary” and therefore includes summaries from movies, books, TV episodes, video games, etc. The trained model consistently came up with titles that were both varied and plausible, like ”Pirates: A Fight Dance Story”, ”Cannibal Spy II” and ”Conan the Pirate”.

2 Method

2.1 Dataset and modules

The data used in the project comes from the Movie Genre from its Poster dataset, available at Kaggle. The dataset contains 40108 items in total, including IMDB Id, IMDB Link, Title, IMDB Score, Genre and link to download the movie posters, obtained from the IMDB website. Each Movie poster can belong to at least one genre and can have at most 3 genre labels assigned to it (IMD). For the first part of this project, pre-trained sub word embeddings were used to create vector representations from the movie titles. The embeddings were generated using BPEmb: Tokenization-free Pre-trained Subword Embeddings in 275 Languages. The embeddings are a collection of pre-trained subword embeddings based on Byte-Pair Encoding (BPE) that are trained on Wikipedia articles (Heinzerling and Strube, 2018). Visual feature vectors of the image posters were also used. The image model used to create the vectors was the ResNet model (He et al., 2016), which is pretrained on the ImageNet classification dataset (Russakovsky et al., 2015). The image representations were generated by feeding the image posters into the model and copying the output of the final layer. This resulted in a 512-dimensional vector representation of each image.

2.2 Method and process

In this section, the method that were used is presented, as well as the processes that were carried out in order to get the results. We start by extracting the movie title, genre and image name from

the original dataset, as well as downloading each movie poster using the URL. Items without title or with a broken URL were removed from the dataset, resulting in a remaining 37742 items that were used in this project.

For the image vectors, a simple approach was to feed an image into the ResNet model in order to get the convoluted image representations. Since ResNet assumes that each image has RGB channels, the grayscale images in the dataset were converted RGB images using Python Imaging Library. For faster processing time, the smallest model ResNet-18 was selected. This model outputs an image representation vector of 512 dimensions for each image, which were all generated and saved into a NumPy file.

The token vectors are generated using English pre-trained embeddings from BPEmb that uses Byte-Pair Encoding (BPE) to generate tokens from a string of characters. BPE is an unsupervised subword segmentation method that iteratively merges the most frequent token pair of a string into a new token (Sennrich et al., 2016). The BPEmb vocabulary size determines the number of BPE generated, therefore a smallest vocab size of 1000 characters was selected to yield the highest number of tokens for a title. Each title string is encoded into a list of integers using the BPEmb module before being fed into the language model.

The main model was a language model implemented in Pytorch. The purpose of the language model is to learn the likelihood that a token occurs based on the previous sequence of tokens used in the input title. This knowledge could then be used to generate plausible movie titles. The model's architecture is modelled after a Recurrent Neural Network (RNN). The RNN model utilizes an embedding layer to vectorize the movie titles and concatenates each token embedding vector with the corresponding image vector. The model accepts a list of movie title encoded to indices mappings using the BPEmb model and the image vectors as input. The token indices are then passed through the embedding layer to generate the vector representations for the tokens. Then, each batch of token vectors are padded before being concatenated with the image vector. The model was fitted against a set of validation data during the training.

The output of the model is a list of sequences that are first concatenated, and then fed into the function calculating the cross-entropy loss between the predicted and the actual token. The weights of the model are then adjusted by the error from the loss function using backpropagation. To conclude if the model regarded the visual inputs at all, a test was made by calculating the loss based on the token representations and a tensor filled with the scalar value 0, of the same size as the image. This was plotted together with the true loss.

Since there was no time to create the decoder, an alternative for evaluation the model was suggested by creating a dataset that included a set of movie posters and fake titles, as well as movie posters and their true titles. The hypothesis is that, when comparing the loss from the true set versus the false set, it should be able to conclude if the model prefers the correct title for the image poster or not.

3 Results

The loss for the data containing the image poster remained around the same values remained constant during training, while the loss for the data without a poster steadily increase, showing that the visual representation is meaningful and that the model is learning from them.

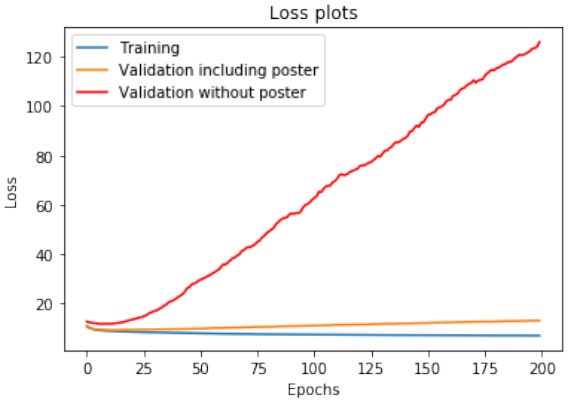

Figure 1: Language model loss

Table 1: Predicted titles for movie posters.

Title	Predicted	p	t	n
Ultra	Adem	15.51	17.11	5.70
Adem	Adem	15.06	15.06	7.53
Gerhard Richter–Painting	Adem	18.85	127.07	12.71
Catch and Release	Adem	15.27	100.68	16.78
Finding Dory	Finding Dory	14.60	14.60	2.92
Skew	Skew	16.72	16.72	5.57
What Women Want	Adem	15.06	28.60	4.09

The model predicted 37% of the titles correctly. Shorter titles produced a lower loss, regardless of image poster. The table shows the true and predicted titled, as well as the loss for the predicted

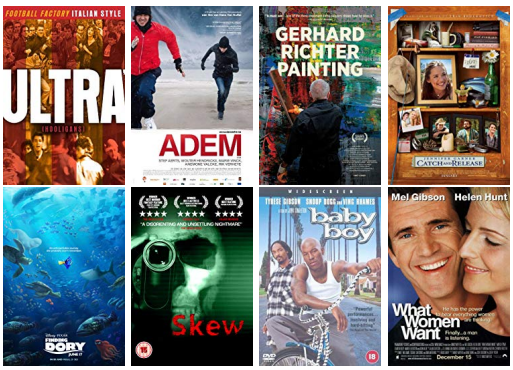

Figure 2: Movie poster images used for evaluation. (a) Ultra; (b) Adem; (c) Gerhard Richter – Painting; (d) Catch and Release; (e) Finding Dory; (f) Skew; (g) What Women Want

titles, the true loss and the normalized loss. p = *predicted* and t = *true* and n = *normalized*.

4 Discussion

The final purpose of the model was to build a language model that would generate movie titles. However, there was no time to implement this part. Instead, the model was evaluated by comparing movie posters with fake titles. The results indicate that the model seems to prefer shorter titles and creates higher predictions for all shorter titles compared to the longer titles, regardless of what image poster is selected. Further investigation is needed to produce better evaluation results and possible improvements of the model. Further work would include implementing this decoder and combining it with the current model in order to generate movie titles.

The selected module for subword embeddings is only trained on English, however there are several non-English titles in the dataset. Changing this module to a multilingual subword segmentation model might improve the model. Other improvements made would be to add genre as a parameter for training to see if genre helps to improve the model's performance. Using different vocab sizes for the BPEmb embeddings compare them to higher dimension ResNet models might also help to improve the model's performance.

References

Movie Genre from its Poster: Predicting the genre of the movie by analyzing its poster. https://www.kaggle.com/neha1703/movie-genre-from-its-poster. Accessed: 2020-01-16.

Wikiplots. https://github.com/markriedl/WikiPlots. Accessed: 2020-01-16.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. pages 770–778.

Benjamin Heinzerling and Michael Strube. 2018. BPEmb: Tokenization-free Pre-trained Subword Embeddings in 275 Languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Olga Russakovsky, J. Deng, Hao Su, J. Krause, Sanjeev Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, and M. Bernstein. 2015. Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.*, 115:1–42.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. pages 1715–1725.

Yan Sun and Bo Ren. 2017. Automatic image description generation with emotional classifiers. pages 748–763.