**Introduction**

In recent years, the field of natural language processing (NLP) 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 its visual content of an image and generating a textual description that verbalizes the most relevant features of the image. (ref) This project investigates the possibility of creating a language model that generates interesting movie titles from posters, similar to how an image captioning model generates textual description from an image.

**Related work (?)**

Janelle Shane created a character-level language model using a multi-layer Recurrent Neural Network to generate story titles from plot summaries (ref). The dataset used was the WikiPlots corpus which has a collection of 112,936 story plots extracted from English language Wikipedia (ref). The dataset searches language article that contains a sub-header with words like "plot" or "plot summary" and thus includes summaries from movies, books, tv episodes, video games, etc. The 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" (ref)

**Dataset**

The data used in the project comes from the Movie Genre from its Poster dataset (2018), available at Kaggle (ref). 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.

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. (reference)

Visual feature vectors of the image posters were also used. The image model used to create the vectors was the ResNet-18 (ref) model, 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 and saving them into a Numpy file. This resulted in a 512-dimensional vector representation of each image.

**Method**

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 (ref) 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 (ref) For faster processing time, the smallest model Resnet-18 (ref) 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 pre-trained embeddings from BPEmb: Subword Embeddings in 275 Languages (ref) that uses Byte-Pair Encoding (BPE) (Sennrich et al, 2016) 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. 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 main model.

The main language model was implemented in Pytorch. 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 using an out-of-vocabulary index before being concatenated with the image vector.

The output of the model is a list of sequences that are 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 using the error from the loss function using backpropagation.

**The generation part (not finished)**

A decoder part should be added to the model generate the output of the statistically most likely title. that should apply to the image.

**Evaluation**

**References**

BPEmb: Tokenization-free Pre-trained Subword Embeddings in 275 Languages

Neural Machine Translation of Rare Words with Subword Units

Deep Residual Learning for Image Recognition

WikiPlots corpus (<https://github.com/markriedl/WikiPlots>)

Story titles, invented by neural network (<https://aiweirdness.com/post/160014619217/story-titles-invented-by-neural-network>)

ResNet

Movie Genre from its Poster: Predicting the Genre of the movie by analysing its poster (<https://www.kaggle.com/neha1703/movie-genre-from-its-poster>)