Project Proposal for Video captioning on GIFs

Anonymous CVPR submission

Paper ID *****

Abstract

The problem of generating captions for a GIF (Graphics Interchange Format) file or video is a topic which connects computer vision and natural language processing. In this project, the group tries to build a generative model based on image recognition convolutional neural network and the Decoder-Encoder type model which is implemented by LSTM [2]. This architecture combines common advances in computer visions and machine translation, which can be used to generate captions describing a video or GIF. Experiments have been taken on several different datasets which show preliminary test results.

1. Problem Formulation

1.1. Input Format and Output Format

The description of a GIF should contain not only the main objects contained in the image, but also tell how the objects relate to each other by their movements and activities they are involved in. In this work, a sequence-to-sequence model is proposed, which is trained end-to-end, and able to learn arbitrary temporal structure. Essentially, the input is a series of GIF frames and the output is a series of words.

1.2. Optimized Variables and Optimization Objectives

The problem can be interpreted by the process of transferring the image sets (frames) to a fixed-length vector and it should be as close as possible to the target sentence vector. The work of this paper is to present an end-to-end system as a neural network model. The pre-trained CNN model will extract the features of the frames data and the encoder-decoder neural network will learn the feature vector and give an output of a fixed length vector and uses stochastic gradient descent method to optimize the parameters. The main goal of the model is to maximize the result of the following equation:

$$P(C|S), \tag{1}$$

where C is the caption that the model generates and S is the target description of the GIF.

2. Related Work

The method in this report is inspired by [7]. The model uses a sequence-to-sequence and encoder-decoder architecture, using LSTM (Long Short Term Memory) [2] as both the encoder and decoder, with series of images as the input. Features are then extracted using VGG16 [4], a pre-trained CNN model. The words in the sentence pair are tokenized.

3. Methodology

3.1. Overall Structure

In terms of model architecture, a sequence-to-sequence model will act as the baseline model. Inside the baseline model, a pre-trained VGG16 [4] model is adopted to generate feature maps that are fed into the sequence to sequence (seq2seq) model. Both the encoder and decoder use LSTM, which is a special kind of recurrent neural network that can store long term dependencies and has a forget gate that selectively forgets information [2].

For the encoder, a sequence of feature maps is fed into LSTM [2] and all the intermediate output from the encoder are discarded. For the decoder, The hidden state of the decoder is the output of the encoder. Therefore "<SOS>" acts as the first input of the decoder. The caption is generated until "<EOS>" is output from the decoder.

In Fig. 1, the video is first divided into frames of PNGs (Portable Network Graphics) and reshaped to 224×224 pixels. Then the resized images are passed to VGG16 [4] model to compute a feature map of size 4096. The latent feature maps are fed into the corresponding LSTM [2] cell inside the encoder until the 80th frame. The final state of the encoder captures all the features inside the 80 frames and all the intermediate outputs from the encoder are discarded. After the encoder processes all the frames, The hidden state of the decoder is the output of the encoder. Then the "<SOS>" token is sent as the input of the first LSTM [2] cell in the decoder to start generating the caption until "<EOS>" is obtained as the output.

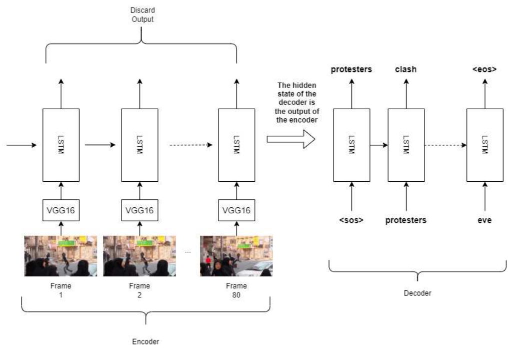


Figure 1. Sequence to sequence model.

3.2. CNN

Convolutional neural networks (CNNs) are deep learning methods for local region feature extraction. CNNs have stacked convolution layers and pooling layers that can be used to reduce the feature dimension [3]. In a convolution network, there is a set of kernels K and additional biases B. A set of new feature maps F can be calculated and a non-line transformation σ is applied to all features in an element-wise form [3]. For each convolution layer x, this process is repeated. A spatial convolution is described as below:

$$F_n^x = \sigma\left(W_n^{x-1} \cdot F^{x-1} + B_n^{x-1}\right) \tag{2}$$

The size of the new feature maps are as follows:

$$H^x = \frac{H^{x-1} - h^{x-1} + 2 \cdot p^{x-1}}{S^{x-1}} + 1 \tag{3}$$

$$W^x = \frac{W^{x-1} - w^{x-1} + 2 \cdot p^{x-1}}{S^{x-1}} + 1, \tag{4}$$

where the size of the new feature map is $H^x \times W^x$, p^{x-1} is the padding size of $(x-1)^{\text{th}}$ layer. S^{x-1} is the stride size of $(x-1)^{\text{th}}$ layer. h^{x-1} and w^{x-1} are the height and width of the convolution kernel respectively.

In terms of the pooling layers, the model aims to reduce spatial dimensions of feature maps. Pooling can help extract features with higher representation ability and increase the speed of convergence. Three types of pooling are applied, including max pooling, L2-norm pooling and average pooling. This usually necessitates the use of a pooling kernel with the same stride length. A pooling layer can be used to create a new feature map. The size of the new feature map is as follows:

$$H^x = \frac{H^{x-1} - h_2^{x-1}}{S^{x-1}} + 1 \tag{5}$$

$$W^x = \frac{W^{x-1} - w_2^{x-1}}{S^{x-1}} + 1, \tag{6}$$

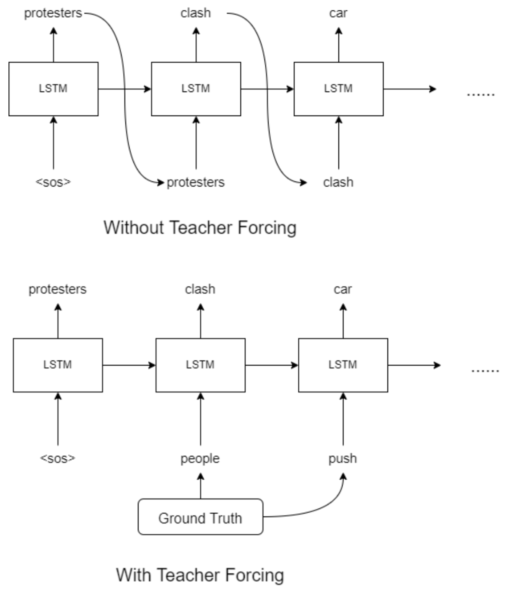


Figure 2. Diagram showing difference between using teacher forcing and not using teacher forcing.

where h_2^{x-1} and w_2^{x-1} are the height and the width of $(x-1)^{\text{th}}$ pooling kernel. For max-pooling, the feature map outputs the maximum value in each sub-region after max pooling.

3.3. Seq2seq

The seq2seq model is used to generate the caption by inputting the numerous frames. The seq2seq model has two LSTM [2] layers to learn about the representation of the features map generated by CNN model. The first LSTM serves as the encoder and receives input from the pre-trained VGG16 model. The second LSTM aims to generate the caption by decoding the output from the encoder. To train the seq2seq model, teacher forcing, masked loss and gradient clipping is used to optimize the model.

3.3.1 Teacher Forcing

Teacher forcing is a training technique used to efficiently train recurrent neural network models that manipulate the ground truth as the input instead of using the previous output as input [9]. Teacher forcing can help the training converges faster since the model in the early stage has poor performance and this technique helps to model to self correct instead of accumulating the error when there is no teacher forcing, as shown in Fig. 2. However, this will create differences between actual inference and training as there are no more ground truth provided during the inference phrase and this will lead to exposure bias.

3.3.2 Masked Loss

In a seq2seq model, padding is a special form of masking which encodes the sequential data into contiguous batches and makes sure all sequences has the same length [8]. Padding is applied to standardize the length of data. In training, calculating the loss is critical and during training, the padded zeros are ignored [8]. Therefore masked loss is applied to tackle the problems raised by masking.

3.3.3 Gradient Clipping

Gradient clipping is a training technique to prevent exploding gradients in a deep learning model [1]. Exploding gradients means a large increase in the norm of the gradient in the training process, and it may caused by the long temporal gap from the input to the output as it is hard for seq2seq model to learn long-distance dependencies.

Gradient clipping prevent gradients from blowing up by rescaling them respective to the norm and the threshold value η.

$$g \leftarrow \frac{\eta g}{\parallel g \parallel} \quad if \parallel g \parallel > \eta, \quad \text{where g is the gradient} \tag{7}$$

The changes to the weights will also be rescaled as a result of rescaling the error derivative, considerably reducing the risk of an overflow or underflow. Even if the model's loss landscape is uneven, this enables gradient descent to behave reasonably.

3.4. Evaluation Metrics

In terms of model evaluation, the BLEU (BiLingual Evaluation Understudy) [6] score is adopted. The BLEU score is a value that measures the similarity between machine generated text and high quality translation references and has low marginal running cost [6].

3.5. Dataset

The dataset used for training and validation is the ACM MM 2020 large-scale video-language pre-training dataset, which contains 163183 GIF videos and 164378 sentences in Auto-captions on GIF, and the vocabulary size is 31662 [5]. This dataset contains a wide range of GIF and sentence pairs, all crawled from the internet. The category of GIFs includes animation, science, sports, movies, memes, animals and humans. The size of GIFs range from 80 × 80 to 1500 × 1500. All pairs are filtered based on the polarity annotations via the Natural Language Toolkit, the repetitions and whether the sentence contains questions or specific names.

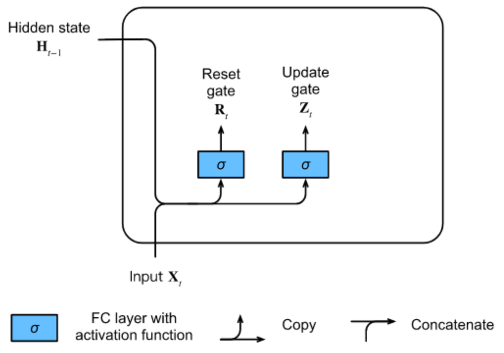

Hidden state $\mathbf{H}_{t-1}$
Reset gate $\mathbf{R}_t$
Update gate $\mathbf{Z}_t$
σ
σ
Input $\mathbf{X}_t$
FC layer with activation function
Copy
Concatenate

Figure 3. Diagram showing the structure of a GRU.

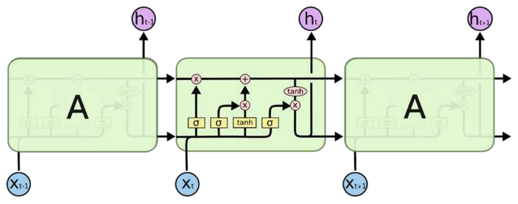


Figure 4. Diagram showing the structure of an LSTM.

4. Experiments and Results

4.1. Experiment Settings

Different settings were experimented with, with the following parameters:

4.1.1 GRU and LSTM

GRU A GRU (Gated Recurrent Unit) is an RNN (recurrent neural network) that targets to solve the vanishing gradient problem which is present in vanilla RNNs. GRU uses update gates and reset gates, which represent short term memory and long term memory respectively. It has a simpler architecture compared to LSTM and only has hidden state.

The GRU takes the input from the time step t and outputs a hidden state which is computed based on the reset gate and the update gate. The update gate enables the model to determine how much of the past information is needed to be remembered. This gate solves the problem of the vanishing gradient as the GRU can decide whether to copy all the past information and eliminate the risk. The reset gate allows the model to decide how much of the past data can be forgotten. With these two gates, the GRU can better capture sequential dependencies with large time step distances and be used in the video captioning pipeline.

LSTM From Fig. 4, each line transmits an entire vector from one node's output to the inputs of other nodes. The pink circles denote pointwise operations, such as vector additions, and the yellow boxes denote learnt neural network layers. Concatenation occurs when lines merge, whereas forking occurs when a line's content is replicated, and the copies are sent to various locations.

The cell state, which is the horizontal line going through the top of Fig. 4, is the key to LSTMs. The state of the cell is similar to that of a conveyor belt. With only a few tiny linear interactions, it flows straight down the entire chain. It is easy for data to travel along it unaltered. The LSTM can add or delete information to the cell state, which is carefully controlled by structures called gates. Gates are a mechanism to selectively allow information to pass through, which are made up of a sigmoid neural net layer plus a pointwise multiplication operation.

4.1.2 Different Methods for Frame Selection

2 methods were used for frame selection, which are "First n Frames", and "Frame Numbers".

First n Frames The first n frames are selected from the GIF as the input for the model. The remaining frames, if n is larger than the number of frames in the GIF, are padded with black images.

Frame Numbers The frame numbers will be selected from the following formula:

$$\left\{ \lfloor \frac{i \times f}{N\_FRAMES} \rfloor | i \in \{0, 1, 2, ..., N\_FRAMES\} \right\}, \quad (8)$$

where f is the number of frames of the GIF and *N_FRAMES* is the number of frames to be selected.

4.1.3 *N_FRAMES*

N_FRAMES refers to the number of frames selected as the input.

4.1.4 Sample Size

The sample size is the number of samples used as for training.

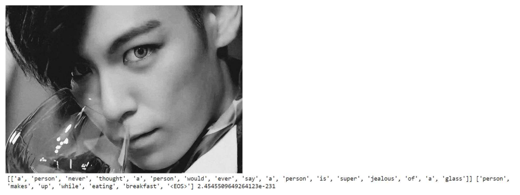
4.2. Experiment Results

The results of the experiments are presented in Tab. 1.

From experiment 1 and experiment 2, we can see that increasing the sample size does not improve the BLEU score. This may indicate that sample size of 2000 is already sufficient for the model to handle the task. By comparing results of experiment 2 and experiment 3, we can see that increasing *N_FRAMES* from 10 to 15 enhances the performance. It can also be seen that the performance of experiment 4 is better than that of experiment 1, 2 and 3. This may indicate that "Frame Number" is better than "First n Frames". Increasing the *N_FRAMES* for experiment 4 further enhances the performance as shown in experiment 5. This is another evidence showing that increasing *N_FRAMES* may provide more information for the model to generate better captions. By comparing results of experiment 2 and experiment 6, it can be seen that there is a slight improvement when GRU is switched to. However, as only one experiment on GRU is carried out and the improvement is not significant, whether GRU is better for captioning generation is still unclear.

	Decoder	Frame Selection	*N_FRAMES*	Sample Size	BLEU
1	LSTM	First n Frames	10	2000	6.65×10^{-81}
2	LSTM	First n Frames	10	4000	3.60×10^{-156}
3	LSTM	First n Frames	15	4000	1.03×10^{-156}
4	LSTM	Frame Number	10	4000	5.70×10^{-81}
5	LSTM	Frame Number	15	4000	1.75×10^{-4}
6	LSTM	First n Frames	10	4000	1.87×10^{-156}

Table 1. Results of the experiments.



```
[['a', 'person', 'never', 'thought', 'a', 'person', 'would', 'ever', 'say', 'a', 'person', 'is', 'super', 'jealous', 'of', 'a', 'glass']] ['person',
'makes', 'up', 'while', 'eating', 'breakfast', '<EOS>'] 2.4545509649264123e-231
```

Figure 5. "a person never thought a person would ever say a person is super jealous of a glass" is the true caption. "person makes up while eating breakfast <EOS>" is the generated caption. The BLEU score is $2.343\,440\,964\,926\,412\,3 \times 10^{-231}$.

5. Future Work

5.1. Attention Mechanism

null

5.2. Filtering Data

When comparing the true caption and the generated caption,it was found that some of the true captions are not well written. As shown in Figs. 6 and 7, some true captions do not correctly describe the GIF and they may be grammatically incorrect as well. Undoubtedly, the performance will be hindered when the model learns from these captions during the training process. Work on filtering the bad quality captions should be done to improve the performance.

6. Conclusion

null

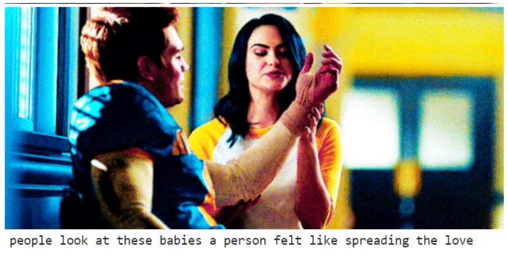

people look at these babies a person felt like spreading the love

Figure 6. One example of bad captions.

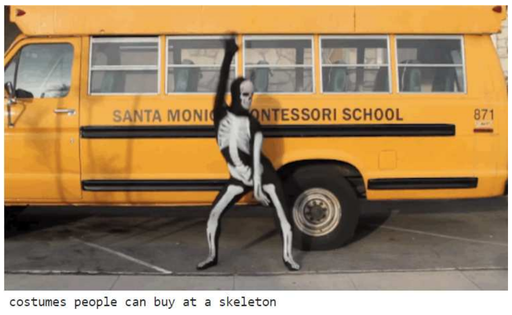

SANTA MONICA MONTESSORI SCHOOL
costumes people can buy at a skeleton

Figure 7. Another example of bad captions.

7. Division of Labour

Yuen Cheuk Heng Marco was responsible for data cleaning and exploration with the reference model. Mak Chak Wing Marco was responsible for literature Review and model architecture design, Cheung Long Sang was responsible for the bulk of the coding, Jiang Wu was responsible for hyperparameter tuning. Finally, Tam Ho Lun Aero, was responsible for typesetting this LATEX document, resolving errors and compiling the document.

References

[1] Roger Grosse. Lecture 15: Exploding and vanishing gradients. *University of Toronto Computer Science*, 2017. 3

[2] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 1, 2

[3] Weikuan Jia, Meili Sun, Jian Lian, and Sujuan Hou. Feature dimensionality reduction: a review. *Complex & Intelligent Systems*, 01 2022. 2

[4] Myeongsuk Pak and Sanghoon Kim. A review of deep learning in image recognition. In *2017 4th international conference on computer applications and information processing technology (CAIPT)*, pages 1–3. IEEE, 2017. 1

[5] Yingwei Pan, Yehao Li, Jianjie Luo, Jun Xu, Ting Yao, and Tao Mei. Auto-captions on gif: A large-scale video-sentence dataset for vision-language pre-training. *arXiv preprint arXiv:2007.02375*, 2020. 3

[6] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 311–318, USA, 2002. Association for Computational Linguistics. 3

[7] Shreyz-max. Video captioning with keras. https://medium.com/analytics-vidhya/video-captioning-with-keras-511984a2cfff, 2021. 1

[8] TensorFlow. Masking and padding with keras: Tensorflow core. https://www.tensorflow.org/guide/keras/masking_and_padding. 3

[9] Ronald J. Williams and David Zipser. A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. *Neural Computation*, 1(2):270–280, 06 1989. 2