

Image Caption Generator in text and audio using Neural Networks.

Sanket Veer^{#1}, Archana Chaudhari^{*2}

[#]Instrumentation and Control Engineering, Vishwakarma Institute of Technology, Pune

Abstract— The job of automatically captioning photos involves the issues of picture analysis and text production. The concept of attention is a crucial part of captioning: how to determine what to describe and in what order. A lot of methods have been practiced on the 2 datasets: Flickr8k and the MSCOCO. This paper presents the algorithm to pre-process the Flickr8k dataset and create a machine learning model that could generate a caption for an input image given by the user. The algorithm is created to convert the generated caption into an MP3 file and play it for better user experience.

Keywords— audio, caption, GTTS, image, keras, LSTM, model, neural networks, ResNet50, tensor flow.

I. INTRODUCTION

Image captioning is an operation that uses deep learning to generate a suitable caption for the given picture. AI has received a good amount of attention since the development of the deep learning algorithms both in image processing and language processing. Image captioning is a hard working task if one performs it manually.

This image captioning can be used in the automatic captioning of the CCTV surveillance system or as a guide for the blind in some applications. They also might prove beneficial for Self Driving Cars or autonomous vehicles. This challenge was unfathomable even to the most advanced researchers in Computer Vision before the recent advent of Deep Neural Networks. However, with the introduction of Deep Learning, if we have the requisite dataset, this problem can be handled quickly.

This paper shows the algorithm and steps performed on a dataset and creation of a machine learning model to perform image captioning upto 91% accuracy. We have performed the experiment on the Flickr8k dataset. We have performed some preprocessing on the data and trained it to create this impressive machine learning model that can generate a caption for any image of the surface of the earth with such great accuracy. Moreover we have taken this text based caption and converted it into an MP3 format for hearing aid applications. We have also saved the files for future scopes of analysis.

II. TOOLS/TECHNOLOGY

Python, Tensorflow, Keras, ResNet50, Neural Networks, GTTS.

III. LITERATURE REVIEW

- Originally, automatic image captioning is only attempted
- to yield simple descriptions for images taken under extremely
- constrained conditions. For example, Kojima et al. [11] used
- concept hierarchies of actions, case structures and verb patterns
- Preprint submitted to Neurocomputing April 13, 2018
- to generate natural languages to describe human activities in
- a fixed office environment.
- Originally, automatic image captioning is only attempted
- to yield simple descriptions for images taken under extremely
- constrained conditions. For example, Kojima et al. [11] used
- concept hierarchies of actions, case structures and verb patterns
- Preprint submitted to Neurocomputing April 13, 2018
- to generate natural languages to describe human activities in
- a fixed office environment.
- Initially, automatic captions are only intended to convey simple descriptions of images taken under extreme
- restricted conditions. For example, Kojima et al. [1] has used action lines, pattern formation and action
- methods to produce natural languages to describe human activities in a dignified environment.

- Hede et al. use a dictionary of objects and language templates to describe images of objects in the background without merging [2]. Such methods are far removed from the requests to describe the images we encounter in our daily lives.
- In [3], inserting a caption for the image Ordonez et al. First use global image descriptions to find a set of images from a web-scale collection of featured images. After that, they use the mantic content of the restored images to make them re-standard and use the caption of the top image as a description of the question.
- Hodosh et al. frame captioning the image as a standard function [4]. Authors use the Kernel Canonical Correlation Analysis [5] [6] to create photographic and textual objects in a common place, where training images and related captions are highly correlated. In the new standard, cosine similarities between pictures and sentences are calculated to select the sentences above to act as descriptions of the image in question.
- Gupta et al. use the Stanford Core NLP toolkit to process sentences in a database to get a list of phrases for each image. In order to determine the meaning of the image in question, image acquisition is first done based on global image features to obtain a set of query images. After that, a model trained to determine the coherence of phrases is used to select phrases from those related to the obtained images. Finally a descriptive sentence is made according to the selected appropriate sentences [7].
- Kiros et al. suggest using a neural language model embedded in an image input to generate image captions [8].
- Li et al. use visual models to make visualization to extract semantic information including objects, attributes and spatial relationships [9]. After that, they described the triplet format $\langle\langle ad\ j1, ob\ j1 \rangle, prep, \langle ad\ j2, ob\ j2 \rangle\rangle$ to get the results of understanding the code.
- Most models rely on a comprehensive, flexible and functional encoder-decoder framework. It is sometimes described as the structure of CNN + RNN. The convolutional neural network (CNN) usually represents the encoder, and the recurrent neural network (RNN) decoder.
- Most functions are tested on Flickr30k and MSCOCO datasets. Both of these data sets are rich in number of images and each image has five shared captions making it ideal for training and testing models.

IV. METHODOLOGY

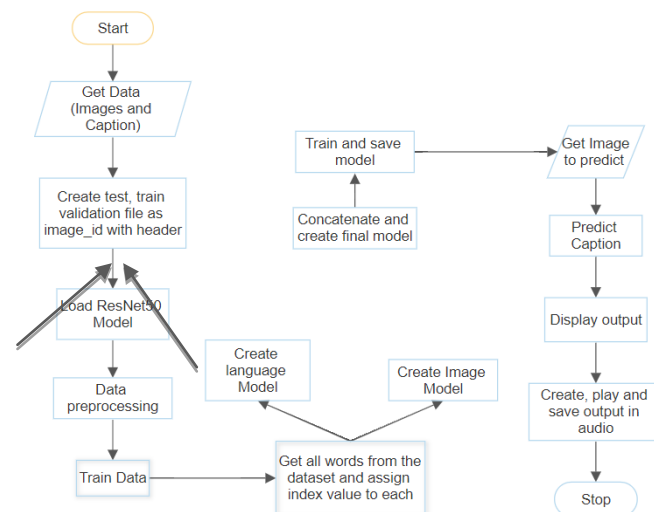


Fig. 1

□ Training

1. Get Data:

We have taken the Flickr 8k set for the experiment. This dataset consists of around 8k images and corresponding captions for each image have been declared beforehand.

2. Create Test, Train files with header name as images_id and captions and populate the data into it by giving caption to each image.

3. Load the ResNet50 Model:

Loading 50 layer Residual Network Model for training purpose.

4. Data Pre-processing:

Give target size to image as (224,224,3) and expand the shape of array for axis = 0

5. Train Data and extract features using ResNet50.
6. Get all unique words from the set of captions in a list and assign key/index value to each using vectorization.
7. Create Image model:
Create a model for images with 2 layers which includes a Dense layer and RepeatVector layer.
8. Create Language model:
A 3 layered language model for captions which includes layers like LSTM, TimeDistributed.
9. Concatenating both models adds a few more layers for the final models.
10. Train the model for 195 epochs and save it and its weights.

❑ Prediction

1. Get the input Image.
2. Pre-process and required image shape and get encoded with ResNet50.
3. Predict the caption using defined function and the model saved. And get the output string printed.
4. Generate the audio file, save and play the file.

❑ About Tools/Resources

● Flickr 8k

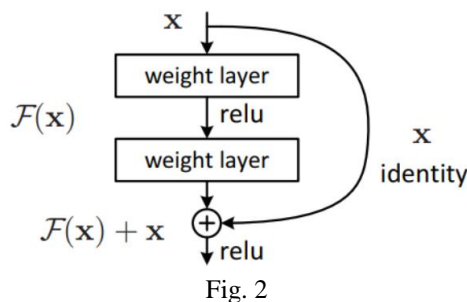
It is a small size freely available dataset where there are 8092 images in jpeg format. Five captions are provided for each image with the 0 to 4. Cleaning and pre-processing the data gives a better idea of the dataset and is helpful for creating models. 6000 images are used for training the model, 1000 for testing and 1000 for development.

● ResNet50

Residual Networks is an architecture proposed by researchers at Microsoft Research.

This architecture introduces the concept of the Residual Network to overcome the problem of the vanishing/exploding gradient. We employ a method called skip connections in this network. The skip connection bypasses a few stages of training and links directly to the output.

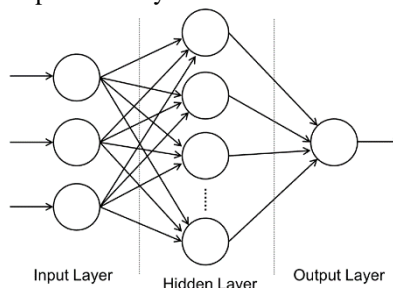
Instead of allowing layers to learn the underlying mapping, we let the network fit the residual mapping. Instead of using $H(x)$ as the initial mapping, use $F(x) := H(x) - x$, which gets $H(x) := F(x) + x$.



The benefit of including this type of skip connection is that any layer that degrades architecture performance will be bypassed by regularisation.

● Dense Layer

The dense layer is a deep-connected neural network layer, i.e. each neuron in the present in the dense layer receives input from all neurons from the previous layer.



Syntax:

```
keras.layers.Dense(units, activation=None, use_bias=True, kernel_initializer='glorot_uniform',  
bias_initializer='zeros', kernel_regularizer=None, bias_regularizer=None, activity_regularizer=None,  
kernel_constraint=None, bias_constraint=None)
```

- RepeatVector

RepeatVector is used to iterate over the input for a specified number of times, n. If you apply RepeatVector with parameter 16 to a layer with the input shape (batch size, 32), the layer's output shape will be (batch size, 16, 32). There is just one argument for RepeatVector.

- LSTM

In the realm of deep learning, LSTM is an artificial recurrent neural network (RNN) architecture. LSTM has feedback connections, unlike normal feedforward neural networks. It can process not only single data points (such as photos), but also complete data sequences (such as speech or video).

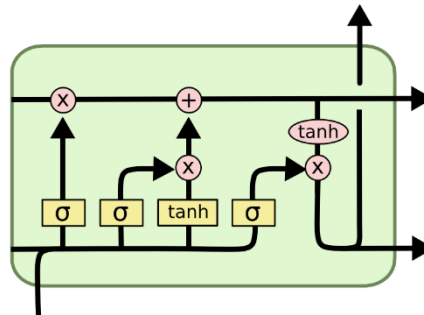


Fig. 4

LSTMs can be applied at the positions of the non-segmented, which is related to recognition, speech-to-text, and the network traffic anomaly detection, or the ids of the s (in the intrusion detection system).

- GTTS

Google text to speech (gTTs) is a python library which is used to interface with API provided by Google for translation. It generates audio files in mp3 format.

- Activation Function

Activation functions like “relu” and “softmax” were used in the neural network model.

Softmax function is generally used at last part of the model. It normalize the input into probability distribution consisting ‘k’ probabilities.

The standard softmax function is given by formula,

$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}} \quad \text{for } i = 1, \dots, K \text{ and } \mathbf{z} = (z_1, \dots, z_K) \in \mathbb{R}^K.$$

V. RESULTS

The model is trained efficiently to predict the caption of the input image upto 91% accuracy. Model contains

Total params: 7,490,110

Trainable params: 7,490,110

Non-trainable params: 0

The caption is also successfully converted into the MP3 audio format and is saved in the directory of the code. It can also be saved into other locations by giving the particular path.

```
model.summary()
```

Layer (type)	Output Shape	Param #	Connected to
embedding_input (InputLayer)	{(None, 40)}	0	
dense_input (InputLayer)	{(None, 2048)}	0	
embedding (Embedding)	(None, 40, 128)	1056512	embedding_input[0][0]
dense (Dense)	(None, 128)	262272	dense_input[0][0]
lstm (LSTM)	(None, 40, 256)	394240	embedding[0][0]
repeat_vector (RepeatVector)	(None, 40, 128)	0	dense[0][0]
time_distributed (TimeDistribut	(None, 40, 128)	32896	lstm[0][0]
concatenate (Concatenate)	(None, 40, 256)	0	repeat_vector[0][0] time_distributed[0][0]
lstm_1 (LSTM)	(None, 40, 128)	197120	concatenate[0][0]
lstm_2 (LSTM)	(None, 512)	1312768	lstm_1[0][0]
dense_2 (Dense)	(None, 8254)	4234302	lstm_2[0][0]
activation (Activation)	(None, 8254)	0	dense_2[0][0]

Total params: 7,490,110
Trainable params: 7,490,110
Non-trainable params: 0

Fig. 5 Model Summary

```
50/50 [-----] - 5s 100ms/step - loss: 0.2397 - accuracy: 0.9089
Epoch 286/280
50/50 [-----] - 5s 100ms/step - loss: 0.2323 - accuracy: 0.9119
Epoch 287/280
50/50 [-----] - 5s 100ms/step - loss: 0.2383 - accuracy: 0.9064
Epoch 288/280
50/50 [-----] - 5s 100ms/step - loss: 0.2357 - accuracy: 0.9101
Epoch 289/280
50/50 [-----] - 5s 100ms/step - loss: 0.2426 - accuracy: 0.9088
Epoch 290/280
50/50 [-----] - 5s 100ms/step - loss: 0.2292 - accuracy: 0.9097
Epoch 291/280
50/50 [-----] - 5s 100ms/step - loss: 0.2394 - accuracy: 0.9079
Epoch 292/280
50/50 [-----] - 5s 100ms/step - loss: 0.2288 - accuracy: 0.9095
Epoch 293/280
50/50 [-----] - 5s 100ms/step - loss: 0.2317 - accuracy: 0.9095
Epoch 294/280
50/50 [-----] - 5s 100ms/step - loss: 0.2336 - accuracy: 0.9072
Epoch 295/280
50/50 [-----] - 5s 100ms/step - loss: 0.2371 - accuracy: 0.9063
Epoch 296/280
50/50 [-----] - 5s 100ms/step - loss: 0.2307 - accuracy: 0.9090
Epoch 297/280
50/50 [-----] - 5s 100ms/step - loss: 0.2262 - accuracy: 0.9114
Epoch 298/280
50/50 [-----] - 5s 100ms/step - loss: 0.2335 - accuracy: 0.9095
Epoch 299/280
50/50 [-----] - 5s 100ms/step - loss: 0.2228 - accuracy: 0.9132
Epoch 300/280
50/50 [-----] - 5s 100ms/step - loss: 0.2322 - accuracy: 0.9088
```

Fig. 6 Training Result.

```
z = Image(filename=img)
display(z)
print(Argmax_Search)
```




Two kids laying on a bed upside down

Fig. 2 Output(a)

```
[65] Argmax_Search = predict_captions(test_img)

[66] z = Image(filename=img)
display(z)
print(Argmax_Search)
```



A man in the top under snowy peak .

Fig. 3 Output(b)

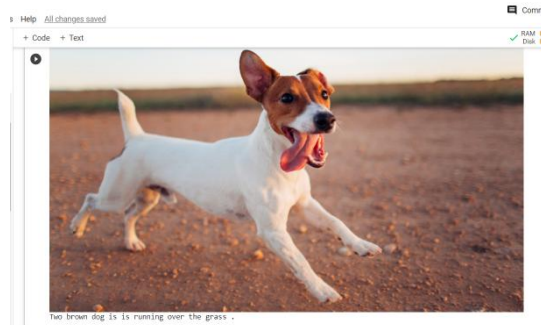


Fig. 4 Output(c)

VI. CONCLUSIONS

The dataset Flickr8k is a pretty convenient dataset being small and workable on laptop/desktop and also which doesn't require a lot of data cleaning and pre-processing. This makes the algorithm faster for further operations. Python makes the coding an easy task for such kind of algorithms where one can think about creativity and less about the errors and syntax of the implementation. Encoding with ResNet architecture and the weights of the model worked well to get above 90 accuracy. The caption was generated using a defined function and the text was successfully converted to audio file in mp3 format which is saved.

VII. FUTURESCOPE

The model can be rebuilt to improve the accuracy. Also it can be implemented in Android or IOS application to be useful for blind people. Output can be translated as per required language. Can be used in defence system with required modifications to generate data for analysis for a mission.

ACKNOWLEDGMENT

I would like to express my sincere gratitude to my guide Prof. Archana Chaudhari for guiding and mentoring me for this project. Thanks to college B.R.A.C.T's Vishwakarma Institute of Technology for introducing such curriculum activities related to project development for all students.

REFERENCES

- [1]. A. Kojima, T. Tamura, K. Fukunaga, Natural language description of human activities from video images based on concept hierarchy of actions, International Journal of Computer Vision 50 (2002) 171–184.
- [2]. P. Hede, P. Moellic, J. Bourgeois, M. Joint, C. Thomas, Automatic generation of natural language descriptions for images, in: Proc. Recherche D'information Assistee Par Ordinateur, 2004.
- [3]. V. Ordonez, G. Kulkarni, T. L. Berg., Im2text: Describing images using 1 million captioned photographs, in: Advances in Neural Information Processing Systems, 2011, pp. 1143–1151.
- [4]. M. Hodosh, P. Young, J. Hockenmaier, Framing image description as a ranking task: data, models and evaluation metrics, Journal of Artificial Intelligence Research 47 (2013) 853–899.
- [5]. F. R. Bach, M. I. Jordan, Kernel independent component analysis, Journal of Machine Learning Research 3 (2002) 1–48.
- [6]. D. R. Hardoon, S. R. Szedmak, J. R. Shawe-Taylor, Canonical correlation analysis: An overview with application to learning methods, Neural Computation 16 (2004) 2639–2664.
- [7]. A. Gupta, Y. Verma, C. V. Jawahar., Choosing linguistics over vision to describe images, in: AAAI Conference on Artificial Intelligence, Vol. 5, 2012.
- [8]. R. Kiros, R. Zemel, R. Salakhutdinov, Multimodal neural language models, in: International Conference on Machine Learning, 2014.
- [9]. S. Li, G. Kulkarni, T. L. Berg, A. C. Berg, Y. Choi., Composing simple image descriptions using web-scale n-grams, in: Proceedings of the Fifteenth Conference on Computational Natural Language Learning, 2011.
- [10]. https://openaccess.thecvf.com/content/ACCV2020/papers/He_Image_Captioning_through_Image_Transformer_ACCV_2020_paper.pdf
- [11]. <https://www.geeksforgeeks.org/residual-networks-resnet-deep-learning/>
- [12]. https://www.researchgate.net/publication/333154356_A_Systematic_Literature_Review_on_Image_Captioning
- [13]. <https://machinelearningknowledge.ai/keras-dense-layer-explained-for-beginners/>
- [14]. https://en.wikipedia.org/wiki/Long_short-term_memory
- [15]. <https://pypi.org/project/gTTS/>
- [16]. https://en.wikipedia.org/wiki/Softmax_function