Volume 9, Issue 4
August 2020

ITEE Journal
Information Technology & Electrical Engineering

ISSN: - 2306-708X

Deep Learning Model Implementation in Web-Based Application for Automatic Image Caption Generator

[1]Ruchitesh Malukani, [2]Nihaal Subhash and [3] Chhaya Zala
Department of Computer Engineering, G H Patel College of Engineering & Technology, Vallabh Vidyanagar, Gujarat, India
Email: [1]ruchiteshmalukani@gmail.com, [2]nihaal.subhash@gmail.com, [3]chhaya20491@gmail.com

ABSTRACT

Image captioning is a big challenge for virtual helpers, editing tools, photo indexing and handicapped assistance. Important progress has been made in recent times in the area of image captioning with the help of deep learning. Deep learning can be applied to real-world datasets which covers frequently faced problems. Image captioning is a crucial job involving linguistic image understanding and the ability to generate an interpretation of sentences with proper and accurate structure. It needs expertise in image processing and natural language processing. This paper describes our experience for creating a web-based deep learning application that can be used to generate automatic image caption from the given input image followed by text to speech translation of the generated caption.

Keywords- Image captioning, Deep Learning, Computer Vision, Natural language processing, web-based application, CNN, RNN.

I. INTRODUCTION

Today, images are ubiquitously present in every aspect of our lives – in magazines, social media, news, books, advertisements, etc. Humans can intuitively understand images without captions, but machines cannot. Image captioning has a wide range of applications. It is used in image indexing, on social media websites, etc.

Image captioning is distinctively separate from the task of image classification and is more complex. While image captioning only focuses on object identification, captioning also involves identifying the properties of the identified objects (like location), the overall context of the image, and the relationship amongst the different objects. Further, it also involves generating properly structured sentences that are both syntactically and semantically correct.

Traditional machine learning approaches usually fail to meet this challenge and therefore techniques like CNNs [1], RNNs [2], Deep Learning, Adversarial Learning, Reinforcement Learning, etc. are used instead. These algorithms can handle the complexities associated with this task well. There have been many different approaches employed for this task, this paper describes an interactive web-based application based on an encoder-decoder model [3, 4].Circumstances where a variable length of input sequence is needed to be mapped to the variable length of the output sequence, the encoder-decoder model is used. Hence for our work, Convolutional Neural Network (CNN) has been used as an encoder to extract features from the images and Long-short-term-memory (LSTM [5]) has been used to encode the text descriptions. After encoding, both the inputs are combined and then given to a simple model for the decoding purpose to get the next word in the sequence. This merged model [4] incorporates both types of encoded inputs whose combination will be decoded to predict the next word. Figure 1 represents general architecture of our project.

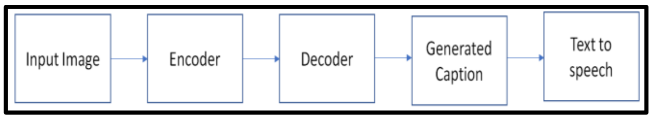


Figure 1. Proposed Model Architecture

II. RELATED WORK

2.1 Image classification

Image classification is a computer vision method that can classify an image by its observable content. For instance, an image classification algorithm can be used to determine whether or not picture includes a human being. Many traditional methods for image classification were designed before but all the previous image classification records were broken in the ImageNet Challenge 2012 using a deep CNN which shows an application of deep learning in the image classification area [6, 7]. A single-label multi-class image classification [8] is concerned with the role of classifying an image with one form of artwork selected from a selection of artwork styles. Besides, deep learning has a major effect on many visual challenges, such as face recognition, image segmentation, object identification, and character recognition.

2.2 Image captioning

Like Image classification, there are various traditional as well as deep-learning-based approaches for image captioning. We can observe good results by using deep learning for the caption generation. We can use various deep-learning-based approaches for getting annotations for images such as encoder-decoder method, visual attention mechanisms, novel object-based image captioning and semantic attention mechanisms. As Vinyals et al. have mentioned in their paper [3] that CNNs can create a rich image interpretation by encoding it in a vector of a fixed length, which further can be used for many visual applications. Hence, one can first train CNN (or can pick pre-trained CNN) for image classification task and then its

ITEE Journal
Information Technology & Electrical Engineering

ISSN: - 2306-708X

last hidden layer can be used as an input to the LSTM (RNN) decoder that generates the sentences. For our work, we use RNN (LSTM) purely for the encoding process of the text. In the attention mechanism, certain factors in data processing are given more attention. Papers [9, 11,12] discuss attention-based aspects used for image captioning with visual or semantic attention mechanisms and paper [10] uses attention mechanisms for machine translation. Papers [13, 14] describe novel object-based image captioning approach. Papers [15-17] use other deep learning based approaches.

In this paper, we showcase an application of the encoder-decoder model trained on the MSCOCO [18] dataset. Here the strategy is to build an effective system that provides a comprehensive user interface to annotate the images. This interface has specifically been designed in such a way that users can easily supply images and the web-based system will return systematic and reliable captions for the given images which reduces the human intervention for such tasks.

Paper	Evaluation Metric			
	Bleu*		Meteor	
[3]	0.277		0.237	
[9]	0.243[a]	0.25[b]	0.239[a]	0.230[b]
[11]	0.375		0.285	
[12]	0.316		0.250	
[13]	-		0.210	
[15]	0.304		0.251	
[16]	0.371[c]	0.281[d]	0.383[c]	0.286[d]
[17]	0.244		-	

[a] Soft Attention
[b] Hard Attention,
[c] Cross Entropy Loss,
[d] Cider Score Optimization
**Blue-4 Scores*

Table 1. Related Work

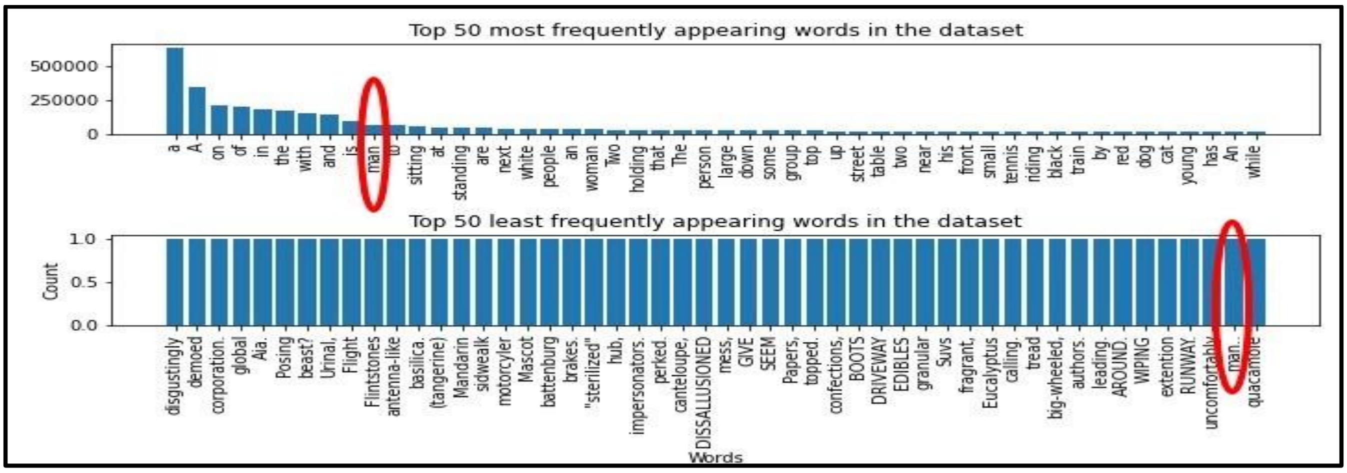


Figure 2. Need of caption preprocessing-MS COCO train2017.json

III. RESEARCH METHODOLOGY

Similar to many other web-based applications our application consists of Angular framework-based frontend and Flask framework-based backend. This section describes and explains the fundamental understanding of our overall work. Our work operates sequentially in several phases described below.

A. Exploring the datasets

For visual challenges like image captioning, there are many open-source datasets available like Flickr8k [19], Flickr30k [20], MS COCO, etc. Flickr8k is relatively small and practical to download and models are easily created on workstations with CPU. It has more than 8,000 pictures and up to 5 image annotations. Flickr30k contains more than 30,000 images and up to 5 image annotations. MS COCO is a comprehensive data collection for object identification, segmentation and captioning. It contains approximately more than 160,000 images. For our application model, we have utilized images and annotations from the MS COCO dataset[1]. Such a large dataset provides more insights about the environment.

We started exploring the dataset annotations by spitting each caption into words, grouping them into a data-frame[2] with the frequency of each word. Then we sorted the data-frame in the descending order of frequency to find frequency wise top k words from the dataset, where k can be any desired integer value. By doing this, we found a problem described in the next sub-section, which led us to the conclusion that pre-processing of the annotations is a must. Next, we affirmed that we need to preprocess the images according to the inception network. This process also has been described in the next subsection.

B. Data Preprocessing

[1]MS COCO dataset is downloaded from http://cocodataset.org/#download
[2]Here, by data-frame we mean an instance of DataFrame class of open-source data analysis tool named pandas

Before providing the data as an input to the mode we preprocessed the data in two preprocessing steps.

Caption preprocessing: As mentioned in the previous subsection the data-frame was built on the annotations which were not preprocessed. We found the following results.

Here as shown in the following Figure 2 word 'man' is appearing in both the top 50 most and least appearing words, which is not correct. The reason is we split the annotations by white space characters but we did not eliminate the punctuations. Hence 'man..' and 'man' are considered as two different words in this case. So here the preprocessing of the annotation before frequency count is necessary. For text preprocessing we did following text cleaning steps:

- Punctuation cleaning
- Removal of a single character
- Eliminating numeric characters
- Converted the captions in lowercase
- Finding and sorting all unique words
- Converting word to numbers
- Adding start sequence and end sequence symbols

Because machine learning operates only on numbers, the next stage of caption preprocessing can be seen as translating unique words into numbers. For example, consider a sentence "she has an apple and an orange" it can be translated in numeric vector by replacing unique words by unique numbers; for instance, an=1, and=2, apple=3, has=4, orange=5, she=6 then the numeric form of the above sentence is [6,4,1,3,2,1,5]. Tokenizer has been created to perform this task.

Image preprocessing: Before fitting the data in the model, the image needs to be converted into a three-dimensional vector (height, width, color channel). In order to match the model's architecture, the image is resized into 299X299 pixels. Then three-color channels RGB (Red, Green and Blue) are added in the last outcome.

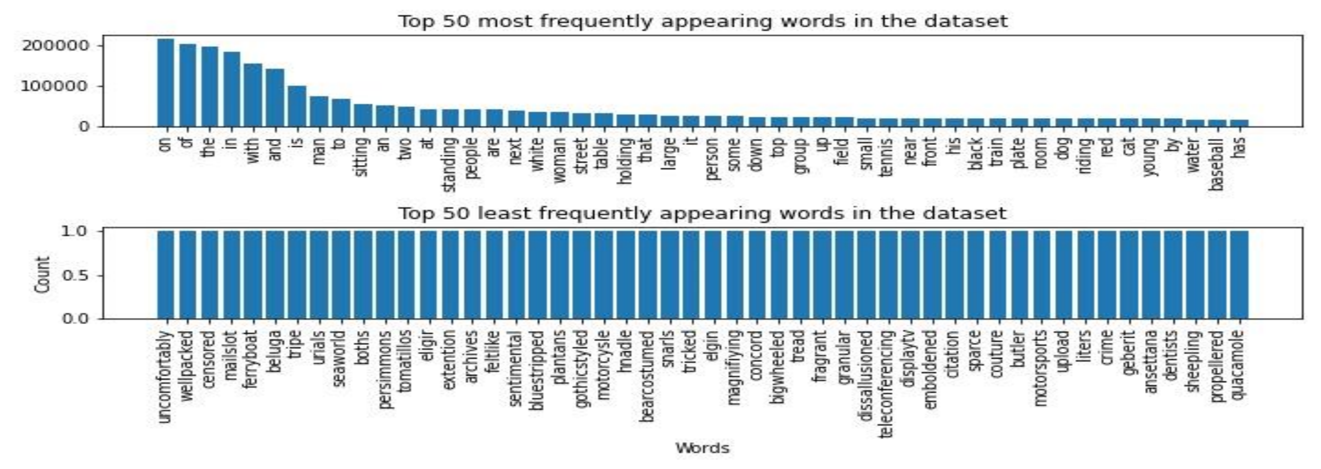


Figure 3. Need of caption preprocessing-MS COCO train2017.json

The next step is to scale the pixel values into a range of [-1, 1] by subtracting mean RGB channels of the ImageNet dataset. This preprocessed image will be given to the Inception model to extract features from the image.

C. Data preparation using data generators

It is a challenging task to fit the dataset in the main memory when it is too large. We faced the same challenge during our work. This subsection describes why we have used data generators to train our model.

The training process utilizes feature vector of the input image, and partial caption generated so far to predict the next word in the expected caption which demands little more memory than any single data point would have required as a whole without remembering the partial caption. We have around 82,000 training images, and each image contains five captions. So, it can be seen as 4,10,000 total image-caption elements. Let us assume we have captions of on an average seven words long. So, we may believe in the worst case, we will have 28,70,000 data points.

Additionally, each word will be mapped to a vector of fixed length (for our work, we have taken word embedding dimension as 200) during the word embedding process described in the word embedding subsection. If we consider a data matrix of size n x m consisting of the image feature vector of length 2048 and partial captions of some maximum length (for our case maximum length is 49) where n is the number of data points and m is the length of each datapoint[3]. For our case value of n will be 28,70,000 and the value of m will be 2048+(200*49) =11,848 and n x m will be 34,003,760,000 blocks. Now even if we assume that each block will take 2 bytes then it

[3]We have used the Inception network to extract features from the image which returns a flatten vector of length 2048. This has been explained in the image embedding subsection.

will require approximately 68 GB of main memory. Such an immense amount is challenging to handle.

We resolved this issue by using Keras API's fit_generator function. The fit_generator is a python generator. fit_generator has an assumption of having a data generator that generates data for it. We have used train data generator and validation data generator in our implementation code associated with this paper[4].

D. Image embedding

CNN can be used as an efficient feature extractor [21]. The low-level features such as edges, shapes, arcs, etc. are extracted by the initial layers of the feature extractor. The transformation of low-level features to high-level features will be performed by the subsequent layers of the network. The output of CNN will be given to another type of FC network. In the case of the InceptionV3 [22], this FC with output size 1000 is designed to solve classification challenges. This will not be useful for our problem. We need a flatten feature vector of the input image having a length of 2048, which is supplied to the last layer of InceptionV3. Hence, we eliminate the last layer of InceptionV3 to use features extracted by this feature extractor model to generate captions.

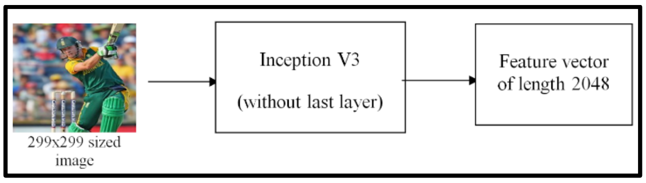


Figure 4 Feature Extraction

E. Word embedding

As we described in the pre-processing subsection, we have transformed words into integers. In this subsection, we describe how we map word index to a word vector. Similar to image embedding the words are also mapped to some fixed length vector before providing them as input to the LSTM layer. This task can be achieved by two steps: 1) One-Hot encoding 2) Embedding using FC.

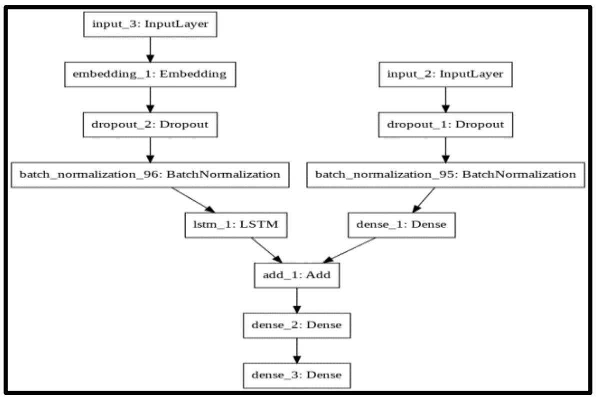


Figure 5 Model architecture

[4]The source code can be found at https://github.com/Rpmalukani22/Automatic-Image-Caption-Generator

One-Hot encoding is a simple method of vectorization but it also leads to the problems faced by tasks in high-dimensional space because it creates a new dimension for each category. Now we can use FC to convert one-hot encoded vectors into fixed dimensional vectors. For word embedding, we have used pre-trained GLOVE [23] model.

F. Model creation

In this subsection, we shall discuss the architecture of our model. For our image captioning model, we have used three types of neural networks. 1) Fully connected neural networks (FC), CNN and RNN. CNN is typically used to obtain spatial invariance in image data. RNN is typically used to handle sequential data such as linguistic concepts. We have used Inception v3 and LSTM as our CNN and RNN respectively.
Our model is mostly based on Vinyals et al. with a few differences [3]:

- We use Inception v3 instead of Inception v1 for CNN.
- We use LSTM to encode the text descriptions. We realize the encoder-decoder architecture using merged model [4].
- We have used different values for some hyperparameters.

The main objective of our model can be defined as "Given an image I from a set of images Di, respective correct image annotation S from given a set of correct annotations Ds for respective images and initial model parameters y find the best suitable values of y such that it maximizes the probability of having correct caption". More mathematically the objective function is

$$y^* = \underset{y}{\operatorname{argmax}} \sum_{(I,S)} \log\left(p(S \mid I; y)\right)$$

In the above function, the probability term can be seen as a joint probability over its words $S_0,\ldots,S_L$.

$$\log(p(S \mid I)) = \sum_{i}^{L} \log\left(p_i(S_i \mid I, S_0, \ldots, S_{i-1})\right)$$

Where L is the length of the caption S. We can define relationship between Si and S as follows:

$$S = \text{concatenate}(S0, S1, \ldots, SN)$$

That is, we can concatenate S_i to get S. Here we have used LSTM for joint probability distribution. Figure 5 represents architecture of our model.

Input layer named input_3 takes sequence of indices of partial caption. Its output is given to the embedding layer where every index gets mapped to a 200-dimensional vector. Dropout layers are used to avoid overfitting. Input layer named input_2 takes a feature vector of length 2048. We have also used batch normalization with the concern of training speed. The next dense layer named dense_1 takes 2048 sized vector and gives 256 sized output. Now both layers lstm_1 layer and dense_1 layer having the same sized output the outputs can be added. This added output is given as an input to the next hidden dense layer named dense_2. The last dense_3 layer is the output layer which generates probability distribution across all words in the vocabulary.

G. Model Training, Validation and Testing

As described in subsection C we have fitted training data and validation data generators to our model. We have used MS COCO2014 train and validation images for training and cross-validation respectively. We have used 'adam' as parameter optimizer and 'cross_entropy' as a loss function**.** Figure 11 in the next section shows results after 50 epochs for training the model. The qualitative and quantitative outcomes of our work have been described in the next section.

H. Web interface development and model deployment

a) Frontend development using Angular framework

The frontend of our web application has been created using Angular framework which is an open-source framework led by Angular team at Google for creating frontends of the web applications. Angular is very efficient and feature-rich for creating single-page applications. Major tools and technologies involved in application frontend development include HTML5, CSS3, JavaScript/TypeScript, Bootstrap3, JQuery and NPM package manager. We have used Web Speech API to convert the predicted caption text to speech.

b) Backend development using flask framework

The backend of this web application has been developed in python using Flask framework. Backend of our application is responsible for accepting, processing and responding to the valid user requests that are sent using application's frontend.

IV. RESULTS AND DISCUSSION

Extensive experiments were performed to evaluate the presented model. The COCO dataset has been used for training and validation of the model. BLEU [24] and METEOR [25] scores have been used as evaluation metrics.

A. Dataset

For the sake of clear understanding, we described the cleaning step using train2017.json annotation file

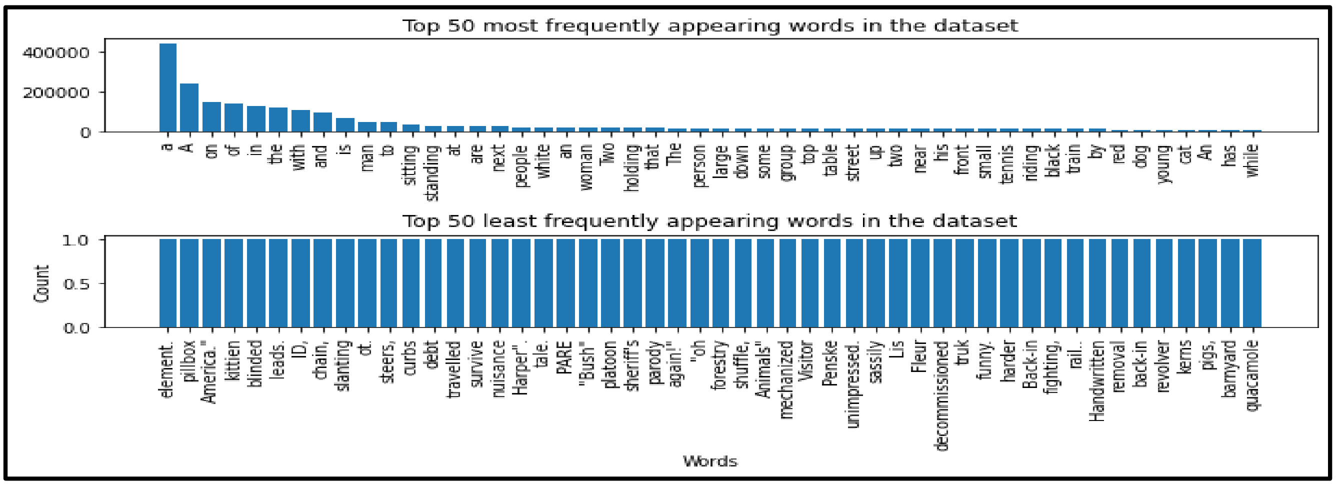


Figure 6. Before cleaning annotations from MS COCO train2014.json

belonging to the 2017 version of the MS COCO dataset in figures 2 and 3. But due to resource constraints, we have used the 2014 version of the COCO dataset for our application. The same annotation cleaning steps explained in the previous section are followed for the training and validation annotations of this version. After the cleaning process on the same annotation, statistics are shown in figures 6 and 7.

B. Model training and validation

We trained our image captioning model with a learning rate of value 0.0001, 160 pictures per batch and a total of 50 epochs. We finished this training on Google's Colaboratory notebook which provided virtual hardware with up to 25 GB RAM and 68.40 GB of secondary storage and GPU/TPU runtime type support. For each epoch, training and cross-validation has been performed and training and validation losses have been recorded. Figure 11 shows 'Loss vs. Epochs' plot after 50 epochs. It can be seen that after 15-20 epochs loss is decreasing comparatively slowly. After 50 epochs training loss was 2.77 and validation loss was 2.94.

C. Quantitative assessment

By taking reference from paper [26] in this subsection we quantitatively evaluate generated captions by taking actual captions for validation given in the dataset as reference captions. We use BLEU and METEOR scores as our model evaluation metrics. We calculate individual bleu score representing n-grams as well as cumulative bleu scores BLEU-1, BLEU-2, BLEU-3 and BLEU-4. The table below shows the quantitative assessment results of our work. Figure 8 represents four graphs showing individual BLEU scores (for n-grams) vs. the number of epochs. It can be observed that the individual score for n-grams is increasing with the betterment of the model. Figure 9 represents four graphs showing cumulative BLEU scores (BLEU-1, BLEU-2, BLEU-3, BLEU-4) vs. epochs. We can observe that the cumulative BLEU score also increases with the increment of the number of epochs. The main reason behind that is the increasing behavior of the individual BLEU scores with the number of epochs. Figure 10 represents a graph showing METEOR scores vs. the number of epochs. This value also increases as the model trains and improves.

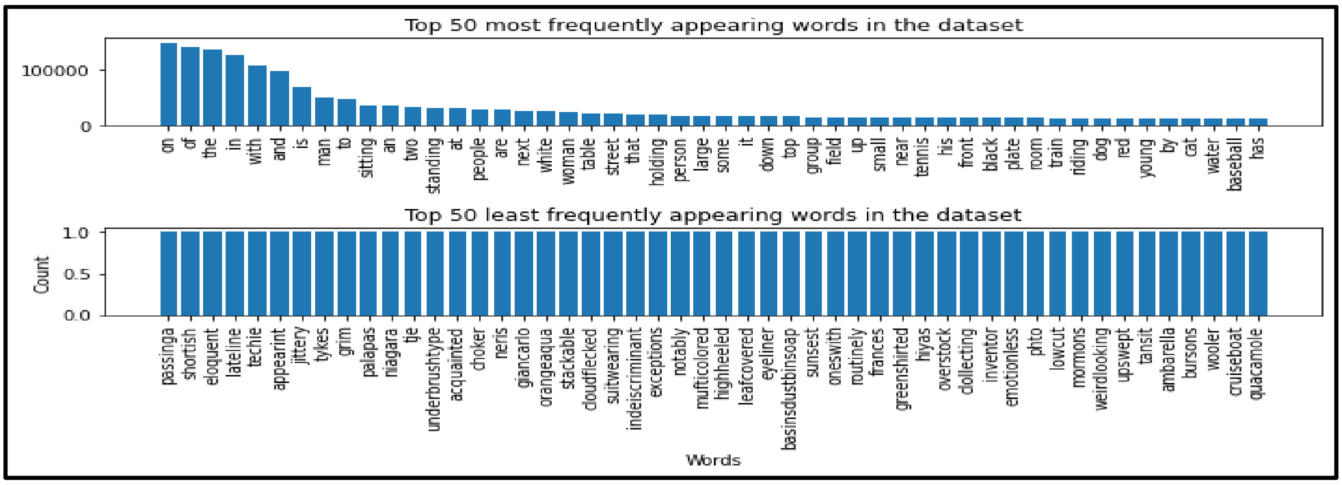


Figure 7. After cleaning annotations from MS COCO train2014.json

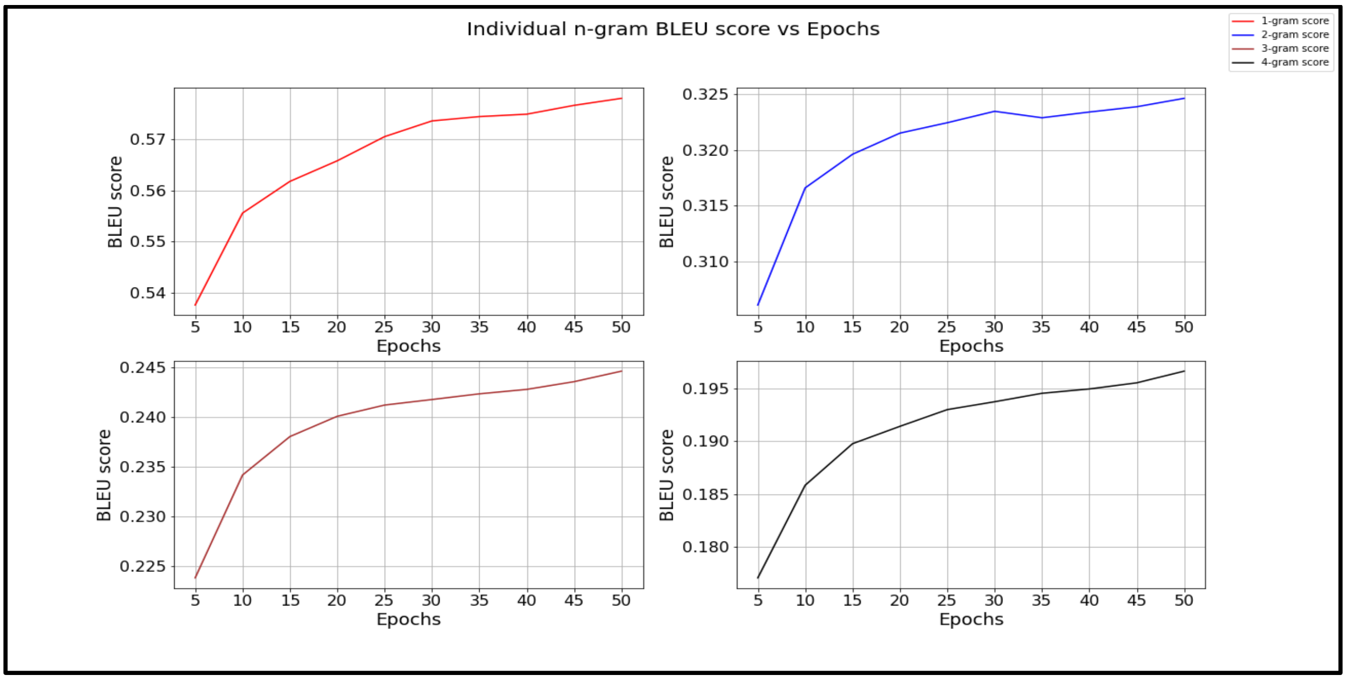


Figure 8. Individual BLEU vs. Epochs

Epochs	1-gram	2-gram	3-gram	4-gram	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR
5	0.538	0.306	0.224	0.177	0.538	0.395	0.322	0.275	0.329
10	0.556	0.317	0.234	0.186	0.556	0.409	0.334	0.286	0.348
15	0.562	0.320	0.238	0.190	0.562	0.413	0.338	0.290	0.355

20	0.566	0.322	0.240	0.191	0.566	0.416	0.341	0.293	0.360
25	0.570	0.322	0.241	0.193	0.570	0.418	0.343	0.294	0.365
30	0.573	0.323	0.242	0.194	0.573	0.420	0.344	0.295	0.368
35	0.574	0.323	0.242	0.195	0.574	0.420	0.344	0.296	0.370
40	0.575	0.323	0.243	0.195	0.575	0.420	0.345	0.296	0.371
45	0.577	0.324	0.244	0.196	0.577	0.421	0.345	0.297	0.372
50	0.578	0.325	0.245	0.197	0.578	0.422	0.346	0.298	0.374

Table 2. Results

D. Qualitative assessment

As we can see below, the captions produced by our model can be categorized in various categories ranging from "Highly relevant" to "Less related". Testing was performed on images from the MS COCO test dataset as well as our local environment. In both types of images, our model predicts considerably good captions. Some of the results are reported in figures 12, 13 and 14. Here figure 12 presents results falling under the "Highly relevant" category. Figures 13 and 14 represent captions predicted by our model with some errors. In figure 13 it can be seen that captions are very relevant but not exact. In figure 14 it can be seen that captions are comparatively more erroneous.For some such erroneous cases, we observed that our model is able to detect some objects correctly but

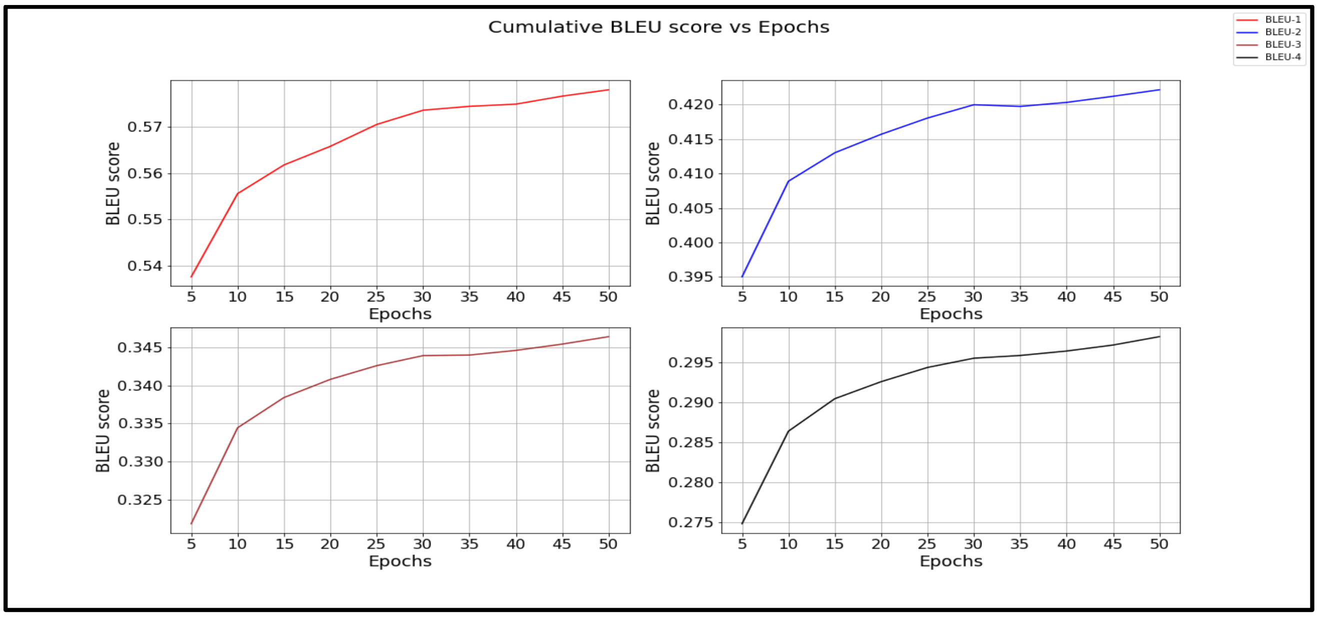


Figure 8. Cumulative BLEU vs. Epochs

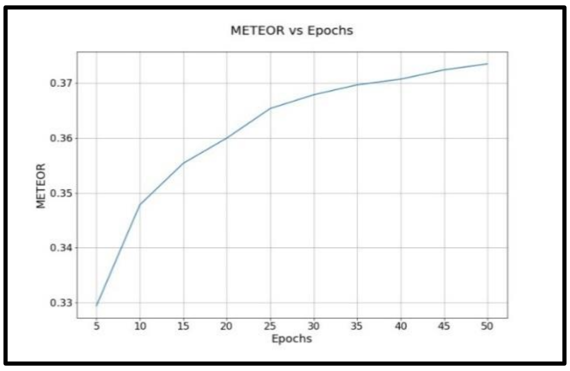


Figure 9. Meteor Score vs. Epochs

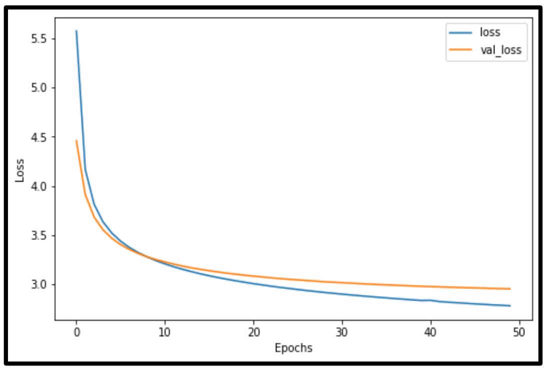


Figure 10. Loss vs. Epochs

Volume 9, Issue 4
August 2020

ITEE Journal
Information Technology & Electrical Engineering

ISSN: - 2306-708X

formed captions are inaccurate. This opens doors to future work in order to improve model accuracy.

E. Web-Application results

In this subsection, we represent results given by our web-application. This application involves an overview section that gives an overview of "What is image captioning". "Captionify" section leads the user to the upload page where the user can upload the image and can listen to the caption that has automatically been spoken aloud by the application. This page also contains a text to speech section that has separate text to speech component where pasted text can be converted into the speech. The described working can be observed in figures 15 and 16.

V. CONCLUSION AND FUTURE WORK

We have presented a web-application to generate annotations for the given image. To develop a web-based application Angular framework has been used in frontend and backend related activities were performed using flask framework. Our application not only generates the captions for the image but also converts them into a speech. This "image to speech" conversion can be highly useful for many applications. To generate fairly accurate captions, we have used InceptionV3 to extract image features and LSTM to handle linguistic information.

In future, we are planning to improve the accuracy of our model by introducing the attention mechanism for

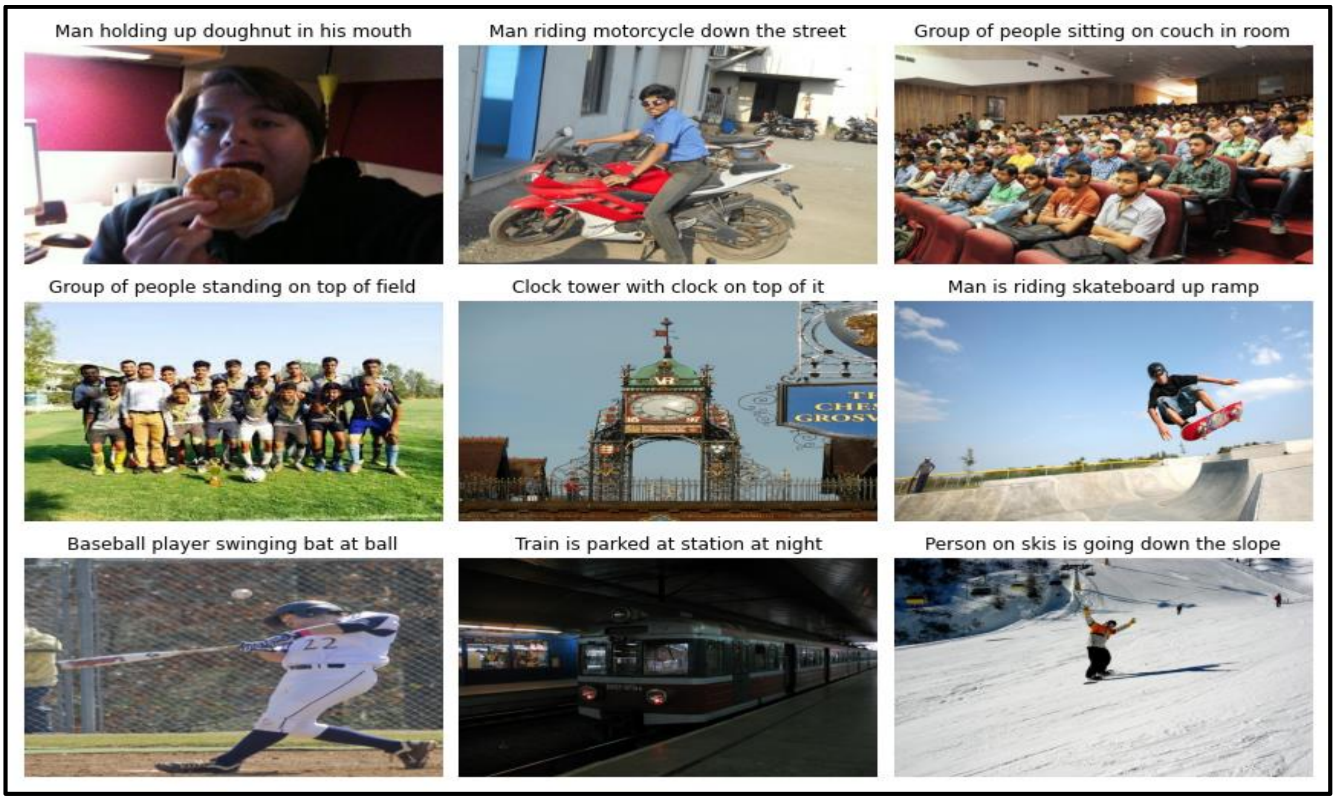

Man holding up doughnut in his mouth
Man riding motorcycle down the street
Group of people sitting on couch in room
Group of people standing on top of field
Clock tower with clock on top of it
Man is riding skateboard up ramp
Baseball player swinging bat at ball
Train is parked at station at night
Person on skis is going down the slope

Figure 11. Predicted captions without errors

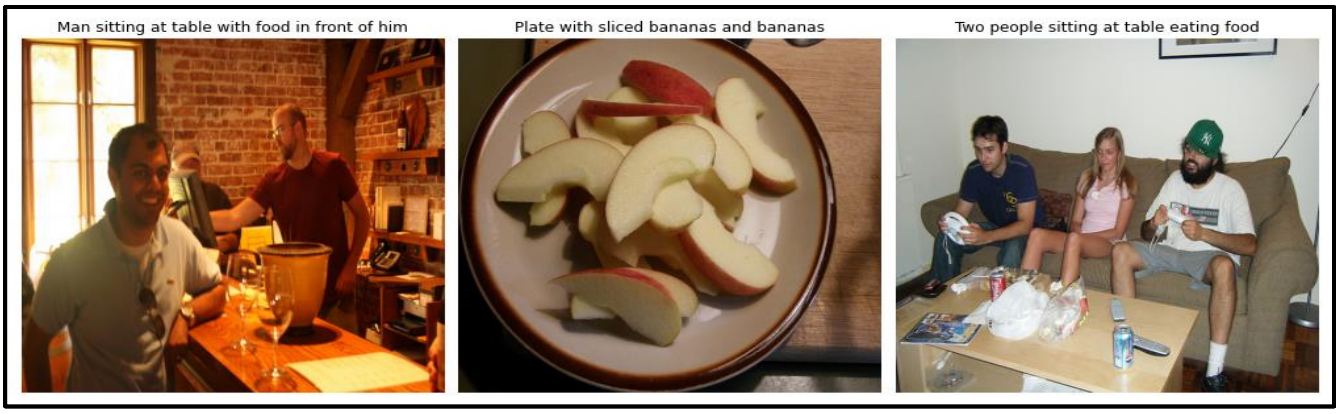

Man sitting at table with food in front of him
Plate with sliced bananas and bananas
Two people sitting at table eating food

Figure 12. Predicted captions with fewer errors

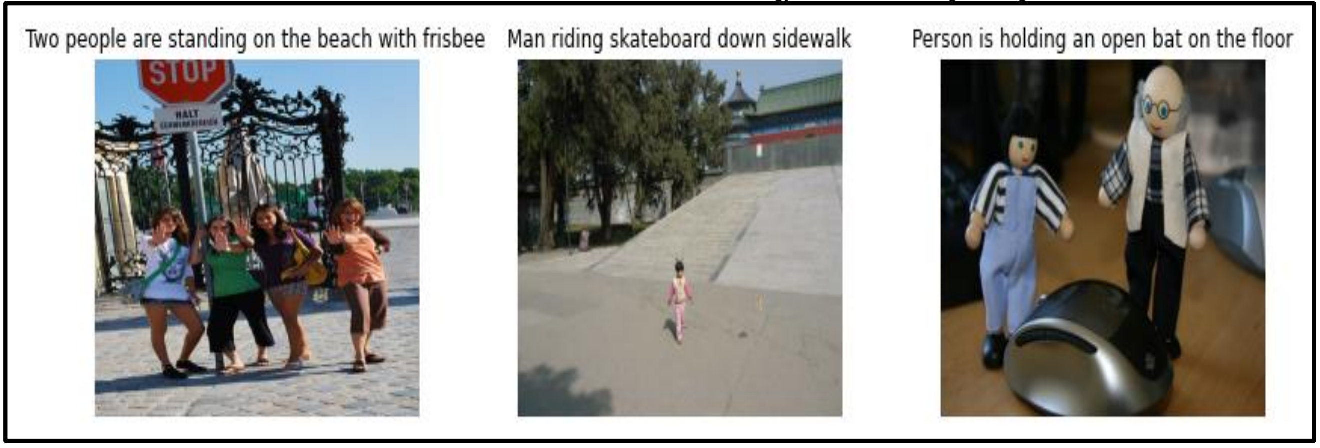
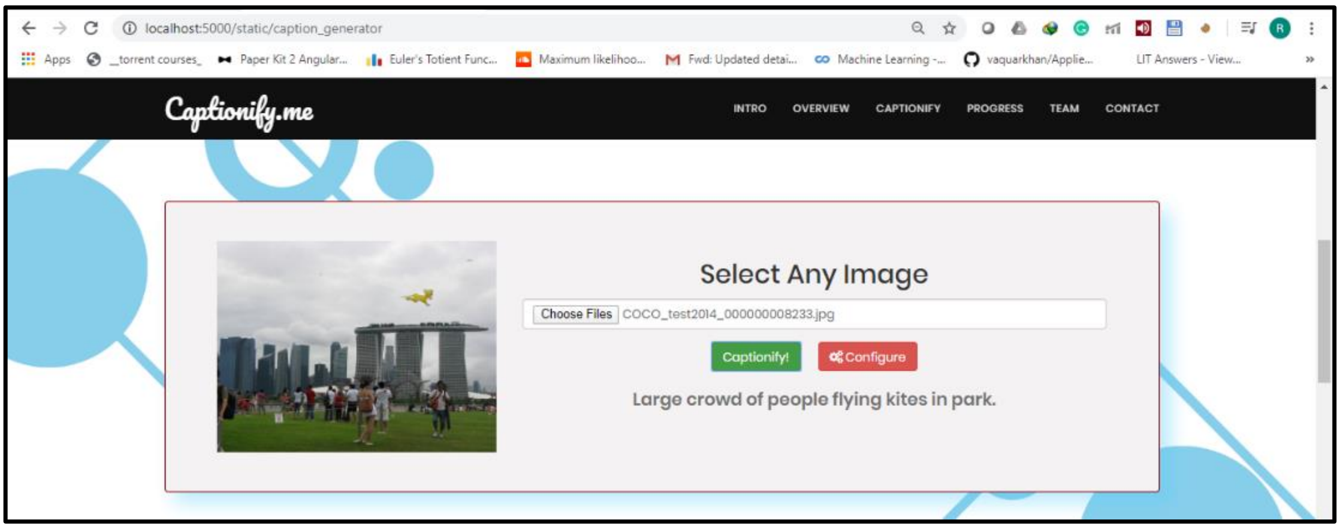

Two people are standing on the beach with frisbee
Man riding skateboard down sidewalk
Person is holding an open bat on the floor

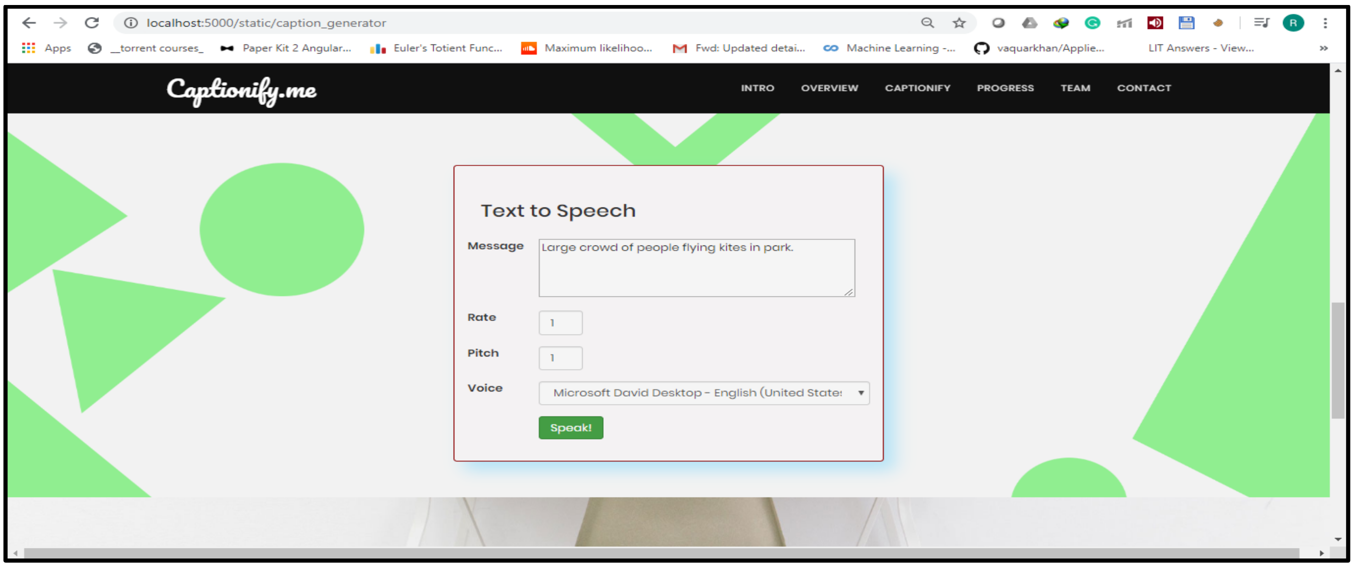
Figure 13. Captions less related to the image

visual as well as language aspects. We are also planning to enhance our application by adding security and privacy features. Response time of the application can be improved by using lightweight deep learning models so that the same model can be used with resource-constrained and mobile devices.



Large crowd of people flying kites in park.

Figure 14. Upload component



Message: Large crowd of people flying kites in park.
Rate: 1
Pitch: 1
Voice: Microsoft David Desktop - English (United States)

Figure 15. Text to speech component

REFERENCES

[1] Wu, Jianxin. "Introduction to convolutional neural networks." National Key Lab for Novel Software Technology. Nanjing University. China 5 (2017): 23.

[2] Boden, Mikael. "A guide to recurrent neural networks and backpropagation." the Dallas project (2002).

[3] Vinyals, Oriol, et al. "Show and tell: A neural image caption generator." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015

[4] Tanti, Marc, Albert Gatt, and Kenneth P. Camilleri. "Where to put the image in an image caption generator." Natural Language Engineering 24.3 (2018): 467-489.

[5] Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." Neural computation 9.8 (1997): 1735-1780.

[6] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems. 2012.

[7] Russakovsky, Olga, et al. "Imagenet large scale visual recognition challenge." International journal of computer vision 115.3 (2015): 211-252.

[8] Qi Dong, Xiatian Zhu, and Shaogang Gong. 2019. Single-label multi-class image classification by deep logistic regression. CoRR abs/1811.08400 (2019).

[9] Xu, Kelvin, et al. "Show, attend and tell: Neural image caption generation with visual attention." International conference on machine learning. 2015.

[10] Luong, Minh-Thang, Hieu Pham, and Christopher D. Manning. "Effective approaches to attention-based neural machine translation." arXiv preprint arXiv:1508.04025(2015).

[11] Gao, Lianli, et al. "Hierarchical LSTMs with adaptive attention for visual captioning." IEEE transactions on pattern analysis and machine intelligence (2019).

[12] You, Quanzeng, et al. "Image captioning with semantic attention." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[13] Anne Hendricks, Lisa, et al. "Deep compositional captioning: Describing novel object categories without paired training data." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[14] Yao, Ting, et al. "Incorporating copying mechanism in image captioning for learning novel objects." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017.

[15] Ren, Zhou, et al. "Deep reinforcement learning-based image captioning with embedding reward." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

[16] Yao, Ting, et al. "Exploring visual relationship for image captioning." Proceedings of the European conference on computer vision (ECCV). 2018.

[17] Wang, Cheng, et al. "Image captioning with deep bidirectional LSTMs." Proceedings of the 24th ACM international conference on Multimedia. 2016.

[18] Lin, Tsung-Yi, et al. "Microsoft coco: Common objects in context." European conference on computer vision. Springer, Cham, 2014.

[19] Rashtchian, Cyrus, et al. "Collecting image annotations using Amazon's Mechanical Turk." Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk. Association for Computational Linguistics, 2010.

[20] Young, Peter, et al. "From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions." Transactions of the Association for Computational Linguistics 2 (2014): 67-78.

[21] Sermanet, Pierre, et al. "Overfeat: Integrated recognition, localization and detection using convolutional networks." arXiv preprint arXiv:1312.6229 (2013).

[22] Szegedy, Christian, et al. "Rethinking the inception architecture for computer vision." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[23] Makarenkov, Victor & Shapira, Bracha & Rokach, Lior. (2016). Language Models with Pre-Trained (GloVe) Word Embeddings. arXiv:1610.03759.

[24] Papineni, Kishore, et al. "BLEU: a method for automatic evaluation of machine translation." Proceedings of the 40th annual meeting on association for computational linguistics. Association for Computational Linguistics, 2002.

[25] Lavie, Alon, and Abhaya Agarwal. "METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments." Proceedings of the second workshop on statistical machine translation. 2007.

[26] Chen, Xinlei, et al. "Microsoft coco captions: Data collection and evaluation server." arXiv preprint arXiv:1504.00325 (2015).

AUTHOR PROFILES

Ruchitesh Malukani is currently pursuing his Bachelor of Engineering in Computer Engineering at G.H. Patel College of Engineering & Technology, Vallabh Vidayanagar, Gujarat, India.

Nihaal Subhash is currently pursuing his Bachelor of Engineering in Computer Engineering at G.H. Patel College of Engineering & Technology, Vallabh Vidayanagar, Gujarat, India.

Chhaya Zala is working as an Assistant Professor in the Department of Computer Engineering at G.H. Patel College of Engineering & Technology, Gujarat, India. She received her M.Tech (Information Technology) from Dharamsinh Desai University, Gujarat, India, where she was the gold medalist. She completed her BE in Computer Engineering from GTU. Her research interests include Data Mining, Machine Learning and Pattern Recognition.