Image Description Generation based on Deep Learning

Miaomiao Zeng1, Xun MuRong2*
1 Education college, Zhaoqing University, Zhaoqing, Guang dong, 526061, China
2 Education college, Zhaoqing University, Zhaoqing, Guang dong, 526061, China
*Corresponding author’s e-mail: 2008020009@zqu.edu.cn

Abstract. The paper firstly explains the importance and research significance of image description generation and makes a brief research summary on the generation of various algorithms for image description; it further introduces the principle and the algorithm process of the LSTM deep learning algorithm in details, specifically describes establishing steps and the model parameter selection process of the LSTM deep learning algorithm of the Microsoft COCO dataset, implements the deep learning models by employing Python, and finally applies the trained LSTM deep learning model to image description generation by using the Microsoft COCO dataset as an instance.

1. Introduction
With the rapid development of Internet, the amount of information and complexity of media such as pictures and videos on the Internet have also rapidly increased, which means that it becomes more and more difficult for users to find the information they need in the vast information. If users want to easily acquire required media materials, they need to describe the media in details. However, due to the explosive growth of videos or pictures, the work amount required to manually add corresponding description tags is too huge and has become impossible. Therefore, the computer is required to help the completion of the work. Automatic description generation from natural images is a challenging problem that has recently received a large amount of interest from the computer vision and natural language processing communities. Figure 1 is a schematic diagram of image description generation.

![Image Description Generation](image.png)

Figure 1. Schematic Diagram of Image Description Generation.

Deep learning is a field of machine learning that is based on learning several layers of representations, typically by using artificial neural networks. Through the layer hierarchy of a deep learning model, the higher-level concepts are defined from the lower-level concepts [1].

Since Hinton et al. (2006) [2] introduced an efficient way of training deep models and Bengio(2009) [3] showed the capabilities of deep architectures in complicated artificial intelligence tasks, deep learning has become an emerging topic in computer science. Currently, deep learning approaches produce the
state-of-the-art solutions to many problems in computer vision, natural language processing, and speech recognition. But it is mostly developed only in a single function in a simple scene. In recent years, researchers have begun to change their research directions to complex cross scenarios. For instance, in the field of image description generation in this paper, the model is required to accurately recognize objects in the picture and describe the relationship between objects using the natural language, which is a cross research direction combining computer vision and natural language processing [4].

2. Review of Image Description Generation Model

2.1. Template-based algorithm
According to the template-based algorithm, local features of the image, including the types and attributes of the objects recognized in the image, are firstly extracted. Then the extracted types and attributes of the objects are put into a pre-written image description template, and certain rules are written to select the optimal description content therefrom. The algorithm is advantageous in that the main content of the image can be reflected, but the generated natural language description is over-rigid, with unvaried contents and lacking in detail attributes.

2.2. Search-based algorithm
According to the search-based algorithm, the initial step is similar to that of the template-based algorithm. Similarly, local features of the image are extracted, including the types and attributes of objects recognized in the image. Then the extracted features are combined into a feature library, the similarity is measured by employing the similarity algorithm, the image with the highest feature similarity is selected, the image description thereof is merged, and a sentence of description is finally output. The disadvantage is that the algorithm depends too much on the feature library, with low generalization ability.

2.3. Encoder-decoder-based algorithm
The encoder-decoder-based method uses the Recurrent Neural Network (RNN)(Yasrab, 2018) [5]. According to the method, firstly image data is encoded into eigenvectors with a fixed length through the Convolutional Neural Network (CNN), wherein the CNN acts as the encoder; then these fixed-length eigenvectors are transferred into the RNN as input, and the information transferred by the CNN is decoded and converted into the natural language, wherein the RNN acts as the encoder. This method features strong generalization ability, varied sentences and high performance. This method is used in this paper.

3. Introduction to LSTM Deep Learning Algorithm
LSTM deep learning algorithm, proposed by Jürgen Schmidhuber and Sepp Hochreiter in 1997., is an improved RNN algorithm which remedies the defect that the RNN is unable to capture the long-term memory. Thus, the algorithm is tremendously suitable for classification and prediction tasks based on time series data.

3.1. Basic principle of RNN
A recurrent neural network (RNN) is a class of artificial neural networks that make use of sequential information[6]. An RNN is specialized to process a sequence of values \( x(0), x(1), ..., x(t) \). The same task is performed on every element of a sequence, while the output depends on the previous computations. In other words, RNNs have internal memory that captures information about previous calculations.

The input, output, and A neural network that allows information to be transferred from a certain step to the next in the network. An RNN can be seen as multiple times of copy and paste of the same simple network. The information is continuously transferred in the network, so it can be considered that the
information has a certain degree of persistence. For instance, can predict, but if \( i \) and \( o \) are far from each other, cannot act on, therefore, the RNN needs to be improved.

### 3.2. LSTM - improved RNN

Despite the fact that RNNs are designed to deal with long-term dependencies, vanilla RNNs tend to suffer from vanishing or exploding gradient. When backpropagation trains the network through time, the gradient is passed back through many time steps, and it tends to vanish or explode. The popular solutions to this problem are Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) architectures. LSTM, having a similar structure to that of RNN, is made to solve the problem of long-term memory loss of the RNN. However, LSTM mainly differs from the RNN in the part of the intermediate neural network \( A \). Firstly, the following symbol descriptions are given:

| Symbols | Description |
|---------|-------------|
| \( i \) | Input gate |
| \( f \) | Forget gate |
| \( o \) | Output gate |
| \( x_t \) | Input quantity |
| \( h_t \) | Output quantity/Hide Status |
| \( C_t \) | Cell state |
| \( \sigma(\cdot) \) | sigmoid function |
| \( \tanh(\cdot) \) | tanh() function |
| \( W \) | Neural network weight |
| \( b \) | Neural network bias |
| \( \vec{c}_t \) | Candidate value vector |

LSTM not only has the hide layer \( h_t \) of the RNN, but also has another hidden layer cell state, namely \( C_t \). The core idea of the LSTM is to control cell state, namely the retention or discarding of \( C_t \) through three gates referring to the input gate \( i \), the forget gate \( f \), and the output gate \( o \). Each of the three gates is composed of a sigmoid neural layer and a dot product unit. The gate outputs 1 or 0 through the sigmoid function, wherein 1 represents completely retaining information, and 0 represents completely discarding information.

**Step 1:** The information firstly flows to the forget gate \( f \) and updates its output:

\[
f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right)
\]

**Step 2:** The step is composed of two parts, firstly the input gate \( i \) determines which values need to be updated to \( C_t \), and secondly the tanh function generates a new candidate value vector \( \vec{c}_t \), and whether the value is input to \( C_t \) depends on the situation. Two parts of the output of the input gate are updated:

\[
i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] + b_i\right)
\]
Step 3: $[h_{t-1}, x_t]$ selected through the forget gate $f$ is multiplied by $C_{t-1}$, $[h_{t-1}, x_t]$ selected by the input gate $i$ is multiplied by the candidate value vector $\tilde{C}_i$, the two products are multiplied to form the current state $C_t$, and the cell state is updated:

$$C_t = f_t \times C_{t-1} + i_t \times \tilde{C}_i$$  \hspace{1cm} (3)

Step 4: Finally, determine which information needs to be retained, $[h_{t-1}, x_t]$ passes through the output gate $o$ and is then multiplied by $C_t$, and finally the output $h_t$ is formed. The output of the output gate is updated:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$  \hspace{1cm} (4)

$$h_t = o_t \times \tanh(C_t)$$  \hspace{1cm} (5)

Figure 2. Schematic Diagram of the LSTM Neural Network.

4. Establishment of Image Description Generation Model

The image description generation model in this paper is based on the encoder-decoder method by using the RNN [7]. According to the method, firstly image data is encoded into eigenvectors with fixed length through the CNN, wherein the CNN acts as the encoder. Then the fixed-length eigenvectors are transferred into the RNN as input, and the information transferred by the CNN is decoded and converted into the natural language, wherein the RNN acts as the encoder.

The objective function of the model is to maximize the probability of correct description, wherein is the hyperparameter, $I$ is the image eigenvector, $S$ is the correct description, and the objective function is as follows:

$$\theta^* = \arg \max_{\theta} \sum_{I,S} \log P(S | I; \theta)$$  \hspace{1cm} (6)

During training, $(S, I)$ is the input training data pair, and the paper adopts the Adam method to optimize and adjust parameters during maximization of the objective function. The objective of the LSTM neural network is to train it so that when the feature data of the image is input, the sentence corresponding to the image, such as $P(S_t | I, S_0, S_1, ..., S_{t-1})$, can be predicted. Considering the LSTM in an expanded form is beneficial to achieve this, therefore, multiple LSTM memory units are copied so that each word of the sentence is received by the corresponding LSTM, and all the LSTMs share the same parameters and the output $m_t - 1$.

$$x_{t-1} = \text{CNN}(I)$$  \hspace{1cm} (7)

$$x_t = W_s S_t, t \in \{0, ..., N - 1\}$$  \hspace{1cm} (8)
\[ p_{t+1} = LSTM(x_t), t \in \{0..N-1\} \]  

(9)

Figure 3. Schematic Diagram of Image Description Generation Model

5. Training and Experiments

Footnotes should be avoided whenever possible. If required they should be used only for brief notes that do not fit conveniently into the text.

5.1. About Microsoft COCO Dataset

The Microsoft COCO dataset, with the full name of Common Objects in Context, is a dataset for image description generation provided by Microsoft Research AI. The dataset is divided into training images, verification images, and test images. The paper adopts the 2017 version, which includes 72,783 training images, 38,504 verification images, and 36,775 test images.

Table 2. Partial Data of Microsoft COCO Dataset.

| Picture name                             | Description of content                                      |
|------------------------------------------|-------------------------------------------------------------|
| 1012212859_01547e3f17.jpg                | White dog with brown ears standing near water with head turned to one side. |
| 1015118661_980735411b.jpg                | A boy smiles in front of a stony wall in a city.            |
| 1015118661_980735411b.jpg                | A little boy is standing on the street while a man in overalls is working on a stone wall. |
| 1015118661_980735411b.jpg                | A young boy runs across the street.                         |
| 1015118661_980735411b.jpg                | A young child is walking on a stone paved street with a metal pole and a man behind him. |
| 1015118661_980735411b.jpg                | Smiling boy in white shirt and blue jeans in front of rock wall with man in overalls behind him. |
| 1015584366_dfcecc3c85a.jpg               | 0 A black dog leaps over a log.                             |
| 1015584366_dfcecc3c85a.jpg               | A grey dog is leaping over a fallen tree.                   |
5.1.1. Data cleaning.

- Extraction of Microsoft COCO Dataset. COCO Dump is downloaded from http://cocodataset.org. The content includes the name of the image file, the context relationship between targets, the location coordinates of the targets, etc. Also, the file is a json file and most tag contents are not needed, so the data extraction tool provided by Python is adopted. The relevant content from the dataset can be easily extracted with the tool, and only the file name and the description are retained.

- Conversion of uppercase and lowercase letters. The presence of uppercase and lowercase in the original dataset can lead to the condition that "hello" and "Hello" are regarded as two words just because they have different first letters, which is inconvenient for subsequent data conversion. In the case, all letters should be converted to lowercase by the adoption of the built-in uppercase to lowercase script of the open-source OpenCC converter.

Table 3. Parameter List of Conversion of Uppercase and Lowercase Letters.

| Parameter name | Parameters | Parameter interpretation |
|----------------|------------|--------------------------|
| -i             | -i coco_dump | File path                |
| -o             | -o coco_dump_new | Output document          |
| -c             | -c c2s.json | Converted configuration file |

- Removal of punctuation marks. There is still a lot of useless information in the simplified file, such as numbers, punctuation marks, non-Chinese language characters, ('%', '$', '#') etc. The punctuation marks in all corpora are matched through regular expressions, and replaced by null characters.

- Filtering of stop words. A preprocessing method in text analysis is stop word filtering. The function is to filter out some functional words, namely, noise. The functional words are extremely common and have no practical meaning compared with other word. In this way, the storage space can be saved, and the search efficiency can be improved. The specific method of filtering out stop words includes loading functions of the new user-defined dictionary, iterating over the stop word list and performing filtering. The specific implementation is shown in the codes.

- Removal of low-frequency words. The description of the image is one sentence, and there are many words in the sentence. If a certain word rarely appears in the descriptions of all images, the word should be regarded as an outlier and removed from the training data. The model in the paper is to be used for prediction so that the probability of model errors can be reduced.

Table 4. List of Some High-frequency Words and their Frequencies.

| Words | Frequency | Words | Frequency |
|-------|-----------|-------|-----------|
| 'in'  | 83224     | 'an'  | 11119     |
| 'the' | 57402     | 'red' | 9857      |
| 'on'  | 45538     | 'sitting' | 9608     |
| 'and' | 44253     | 'girl' | 9342      |
| 'is'  | 41108     | 'boy'  | 9135      |
| 'man' | 40277     | 'standing' | 9103     |
There are 8,763 non-repeated words in 72,783 training images. It can be seen that words such as 'in', 'the', and 'on' appear frequently, while words such as 'rockstar', 'foolish', and 'stereoscope' scarcely appear, so we can remove these words as outliers. If the words which appear less than 10 times are removed, we can get 1,651 non-repeated words.

- Thesaurus. For convenient passing to the LSTM network, each of the 1,651 non-repeated words after removing the low-frequency words is replaced by an integer, that is, 0-1650 represents 1,651 words.

### 5.1.2. Image data preprocessing

The field of deep learning depends heavily on data driving. For CNN models such as Inception V3, only a company like Google has enough hash rate to complete the training of such a complex model. The Inception V3 model is obtained by training on the Imagenet dataset, hyperparameters have been persistent, and the parameters can be well applied to other datasets through simple adjustment. The method is called transfer learning.

The Inception V3 model is obtained through training of the Imagenet dataset and established with the target of image classification, so the last network of the final Inception V3 model is a 1,000-dimensional softmax bottleneck layer which can classify images into 1,000 categories. However, the objective of this paper is not image classification, so the last layer of structure can be removed, and only the input layer to the layer before the output layer is retained. The preprocessing steps are as follows:

```python
# Obtaining the InceptionV3 model trained by employing Imagenet
model = InceptionV3(weights='imagenet')
```
# Removing the final layer of the InceptionV3 model, namely the softmax layer
model_new = Model(model.input, model.layers[-2].output)
# Converting all the images to the expected size 299*299
img = image.load_img(image_path, target_size=(299, 299))
# Converting the PIL image to the three-dimensional numpy array
x = image.img_to_array(img)
# Add another dimension
x = np.expand_dims(x, axis=0)
# Preprocessing the image by employing preprocess_input() from the inception module
x = preprocess_input(x)
# Converting the vector from the shape (1,2048) to (2048,)
x = np.reshape(x, x.shape[1])

Figure 4. Schematic Diagram of Image Data Pre-processing.

5.1.3. **Image and description data merging and preprocessing**
The section mainly introduces how to merge images and description data, input the merged data into the model in a simple way, and finally train the model through the data.

The above two images are two training images, the first one is described as 'The black cat sat on grass', and the second one is described as 'The white cat is walking on road'. The images can be easily segmented into one word through the split function of Python and then put into the vocab list, wherein vocab = {black, cat, endseq, grass, is, on, road, sat, startseq, the, walking, white}, so the non-repeated corpus formed by descriptions of the two images can be obtained. The descriptions of the above pictures are put into the non-repeated word lexicon for retrieval, and the result is shown in Table 6.

| Words | Lexicon position | Words | Lexicon position |
|-------|-----------------|-------|-----------------|
| black | 1               | cat   | 2               |
| grass | 3               | is    | 4               |
| on    | 5               | road  | 6               |
| sat   | 7               | startseq | 8       |
| the   | 9               | walking | 10         |
| white | 11              |       |                 |

Table 6. Training image 1.

Figure 5. Training Image 1.  Figure 6. Training Image 2.
In order to use the data in the neural network, the problem of image description generation should be regarded as a supervised learning problem to solve, \( x^i \) refers to the eigenvector and partial description of the image. The objective of the model is to generate description, so the generated natural language description is the objective function. \( y_j \) is the objective function, namely, the next word.

Table 7. Before the word is converted to the vector.

| i | Image eigenvector | Partial description | \( y_j \) |
|---|-------------------|---------------------|----------|
| 1 | Image 1           | startseq            | the      |
| 2 | Image 1           | startseq the        | black    |
| 3 | Image 1           | startseq the black  | cat      |
| 4 | Image 1           | startseq the black cat | sat    |
| 5 | Image 1           | startseq the black cat sat | on |
| 6 | Image 1           | startseq the black cat sat on | grass |
| 7 | Image 1           | startseq the black cat sat on grass | endseq |
| 8 | Image 1           | startseq the black cat sat on grass endseq |   |
| 9 | Image 2           | startseq            | the      |
| 10| Image 2           | startseq the        | white    |
| 11| Image 2           | startseq the white  | cat      |
| 12| Image 2           | startseq the white cat | is    |
| 13| Image 2           | startseq the white cat is | walking |
| 14| Image 2           | startseq the white cat is walking | on |
| 15| Image 2           | startseq the white cat is walking on | road |
| 16| Image 2           | startseq the white cat is walking on road |   |
| 17| Image 2           | startseq the white cat is walking on road endseq | endseq |

Table 8. After the word is converted to the vector.

| i | Image eigenvector | Partial description | \( y_j \) |
|---|-------------------|---------------------|----------|
| 1 | Image 1           | [9,0,0,…,0]         | 10       |
| 2 | Image 1           | [9,10,0,…,0]        | 1        |
| 3 | Image 1           | [9,10,1,…,0]        | 2        |
| 4 | Image 1           | [9,10,1,2,…,0]      | 8        |
| 5 | Image 1           | [9,10,1,2,8,…,0]    | 6        |
| 6 | Image 1           | [9,10,1,2,8,6,…,0]  | 4        |
| 7 | Image 1           | [9,10,1,2,8,6,4,…,0]| 3        |
| 8 | Image 1           |                     |          |
| 9 | Image 2           | [9,0,0,…,0]         | 10       |
| 10| Image 2           | [9,10,0,…,0]        | 12       |
| 11| Image 2           | [9,10,12,…,0]       | 2        |
| 12| Image 2           | [9,10,12,2,…,0]     | 5        |
| 13| Image 2           | [9,10,12,2,5,…,0]   | 11       |
| 14| Image 2           | [9,10,12,2,5,11,…,0]| 6        |
| 15| Image 2           | [9,10,12,2,5,11,6,…,0]| 7    |
| 16| Image 2           | [9,10,12,2,5,11,6,7,…,0]| 3    |
| 17| Image 2           |                     |          |
Above is the simplest case, namely, there are only 2 images for combination. In the real test, there are 6,000 training images. Each image has 5 descriptions, which is equivalent to 30,000 images, and the descriptions have an average of 8 words, which is equivalent to merging of 210,000 data into the vector.

5.2. Process and results of image description generation based on LSTM algorithm

5.2.1. Image prediction description result

Five images are randomly crawled from the Internet and put into the image description generation model for prediction, and the following results are generated. The images are not real training images, so there is no accurate manual annotation. Here examples and results of the image description generation model algorithm are provided, and the generalization of the algorithm is displayed.

| Test image     | Description                                |
|----------------|--------------------------------------------|
| Test image 1   | a train traveling down a track next to a forest |
| Test image 2   | The black cat is walking on grass           |
| Test image 3   | A little girl climbing the stairs to her playhouse |
| Test image 4   | A dog running through snow                  |
| Test image 5   | A brown and white dog is running through the snow. |

5.2.2. Selection of final model

![Figure 9. Training Error Graph within 91 Rounds.](image)
The optimal parameters are determined by different iteration numbers. From the above table, it is obvious that the training error is smaller and smaller as the iteration number increases; and when the iteration number of the model reaches 91 rounds, the training error of the model has dropped to 0.076408. It is considered that the time consumption of each iteration is huge and the decreasing training error gradually decreases, so the current parameters are saved as the final parameters of the model, as shown in Table 10.

| Layer                | Output dimensions | Parameter number | Connection             |
|----------------------|-------------------|------------------|------------------------|
| input_4 (InputLayer) | (None, 34)        | 0                | 0                      |
| input_3 (InputLayer) | (None, 2048)      | 0                | 0                      |
| embedding_2 (Embedding) | (None, 34, 200)  | 330400           | input_4[0][0]          |
| dropout_3 (Dropout)  | (None, 2048)      | 0                | input_3[0][0]          |
| dropout_4 (Dropout)  | (None, 34, 200)  | 0                | embedding_2[0][0]      |

6. Conclusion
According to the paper, in the process of establishing the image description generation model, firstly the input corpus data is processed in details, then the image features are extracted through the excellent pre-training CNN model Inception V3, and the problem of slow network convergence is optimized and avoided by employing the Adam method in establishing the model. From the inspection of random network images, the prediction precision is still within a reasonable range so that the prediction results have a certain reference value. The image description generation model in the paper also has a certain reference applicability. However, considering that the training data is not huge enough and the sample input is still inadequate in the paper, so the generation description of the images that appear less frequently in the training set is inaccurate.

Acknowledgments
This paper is a result of the “13th Five-Year Plan” project (2018GXJK206) of Guangdong education and science in 2018, and a phased result of the project (ZQJYY2019008) undertaken by Zhaoqing Education Development Institute.

References
[1] Li Deng and Dong Yu (2014), "Deep Learning: Methods and Applications", Foundations and Trends® in Signal Processing: Vol. 7: No. 3–4, pp 197-387. http://dx.doi.org/10.1561/2000000039
[2] Hinton GE, Osindero S, Teh YW (2006) A fast learning algorithm for deep belief nets. Neural Compute,18(7):1527–1554. https://doi.org/10.1162/neco.2006.18.7.1527
[3] Bengio Y (2009) Learning deep architectures for ai. Found Trends® Mach Learn 2(1):1–127. https://doi.org/10.1561/2200000006
[4] Bernardi, R., Cakici, R., Elliott, D., Erdem, A., Erdem, E., Ikizler-Cinbis, N., … Plank, B. (2017). Automatic description generation from images: A survey of models, datasets, and evaluation measures. IJCAI International Joint Conference on Artificial Intelligence, 0, 4970–4974.
[5] Yasrab, R. (2018). ECRU: An Encoder-Decoder Based Convolution Neural Network (CNN) for Road-Scene Understanding. Journal of Imaging, 4(10), 116. doi: 10.3390/jimaging4100116
[6] Donahue J, Anne Hendricks L, Guadarrama S, Rohrbach M, Venugopalan S, Saenko K, Darrell T (2015)
[7] Long-term recurrent convolutional networks for visual recognition and description. In: Proceedings of the 28th IEEE conference on computer vision and pattern recognition, Boston, MA, USA, 2625–2634
[8] Wu Q, Shen C, Wang P, et al. Image Captioning and Visual Question Answering Based on Attributes and External Knowledge[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2018, PP(99):1-1: 41-42.