A Cross Modal Deep Learning Based Approach for Caption Prediction and Concept Detection by CS Morgan State

Md Mahmudur Rahman

Morgan State University,
Baltimore MD 21251, USA
md.rahman@morgan.edu

Abstract

This article describes the participation of the Computer Science Department of Morgan State University, Baltimore, Maryland, USA in the ImageCLEFcaption under ImageCLEF 2018. The problem of automatic image caption prediction involves outputting a human-readable and concise textual description of the contents of a figure appeared in a biomedical journal. It is a challenging problem in biomedical literature as it requires techniques from both the fields of Computer Vision to understand the visual contents of the image and Natural Language Processing (NLP) to turn the understanding of the image into words in the right order to generate the textual description for caption. Recently, deep learning methods have achieved state-of-the-art results on this challenging problem in general domain of natural photographic images. For the tasks of caption prediction and concept detection, we used the merge architectures for the encoder-decoder recurrent neural network (RNN) due to it's success over inject for caption generation. The merge model combines both the encoded form of the image input with the encoded form of the text description generated so far. The combination of these two encoded inputs is then used by a very simple decoder model to generate the next word in the sequence. In this article, we present main objectives of experiments, overview of these approaches, resources employed, and describe our submitted runs and results with conclusions and future directions.

Keywords: Caption generation, Concept Detection, Cross Modality, Deep Learning, Encoder-Decoder, Merge Architecture, Performance evaluation

1 Introduction

This article describes our second year's participation in ImageCLEF 2018 [1] for the ImageCLEF caption track which consists of both Concept Detection and Caption Prediction Tasks [2]. For the Concept Detection, participating systems are tasked with identifying the presence of relevant UMLS concepts in images appeared in biomedical journal articles (PubMed Central). For the Caption Prediction, participating systems are tasked with composing coherent captions for the entirety of an image based on the interaction of visual information content and the detected concepts from the first task.

There has been a lot of interest recently from Computer Vision, Machine Learning and NLP community in developing deep learning-based approaches for automatic caption generation of natural photographic images [3,4]. The problem of image caption generation involves outputting a readable and concise description of the contents of a photograph. It is a challenging artificial intelligence problem as it requires both techniques from computer vision to interpret the contents of the photograph and techniques from natural language processing to generate the textual description. Recently, deep learning methods have achieved state of the art results on examples of this problem. In general, a standard encoder-decoder recurrent neural network architecture is used to address the image caption generation problem. Generally, a pre-trained convolutional neural network (CNN) is used to encode the images and a recurrent neural network, such as a Long Short-Term Memory (LSTM) network, is used to either encode the text sequence generated so far, and/or generate the next word in the sequence [5,6].

2 Our Approach of Developing Deep Learning Model

The provided datasets have separate training and test sets, which are really just different groups of image identifiers in separate text files respectively (such as CaptionTraining2018-List.txt and CaptionTesting2018-List.txt in case of caption prediction task). From these file names, we extracted the photo identifiers and use these identifiers as keys to filter photos and descriptions for each set.

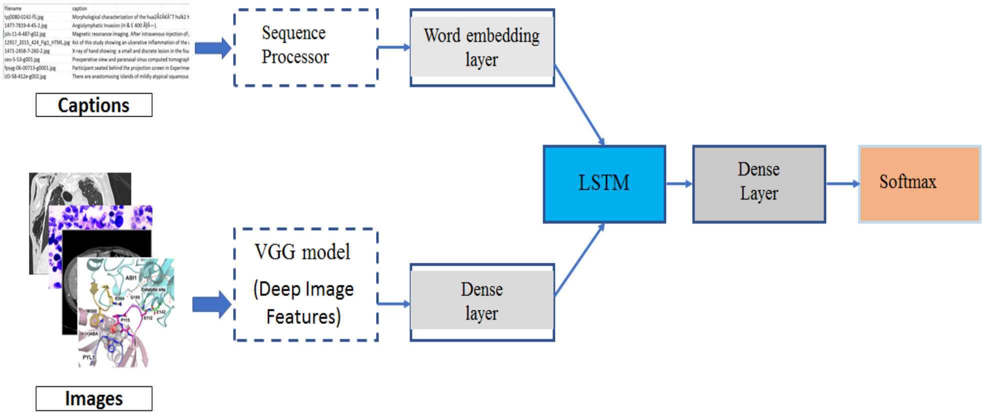


Fig. 1. Merge Architecture for Encoder-Decoder Model [9]

The merge model (Fig. 1) combines both the encoded form of the image input with the encoded form of the text description generated so far, which is handled by Recurrent Neural Networks (RNNs). This separates the concern of modeling the image input, the text input and the combining and interpretation of the encoded inputs [8]. It is experimentally found [9] that the merge architecture is more effective compared to the inject approach, where both image and word information is injected into the RNN. Hence, we followed this approach for both tasks in ImageCLEFcaption.

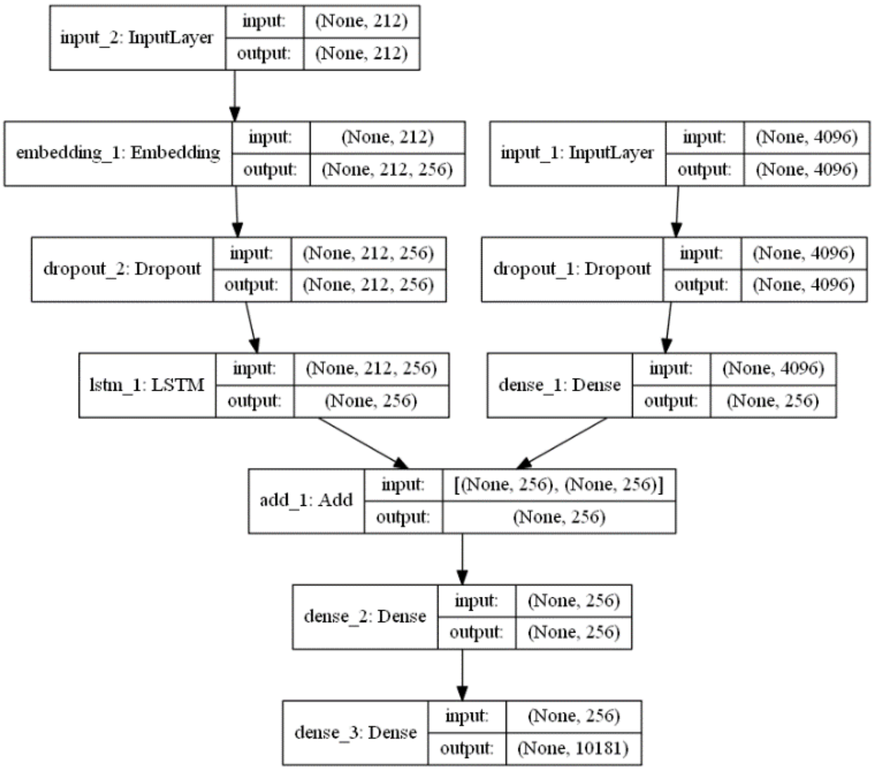


Fig. 2. Plot of the Deep Learning Model for Caption/Concept Generation

The entire process of the model generation is shown as a plot in Fig. 2 and can be described in three parts as follows:

2.1 Visual Feature Extractor

We loaded all the images from a subset of the original training set, extract their features using a pre-trained 16-layer VGG model [5], and store the extracted features keyed on the image id to a new file that is later loaded and used as input for training a language model. We pre-processed the images with the VGG model (without the output layer) trained on ImageNet dataset [7] and used the extracted features predicted by this model as input. Keras also provides tools for reshaping the loaded photo into the preferred size for the model (e.g. 3 channel 224 x 224 pixel image). The Visual Feature Extractor model expects input photo features to be a vector of 4,096 elements. These are processed by a Dense layer to produce a 256-element representation of the photo.

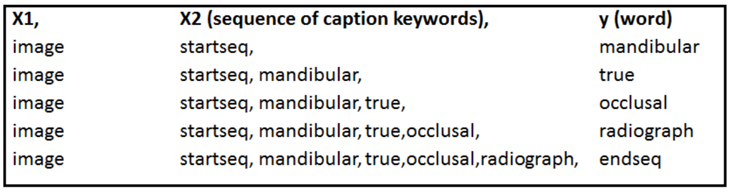
2.2 Text Feature Extractor and Sequence Processor

The training dataset contains caption for each image, which requires some minimal cleaning. Each image name without the file extension (e.g., .jpg) is acted as a unique identifier in a processed text file of descriptions. The descriptions are cleaned and tokenized in order to reduce the size of the vocabulary of words; this means that each token is comprised of words separated by white space where words are transformed to lowercase, all punctuation are removed from tokens and tokens that contain one or fewer characters are also removed. Finally, we save the dictionary of image identifiers and descriptions to a new file named descriptions.txt, with one image identifier and description per line.

This is a word embedding layer for handling the text input, followed by a LSTM layer. The model expects input sequences with a pre-defined length which are fed into an embedding layer that uses a mask to ignore padded values. This is followed by an LSTM layer with 256 memory units. Both the input models produce a 256-element vector. Further, both input models use regularization in the form of 50% dropout. This is to reduce overfitting the training dataset, as this model configuration learns very fast. The Decoder model merges the vectors from both input models using an addition operation. This is then fed to a Dense 256 neuron layer and then to a final output Dense layer that makes a softmax prediction over the entire output vocabulary for the next word in the sequence [6].

2.3 Decoder

Both the feature extractor and sequence processor output a fixed-length vector. These are merged together and processed by a Dense layer to make a final prediction. The model we will develop will generate a caption given a photo, and the caption will be generated one word at a time. The sequence of previously generated words will be provided as input. Therefore, we will need a first word to kick-of the generation process and a last word to signal the end of the caption. We used the strings ***startseq*** and ***endseq*** for this purpose.



X1,	X2 (sequence of caption keywords),	y (word)
image	startseq,	mandibular
image	startseq, mandibular,	true
image	startseq, mandibular, true,	occlusal
image	startseq, mandibular, true,occlusal,	radiograph
image	startseq, mandibular, true,occlusal,radiograph,	endseq

Fig. 3. Example of transforming into input and output Sequences from image description [6]

Each caption of an associated image is tokenized into words after performing some text processing and cleaning operations. The model is provided one word and the photo and generate the next word. Then the first two words of the description will be provided to the model as input with the image to generate the next word. For example, the input sequence *“Mandibular true occlusal radiograph"* would be split into five input-output pairs to train the model:

The Decoder model merges the vectors from both input models using an addition operation. This is then fed to a Dense 256 neuron layer and then to a final output Dense layer that makes a softmax prediction over the entire output vocabulary for the next word in the sequence. Later, when the model is used to generate descriptions, the generated words are concatenated and recursively provided as input to generate a caption for an image. The Keras functional API [10] is used to define the model (Figure 2) as it provides the flexibility needed to define a model that takes two input streams and combines them.

3 Software and Experimental Environment

For implementation of our deep learning model and submission of the runs, we used Keras (version 2.1.5) library in a Python 3 (version 3.5.2) SciPy environment on top of TensorFlow (version 1.1.0) backend. Keras provides a clean and convenient way to create a range of deep learning models on top of both Theano or TensorFlow backend which can be executed on both GPUs and CPUs given the underlying frameworks. Other necessary libraries, such as scikit-learn, OpenCV, Pandas, NumPy pickle, and Matplotlib are also installed and used to support visual and text feature extraction for model inputs as well as model generation, and saving and plotting the model. For training, the model was fitted for 30 epochs and given the amount of training data each epoch took around 3 hours in a window-based workstation.

4 Submitted Runs and Results

This section provides descriptions of our submitted runs and analysis of the result. We (morgan_cs) tried to submit few runs for each task of caption prediction and concept detection. However, only a single run for each task was graded successfully, whereas others were failed due to processing errors.

Both runs are generated based on the deep learning method (merge model) described in previous sections. We almost implemented the same process of feature (textual caption and visual) extraction and model generation for training for both tasks. Results are generated based on the test sets (9,938 images) for captions and concepts. However, for training we only used only a small subset (around 4K images) of the original training set (> 222K images) as provided by the organizers. This limitation is due to the limited resources (both memory and computing power) that was available during the model generation and result submission steps in this evaluation campaign.

Table 1. Results (Top 3) of the Caption Prediction Task.

Group ID	Entries	BLUE

ImageSem	10	0.25
UMMS	6	0.18
morgan_cs	5	0.172

Our successful submission (morgan_cs) of caption prediction (ID: 6202) received a BLUE score of 0.1724, which is ranked third (group submission wise) as shown in Table 1. We also submitted mistakenly a run for caption prediction (which was graded successfully with a 0.0 score although it was meant for the submission of concept detection task). We failed to submit few other runs. However, there were some problems in our runs (result files) and received errors while the files were parsed by the online evaluation tool under crowdAI provided by the CLEF organizer.

For the sole submission (ID: 6216) of concept detection, we received a F1 score of 0.0417 (ranked 4th group wise) which is comparatively lower compared to the leading group (UA.PT_Bioinformatics) with a score of 0.11.

5 Conclusions

This article describes the strategies of the second year's participation of the Morgan CS group for the concept detection and caption prediction tasks of ImageCLEFcaption [2]. Motivated by the state-of-the-art results on caption generation problems in general domain, we developed a cross modal deep learning model by using both image and text features in a merge architecture to accomplish these tasks. We achieved comparable results (ranked in the middle group wise for both tasks) considering the limited resources (computing and memory power) we had at the time of the submission. In future, we plan to experiment with larger pre-trained models (such as, Inception, ResNet, Google Net, etc.) for photo feature extraction that offer better performance on the ImageNet dataset [**7**]. Also, better performance might be achieved by using word vectors trained on a much larger corpus of text, such as news articles or Wikipedia. We will also perform our experiments (e.g. model generation and prediction) in a cloud infrastructure (such, as Amazon AWS) by exploring alternate configuration and exploiting to full capacity of the ground truth dataset (e.g., training sets).

Acknowledgment

This work is supported by an NSF HBCU-UP Research Initiation Award (ID. 1601044).

References

1. Bogdan Ionescu, Henning Müller, Mauricio Villegas, Alba García Seco de Herrera, et all. Overview of ImageCLEF 2018: Challenges, Datasets and Evaluation, Proceedings of the Ninth International Conference of the CLEF Association, CLEF 2018, (2018).
2. Alba García Seco de Herrera, Carsten Eickhoff, Vincent Andrearczyk and Henning Müller. Overview of the ImageCLEF 2018 caption prediction tasks, CLEF working notes, CEUR, (2018).
3. Oriol Vinyals, Alexander Toshev, Samy Bengio, Dumitru Erhan, Show and Tell: A Neural Image Caption Generator, CVPR 2015: 3156-3164 (2015)
4. Oriol Vinyals, Alexander Toshev, Samy Bengio, Dumitru Erhan: Show and Tell: Lessons Learned from the 2015 MSCOCO Image Captioning Challenge. IEEE Trans. Pattern Anal. Mach. Intell. 39(4): 652-663 (2017)
5. Karen Simonyan and Andrew Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: CoRR abs/1409.1556 (2014). URL: http://arxiv.org/abs/1409.1556 (cited on pages 32, 86, 171).
6. Jason Brownlee Blog: https://machinelearningmastery.com/develop-a-deep-learning-caption-generation-model-in-python/
7. J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In CVPR, 2009.
8. Marc Tanti, Albert Gatt, Kenneth P. Camilleri, Where to put the Image in an Image Caption Generator, Natural Language Engineering, 24(03),
DOI: 10.1017/S1351324918000098, (2017)
9. Marc Tanti, Albert Gatt, Kenneth P. Camilleri, What is the Role of Recurrent Neural Networks (RNNs) in an Image Caption Generator?, Proc. 10th International Conference on Natural Language Generation (INLG'17), cited as arXiv:1708.02043 [cs.CL], (2017)
10. Keras Applications API, https://keras.io/applications/