Name: Nour eldin Ibrahim 20101066
Nada Mohamed Samir 20100752

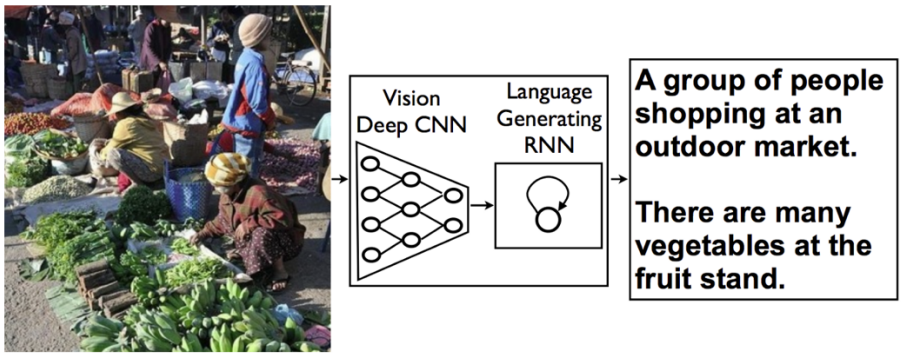
End-to-End Neural Network Models for Arabic Image Captioning

A group of young people playing a game of frisbee.

Two dogs play in the grass.

Abstract

- Image Captioning (IC) is the process of automatically augmenting an image with semantically-laden descrip- tive text. While English IC has made remarkable strides forward in the past decade, very little work exists on IC for other languages. One possible solution to this problem is to boostrap off of existing English IC systems for image understanding, and then translate the outcome to the required language. translated IC is lacking due to the error accumulation of the two tasks; IC and translation. we address the problem of image captioning in Arabic. We propose an end-to-end model that directly transcribes images into Arabic text.
- Image captioning requires extracting meaningful information about the content of the image and expresses the extracted information in a humanreadable sentence. As a result, image captioning models need to achieve several objectives including object detection, extraction of relationships among objects, inference of the pragmatic information within the image, transcribing the information into coherent textual forms with correct syntactical and semantic structures.



- Despite these significant challenges with image captioning, tremendous achievements have been accomplished recently with deep neural networks. In- spired by recent advances in neural machine translation the encoder-decoder approach has been adopted in several proposed image captioning methods. The intuition is that image captioning can be thought of as translation from a set of image pixels to a natural sentence. In machine translation, typically, both the encoder and the decoder include Recurrent Neural Networks (RNN), or one of its variations (e.g., LSTM or GRU)
- instead of RNN, the encoder in image captioning is a Convolution Neural Network which is considered the ideal solution when dealing with unstructured, spatial data such as images.
- we address the challenge of image captioning in Arabic including the lack of Arabic re- sources. We develop a new AIC dataset and propose two separate models for the evaluation of AIC. The first model uses English image captioning mod- els then

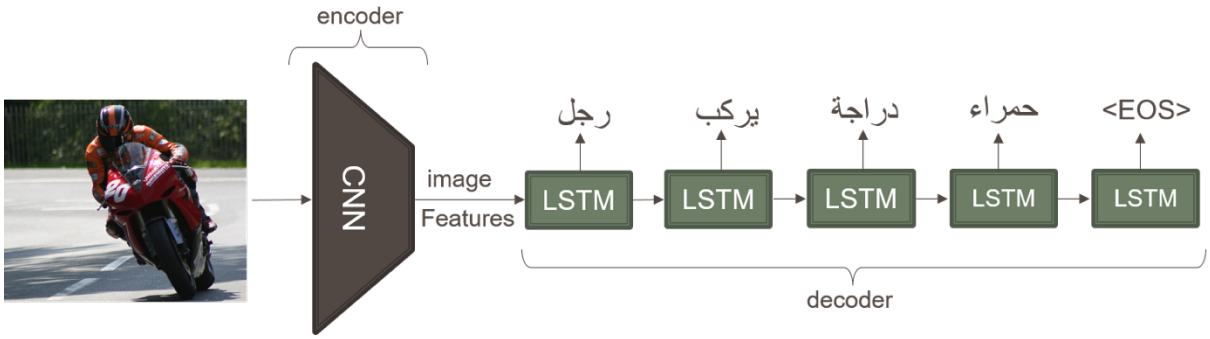
translates the English text to Arabic. The second model is an end-to-end model that directly transcribes Arabic text from images. The models are compared using our AIC dataset, and the results show the superiority of our end-to-end AIC model.

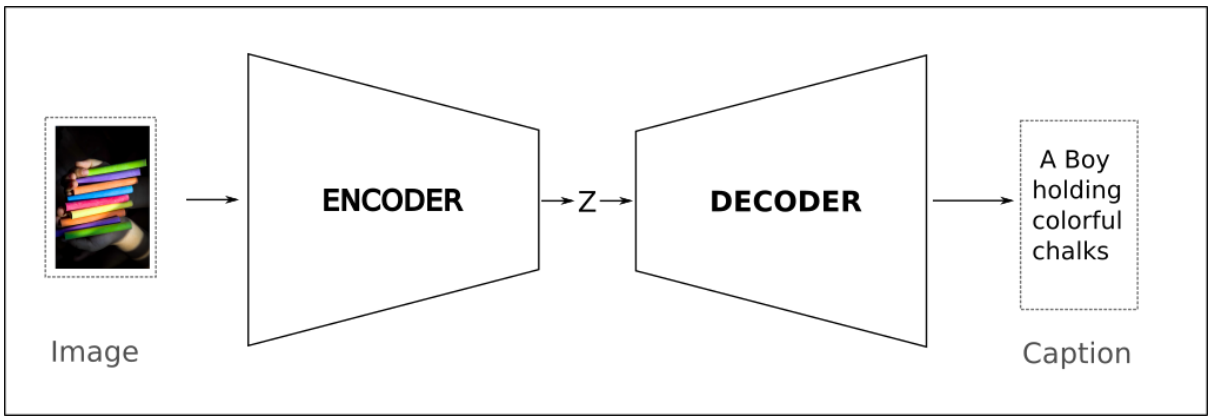
- For the tools/libraries we used in the implementation:

    - Keras: for creating deep models
    - Arabic preprocessing: for manipulating Arabic sentence
    - Tokenizer: for change text to numerical

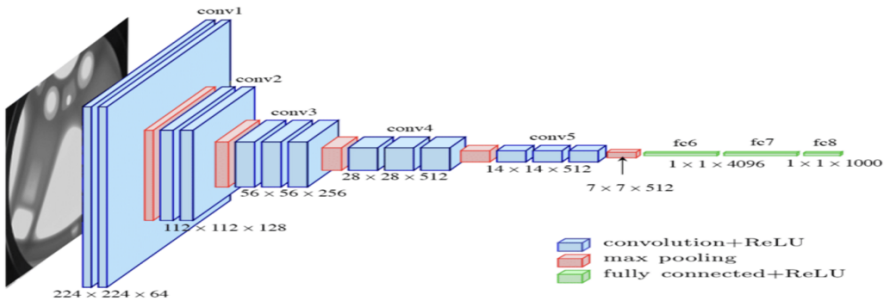
high level architecture Diagram

- model is based on a CNN encoder and an RNN decoder. The CNN extracts relevant features form an image and encodes them into a vector; on the other hand, the RNN aims to decode the encoded vector into a sentence (e.g., LSTM or GRU)





- CNN: Transfer Learning instead of initializing our decoder CNN weights randomly and train from scratch, we will use the weights of a pre-trained CNN. This is known as transfer learning For our CNN, we use VGG16. VGG16 contains thirteen convolution layers and three fully connected layers, and is able to detect approximately one thousand different objects .



- RNN: we used it as Decoder:
  - Two models:
    - GRU
    - LSTM

Description of the dataset

- The proposed Arabic captioning dataset is built by translating the Flickr8K dataset, containing 8000 images, each captioned five times by humans. Images are extracted from flickr and mainly contain humans and animals. Translation to Arabic is performed in two steps: first, all English captions are translated using the Google Translate API During a second stage, all translated captions are edited and validated by a professional Arabic trans- lator. This is necessary because of the many contex- tual errors Google Translate performs.

(A skier jumps high in the air with a view of the mountains)
متسلق يقفز عاليا في الهواء مع منظر للجبال

(A cyclist wearing a red helmet is riding on the pavement)
راكب يرتدي خوذة حمراء يقود على الرصيف

(A black dog is running after a white dog in the snow)
كلب أسود يركض خلف كلب أبيض في الثلج

explanation of the algorithm

1.import libraries:

- Kears – matplotlib – pickle - arabic_reshaper – numpy - pandas

2.Define variables:

- Here, we define some constant variables that we will use later. Variables include the path of the dataset (images & captions) and

the path of other important resources such as the pre-trained VGG16 weights and images embeddings (the output of VGG16 (without the last layer) when fed with Flickr8K dataset).

3. Read & visualize dataset:

- To read the text from the path file captions and visualize it with their images using matplotlib for visualizing and Arabic reshaper for printing Arabic sentence

4. Preprocessing & cleaning captions:

A. clean captions to reduce vocabulary size we need to work with:

- lowercase
- remove punctuations
- remove one-character words
- remove words with numbers

B. clean captions to get rid of useless textual info & reduce vocabulary size. Preprocessing includes:

- remove punctuations & diacritics
- normalize (or standarize) Hamza & Ha2
- remove repeating characters
- remove english characters
- remove one-character words

C. precede each caption with <START> and end each caption with <END>
D. loading feature map of VGG16
E. splitting the images and text into training and test dataset

5. Tokenize text: convert from text to numbers

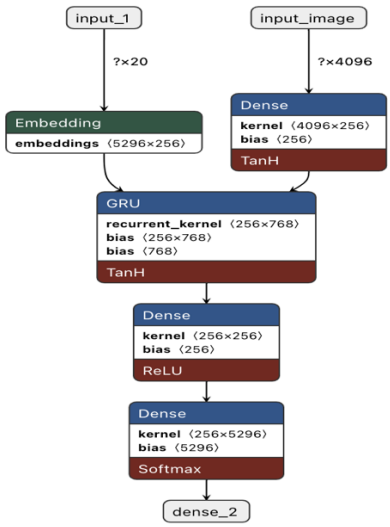
- to break up text into manageable pieces we use keras.Tokenizer for that which contains (num_words : the maiximum number , oov_token: to will added to word index)
- splitting the tokens into training and test

6.Building the model:

A. seq2seq encoder-decoder (with GRU)

- We are using GRU decoder to decode the feature of image into text
- We used as follow for the layers:
  - Input layer

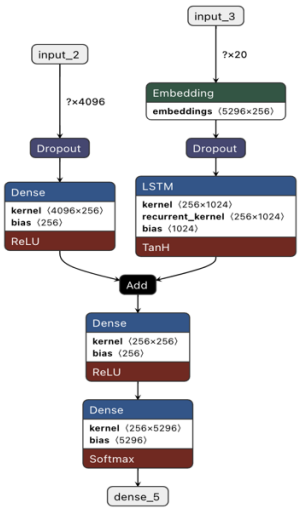
- Embedding layer : Embedding layer enables us to convert each word into a fixed length vector of defined size. The resultant vector is a dense one with having real values (input dim = 20 , output dim = 256)
- Gru layer
- Dense layer
- Dense layer



- we are using Adam for our optimizer
- we trained our model in 10 epochs and batch size 1024

B. seq2seq encoder-decoder (with LSTM):

- We are using LSTM end to end
- We used as follow for the layers:
  - Input layer
  - Embedding layer : Embedding layer enables us to convert each word into a fixed length vector of defined size. The resultant vector is a dense one with having real values (input dim = 20 , output dim = 256)
  - Dropout layer
  - LSTM layer
  - Dense layer
  - Dense layer



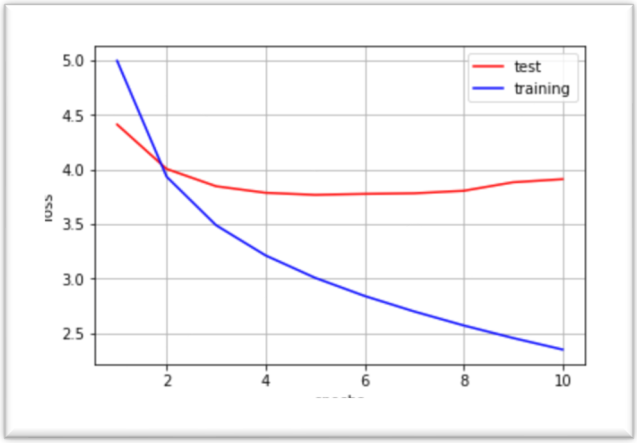
7. extract the English captioning and then translate it to Arabic and then calculate the blue score

results view & results Analysis

A. Predict &Evaluate:

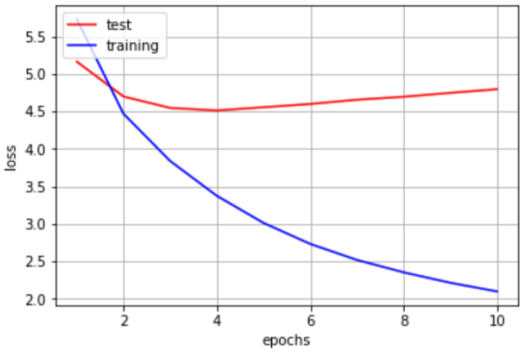
- We predict from the test dataset, and we made comparison between test training Based on number of epochs and loss

A. GRU:



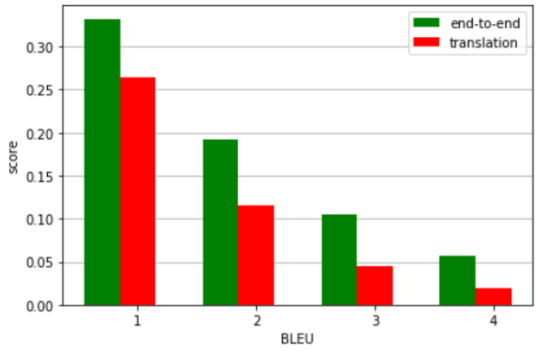
- As we can see the loss reached in test set 40 precent and training set reached below 25 percent

B. LSTM:



- As we can see the loss reached in test below 50 precent and training set reached below 25 percent

C. Compare with translate caption with blue score



- As we can see the end to end it better than the translation

B. predicted output:

A. GRU

<START> كلب بني و اسود يقفز فوق حاجز <END>

<START> فتاه صغيره في ثوب و ردي و قبعه بيضاء و قبعه بيضاء <END>

<START> كلب ابيض و اسود يقفز فوق حاجز <END>

B. LSTM

<START> الكلب يركض في الغابه <END>

<START> طفل في القميص البرتقالي يلعب في زحليقه قابله للنفخ <END>

<START> كلب يقفز في الهواء مع كلب ابيض واسود <END>

Future work

We are going design a model that uses Arabic speech to text then input the output to Arabic to English translation with seq2seq then take the output entered to deep ai API to generate a photo then we will caption the image with voice from text to speech model.

References

Thanks to the paper we able to know to understand image captioning works

- https://www.insticc.org/Primoris/Resources/PaperPdf.ashx?idPaper=88812

download the flicker dataset

- https://www.kaggle.com/datasets/kanishkme/flicker-8k-image-dataset-captionstxt

download VGG16 weights

- https://github.com/fchollet/deep-learning-models/releases/download/v0.1/vgg16_weights_tf_dim_ordering_tf_kernels.h5

download flicker8k features

- https://drive.google.com/file/d/1hAbpX0EQvOWk6EnSHsSKoqP3Ncm167GB/view?usp=sharing