B.M.S College of Engineering

Department of Information Science and Engineering

Deep Learning
20IS6PEDLG

Medical Report Generation

Presented To
Mrs. Shobhana TS (Assistant Professor)

By
Pranava Adiga (1BM19IS103)
Prathviraj Prabhu (1BM19IS111)
Niranjan Hegde (1BM19IS103)
Samartha S (1BM19IS219)

Table of Content

Problem Statement

Generating medical reports from chest x-rays using an encoder-decoder model.

Abstract

Image Captioning is a challenging artificial intelligence problem which refers to the process of generating textual description from an image based on the image contents. Image captioning has many applications such as virtual assistants, editing tools, image indexing and support for the disabled.

In our project we aim to aid radiologists in generating medical reports from x-rays of the human chest. We have built an image captioning system using Deep Learning to achieve our objective. We have used an encoder-decoder architecture, which uses abstract image feature vectors as input to the encoder. We have also used Convolutional Neural Networks and a Recurrent Neural Network powered by LSTM (long short term memory) units.

Introduction

.

Radiologists are medical professionals who focus on employing medical imaging (radiology) techniques such as X-rays, computed tomography (CT), magnetic resonance imaging (MRI), and ultrasound to diagnose and treat injuries and diseases. For the purpose of our project, let us consider radiologists working with x-rays. They observe X-rays and draw meaningful insights from them. They write reports indicating whether the region of the body captured in the X-ray was found to be normal, abnormal or potentially abnormal. Writing medical imaging reports is challenging for less-experienced radiologists, particularly those operating in rural locations where the standard of healthcare is rather low. And for the experienced radiologists, writing these medical reports can be a laborious and time-consuming task. Therefore, in order to overcome all of these issues, wouldn't it be amazing if a computer could input an image of an X-ray and output a medical report, just like a radiologist would? Yes, computers can do this. This field of study is called Image Captioning and it refers to the process of generating textual description of an image.

We have implemented several different techniques to achieve the objective of our project. These techniques are Deep Learning, Convolutional Neural Networks, Recurrent Neural Networks, Encoder-Decoder model, and LSTM.

Deep Learning

Deep learning is a type of machine learning that is trained on massive amounts of data and uses a large number of compute units to make predictions. The goal is to build a system that is similar to how humans think and learn, and hence the underlying architecture for Deep Learning was inspired by the structure of a human brain. Deep learning is entirely based on artificial neural networks. Similar to how the fundamental building blocks of a brain is the neuron, Deep learning architecture includes a computational unit called a perceptron that permits modeling of nonlinear functions. The perceptron receives a list of input signals and changes them into output signals in the same way that a neuron in the human brain transmits electrical pulses throughout our nervous system.

Convolutional Neural Network

The Convolutional Neural Network works by taking an image, assigning it a weightage depending on the image's different objects, and then distinguishing them from one another. In comparison to other deep learning algorithms, CNN requires very minimal data pre-processing. One of CNN's strongest features is that it uses primitive methods to train its classifiers, allowing it to learn the characteristics of the target object. CNNs are most commonly utilized in the field of pattern recognition within images. This enables us to encode image-specific properties into the architecture, making the network better suited for image-focused tasks while also lowering the number of parameters needed to set up the model.

Recurrent Neural Network

Recurrent neural networks (RNNs) are a type of neural network in which the results of the previous step are fed as input to the current step. It is used to solve problems such as image captioning, language translation, natural language processing, speech recognition etc. For example, to predict the next word of a sentence, the previous words are required and hence there is a need to remember the previous words. The main and most important feature of RNN is Hidden state, which remembers some information about a sequence. RNN have a “memory” which remembers all information about what has been calculated. It uses the same parameters for each input as it performs the same task on all the inputs or hidden layers to produce the output.

Long Short Term Memory (LSTM)

Long Short-Term Memory (LSTM) networks are an extension of RNN that extend the memory. LSTM are used as the building blocks for the layers of a RNN. LSTMs assign data “weights” which helps RNNs to either let new information in, forget information or give it importance enough to impact the output. LSTM is well suited to learn from important experiences that have very long time lags in between. LSTMs enable RNNs to remember inputs over a long period of time. This is because LSTMs contain information in a memory, much like the memory of a computer. The LSTM can read, write and delete information from its memory.

Encoder-Decoder Model

Encoding means to convert data into a required format. The encoder is built by stacking recurrent neural network (RNN). We use this type of layer because its structure allows the model to understand context and temporal dependencies of the sequences. The last hidden state of the CNN is connected to the Decoder. To decode means to convert a coded message into intelligible language. The Decoder is a Recurrent Neural Network(RNN) which does language modelling up to the word level. The first time step receives the encoded output from the encoder and also the <START> vector. One of the major advantages of this model is that the length of the input and output sequences may differ. This opens the door for very interesting applications such as video captioning or question and answer.

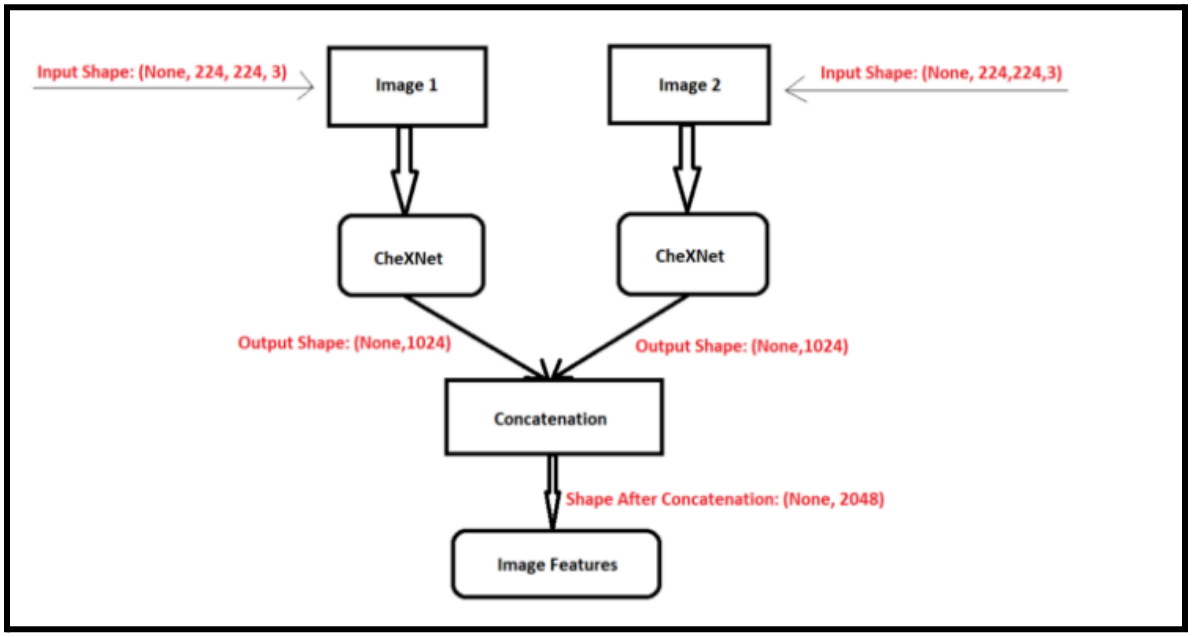
Dataset Used

Indiana University - Chest X-Rays (PNG Images and XML Reports): 1000 radiology reports for the chest x-ray images from the Indiana University hospital network.

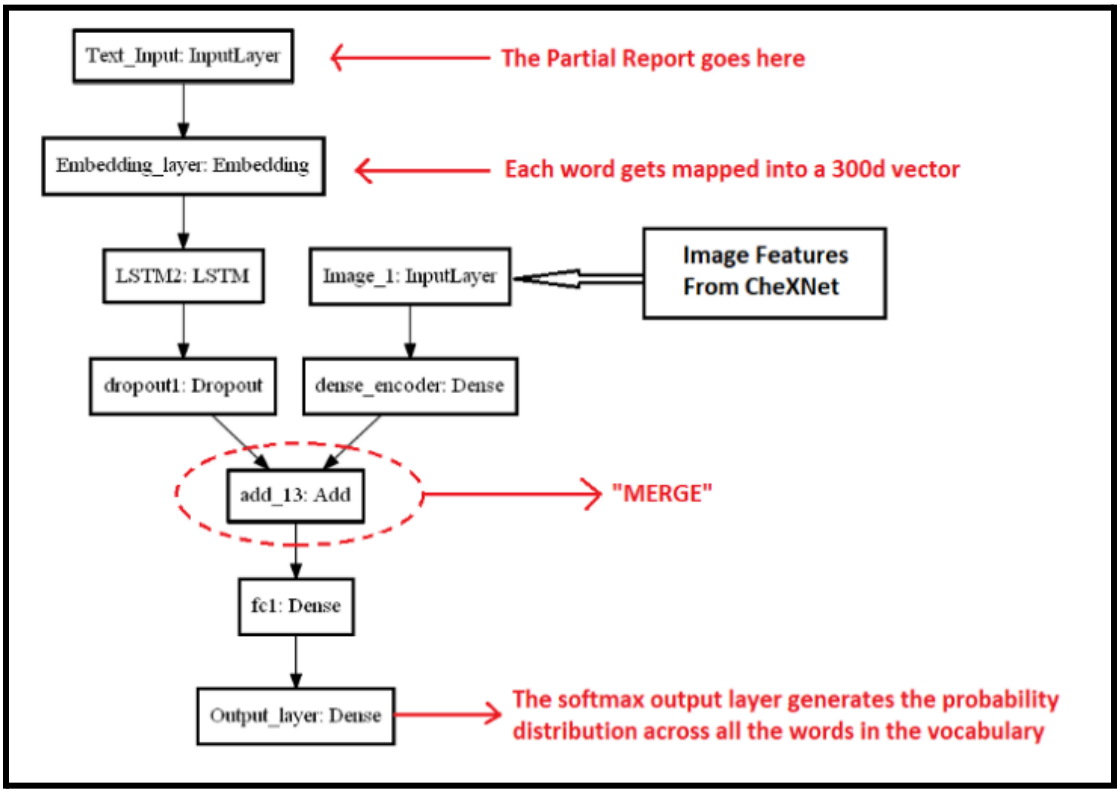
The image dataset contains multiple chest x-rays of a single person. For instance: side-view of the x-ray, multiple frontal views etc. The model will use all these various images and generate a medical report. There are 3955 reports present in the dataset, with each report having one or more images associated with it.

Architecture

Image Feature Extraction:



Architecture:



Implementation

Transfer Learning For Chest Xnet

```
chex = densenet.DenseNet121(include_top=False, weights =
None, input_shape=(224,224,3), pooling="avg")

X = chex.output
X = Dense(14, activation="sigmoid", name="predictions")(X)
model = Model(inputs=chex.input, outputs=X)

model.load_weights(r'./utils/Image_features_enc.h5')
model = Model(inputs = model.input, outputs =
model.layers[-2].output)
```

Converting Word to 300d vector(Text Embedding)

```
try:
    with open(r"embedding_matrix.pickle", "rb") as
output_file:
        embedding_matrix = pickle.load(output_file)
    print("loaded")
except:
    embedding_matrix = np.zeros((vocab_size,300))
    filepath = r"./utils/glove.6B/glove.6B.300d.txt"
    with open(filepath, encoding="utf8") as f:
        for line in f:
```

```
            word, *vector = line.split()
            if word in tokenizer.word_index.keys():
                idx = tokenizer.word_index[word]
                embedding_matrix[idx] = np.array(
                    vector, dtype=np.float32)[:300]
    with open(r"embedding_matrix.pickle", "wb") as
output_file:
        pickle.dump(embedding_matrix, output_file)
```

Main Model

```
input1 = Input(shape=(2048), name='Image_1')
dense1 = Dense(256,
kernel_initializer=tf.keras.initializers.glorot_uniform(seed =
56), name='dense_encoder')(input1)

input2 = Input(shape=(153), name='Text_Input')
emb_layer = Embedding(input_dim = vocab_size, output_dim = 300,
input_length=153, mask_zero=True, trainable=False,
                weights=[embedding_matrix],
name="Embedding_layer")
emb = emb_layer(input2)

LSTM1 = LSTM(units=256, activation='tanh',
recurrent_activation='sigmoid', use_bias=True,

kernel_initializer=tf.keras.initializers.glorot_uniform(seed=23),

recurrent_initializer=tf.keras.initializers.orthogonal(seed=7),
            bias_initializer=tf.keras.initializers.zeros(),
return_sequences=True, name="LSTM1")(emb)
#LSTM1_output = LSTM1(emb)

LSTM2 = LSTM(units=256, activation='tanh',
recurrent_activation='sigmoid', use_bias=True,
```

```
kernel_initializer=tf.keras.initializers.glorot_uniform(seed=23),

recurrent_initializer=tf.keras.initializers.orthogonal(seed=7),
            bias_initializer=tf.keras.initializers.zeros(),
name="LSTM2")
LSTM2_output = LSTM2(LSTM1)

dropout1 = Dropout(0.5, name='dropout1')(LSTM2_output)

dec =  tf.keras.layers.Add()([dense1, dropout1])

fc1 = Dense(256, activation='relu',
kernel_initializer=tf.keras.initializers.he_normal(seed = 63),
name='fc1')
fc1_output = fc1(dec)
dropout2 = Dropout(0.4, name='dropout2')(fc1_output)
output_layer = Dense(vocab_size, activation='softmax',
name='Output_layer')
output = output_layer(dropout2)

encoder_decoder = Model(inputs = [input1, input2], outputs =
output)
encoder_decoder.summary()
```

Future Enhancements

1. Hosting our project as a website, where any person can upload images of X-rays and obtain the medical report generated by our model.

2. Bettering the performance of our model by using Attention Mechanism

Research Papers Referred

1. Medical Report Generation Using Deep Learning by Vysakh Nair
2. Image Captioning in Deep Learning by Pranoy Radhakrishnan
3. Convolutional Image Captioning by Jyoti Aneja, Aditya Deshpande, and Alexander G Schwing
4. Image Captioning: Transforming Objects into Words by Simao Herdade, Armin Kappeler, Kofi Boakye, Joao Soares
5. Convolutional neural networks: an overview and application in radiology by Rikiya Yamashita, Mizuho Nishio, Richard Kinh Gian Do and Kaori Togashi
6. An Efficient CNN Model Based on Object-level Attention Mechanism for Casting Defects Detection on Radiography Images by Chuanfei Hu and Yongxiong Wang