# RADIOLOGY REPORT GENERATION USING DEEP LEARNING

*Ruochen Liu, Ruosi Feng, and Zixi Wang*

University of California San Diego, La Jolla, CA 92093-0238

## ABSTRACT

With the increasing demand for radiology diagnoses in pneumonia outbreak like COVID-19, automated radiology report generation could be an great aid to hospitals.

***Index Terms*—** Deep Learning, Natural Language Generation, Radiology Report Generation, CNN, RNN

## 1. INTRODUCTION

In diagnostic radiology, clinicians often need to identify diseases from radiological images such as CT & chest X-ray and generate a report of such findings for clinical communication. As computer-aided diagnostics prevails, various research has been done in recent years to automatically generate descriptive texts from radiological images using deep learning models, which can greatly expedite the workflow of radiologists. During a large-scale pneumonia outbreak such as COVID-19, such automated radiology report generation can become a crucial aid to healthcare workers by accelerating the diagnosis.

As shown in a recent study[1], in 9 comparable countries, the U.S. has fewer doctors per capita than the average of comparable countries, however, the U.S. has three times as many MRI machines than comparable countries on average. With plenty of medical equipment, the lack of professional healthcare workers can leads to excessive workload for doctors and prolonged waiting time for patients. With computer-aided radiology report generation, the level of automation in radiology examination can be increased, there for the pressing demand for professional healthcare labor can be alleviated.

The input to our algorithm is an image. After our CNN-RNN model is trained by a set of radiology images and their corresponding text reports, it can output a predicted text report for a given radiology image. The assumption is that the descriptive words in the report correspond to details or latent features in the image, and thus can be learned and predicted by a machine learning model.

## 2. RELATED WORK

### 2.1. Deep Learning in Generating Radiology Reports: A Survey[2]

This survey introduced the background of radiology, we used it as a guide line for our project.

A radiology report has two main parts: findings and impressions. The finding section is a descriptive observation of the radiology image. The impression section is the diagnosis from the radiologist. Convolutional neural networks (CNN) are multi-layer neural networks that are efficient in extracting visual features from pixel images. Recurrent neural networks (RNN) are neural networks that can process sequential information (such as words in a sentence). In radiology report generation, CNNs are usually used to extracting features of the radiology image (encoding the feature), RNNs are usually used for generate texts (decode feature into text).

### 2.2. Image Feature Extraction and Language Generation

#### *2.2.1. How to Develop a Deep Learning Photo Caption Generator from Scratch[3]*

The approach of this paper is first use a pre-trained CNN (VGG16) to extract the image feature, then tokenize the text and train a RNN consists of gated recurrent unit (GRU) layers. With the trained RNN model, a caption can be generated for a given image. Compared with long short-term memory (LSTM), the strengths of GRU is it don't need to have a memory unit, thus more computationally efficient. We chose to use GRU for the same reason. The difference between this work and our work is that this work uses a VGG16 CNN to encode the image feature, whereas our work uses a DenseNet-121, which has a better parameter efficiency.

#### *2.2.2. Clinically Accurate Chest X-Ray Report Generation[4]*

The approach of this paper is first use a pre-trained CNN (DenseNet-121) to extract the image feature, then tokenize the text and train a LSTM RNN. With the trained RNN model, a caption can be generated for a given image. Compared with GRU, the strengths of LSTM is it can remember longer sequences, therefore good for tasks requiring modeling long-distance relations. We also used DenseNet-121 since it has a better parameter efficiency.

*2.2.3. Multimodal Recurrent Model with Attention for Automated Radiology Report Generation[5]*

Similar with the papers above, the approach of this paper used pre-trained CNN to extract the image feature, then used LSTM to generate the text. The difference is this paper used attention mechanism in the RNN. The attention mechanism allows the RNN to focus on important part that conveys the main idea of the input image.

### 2.3. Evaluating Text Output in NLP: BLEU at your own risk[6]

The evaluation of language generation is different from most other applications of deep learning since there is no guaranteed way to evaluate the quality of the generated text without the help of a human. Bilingual evaluation understudy (BLEU) is a popular metric for image captioning. It can evaluate the quality of the generated human language, however, it have poor correlation with human judgments.

## 3. DATASET AND FEATURES

We used *Open-I Indiana University Chest X-ray Collection*[7] from Indiana Network for Patient Care as our dataset. It contains 7466 frontal and lateral Chest X-ray images of 3,999 patients. Each patient has a diagnostic reports including "findings" and "impressions" provided by radiologists. Some "findings" or "impressions" items are empty, 3852 out of 3999 patients has both "findings" and "impressions" in their reports. We currently work with "impressions". An example of the data is shown in Fig.1 of The training/validation ratio was set to 8/2, cross validation shows no bias. We centered and cropped the images to the the size of 256*256. We didn't apply normalization nor data augmentation to the images. We divided the text reports into words and then tokenized them into integer token strings. The word frequency was embedded in the string.

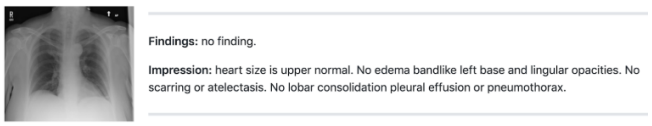


**Fig. 1**. Example data[7]

## 4. METHODS

### 4.1. Text Generation Model Overview

In this work, we focus on predicting the "impressions" section in the reports because the dataset contains more instances of impressions for better training compared to "findings". First, the hierarchical generation via our CNN-RNN model will be introduced. Then each DL architectures and algorithms involved will be explained with more details.

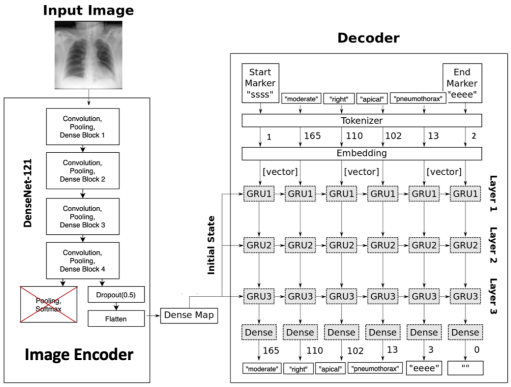


**Fig. 2**. CNN-RNN encoder-decoder architecture used for text sequences generation (training phase schematic)[8]

The encoder-decoder architecture commonly used in image captioning tasks. The assumption is that the descriptive words in "impressions" correspond to details or latent states in the image, and thus can be learned and predicted by a deep learning model. As shown in Fig.2, the model consists of two parts: an image encoder for generating the hidden image representation, and a word decoder that uses image embeddings and word embeddings to predict the sequence of words correspond to the input image. In other NLP models, an additional sentence encoder can be used to bridge the image encoder and word decoder to extract the topic in sentences and generate multiple sentences.[9] The sentence decoder is not required for our work because the "impressions" instances in our dataset mostly contain one sentence each.

During model training, the input chest X-ray images are fed into the image encoder CNN to generate a visual feature map, which is then flattened to a vector that compactly represents the hidden representation of the input image. This mapping vector will then be used as the initial state of the connecting word decoder RNN model. Report text corresponding to the input image is first tokenized to integers, with word frequency information included and then converted to word embedding vectors. The RNN takes the word embeddings generated from sequences of training text strings and recurrently generates output texts. As shown in Fig. X, the vertical arrows in Decoder highlight the process of each input word tokenized and converted to word embeddings, and fed into 3 layers of gated recurrent units (GRUs). The classification layer outputs result based on the predicted probability distribution of the next word following the input word in an input sequence. The horizontal arrows show how the incoming states feeding into the GRUs and becoming inputs to the next GRUs in the pipeline. Each column of GRUs predicts the

next word in sequence until it outputs the end marker token.

### 4.2. Image Encoder CNN Architecture

This part of the model is based on DenseNet-121. DenseNet architecture is used because this architecture solves the problem of vanishing gradient by Dense blocks. These blocks generate a feature map from the concatenated outputs of incoming layers. The representation of the feature map at each layer is described by the following equation:

Thus, the 4 dense blocks in DenseNet-121 allow this model to support very deep network architecture with stable training. Our model 40,592,448 parameters in total.

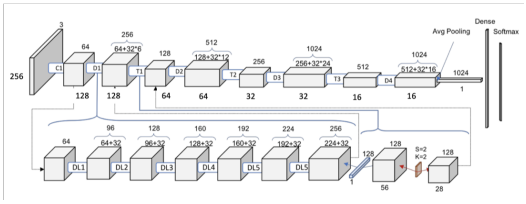

**Fig. 3**. DenseNet-121 Architecture with DenseBlocks)[10]

The input is an image with size 256*256. The output of the last layer before global pooling, (1024) 8*8, is then connected to a dropout layer with a dropout rate r = 0.5, and then flattened and mapped through a dense layer to an image embedding vector of dimension d = 512, corresponding to the size of states in the GRUs in RNN. The activation function used in the output layer is the tanh function, as we found the resulting vector with range [-1, 1] to work better as the input to the decoder RNN.

### 4.3. Word Decoder RNN Architecture

The decoder RNN consists of an input word embedding layer, 3 layers of GRU cells, and a dense output layer with a softmax activation function to select one integer corresponding to the most likely word among the possibilities. Each GRU layer has 30 cells, corresponding to the maximum size of a sentence in our dataset. GRU network is a commonly used RNN architecture, and it has a more compacted architecture than Long Short Term Memory (LSTM) networks. Thus, it is faster to train than LSTM.

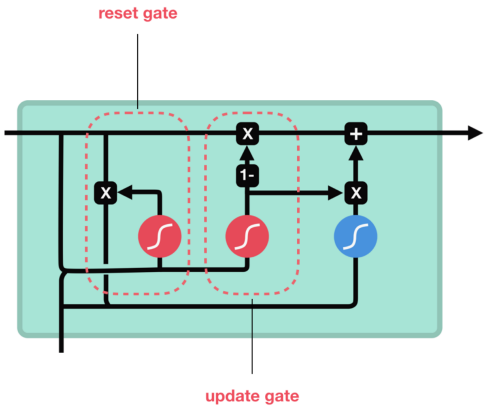


**Fig. 4**. GRU cell architecture)[11]

As the equations shown below, the GRU consists of 4 important components: update gate, reset gate, candidate activation, and the output state.

The update gate and reset gate hold their own weights and biases, use a sigmoid function to generate output. The candidate activation applies a tanh function to represent current memory, and the output vector describes the final memory at the current time step.

## 5. EXPERIMENTS, RESULTS AND DISCUSSION

### 5.1. Hyperparameters

There were 7 hyperparameters that we experimented on in order to get the best performance: CNN's dropout rate, output size and activation function, batch size, RNN's state size and embedding size, and number of epochs.

From our experiments, we noticed that as the output size of the dense net in CNN getting larger, i.e. larger transfer value size, it was more likely to overfit. Thus, we needed to adjust the dropout rate accordingly. We tried two activation function for the CNN dense layer, ReLu and tanh. We found that when using tanh, the transfer values were polarized to 1 and 0. Although non-polarized transfer value should store more information in it, tanh worked good occasionally as well. For the batch size, we tried between 32 to 256, and it did not affect the performance a lot. The state size should be the same as the transfer value size, and the reasonable ratio between it and embedding size was 4. We did not trained our model for more than 50 epochs to avoid overfitting.

Eventually, we used a dropout layer with dropout rate of 0.3 and a dense layer of size 256 and ReLu activation function for the CNN, batch size of 128, state size of 256, and embedding size of 128 for the RNN for our model. We trained this model for 50 epochs.

### 5.2. Quantitative Evaluation

It was not very straightforward to quantitatively evaluate the report we generate, so we chose several metrics: precision, recall, f-measure and BLEU (bilingual evaluation understudy). Let $correct$ be the number of words in prediction sentence that matches the ground truth,

$$precision = \frac{correct}{\text{length of prediction}} \tag{1}$$

$$recall = \frac{correct}{\text{length of the ground truth}} \tag{2}$$

$$f\text{-}measure = \frac{2 \times precision \times recall}{precision + recall} \tag{3}$$

$$\text{BLEU} = \min\left(1, \frac{\text{length of prediction}}{\text{length of the ground truth}}\right)\left(\prod_{i=1}^{4} precision_i\right)^{\frac{1}{4}} \tag{4}$$

The evaluations are shown below in Table 1

| Metric | 53 | 103 | 3414 | 3422 | Overall |
|---|---|---|---|---|---|
| Precision | 0.63 | 0.25 | 0 | 0 | |
| Recall | 0.42 | 0.04 | 0 | 0 | |
| f-measure | 0.5 | 0.06 | 0 | 0 | |
| BLEU | 0.68 | 0.12 | 0.14 | 0.08 | |

**Table 1**. Quantitative Evaluation for Patients 53, 103, 3414, 3422

Patients 53 and 103 were in our training set, while the other two were in the validation set. In the training set, the accuracy for patients without any abnormal in their X-ray in the training set was usually high, as it for patients with abnormal was lower. In the validation set, the precision, recall, and f-measure metrics did not work because the predicted report may using different words describing the same meaning as the ground truth. Therefore, we need to do qualitative evaluation as well.

### 5.3. Qualitative Evaluation

Unfortunately, we could not find a good way to do qualitative evaluation systematically. So, we went though a lot of data to identify some representative ones, such as patients 53, 3414, 3422. Examples are shown below in Fig 5, Fig 6 and Fig 7.

From these results, we can see that most of the reports we generate could match the meaning of the ground truth, but wording differently, and some times did not that close to natural language, which can be improved in our future work.

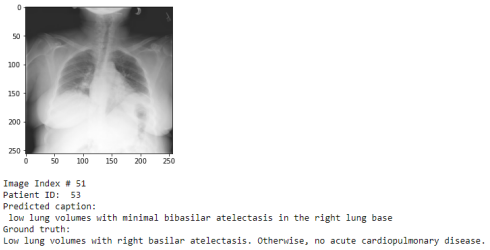


**Fig. 5**. Patient 53

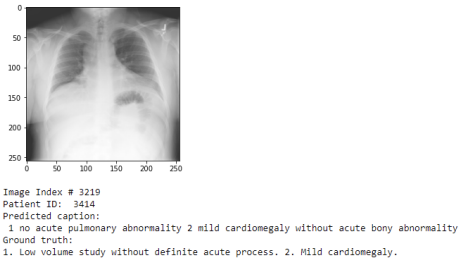


**Fig. 6**. Patient 3414

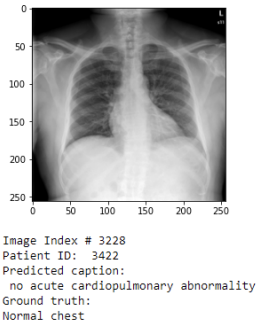


**Fig. 7**. Patient 3422

## 6. CONCLUSION AND FUTURE WORK

Overall, our implementation reached our preliminary goal, which was generating a readable and reasonable radiology report based on a X-ray image. For CNN, compared with VGG16, DenseNet121 can increase the parameter efficiency when extracting the image feature. Compared with LSTM, GRU has is more time-efficient in the training process. For the circumstances that time is a pressing factor, the combination of DenseNet121 and GRU could be a better choice.

For future work, if we had more time, we could try more models for CNN and RNN, such as LSTM, and different architectures. Also, we could experiment more combination of

parameters to find a more optimal one. If we have more computational resources, we could use larger dataset to train the model, as the current dataset has about 8000 images for only 4000 patients. Moreover, we could add phase decoder and paragraph decoder rather than using just one word decoder, which should be able to generate results that are more accurate and closer to human language.

## 7. CONTRIBUTIONS

### 7.1. Ruochen Liu

For the programming part, she worked on dataset preprocess, hyperparameter adjustment and results generation. For presentation, she explained the results and future work. For this report, she mostly worked on Section 5 & 6.

### 7.2. Ruosi Feng

For the programming part, she worked on CNN encoder and image embeddings visualization. For presentation, she mainly worked on model architecture. For this report, she worked on Section 4.

### 7.3. Zixi Wang

For the programming part, he worked on RNN decoder model and hyperparameter adjustment. For this report, he worked Section 1-3 & 6.

## 8. REFERENCES


[1] Bradley Sawyer and Nolan Sroczynski nbsp; KFF. How do u.s. health care resources compare to other countries?

[2] Maram Mahmoud A Monshi, Josiah Poon, and Vera Chung. Deep learning in generating radiology reports: A survey, May 2020.

[3] Jason Brownlee. How to develop a deep learning photo caption generator from scratch, Apr 2020.

[4] Guanxiong Kiu. Clinically accurate chest x-ray report generation, 2019.

[5] Yuan Xue, Tao Xu, L. Rodney Long, Zhiyun Xue, Sameer Antani, George R. Thoma, and Xiaolei Huang. Multimodal recurrent model with attention for automated radiology report generation, Sep 2018.

[6] Rachael Tatman. Evaluating text output in nlp: Bleu at your own risk, Jul 2019.

[7] Rosenman M Kohli MD. Indiana university chest x-ray collection, 2013.

[8] Hvass-Labs. Hvass-labs/tensorflow-tutorials, Apr 2020.

[9] Liu, Hsu, Tzu-Ming Harry, McDermott, Matthew, Boag, Willie, Weng, Wei-Hung, Peter, and et al. Clinically accurate chest x-ray report generation, Jul 2019.

[10] Pablo Ruiz. Understanding and visualizing densenets, Oct 2018.

[11] Michael Phi. Illustrated guide to lstm's and gru's: A step by step explanation, May 2020.