IMAGECLEF2019 VISUAL QUESTION ANSWERING IN MEDICAL DOMAIN

CS4090 PROJECT

Done by:
K.A.Siva Vardhan (B160333CS) B.Maheswararao(B160349CS)
P.Satyanarayana (B160340CS) R.Vamsi Krishna(B160109CS)

Under the guidance of:
Ms.Lijiya A
Assistant Professor

Outline

Topics Covered:

1. Introduction
2. Problem Definition
3. Literature Survey
4. Proposed Method
5. Implementation
6. Results and Discussion
7. Conclusion and Future Work
8. References

Introduction

- Visual Question Answering is a recent and exciting problem at the intersection between Computer Vision and Natural Language Processing
- Input of VQA is an image and a question related to it written in a natural language and the output is the correct answer to the question.
- The traditional CV techniques and NLP techniques are very far from each other.
- The ability to construct a useful answer based on such multi-modal input adds to the complexity of the problem.
- Luckily, recent advances in Deep Learning(DL) have paved the way to build more robust VQA techniques.

Introduction

- This project is about building a model, where the input is an image and a question related to it and output is correct answer to the question.
- For Image Processing, we use pretrained VGG Architecture.
- We use embedding layer to map input words to dimensional features.
- Finally, we concatenate them by LSTM to generate the answers.

Problem Definition

- Given a medical image accompanied with a clinically relevant question, our desired model is supposed to answer the question based on the visual image content.
- This may vary from simple problem such as classification of the image to a complex one such as answer generation.

Literature Survey

NLM at IMAGE CLEF 2018 VQA in the Medical Domain

- Studied Deep learning techniques with state of-the-art performance in open domain VQA.
- Selected Stacked Attention Network(SAN) and Multimodal Compact Bilinear Pooling(MCB) for their official runs.

Literature Survey

Stacked Attention Network(SAN):

- Proposed to allow multi-step reasoning for answer prediction.
- Includes three components
  1. Image model based on a CNN to extract high level image representations.
  2. Question model using an LSTM to extract a semantic vector representation of the question.
  3. Stacked attention model which locates the image regions that are relevant to answer the question.

Literature Survey

Multimodal Compact Bilinear Pooling(MCB):

- It is an attention mechanism that implicitly computes the outer product of visual and textual vectors.
- MCB architecture contains
  1. CNN image model
  2. LSTM question model
  3. MCB Pooling that first predicts the spatial attention and then combines the attention representation with the textual representation to predict the answers.

Literature Survey

JUST at VQA-Med: A VGGSeq2Seq Model

- The encoder consists of two main components.
  1. The first component is an LSTM network with a pretrained word embedding layer.
  2. The second component is a pretrained VGG network that takes the image as an input and extracts a vector representation for it.
- The outputs of the two components are concatenated together into one vector called thought vector.
- The decoder consists of LSTM network that takes the thought vector as initial state and start token as input in the first step and try to predict the answer using softmax layer.

Literature Survey

JUST at ImageCLEF 2019 Visual Question Answering in the Medical Domain

- used different models for different categories of data.

Plane Model:

- The questions format on this category are repetitive and all questions have the same meaning even if they use different words.
- So, it is expected that questions would not contribute anything in answer predictions.
- Hence, they deal with this part as an image classification task.

Literature Survey

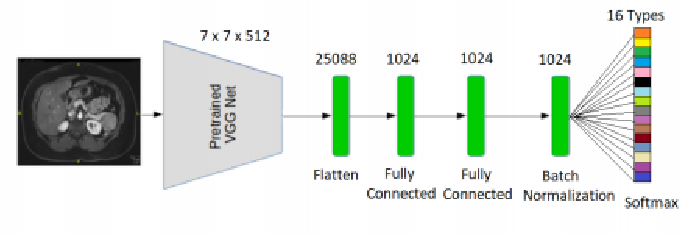


Figure: Plane Model architecture

Literature Survey

- They used the pre-trained model VGG16 with the last layer (the Softmax layer) removed and all layers (except the last four) frozen.
- The output from this part is fed into two fully-connected layers with 1024 hidden nodes followed by a Softmax layer with 16 plane classes.
- Since the data is unbalanced, they used class weights in order to give the classes with smaller numbers of images higher weights.

Literature Survey

Organ model:

- The question format in organ system are also repetitive and have the same meaning.
- So, they rely on the images only to get the organ system answer, i.e., as an image classification task.
- They used the same model architecture for plane model except that the last layer, which is the Softmax layer, has the ten organ systems classes.

Proposed Method

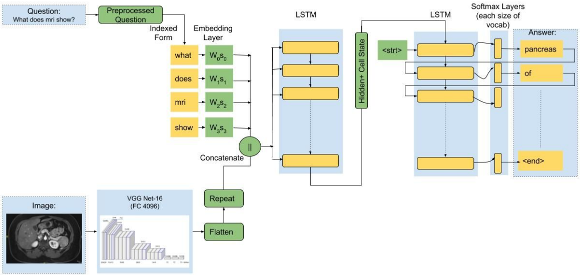


Figure: BaseLine architecture

Proposed Method

- Inputs to the model are pre-processed representations of our images and questions.
- The architecture of our model consists of four modules: Image module, Question module, Encoder module, Decoder module.

Image module:

- Input to the image module are features extracted using VGG-net.
- Initially these features are of size (1, 4096).
- We then use the operation repeat to convert the features to size (maximum question length , 4096).

Proposed Method

Question module:

- We use an embedding layer that learns to map input words to dimensional features(or word-embeddings).
- Word-embeddings help us to represent our words as vectors where semantically similar have similar word vectors.
- Input to this layer is: each question has dimensions(maximum question length,1).
- Output of this layer will be (maximum question length, embed layer size).

Proposed Method

Encoder module:

- The output of the question module is concatenated with the output of the image module.
- The idea of using repeat block for the features and merging features of the image with every word embedding of the question is to make the model learn which word corresponds to what part of the image.
- We then pass this merged output as input to LSTM.

Proposed Method

LSTM:

LSTM(Long Short Memory Networks) are a special kind of RNN, capable of learning long-term dependencies.

LSTMs have chain like structure with four network layers interacting in a very special way.

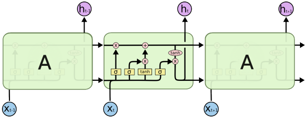


Figure: LSTM Networks

Proposed Method

Different layers in LSTM:

- LSTM does have the ability to remove or add information to the cell state, carefully regulated by structures called gates.
- The key to LSTM is the cell state, the horizontal line running through top of the diagram.

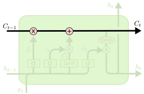
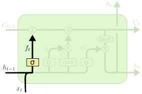


Figure: Cell State

Proposed Method

Step 1:

- The first step in LSTM is to decide what information we are going to throw away from the cell state.
- The decision is made by a sigmoid layer called "forget gate layer".
- It looks at ht-1 and xt and outputs a number between 0 and 1 for each number in the cell state ct-1.



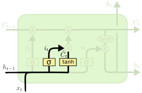
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \;+\; b_f\right)$$

Figure: first layer in LSTM

Proposed Method

Step 2:

- In this step, we have to decide what new information we are going to store in the cell state.
- This step has two parts:
  1. A Sigmoid layer called the "input gate layer" decides which values we will update.
  2. A tanh layer creates a vector of new candidate values, c't, that could be added to the state.



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

Figure: second layer in LSTM

Proposed Method

LSTM:

- LSTMs are used for sequence modelling and capturing long term dependencies.
- Last layer of LSTM will encode question sequence information with image information.

Proposed Method

Decoder module:

- The output of the encoded layer is passed as the initial state to another LSTM.
- During training, the outputs of the decoder of the LSTM are probability distributions(over all the words in the vocabulary) generated by the model for the next word in the sentence.
- The model is trained to minimize the negative sum of the log probabilities of each word.
- During interference part, to get the next word, we pick the word with maximum probability at each time step.

Implementation

Image Pre-processing:

- Resize

Images are resized such that height and width of the target image is equal to 224.

- Feature Extraction

Pre-trained VGG-Net model is used for computing the image features.

The activations from the last fully connected layer of the model are extracted and used as features.

The features created are of size 4096.

Implementation

Question Pre-processing:

- Tokenize
  1. Here our task is to convert text input sequence into numerical form so that each word in the sequence is represented by a number that can be used to index vocabulary.
  2. We start by first parsing the sentence to remove punctuations, decapitalize words etc.
  3. For instance, if input sequence is of the form: "what does mri show?". Then we tokenize this sequence to obtain: [“what", "does", "mri", "show"]

Implementation

- Vocabulary creation:
  1. Next step is to create a vocabulary.
  2. We take the total number of unique words present in the question and answers of the given dataset which turns out to be the size of 3499 in our case.
  3. We have chosen to taken all the words present in the QA in the vocabulary instead of taking the most frequent ones.
  4. As the answers contain medical terms, which have low frequency but are relevant in answering the question.

Implementation

- Additional Tokens

  Also we have added 4 new tokens: [‘NULL’, ‘START’, ‘END’, ‘UNK’] that denotes “no word”, “start of the sequence”, “end of the sequence”, “unknown word” respectively.
- Fixed length sequences
  1. Input sequences would be of variable lengths, so the task here is to transform all of them to fixed size.
  2. We took max question length and max answer length to be 21.
  3. The questions and answers with smaller length are padded with NULL token.

  So, tokenized input gets converted to [“START”, “what”, “does”, “mri”, “show”, “END”, “NULL”, “NULL”, “NULL”, “NULL”]

Implementation

- Create two dictionaries
  1. ‘word_to_idx’ and ‘idx_to_word’.
  2. As the name suggests, ‘word_to_idx’ dictionary maps words to index of that word in the vocabulary and ‘Idx_to_word’ does the opposite.
- Change to numerical form
  1. Now we have to transform our sequence to sequence of numbers where each number represents the index of that word in the dictionary.
  2. We can do that using ‘word_to_idx’ dictionary.

So, tokenized input changes to [1, 7, 28, 55, 6, 2, 0, 0, 0, 0].

Implementation

Training and Modelling parameters:

- Batch Norm
  1. Batch Normalisation is used to stabilize the network and it helps network converge faster.
  2. We applied Batch Normalisation in the encoder module LSTM that is after merging the outputs of question and image module.
- Drop Out
  1. It is a regularization technique used to ensure that model does not overfit.
  2. The value of dropout parameter is 0.3

Implementation

- Learning Rate
  1. Learning rate controls how model weights are adjusted with respect to the loss gradient.
  2. When learning rate is small, the model trained is more reliable but it takes time to get the optimal parameters as the parameters are updated by a smaller value.
  3. The optimiser used by us changes the learning rate dynamically during the training, but we can still set the initial learning rate

Implementation

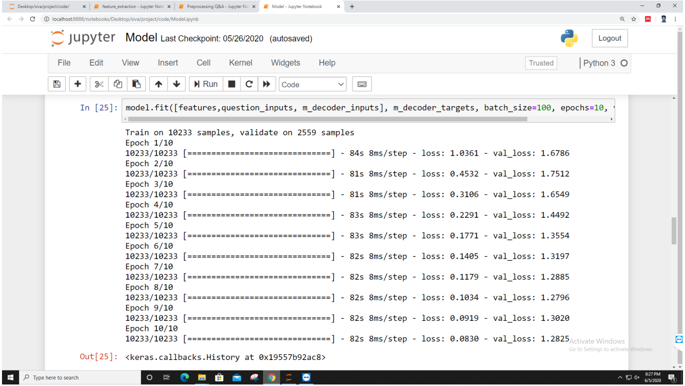
- Dropout is applied before getting outputs from the encoder and decoder module to avoid overfitting.
- Cross entropy is used as loss function for training the model.
- RMSProp optimisation algorithm is used for updating the parameters of the model.
- TanH function is used as activation function in LSTM units. 20% of the training dataset is used as validation dataset.

Implementation

The important model parameters and their values are used for the parameters:

- Embedding size=150
- Dimension of hidden unit=150
- Maximum question answer length=21
- Learning rate=0.001
- Batch size=100
- No of epochs=10
- Dropout=0.3

Implementation


Desktop/siva/project/code/
feature_extraction - Jupyter Note
Preprocessing Q&A - Jupyter No
Model - Jupyter Notebook
localhost:8888/notebooks/Desktop/siva/project/code/Model.ipynb
jupyter
Model Last Checkpoint: 05/26/2020 (autosaved)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3
Run
Code
Activate Windows
Go to Settings to activate Windows.
Type here to search
8:27 PM
6/5/2020

```
In [25]: model.fit([features,question_inputs, m_decoder_inputs], m_decoder_targets, batch_size=100, epochs=10,

         Train on 10233 samples, validate on 2559 samples
         Epoch 1/10
         10233/10233 [==============================] - 84s 8ms/step - loss: 1.0361 - val_loss: 1.6786
         Epoch 2/10
         10233/10233 [==============================] - 81s 8ms/step - loss: 0.4532 - val_loss: 1.7512
         Epoch 3/10
         10233/10233 [==============================] - 81s 8ms/step - loss: 0.3106 - val_loss: 1.6549
         Epoch 4/10
         10233/10233 [==============================] - 83s 8ms/step - loss: 0.2291 - val_loss: 1.4492
         Epoch 5/10
         10233/10233 [==============================] - 83s 8ms/step - loss: 0.1771 - val_loss: 1.3554
         Epoch 6/10
         10233/10233 [==============================] - 82s 8ms/step - loss: 0.1405 - val_loss: 1.3197
         Epoch 7/10
         10233/10233 [==============================] - 82s 8ms/step - loss: 0.1179 - val_loss: 1.2885
         Epoch 8/10
         10233/10233 [==============================] - 82s 8ms/step - loss: 0.1034 - val_loss: 1.2796
         Epoch 9/10
         10233/10233 [==============================] - 82s 8ms/step - loss: 0.0919 - val_loss: 1.3020
         Epoch 10/10
         10233/10233 [==============================] - 82s 8ms/step - loss: 0.0830 - val_loss: 1.2825

Out[25]: <keras.callbacks.History at 0x19557b92ac8>
```

Figure: Training the model

Results and Discussion

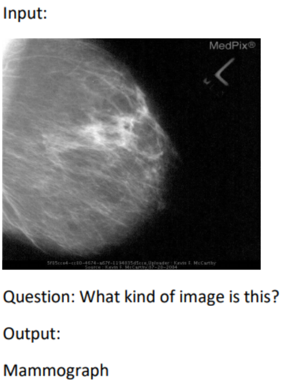
- The results of the model are tabulated as follows:

Table: Results

Sl. No	Category of data	Accuracy
1	Modality	0.444
2	Plane	0.734
3	organ	0.684
4	Entire data	0.5155

Accuracy of the model increases with increase in the depth of the network.

Results and Discussion



Input:

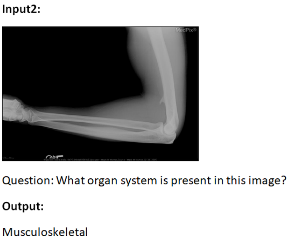
Question: What kind of image is this?

Output:

Mammograph

Figure: example1

Results and Discussion



Input2:

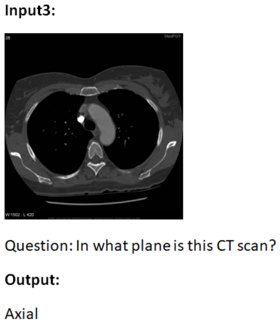
Question: What organ system is present in this image?

Output:

Musculoskeletal

Figure: example2

Results and Discussion



Input3:

Question: In what plane is this CT scan?

Output:

Axial

Figure: example3

Conclusion and Future Work

- We reviewed popular methods in deep learning and built a VQA model for ImageCLEF 2019.
- We adjusted some learning parameters in our model to increase accuracy.
- Since deep learning techniques are significantly improving, we can reasonably expect that VQA is going to be more and more accurate in the next years.

References

- A. B. Abacha, S. Gayen, J. J. Lau, S. Rajaraman, and D. Demner-Fushman, “Nlm at imageclef 2018 visual question answering in the medical domain,” 2018.
- B. Talafha and M. Al-Ayyoub, “Just at vqamed: A vgg-seq2seq model”
- Aisha Al-Sadi1, Bashar Talafha1, Mahmoud Al-Ayyoub1, Yaser Jararweh1 and Fumie Costen2 ”JUST at ImageCLEF 2019 Visual Question Answering in the Medical Domain”
- Akira Fukui, Dong Huk Park, Daylen Yang and Anna Rohrbach ”Multimodal Compact Bilinear Pooling for Visual Question Answering and Visual Grounding”

Thank You.