Medical Images Captioning

Alessandro Calò, Edoardo Procino NLP and Text Mining, University of Oulu

In this project we focused on analyzing a medical image dataset, specifically comprised of radiology images and their captions. Our goal was to build a deep learning model, composed by an encoder and a decoder, to be able to generate captions for the radiology images. Along with that, this project includes a natural language processing section in which we utilized specific tools in order to analyze and visualize the word space of the caption dataset.

1 Introduction

One of the most cutting-edge techniques which has seen a significant rise during this period is the use of artificial intelligence developed specifically for the purpose of aiding the medical worklfow, including diagnosis and treatment of diseases, being implemented with the use of machine learning or deep learning algorithms. The use of these technologies is very promising and already showed great potential, as some of them are already being employed in professional settings. [4]

In medical standard practice it is very often required to study and analyze radiology images for many purposes, diagnostics being one of the most important ones. In order to perform such a task, the presence of a specialist is required, in this case a radiologist or another kind of physician. This operation is not immediate and can sometimes be very time consuming, it is also reasonable to consider that these specialists could make mistakes and may need a second opinion. In order to reduce the human error factor as much as possible and avoid time waste, different methods to aid these professional figures have been (and are being) developed during the last years.

Our interest is now focused on medical images captioning, which is an ever growing field, its aim is to provide the visual information of biomedical images (radiology in our case, example in Figure 1) in the form of a human-readable, meaningful and descriptive text, actively emulating what an human being could be capable of doing with the proper academic background and experience. This research field provides aid to the search of finding a secure, reliable and cheap method to perform medical reports.

However it comes with some limitations: the available data is not always adequate, as there are not many sources that are accessible publicly, and there are no agreed-upon baselines for the performances of the models, which are all very complex and difficult to compare. These limitations play a huge role in preventing automated image captioning from being clinically accepted, not being able to achieve adequate results.

In our work we build and implement a model to perform this task, following the recent literature which shows that the preferred method to tackle this problem is utilizing an encoder-decoder model, as it achieved significant results in extracting the captions of the images in an efficient way.

2 Methodology

2.1 Dataset

The dataset we studied in this project is called ROCO (Radiology Objects in COntext) Dataset, a large-scale medical and multimodal imaging dataset. It was made with the data found in PubMedCentral, a public biomedical database, and is composed by more than 80k samples considering both radiology and non-radiology. Along with the images, the dataset also contains the captions of each one of them, which describes what the im-

Figure 1: An example of a radiology image in the ROCO dataset

age is showing.

As stated above, the full dataset comprises both radiolgy and non-radiology, but we only considered the former, as the captions for the non-radiology images were not correctly matched. In fact, for the radiology images the matching was done trough a .csv file containing all the image-caption pairs, enabling us to create a pandas dataframe with the correct samples: this file was not present in the non-radiology folder, preventing the correct matching of the samples.

It is split into three categories as can be seen in Figure 2

2.2 Pre-Processing

While analyzing the dataset it became clear that there were some problems with how the images were labeled and with the images themselves, not only in the non-radiology section, but also in the radiology one. More specifically, it was noticed that in the test set, composed of around 8k samples, the names of the images found in the .csv file were different than the ones actually present in the images folder, so not a single image-caption match could be done inside that fraction of the dataset. A similar, but less severe problem was found in the validation set, while the train set was perfectly matched throughout its entirety. This is why we decided

- dataset
  - test
    - non-radiology
      - images
      - captions.txt
    - radiology
      - images
      - captions.txt
  - train
    - non-radiology
      - images
      - captions.txt
    - radiology
      - images
      - captions.txt
  - validation
    - non-radiology
      - images
      - captions.txt
    - radiology
      - images
      - captions.txt

Figure 2: An example of a radiology image

to slice a part of the train set to obtain a new test set of 10k samples.

One of the features of this dataset is that the image size is not coherent among its entirety, resulting is some images with very high resolution and size, opposed by other images with very low resolution and relative small size.

In order to prevent this size mismatch along the data, a normalization strategy was implemented by resizing every image to 299x299 (as required by our network standards): Figure 3 shows an example of implementation of this method. As the figure shows, images were resized without maintaining the aspect ratio.

Another crucial factor related to data is the difference in the captions lengths (in terms of number of word). Some images have a captions of 0, 1 or 2 words while others have hundreds of words and this fact obviously can worsen the performance of the model. For this reason and also because same of the very short and some of the very long captions were meaningless, we decided to keep only the data with a caption length between 9 and 33 words. As an example, the caption of a radiography of an hand describe the fact that now the patient, after

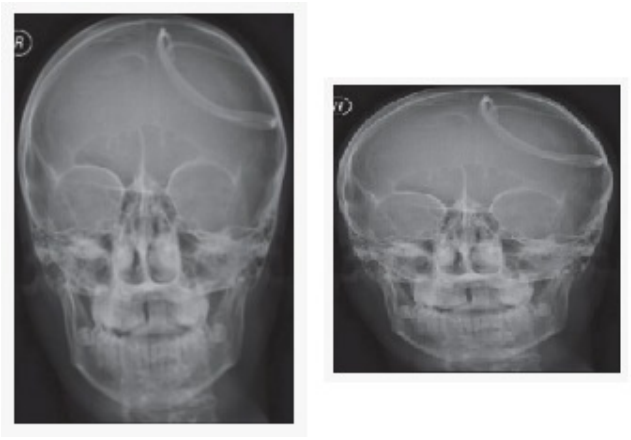

Figure 3: Comparsion between original image and resized image

one year from the surgery, feels better. This caption is obviously an outlier – the majority part of the other captions describe only the current radiography without having cognition of a precedent state of the patient – and thus can only confuse the network. The final dimension of our sets are 40000 images for the train and 4000 images for both the train and validation datasets.

Another pre-processing step was the addition of the *startseq* and *endseq* tokens at the beginning and at the ending of each caption to make the network able to understand when a caption starts and when it ends.

2.3 Visualizing Word Space

This section includes the work done on the captions in order to perform the embedding of the words. Word embedding is a technique which is used to represent the semantic space of a document's vocabulary, comprehending dependencies between linguistic units. Furthermore, it is able to perform the acquisition of a word inside a document and its relation with other words in the same document from a syntactic and semantic point of view. This is achieved trough a transformation of the word into a vector representation: similar vectors, called embeddings, (similar words) are gonna be gathered in semantically-related areas in a plot, as their vectorial representation is gonna be similar. The vector space in which these words are embedded can reach very high dimensionality, above 300: due to this factor, in order to visually be able to read such a space, it is foundamental to apply dimensionality reduction techniques before projecting the embeddings.

When studying a domain, it is often very useful to graphically visualize the word space, to better understand how the words in the vocabulary

2.3.1 Data Cleaning

This step was not done on the entirety of the datasets, but to our, above described, reduced datasets. The steps taken in order to complete this process were the following:

- Stop word removal
- Punctuation removal
- Lemmatizaion
- Stemming

The input data before the cleaning step is essentially a list of strings, where each string represents the actual caption of each medical image. After cleaning, the list is transformed into a list of lists: the outer list contains all the captions, while the inner list is composed of the single words in the original phrase, with all the above mentioned modifications applied to them. An example is shown below:

Original string:

```
"Computed tomography scan in axial
view showing obliteration of the left
maxillary sinus"
```

After cleaning operations:

```
['comput', 'tomographi', 'scan',
'axial', 'view', 'show', 'obliter',
'leav', 'maxillari', 'sinus']
```

Then, starting from these lists, the captions with the cleaned words were computed and they were used to train and test the model.

2.3.2 Word Distribution

Word occurrence frequency is a method to evaluate the importance of the terms in a document and discern their discriminatory power. Figure 4 and Table 1 show the word distribution and ranking of the studied dataset. The leftmost words are the most common, while descending on the curve we can find words that appear less often, following a logaritmic function.

	words	count
1	show	21068
2	arrow	9692
3	leav	8446
4	imag	8378
5	right	8231

Table 1: Most common five words in the vocabulary

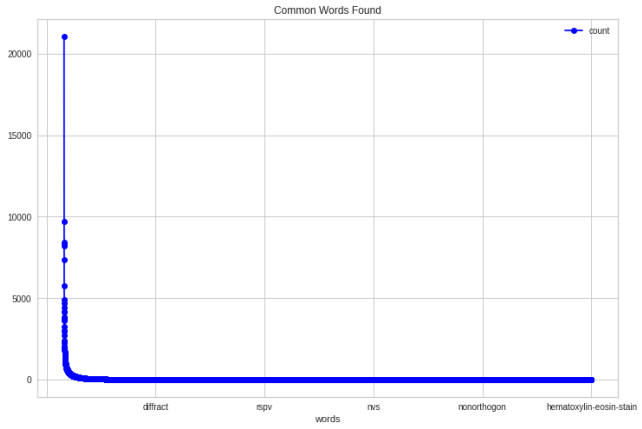

Figure 4: Word distribution of the caption dataset

In order to understand if a power law can be fitted to this vocabulary it is necessary to plot a log-log scale graph, as seen in Figure 6. For this kind of curve we can say that a power law could be fitted. Even if the line doesen't seem to be straight, this is due to the fact that Zipf's law makes most errors at the lowest and highest frequencies.

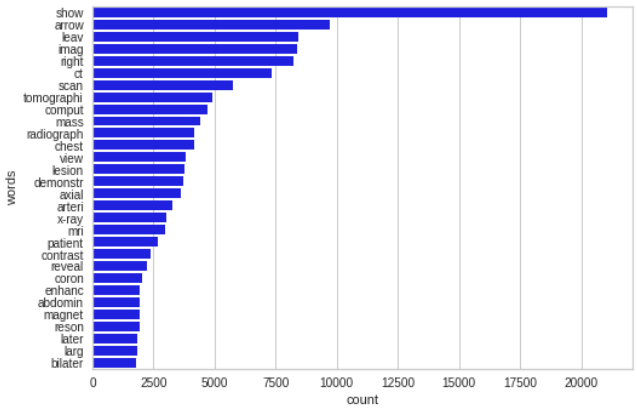

Figure 5: Top 30 words showed trough a barplot

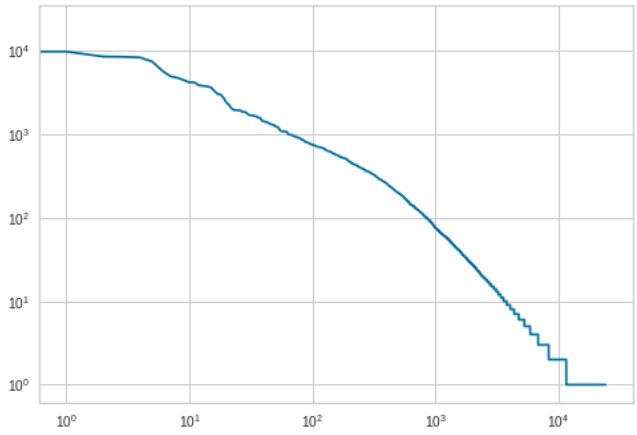

Figure 6: Log-log scale representation of the word distribution plot

2.3.3 Word2Vector

Word2Vector represents one of the ways it is possible to implement and obtain an embedding of the words inside a vocabulary, by utilizing a shallow neural network that needs to be trained on the data of the selected domain. Using word2vector, we can visualize the word space by plotting the embedded words. This technique counts on local statistics to obtain local semantics of a certain word.

In this method we are training the model on the ROCO dataset and then calculating the embeddings. Figure 7 shows a restricted space of the vocabulary, in particular the space of the words: 'hand', 'leg', 'tumor'.

As it can be seen, semantically close words are correctly placed next to each other, while a poorly related word is represented on the other side of the space. It

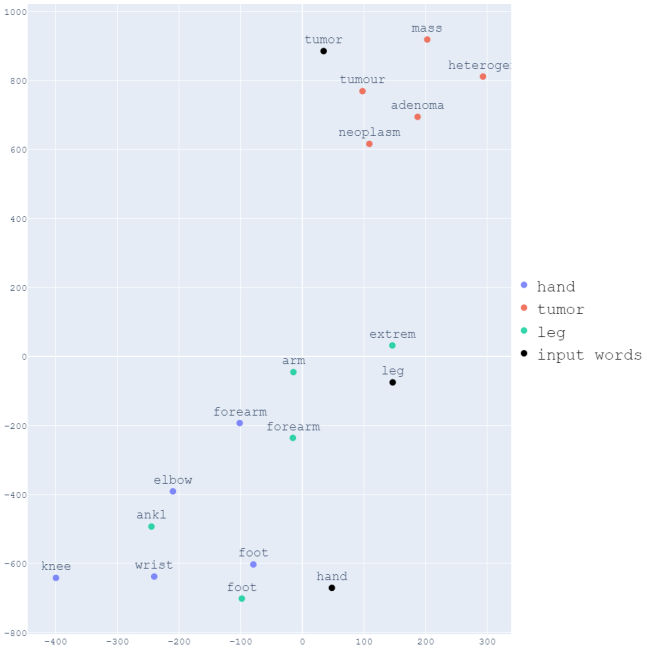

Figure 7: Word space visualization using Word2Vector

is also possible to notice that the cluster of words surrounding the input is composed of elements that fit in a proper manner to the specific context. In 2.3.5 cosine similarities will be discussed.

Figure 8 shows the space of words which are semantically related to the word 'hand': as it can be seen, Word2Vector is capable of understanding the meaning of the words of a specific domain and is able to perform the task of visualizing the space of logical units. This is because all the words that surround our searched word represent body parts that can be found near the hand.

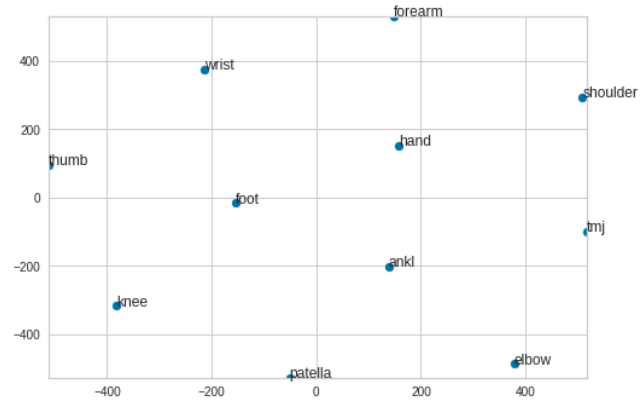

Figure 8: Word space representation of semantically related words to the word 'hand'

For the purpose of obtaining a new representation, another technique can be adopted, as described in the following section.

2.3.4 GloVe

Rather than relying on local statistic, GloVe utilizes globals statistics as Latent Semantic Analysis to derive the global linguistic affinities between words. Using GloVe (Global Vectors) can be a key factor to better visualize the word space of this domain. In fact, this method allows us to utilize pre-trained vectors to embed words, in this case the vector is composed of 400.000 words. As expected, Figure 9 shows how this method can be used to better represent the word space, because the input words are surrounded by semantically-related words.

Figure 9: Word space visualization using GloVe

2.3.5 Cosine similarity

Cosine similarity is a typical approach for measuring the affinity of two vectors of an inner product space. It takes into account the cosine of the angle between the two vectors and detects whether or not the vectors have similar directions: the smaller the angle, the most simi-

lar are the two vectors, obtaining a greater value (value equals to 1 when two identical vectors are being compared). Equation 1 shows the mathematical formula of cosine similarity.

$$sim(x, y) = \frac{x * y}{||x|| * ||y||} \quad (1)$$

In our case it can be used to compare how Word2Vec and GloVe perform on the selected words.

Table 2: Cosine similarity between words using Word2Vect

	hand	leg	tumor
hand	1	0.86	0.36
leg	0.86	1	0.41
tumor	0.36	0.41	1

Table 3: Cosine similarity between words using GloVe

	hand	leg	tumor
hand	1	0.61	0,25
leg	0,61	1	0,26
tumor	0,25	0,26	1

Tables 2 and 3 show the different performances obtained used the two different methods: the values in the tables are in line with the visual representation, showing how word2vec exhibited different performances w.r.t. GloVe, for the above mentioned reasons. In fact, Word2Vect is able to better understand that 'hand' and 'leg' are semantically close words, finding an affinity factor of 0.86, but finds a pretty high similarity of 0.41 between 'leg' and 'tumor'. On the other hand, GloVe calculates a similarity of just 0.61 between 'hand' and 'leg', but correctly discriminates very well between semantically distant words.

2.4 Model

The presented model presents an encoder-decoder architecture (Figure 13), which has been the preferred one from the literature, as it achieved generally better results w.r.t. other approaches.

2.4.1 Others preprocessing steps

To prepare the captions to be feeded to the network, others preprocessing steps were needed. Starting from the cleaned captions we saw that, in the train set captions, there were 22927 unique words, too much to train our network, also because some of them are repeated a few times. Thus, we limited the train set vocabulary to that words that occur at least 15 times overall obtaining a new vocabulary size of 2778 words.

The last preprocessing steps are the creation of two dictionaries, one to translate the words in the dictionary to numbers (understandable by the model) and one for the inverse operation, and the creation of a GloVe embedding matrix which values will be then used in a layer of the neural network.

2.4.2 Encoder

The encoder is the first part of our model. It performs some convolutions on each image generating as output a *latent feature vector* (one for each image). To do so, a pre-trained neural network is used. The network that we chose is InceptionV3. It's main characteristic is that it avoids the overfitting generated by stacking a lot of layers using instead parallel layers reducing the need to go deep (see Figure 10). Also other techniques such as Spatial Factorization into Asymmetric Convolutions and Factorization into Smaller Convolutions are used [5]. We chose this network since it has very good performances and it is also quite efficient w.r.t. other popular networks for the same task.

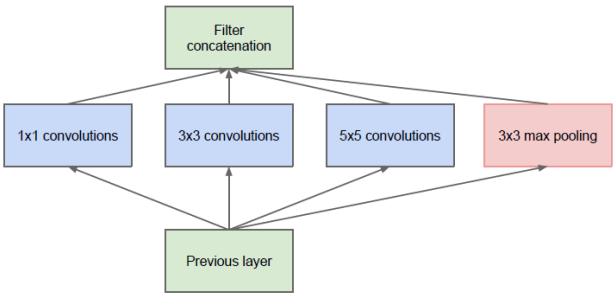

Figure 10: Basic visual description of an inception layer

2.4.3 Data generators

Given the huge amount of data we used data generators to feed our model during the train (with both train and validation data).

The generator, for each image in the current set performs the following steps:

- Extract from a dictionary the current image features vector which will be the first input of the network;
- Create a sequence of integers – using the proper dictionary (see 2.4.1) – with the words in the current image caption which are also inside the dictionary;
- Starting from that sequence, the first word (inside a padded vector) is used as seed input text in the model and the second word (in form of categorical vector) is used as ground truth; then, the first and the second words together are used as input and the third one is used as ground truth and so on. Thus, each feature vector, with different seed text and different ground truth is feeded to the network with a batch of length length_of_its_caption - 1 seed texts and ground truths.

2.4.4 Decoder

In order to improve the performances, we have modified the structure that the tutor suggested us as a starting point [3]. As can be seen in figure 11, our decoder is composed from the following blocks:

- Two input layers, one for the feature vectors and one for the seed text produced by the generator;
- The feature vector passes through a dense layer with a ReLU activation function with the main purpose of changing its size to match that of the text features;
- The text vector passes first of all through an Embedding layer – which is not trainable since it uses the weights of GloVe – which enables us to convert each word into a fixed length vector of defined size;
- The output of the embedding layer is given as input to 3 sequential LSTM layers (see figure 12) which use also dropout and BatchNormalization to prevent respectively overfitting and the vanishing gradient problem;
- At the end 2 Dense layers – the former with a ReLU activation function and the latter with a softmax activation function – are used to put together the outpus of the 2 branches and to give in output a probability distribution over the words in the reduced vocabulary.

We tried a lot of configurations before ending up with this one, for example we tried to use more dense layer at the end, to use more dense layers to process the features vector and we tried also to use Bidirectional LSTM but all these approaches did not improve the performances (or did it by such a small factor that it did not justify the overhead of resources and time).

2.4.5 Predictions

To make our predictions, we used a greedy algorithm. Basically, a greedy algorithm orders all the possible values in ascending or descending order and than takes the first one, which is the best one in that moment.

To do so, we make a first prediction giving to the model the tuple *(test_img_feature_vector, startseq)* as input, from the output probability distribution we take the most probable word which will be concatenated to *startseq* to form a new input for a new prediction. We repeat these steps until the next most likely word is *endseq*.

3 Results and Discussions

The chosen evaluation methods were BLEU, precision, recall, f-score.

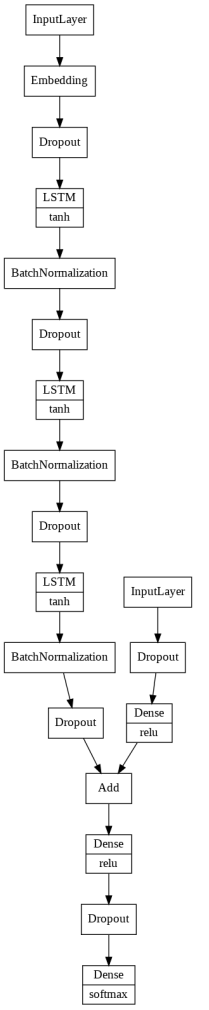

Figure 11: Structure of the decoder

These results are heavily influenced by the modifications that were performed on the original dataset in the pre-processing steps due to our hardware limitations (see section 2.2). In fact, due to the complex architecture of our model and original image resolution and overall dataset size, training time was unbearably long, so these modifications were mandatory in order to complete a full training session of 20 epochs in a manageable amount of time.

The final training session was 20 epochs long, which is much less than what is used in the state of the art, but the computational capacity at our disposal limited this phase. Figure 14 shows the loss function (set to categorical cross entropy) evolution. As can be noticed, the model showed little signs of overfitting, that were also tamed by adding dropout layers. Overall, the training

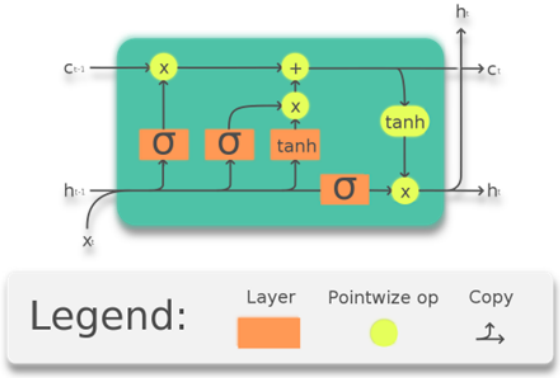

Figure 12: LSTM's working principle

of the model was pretty satisfactory.

Figure 15 shows a comparison between predicted captions and ground truth. The labels are cleaned for manageability purposes. As we can see by the figure, the predicted labels are longer and often present the repetitions of the same words: this is because the network predicts each word one by one, choosing the most probable one each time, until the word with highest probability is the end token. Tab 4 shows the scores of the afore mentioned images.

Compared to them, average scores are of course lower, with a BLEU score of 14.28%, precision of 0.16, same for recall and f-score. This is probably due to the word repetition pattern at the end of each predicted label, that lowers the overall scores among the entirety of the test set.

Table 4: Scores of images in Figure 15

	BLEU	precision	recall	f_score
image 1	58.30%	0.63	0.58	0.60
image 2	50.00%	0.77	0.50	0.60
image 3	57.14%	0.66	0.57	0.61

4 Overall discussions

This was an exploratory work aimed at studying and understanding the underlying structure and methods of medical image captioning, while also applying Natural

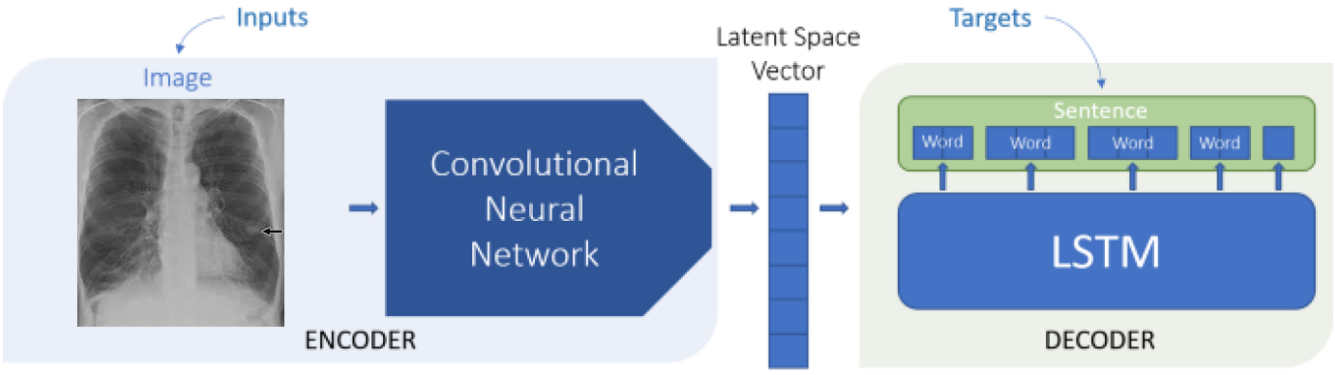

Figure 13: Implemented model

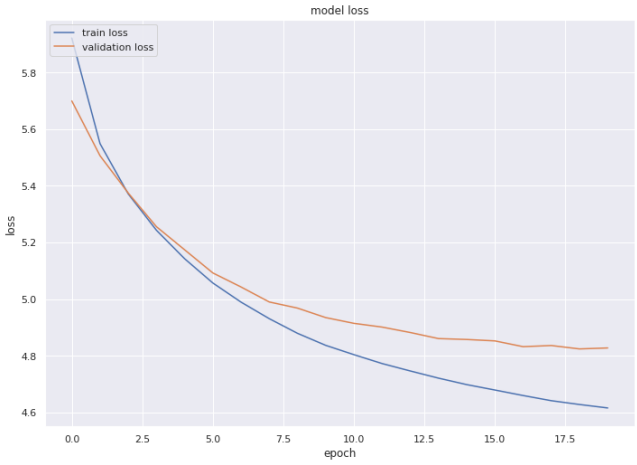

Figure 14: Training evolution of the model

Language Processing methods to this specific domain to obtain a broader and clearer visualization of it.

Our results were, as expected, worse than the state of the art,which for example presents a BLEU score range that goes from 25.00% to 51.00% [2][1], also due to the heavy dataset reductions and resizing transformations. In order to improve the performances of our model, a different pre trained network could be adapted to this task, along with the use of transformers, which represents the most recent architecture (called sequence-to-sequence) for NLP tasks. Another improvement could be trying different methods to avoid word repetitions inside the predicted labels.

5 Conclusions

Image captioning is a field of research which gained much interest during the last years, with many research groups focusing on it. In our work we studied this field and provided our take on it, trying to implement widely used methods. The dataset at our disposal was the ROCO dataset, which did not receive much attention from literature, probably due to the fact that it presents many limitations, making its evaluation problematic all things considered. We encountered many issues with it, and in order to make the samples contained in it ready to be fed to our model we had to invest a big share of our time, not being able to actually focus on improving the model in an effective way.

However, we were able to study and analyze the world of image captioning in an immersive way, searching for various implementation methods and ways to solve the problems that this task presented to us along the way. We found this world very interesting and captivating, due its possible implementations that can be applied to many fields of research and improve our lives.

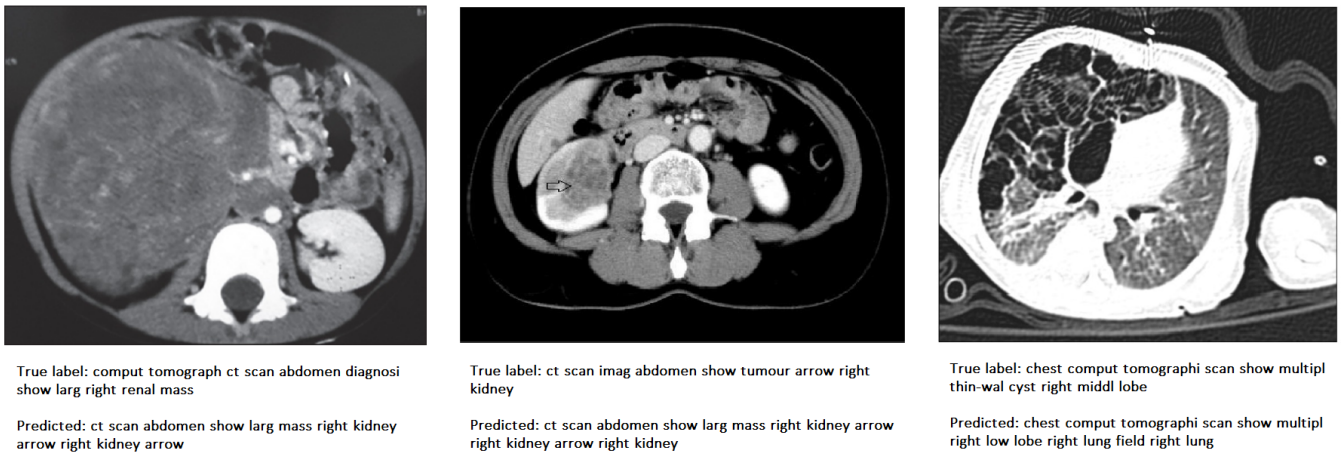

Figure 15: Comparison of three test sample's true captions and generated ones

References

[1] Djamila Romaissa Beddiar, Mourad Oussalah, and Tapio Seppänen. "Automatic captioning for medical imaging (MIC): a rapid review of literature". In: *Artificial Intelligence Review* (). URL: https://doi.org/10.1007/s10462-022-10270-w.

[2] Djamila Romaissa Beddiar et al. "ACapMed: Automatic Captioning for Medical Imaging". In: (). URL: https://www.mdpi.com/2076-3417/12/21/11092.

[3] *Captioning Images with CNN and RNN, using PyTorch*. URL: https://medium.com/@stepanulyanin/captioning-images-with-pytorch-bc592e5fd1a3.

[4] John R. Allen Darren M. West. "How artificial intelligence is transforming the world". In: (1) (Jan. 2018), p. 1. URL: https://www.brookings.edu/research/how-artificial-intelligence-is-transforming-the-world/.

[5] *Inception V3 Model Architecture*. URL: https://iq.opengenus.org/inception-v3-model-architecture/.