CAP 6640 Final Report

MICHAEL FIELDER, University of Central Florida, USA
MICHAEL CRUZ, University of Central Florida, USA

ACM Reference Format:
Michael Fielder and Michael Cruz. 2023. CAP 6640 Final Report. 1, 1 (April 2023), 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 ABSTRACT

Transformers and other Encoder-Decoder architectures have been extremely significant in the progression of the field of Natural Language Processing and Computer Vision. These technologies have broadened the scope of the proposed capabilities of data science in general. In particular, the medical field has benefited unimaginably by using technology such as these. This experiment was inspired by these previous works. However, there's a natural bottleneck that occurs at the intersection of deep learning and the medical field. Specifically, HIPPA laws restrict the amount and type of information that can be used for research purposes, especially outside of the medical community. Any sensitive information that can be used to identify a patient, or is not publicly available can't be used in deep learning models. This restriction has limited the incorporation of deep learning within medical spaces. However, this experiment aims to solve the dilemma. By training multiple models, each of which is defined by a CNN Encoder and an LSTM Decoder, a global model can be obtained from the various independent models to create a more accurate model for collective use. Consequently, the amount of data needed to train each model would be decreased along with the computation resources needed for further training.

2 INTRODUCTION

Natural Language Processing is a discipline of deep learning that's constantly evolving. Data scientists are constantly seeking ways in which to refine techniques as well as create new methodology in efforts to derive new technology. Due to the ever-increasing popularity of deep neural networks, there has been an increase in research concerning image neural networks and the endless possibilities of technology that can utilize these types of models. Models such as *ResNet* have demonstrated the capabilities of such neural networks in image recognition, feature extraction, and many other crucial deep-learning tasks. However, Convolutional Neural Networks and their applications have not been fully researched and therefore have not properly been utilized in various sectors. In particular, the medical field has a plethora of areas in which deep learning models would vastly improve current practices. For instance, the process of creating clinical reports for each patient is a tedious, but imperative task that all doctors must complete in order to maintain proper medical records. These documents are used by all medical professionals to properly diagnose and treat patients, so the information recorded must be accurate and up to date. Yet, the accuracy of said documents is completely reliant on the accuracy of the hospital staff. This flaw often leads to inaccurate medical records as well as misdiagnoses. If the process of clinical reporting was partially or fully automated utilizing deep learning, human error within medical documentation could be dramatically reduced. Furthermore, it could restore time and resources to doctors which would allow doctors to see more patients and ensure that all medical professionals have the correct information to accurately diagnose their patients.

3 RELATED WORK

This section focuses on the two types of prior research related to our project, clinical report generation and federated learning in a medical setting.

3.1 Clinical Report Generation

Clinical report generation is a well-researched field that aims to automatically generate reports for medical imaging data. The objective is to provide practitioners with a valuable tool that facilitates the diagnoses of medical imaging data such as radiology reports. According to [3], there are a number of properties in clinical report generation datasets that make it difficult to accurately generate accurate reports. These include things like dataset bias, where sentences outlining normalities greatly outweigh those highlighting potentially dangerous abnormalities. This can be seen as an imbalanced dataset which can severely impact the performance of the model. One solution to this problem is outlined in [5] which uses an architecture called *AlignTransformer*. This model combines the two modules: align hierarchical attention and multi-grained transformer to overcome the problems associated with dataset bias and achieves good results MIMIC-CXR and IU-Xray. In general, many of the advancements in the field revolve around the implementation and alterations to Transformer architectures which have been shown to achieve good results in the area of Natural Language Processing. This can be seen in papers such as [2], which uses an RNN encoder and Transformer based decoder.

3.2 Federated Learning

Medical Imaging has a long-standing relationship with Federated Learning architectures. This is due to the sensitive nature of medical data which prevents it from being easily distributed. [4] is a survey that outlines some of the advancements made toward training image classifiers on different sets of medical data. Oftentimes, this involves using one of the most widely-implemented Federated Learning networks, FedAvg, to handle the distribution and training of multiple

Authors' addresses: Michael Fielder, University of Central Florida, Orlando, FL, USA, MichaelFielder@Knights.ucf.edu; Michael Cruz, University of Central Florida, Orlando, FL, USA, mbcruz96@Knights.ucf.edu.

XXXX-XXXX/2023/4-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

models. The general outline is to create multiple clients which are trained individually and then manipulated in some way to create a global model. Some examples of experiments on different medical imaging datasets include brain tumor segmentation [6], liver segmentation [7], and lung cancer segmentation [8]. In all of these examples, a *FedAvg* backbone was used for the Federated Learning aspect of the network.

4 PROBLEM STATEMENT

The primary objective of the experiment is to research different methodologies with respect to clinical report generation in a federated learning setting. Clinical report generation seeks to take medical imaging with various modalities (such as chest x-rays) and generate readable reports that can be used by medical professionals as a preliminary diagnostic tool. The main advancement proposed by this experiment is the inclusion of multiple clients in a federated learning environment in an effort to train the model in a manner that protects the confidential information of each patient. In this manner, the experiment aims to create a technique that would allow multiple hospitals to participate in providing data for medical models, allowing for the construction of a superior global model, while simultaneously upholding HIPPA laws.

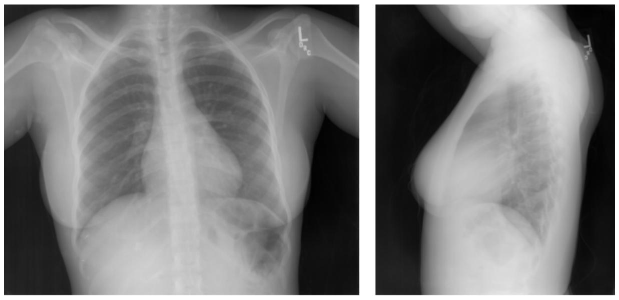
5 TECHNIQUE

The experiment utilized various techniques in order to accomplish creating the model. The library *PyTorch* will be used to train the model, as it affords us the most flexibility when altering our model architecture. There were several methods used to pre-process and store the images and reports, but the main library utilized for corpus pre-processing was the `nltk` library for tokenization in conjunction with various `torch` libraries to load, shuffle, and batch the data. The `resnet` pre-trained CNN was used to train the model on the medical images. As for the model itself, the `Modules` library was used to implement an Encoder and Decoder class. The Encoder class accepts image tensors as input, while the Decoder class accepts vocabulary indices to train an LSTM. Architecture specifics will be discussed in the next sections.

5.1 Dataset

The model uses the *IU X-Ray* dataset, which is a collection of Chest X-Rays and accompanying medical reports. For a large portion of the reports, there are both frontal and lateral images. All lateral imaging examples are removed from the dataset for simplification as well as to reduce possible complications. This resulted in 3691 examples of frontal X-ray images, which is a sufficiently sized dataset for the scope of this experiment. The data was initially obtained from Kaggle, but further research yielded other models using the same data. Since the images were already extracted and stored, those pre-processed images were used in this project instead. After experimentation, we concluded that excluding all lateral imaging might have had adverse affects on our models. Excluding the images may have decreased the overall accuracy of the model by decreasing the amount of data the encoder had to process. **Figure 1** shows one example of the data.

The dataset includes projections and reports CSV files in addition to the X-rays associated with each report. All files were initially downloaded to local storage. Out of necessity and lack of storage, all the data was migrated to OneDrive for communal use. Excluding all Lateral imaging, all X-rays were converted into torch tensors, and then pre-processed for the deep learning. After processing, the tensors were normalized to attempt to mitigate over-fitting or under-fitting of the data. Unfortunately, the image tensors still seem to be over-fitting, but this has not caused significant problems with the model implementation. The CSV files were loaded into pandas dataframes for pre-processing and to create a Bag of Words representation of each report. Each report and image have a unique ID number, so dictionaries were created to map image names to ID numbers and vice versa.



COMPARISON None.
INDICATION Positive TB test
FINDINGS The cardiac silhouette and mediastinum size are within normal limits. There is no pulmonary edema. There is no focal consolidation. There are no XXXX of a pleural effusion. There is no evidence of pneumothorax.
IMPRESSION Normal chest x-XXXX.

Fig. 1. One image-report pair from the IU-Xray dataset

As for the physical reports, a rudimentary pre-processing pipeline was developed in order to clean the text. Since this project seeks to emulate the writing of medical professionals as closely as possible, stop words were not removed from the corpus in an attempt to replicate authenticity. Currently, the pipeline involves making all text lowercase, removing non-ASCII characters and punctuation, and finally separating words that are connected with no spaces inadvertently. It was observed that many words within the corpus were actually a conjunction of two words that must have been erroneously transcribed when transferring the doctor's clinical report to a digital medium. One of an example of two conjoined words is "thickeningsclerosis" which should be "thickening sclerosis". Initially, the library `wordninja` was used to separate these words, but the library failed to accomplish the goal. For example, it splits "thickeningscars" into "thickenings car". It was determined that the best method to accomplish the separation was to manually separate the words, and join them back together with a space in between them. Finally, all data in the corpus which previously contained sensitive patient information was redacted and replaced with XXX's, so the final step of the pipeline was the removal of the redacted text. Any word that contains two or more X's is removed from the document using a regular expression to identify them.

In order to a word word-level text generation, a vocabulary must be constructed. We used `CountVectorizer` to construct the dictionary which is composed of the words present in the reviews. Currently, we have 1700 words in the vocabulary. This was the original way it was done, but after looking into what PyTorch has to offer in terms of tokenization, this seemed more suitable for the experiment. After tokenization, two dictionaries were created: a dictionary that indicates the index given a word, and a dictionary that indicates the word given an index.

5.2 Model Architecture

The model used for each individual client is an encoder-decoder based network where the encoder is a Convolutional Neural Network architecture and the decoder is some Recurrent Neural Network that utilizes LSTMs, which is used to generate text from a given vocabulary. **Figure 2** shows a high-level representation of how our model operates. For the encoder, the pre-trained `ResNet` model was used and fine-tuned for the medical images used in the experiment. More specifically, we will be using the `resnet152` architecture from the `torchvision` library. After removing the classifier, the model returns a latent representation of each image which can be used by the decoder to generate clinical reports. Our final model was influenced by [1] and [9] which use a similar CNN-RNN based network to achieve their results.

5.2.1 Encoder. By using `resnet152` as the pre-trained model for the encoder, it was a simple process to fine-tune the model for the downstream task of medical report generation. The images used for the experiment had to be altered in order to fit the expected input of the pre-trained model. Furthermore, the `Relu` activation function was added to encoder since there are multiple classes of images being predicted. Finally, the dropout rate was set to 0.5 in order to reduce over-fitting during training. The goal of the encoder is to create a linear representation of each image tensor to embed within the deep network to later be used by the LSTM decoder.

5.2.2 Decoder. As for the decoder, the original architecture was based on the Transformer's architecture, as it's standard practice for encoder-decoder downstream tasks. However, due to the limited amount of time and resources for this project, an LSTM was chosen as the decoder for ease of training. In a federated learning setting, the model would have to train at minimum three Transformer decoders. Therefore, an LSTM-based decoder was the natural choice. Additionally, a linear layer and embedding layer were added to the decoder.

5.2.3 Hyper-parameters. The model was implemented using a batch size of 64, which was chosen in order to decrease the amount of time required for training. Other hyper-parameters were also tuned to produce the most accurate results. Two layers were added to the input layer: an embedding layer of size 256, and a hidden layer of size 128. The learning rate was set to 3e-3 to decrease the step size when calculating the gradients for the layer weights. These parameters resulted in the best-generated text output.

Each model was trained for 10 Epochs. For the experiment, the cross-entropy loss function was to calculate the loss after each epoch. The optimizer used was Adam, with the previously stated learning rate. There were 2878 training examples and 319 test examples which were randomly shuffled and batched. Although these were the parameters that caused the best text generation, they certainly weren't the best parameters for the model in all. The text generated by the test samples were not exactly what we were expecting. However, there was no more time to tune the hyperparameters for this experiment.

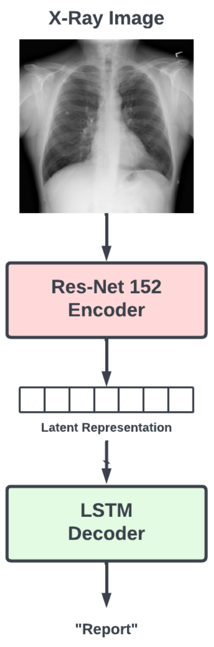


Fig. 2. Overview of model architecture

Figure 2 Demonstrates the overall architecture used to train the model. Each image tensor was passed to the pre-trained encoder after being pre-processed to the expected input of the pre-trained model. Unlike the diagram, which uses the DesNet-121 pre-trained image encoder, the model of this experiment utilizes the `ResNet` pre-trained encoder as previously stated. As each model is processed through the encoder and their losses are calculated, part of the output of the encoder along with the decoder input is passed to the decoder. For this experiment, the decoder is an LSTM instead of a traditional Transformer decoder. The encoder produces a linear representation of the image tensors which are embedded into space. Said linear representation is passed to the LSTM decoder.

5.3 Federated Learning

The key novelty of the experiment involves using Federated learning to optimize the models while preserving patient anonymity. In particular, the FedAvg architecture was implemented as the basis for the experiment. This architecture is relatively simple and involves creating multiple clients(i.e. models) and training them individually with different training sets. Once trained, each client's layer weights are sent to a global model. The task of the global model is to take the individual weights of each model at each layer and average the weights. The global model then returns the averaged layer weights back to each client, ensuring each client will have more accurate weights. **Figure 3** shows an high-level overview of our architecture.

In the diagram, each client is a model trained on medical diagnostic data. The black lines indicate the model weights being sent over the network to a federated server with the goal to average all of the weights at each level of the clients. The red lines demonstrate how the backpropogation process of the averaged weights back to each client

In real-world practice, each client would be an entire medical facility that would collect images, reports, and other diagnostic documentation and train their models with the cumulative data. The functionality of the global model would be implemented via a dedicated server, as described in **Figure 3**, that would intermittently collect all of the medical data, average the layer weights, and backpropagate those weights to the facilities. Because only weights are involved in creating this type of machine learning model, HIPPA regulations would still be enforced and patient confidentiality would be protected. Although a minor change from previous models, the use of Federated learning dramatically alters the type as well as the amount of medical data that can be trained and stored for use in the medical field. These new models will have numerous applications and will inevitably help the overall accuracy of diagnostics.

For the experiment, three clients were created, each with a subset of images and report data. The clients were individually trained on their respective datasets for 10 epochs each, and their Cross-entropy was measured to calculate the loss of each model. Then, as previously discussed, each layer's weights were averaged, and the resulting layer weights were propagated back to all of the clients and then tested again. Once training had concluded, the models were tested for accuracy in correctly generating medical reports given a specific X-ray image.

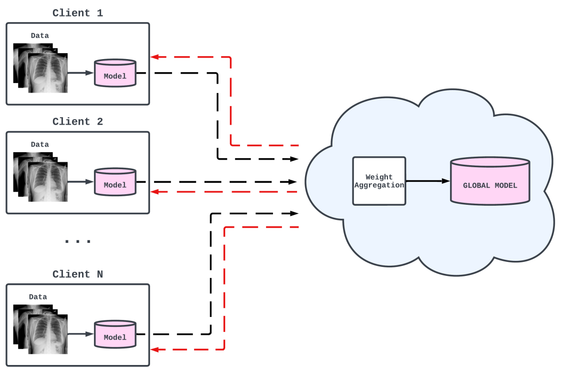


Fig. 3. An example of our federated learning architecture

6 RESULTS

During experimentation, all five models were successfully trained individually using the previously described architecture. As previously stated, there was an error involving the image data being over-fitted while training, but the issue was not significant enough to alter the image tensors. From pre-processing and analyzing the dataset, we concluded that implementing an LSTM decoder would be more suitable for the scope of the experiment. Initial research indicated that a traditional transformer encoder-decoder architecture would produce the most accurate model, but further research has led to the determination that a decoder with an LSTM-based architecture would be more beneficial. In addition, pre-processing the images uncovered that using both Lateral and Frontal images from the dataset was too exhaustive with regard to the resources and computing power currently available. Google Colab only allocates 25GB of RAM per session without crashing, therefore it was imperative to decrease the number of images in the data. Consequently, all Lateral images were removed from the dataset.

In terms of formal experiments, **Figure 4** shows the loss graph for training one model for 25 epochs. As you can see, the training process returns a desirable loss,

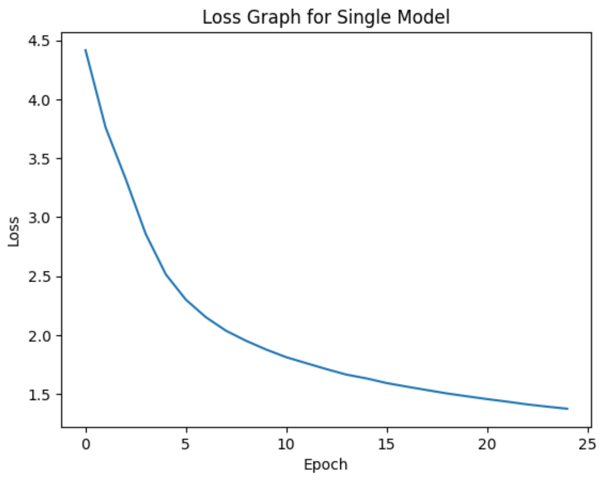


Fig. 4. Loss graph for training one model in non-Federated setting

However, when testing the model on test data to evaluate its results, it exhibited severe over-fitting, defaulting to one of several outcomes. One example that appears frequently is,

The heart is normal in size. The mediastinum is unremarkable. The lungs are clear.

This kind of sentence will be used for multiple different report generation, and will sometimes be used for reports where there is an abnormality present (when the lungs are not clear or the heart is not a normal size). Even when randomly sampling the generated token, it seemed to collapse towards these same few examples. For comparison, here is an example of the ground-truth report to compare,

Cardiac size, mediastinal contour, and pulmonary vascularity are within normal limits. The right heart XXXX appears obscured, and there are streaky right medial basilar airspace opacities, possibly due to airspace disease or atelectasis. Otherwise, no focal consolidation, pleural effusion, or pneumothorax. The visualized osseous structures appear intact.

When comparing it to the generated text by the model, it is clear that the generated text is missing a lot of important information. This includes most of the information concerning the actual disease or ailment shown by the test image. Most of the generated text concerned normal projections seen by the doctor. This is another issue incurred due to the overfitting of our data. Although this isn't conclusive, it would explain the incorrectly generated text.

7 DISCUSSION

Initial testing on the first round of training yielded a model which could discern markers and generate accurate medical reports. Each report generated contain two sections. The first of which is for the doctor's initial impressions which indicated the results of preliminary medical testing. The generated text for this section is very similar for every test set. However, this is most likely due to the nature of the sets. On average, most of the impressions are normal for the majority of testing criteria. After analyzing the raw data, this section of the text seemed to be generated correctly. The second section of generated text was for findings and described the X-ray images associated with the patient. This section was different for every test set. Consequently, this will probably be the section of text used to generate the clinical report.

The collapse of our model is likely due to a complex combination of various things. As explained in the *Related Work* section, there is an inherent data imbalance present in these models where there are more examples of sentences describing normalities than those describing abnormalities. Our model could be over-fitting to the normal examples to achieve the highest possible accuracy. This is a common problem in long-tailed datasets where the model "maximizes" performance by guessing that most things will be normal. This could also be directly tied to the simplicity of our model. There is a good reason why most state-of-the-art text generation models these days utilize Transformers, and it is because of their ability to create deep connections between words as a result of self-attention modules. Unfortunately due to the time frame of the assignment and the resources available to us, we weren't able to test and see if a model like this would be able to achieve better results and create better connections with words. It seemed as though no combination of different embedding sizes, hidden sizes, epochs, or a number of LSTM layers was able to achieve a desirable result.

It is worth mentioning that it is worth doing more testing for a single model than in a federated learning setting. This is because a model should perform the same and most times better than one trained on multiple models. Since we weren't able to successfully train a model in a non-federated setting, it was not worth testing it with multiple clients. However, that's not to say the model wasn't fully trained for all epochs, only that the model did not generate text that was up to par with medical record standards. With small adjustments and a little more time, we believe this model would have succeeded in the overall goal.

After discussing Assignment 3, we noticed a similar thing occur during the training of that network when compared to this one. There was some over-fitting present in the training of that one which was most likely due to not having enough samples to train on. This could make sense considering that the IU-Xray dataset has only several thousand data points while other similar datasets, like MIMIC-CXR, contain over 100,000 examples.

One possible avenue that we could've explored was the implementation of *attention* into our model which could have improved its performance. As described earlier, Transformers have had incredible success in encoder-decoder architectures. Initially, we felt as though our model would not be complex enough to warrant the use of Transformers. In retrospect, it might have helped in the aforementioned abnormalities observed through testing. However, our model has many similarities to a Transformer, but with the limited amount of time for the project, it was deemed impractical to switch to an attention based deep network. Unfortunately, we weren't able to run any experiments using this mechanism in our LSTM decoder.

Even with all of this in mind, however, it's possible that there is something fundamentally wrong with the model that is causing it to over-fit. It would require more extensive testing to diagnose. The problem could have come from many sources, which is why it was so hard to pinpoint the main issue. Most likely, the architecture we decided to implement was not conducive to a federated learning application. More test are necessary to conclude where the real issue occured.

8 CONCLUSION

There were many challenges that we encountered while conducting our experiment. For instance, the nature of the data caused an overfitting issue which we were unable to overcome within the time constraints. Furthermore, the skewed data which represented more normal projections than irregular projections caused the generated text to be similar between test samples, a result that we did not expect. Finally, the intricacies of the implementation of Federated learning was not fully calculated in the length of time we had to complete the full project. While we were not able to achieve the end goal of successful clinical report generation for our project we still learned a lot about the text-generation through the necessary background research that needed to be constructed and the complexity that is present in a lot NLP tasks. We were able to use the techniques we learned in class such as creating a vocabulary, designing a model using what we learned (LSTMs), and text pre-processing on a complex dataset. Applying these fundamental techniques to a larger-scale project with various moving parts definitely illuminated the job of a Data Scientist. Although the experiment was technically a failure, with further research, we hope to finish this project with a positive result.

9 WORK COMPLETED

The next section outlines the completed tasks by the group. Please note that in some aspects of the project, we both worked on the same section of the project. The following is the procedure used in the experiment:

- Linked the Google Colab to OneDrive
- Imported all data (projections, reports, and images) into OneDrive
- Imported reports and projections into dataframes and NumPy arrays
- Pre-processed images into a dictionary of tensors with the file name as the key
- Processed the reports into BOW representations

- Pre-processing Images (Resizing, removing the lateral views, etc.)
- Exploration of encoder(Testing the input size, removing the classifier, finding models)
- Text Pre-processing (Removing unnecessary characters, splitting words, removing NaNs etc.)
- Vocabulary generation (Exploring and testing different types of tokenizers including)
- Trained model on various hyperparams
- Tested the model on a test set
- Evaluated the accuracy of the generated text using BLEU (Although was not useful due to results of model)

10 REFERENCES

(1) github.com/tengfeixue-victor/Medical-Report-Generation
(2) arxiv.org/pdf/2107.02104.pdf
(3) arxiv.org/pdf/2201.09873.pdf
(4) arxiv.org/pdf/2208.03392.pdf
(5) arxiv.org/pdf/2203.10095.pdf
(6) `link.springer.com/chapter/10.1007/978-3-030-11723-8_9`
(7) https://arxiv.org/pdf/2205.11096.pdf
(8) https://www.nature.com/articles/s41598-022-05539-7
(9) github.com/aladdinpersson/Machine-Learning-Collection