TÉCNICO LISBOA

Multi-Task Training of Transformer Language Models for Processing Radiology Reports

Nuno Miguel Branquinho Infante

Thesis to obtain the Master of Science Degree in

Biomedical Engineering

Supervisor(s): Prof. Bruno Emanuel Da Graça Martins
Dr. Nuno André da Silva

Examination Committee

Chairperson: Prof. João Miguel Raposo Sanches
Supervisor: Prof. Bruno Emanuel Da Graça Martins
Member of the Committee: Prof. Francisco Moreira Couto

October 2021

To my brother

Declaration

Declaro que o presente documento é um trabalho original da minha autoria e que cumpre todos os requisitos do Código de Conduta e Boas Práticas da Universidade de Lisboa.

I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

Preface

The work presented in this thesis was performed at INESC-ID research center and Instituto Superior Técnico (Lisbon, Portugal), during the period March-October 2021, under the supervision of Prof. Bruno Emanuel Da Graça Martins and Dr. Nuno André da Silva.

Acknowledgments

I would like to thank professor Bruno Martins and Dr. Nuno André da Silva, for giving me the opportunity to pursue a topic that was dear to me. In particular, I would like to thank Prof. Bruno for the incredible patience and dedication to supervise my work closely during this thesis. His guidance and teachings helped me find a path for my professional future and goals. To my parents and family for being so supportive of my dreams and ambitions. To all my colleagues at Unbabel, to João Maria Janeiro and my incredible manager Catarina Farinha, who helped me understand the basis of applied NLP and that we should never give up to challenges that life presents us.

Resumo

Nos últimos anos, modelos pré-treinados baseados na arquitetura de ”Transformers” têm atingido resultados verdadeiramente impressionantes no que toca a tarefas de processamento biomédico de linguagem. Em particular, a área de radiologia ganhou especial relevância. Os relatórios de radiologia têm por norma o fim de captar duas seções especialmente relevantes, a descrição e as observações de um determinado exame de radiologia. Neste trabalho propomos um modelo de linguagem multitarefas baseado no ”Text-to-Text Transfer Transformer”, conhecido como T5. Realizámos uma abordagem de treino em diversas tarefas como geração de texto, classificação e inferência, relacionadas com relatórios de radiologia. O modelo multitarefas proposto obteve resultados promissores, próximos do estado da arte, nas tarefas de sumarização, inferência de linguagem natural, classificação de semelhança semântica e deteção de paráfrases. Os resultados obtidos confirmam o potencial de modelos multitarefas e de transferência de aprendizagem num contexto de processamento biomédico de linguagem natural.

Palavras-chave: Processamento Biomédico de Linguagem Natural, Relatórios de Radiologia, Modelos de Linguagem baseados em Transformers, Aprendizagem Multitarefas, Transferência de Aprendizagem

Abstract

Pre-trained language models based on the Transformer architecture have achieved impressive results in biomedical natural language processing tasks. One specific area that has raised interest in recent years is radiography, where textual reports are usually generated to describe findings and impressions from radiography exams. There are several practical applications related to the processing of these documents, such as automated classification, summarization or extraction of key findings. This work proposes a multi-task language model based on the Text-to-Text Transfer Transformer, commonly known as T5, that was trained on different text generation, classification, and inference tasks involving text from radiology reports. We discuss the data pre-processing and the model training strategy together with its evaluation. The proposed multi-task model achieved very good results, close to state-of-the-art results from models that were individually trained for the summarization, natural language inference, semantic text similarity, and paraphrase identification tasks. The results confirm the potential of multi-task and transfer learning for biomedical natural language processing.

Keywords: Biomedical Natural Language Processing, Radiology Reports, Transformer-Based Language Models, Multi-Task Learning, Transfer Learning

Contents

List of Tables

List of Figures

Nomenclature

Acronyms

C4	Colossal Clean Crawled Corpus
EHR	Eletronic health record
GPU	Graphics Processing Unit
LCS	Longest Common Sequence
LSTM	Long Short-Term Memory Network
MLP	Multi-Layer Perceptrons
NLI	Natural Language Inference
NLP	Natural Language Processing
PD	Paraphrase Detection
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
ROUGE	Recall Oriented Understudy for Gisting Evaluation
RRS	Radiology Report Summarization
SOTA	State-Of-The-Art
STS	Semantic Text Similarity

Chapter 1

Introduction

1.1 Motivation

The field of Natural Language Processing (NLP) has seen significantly advances in the past couple of years. The release of the paper “Attention Is All You Need” [1] has revolutionized the way we construct models and introduced a new type of neural network architecture, the Transformer, that is now vastly used. The paper demonstrated that an architecture solely based on attention layers could produce better results than the standard approaches that used recurrent neural networks.

A new generation of models based on Transformers has appeared since then. These include BERT, T5, BART, and GPT-2, which consequently opened a wide range of new possibilities regarding applications of neural models for language understanding and generation. Such models are today’s standard for most tasks and applications .

One of the areas where such models have raised significant interest is the medical domain. This area raises several challenges regarding data, since acquiring large amounts of labeled medical data tends to be very hard due to the cost associated with expert labeling and the time required for such annotations. Few medical datasets are publicly available, and the quality of such data tends to vary significantly. Anonymization is also a very important topic when referring to medical data, where real patient data is being handled. Public tasks such as MEDIQA [2], contribute to reducing the shortage of training and testing data by releasing new datasets.

Radiology and radiography exams are particularly interesting in terms of possible natural language processing application. This is mostly related with the textual nature of the reports describing radiology studies. Radiology reports also present a consistent structure that is usually used to describe the impressions and findings of an exam. The practical applications of processing information on these documents can be automatic classification and labeling, summarization and extraction of key findings, or disambiguation of text. Chest X-ray is the most common imaging study performed worldwide. A large and publicly available dataset is MIMIC-CXR [3], i.e. a radiology report dataset that contains a total of 227,835 radiography studies of chest x-rays.

Although there is a relatively good number of radiology report examples publicly available, very few

work is performed on radiology tasks that are not summarization or information extraction. This creates an immense gap between data available and possible uses for it. In particular, data that can be used for inference tasks like natural language inference or paraphrase detection, is not processed or created.

A considerable gap between currently available models and their proven application in the medical field still exists. Again, due to the circumstances mentioned above.

1.2 Thesis Proposal

This work is focused on exploring the application of recent language models, using the Text-to-Text Transfer model to demonstrate the capabilities of such architectures when applied to the medical domain, and radiology in particular. The decision to use this model is supported by recent literature that indicates the potential of T5, and the fact that pre-trained T5 models currently hold state-of-the-art [4] results in several medical tasks such as natural language inference.

We devised several experiments using the MIMIC-CXR dataset of radiology reports, where we intend to use multi-task training strategy for the following tasks: summarization, natural language inference, paraphrase identification, semantic text similarity, and classification. The performance of the resulting models will be compared to current state-of-the-art models [4] with similar training strategies and training tasks.

This document is organized as follow. We start by providing a thorough description on all the important literature and concepts for our background on Chapter 2. This was the base for our proposed approached that can be found on Chapter 3, where we detail the proposed strategies, models developed, training strategies and data augmentation approaches. On Chapter 4, we present our experimental results and highlight the main characteristics and conclusions. Finally, our conclusions and remarks about our work are presented together, with remarks to lay the foundation for further exploratory work on this field, on Chapter 5.

1.3 Summary of Contributions

On our work we present a multi-task learning model, trained with a task specific training scheme. Our model takes full advantage of the training datasets characteristics and achieves promising results for the tasks of summarization, natural language inference, paraphrase detection and semantic text similarity on radiology reports. We demonstrate the potential of multi-task models and multi-task training schemes for radiology related tasks and hopefully, by doing so, we encourage further exploration on this field.

Chapter 2

Background

In this chapter we will present the underlying knowledge of the machine learning models used on the work. We introduce the concept of neural networks, attention layers and finally we give an overview of the transformer architecture.

2.1 Introduction to Neural Networks

The concept of Artificial Neural Networks is a very intuitive one, it represents an approach based on models inspired by the way the human brain and biological neural networks actually process information. The brain's computational unit is the neuron, which inspired McCulloch and Pitts [5] to propose a simplified computational element that was based on propositional logic.

This simple unit is commonly know as Perceptron, which can be formally define in the following way:

$$perceptron(x) = \sigma(b + \sum_i w_i x_i) = \sigma(w \cdot x + b) \tag{2.1}$$

This unit receives a set of n input values represented by x, to which it applies a set of weights, represented by w. After this operation, an additional value b in summed, turning this computation into an weighted sum. This final term b is called the bias term. This expression can also be described using a matricial notation that can be seen in the left part of equation 2.1.

After this computation, a non-linear function $\sigma()$ is applied to the perceptron output that is normally referred to as an activation function. Common activation functions are the sigmoid, the tanh and rectified linear ReLU.

Using the idea of a basic unit similar to perceptrons, we can construct more complex architectures like Multi-Layer Perceptrons (MLP). MLPs like feed-forward networks, are multi-layer networks, which units are connected but only pass their outputs to an higher layer and never back to lower layers, hence the name feed-forward. Simple feed-forward networks are composed by only three layers, an input layer, an hidden layer and an output layer. The input passed by the input layer is received in the hidden layer, where the weighted sum described in equation 2.1 takes place. The output generated is them passed

to next layer after the application of the activation function $\sigma()$. This continues until the last output value is passed to the output layer.

2.2 Gradient Descent

The parameters of our model, that we may refer to as θ (this is the weight matrix w and our bias factor b) can be optimized by applying a common strategy called Gradient Descent. This algorithm is intended to find the optimal parameters for our model by minimizing what is called a loss function. This function $L(\widehat{y}, y, \theta)$ can be defined by the difference between predictions given by the model $\widehat{y}$ and the expected value y, for the model parameters θ.

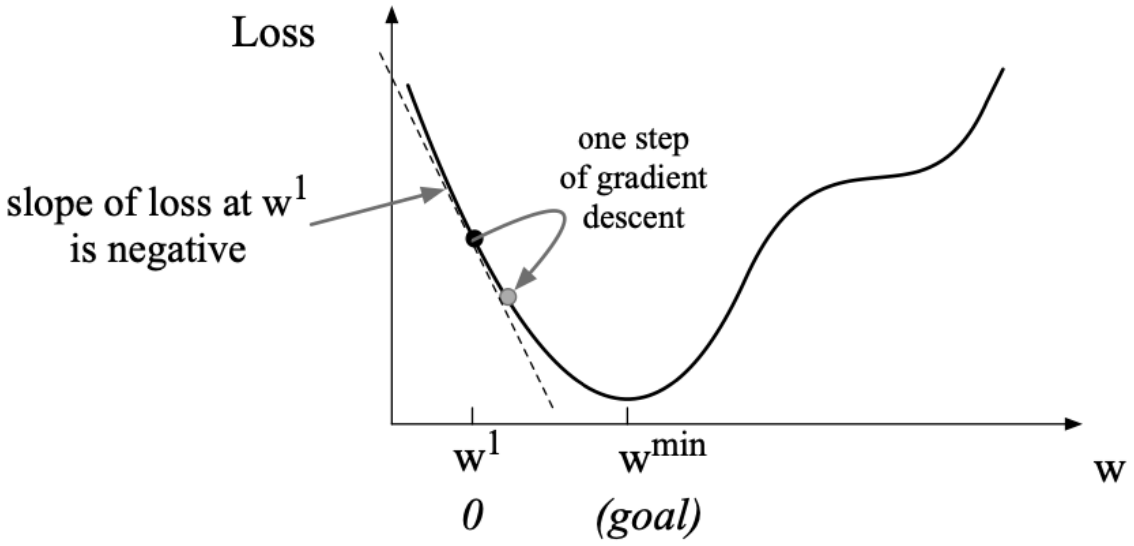

Figure 2.1: Gradient Descent Visual Illustration taken from Jurafsky et al. [6]

The idea behind this algorithm is that we know that the calculated gradient vector of a function at a certain point indicates the direction of the greatest increase for that function. To reach the minimum value that algorithm directs us to the opposite direction of the gradient. By iteratively following this approach we will eventually reach a local or global minimum.

2.3 Recurrent Neural Networks

A huge limitation of the previously presented neural network architecture is that previous layers do not have access to the output of next layer, something that is particularly relevant for processing sequential data like a sentence. Recurrent Neural Networks (RNN) on the other hand, contain a cycle in its network. This approach doesn't need a fixed input length and instead of only taking into account the current state, the value at the current state is also dependent of the previous step.

The Elman Network architecture [7] is considered the simplest RNN architecture and is the basis for the Long Short-Term Memory (LSTM) networks. Each input sequence is passed, item by item, being x_t

the item at time t. The value of the previous hidden layer weights h_{t-1} is used to influence the current hidden layer weights h_t, which will impact the value of the next input value x_{t+1}.The hidden layer from the previous step provides a sense of memory to the model, giving it a certain context starting from the beginning of our input sequence.

$$h_t = \sigma(Uh_{t-1} + Wx_t + b) \tag{2.2}$$

To calculate our output y_t we multiply the input x_t with the weights matrix W, and we sum it with the multiplication of the values of the previous step hidden layer h_{t-1} with the weight matrix U. We also add the b that is the bias factor. We then pass the value of the previous sum to the activation function to finally get the current hidden layer value h_t.

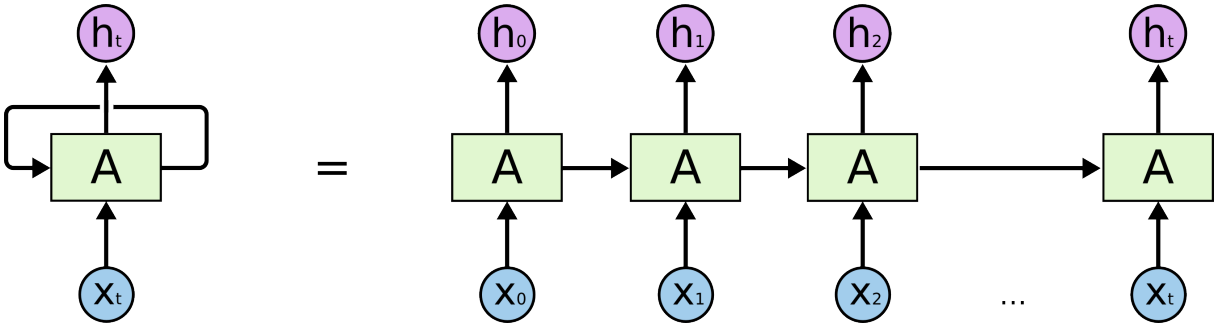

Figure 2.2: RNN architecture illustration and computations for the input vector x_t, from `https://tinyurl.com/cnsz3wdw`

Although very useful for some tasks, RNNs present several disadvantages when it is required information distant from the current point of processing. This problem arises from two reasons. The first one, is related with the fact that RNNs need to carry forward relevant information for future decisions while they also need to provide equally useful information for the current state. The second reason is known as the vanishing gradient problem, since hidden layers suffer from repeated multiplication which at some point can drive gradients values during training to zero.

2.4 Long-Short Term Memory Network (LSTM)

The solution for the limitations of the RNNs mentioned before was the Long Short-Term Memory Network (LSTM) [8]. To surpass the problems that RNNs had, the LSTM architecture performs two operations, removes information that is no longer required from context and, adds information that will likely be used. The elements that perform these operations are the cell/memory state and a special neural unit known as gate. The first one is a component that preserves a memory of gradients across time with very few changes. The gates are feed-forward layers followed by a sigmoid activation function. Basically, there are two major types of gates , the forget gate that has the purpose of deciding which information to delete from our context and, the add gate, which is responsible for choosing relevant information to add to the current context.

We can formally define the LSTM architecture as the following set of equations:

$$
\begin{aligned}
f_t &= \sigma(U_f h_{t-1} + W_f x_t]) \\
k_t &= c_{t-1} \odot f_t \\
g_t &= \tanh(U_g h_{t-1} + W_g x_t) \\
i_t &= \sigma(U_i h_{t-1} + W_i x_t) \\
j_t &= g_i \odot i_t \\
c_t &= j_t + k_t \\
o_t &= \sigma(U_o h_{t-1} + W_o x_t) \\
h_t &= o_t \odot \tanh(c_t)
\end{aligned}
$$

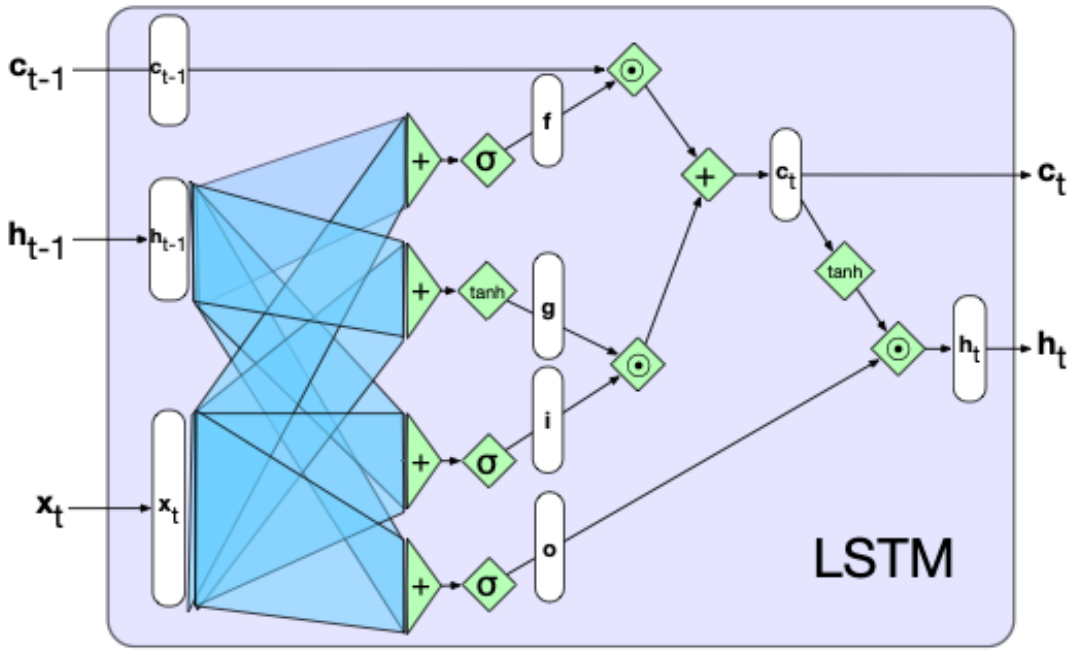

Figure 2.3: A single LSTM unit displayed as a computation graph. The inputs to each unit consist of the current input, x_t, the previous hidden state, h_{t-1}, and the previous context,c_{t-1}. The outputs are a new hidden state, h_t and an updated context, c_t. Taken from Jurafsky et al. [6]

The first two equations are respective to the forget gate. The weighted sum of the previous state's hidden layer $U_f h_{t-1}$ and the current input $W_f x_t$ is passed by a sigmoid activation function. The value from this function is then multiplied by the context mask c_{t-1} to get rid of any unnecessary context. On the third equation formulates how to decide which information to retain from the previous state and the current input, identical to equation 2.2. The fourth and fifth equation regard the add gate, which we mention that was responsible for retaining useful information for our context. In the sixth equation we add the respective elements of the modified context to generate the new context vector. The remaining 2 equations regard the final output gate, where it decides the information required for the current hidden state.

2.5 Transformer Architecture

A recent architecture of models has become the norm when it comes to language models. The transformer, presented by Vaswani et al. [1] takes full advantage of the attentions layers. The novelty behind this new architecture comes from the fact that it completely discards recurrent or convolutional neural networks, replacing them with attentions layers.

The overall structure is composed by two main structures, the encoder and decoder. The encoder is responsible for mapping the input into higher dimensional space, creating a contextualized representation of this of it. The decoder module receives this contextualized representation and converts it to an output sequence.

A simple analyses of the encoder and decoder can be done by creating an abstraction called the transformer block/stack. When we refer to transformer block/stack, we are only referring to the basic encoder stack that is very similar with the decoder one, with slight differences. Both encoder and decoder are composed by 6 stacks each, according to the original article.

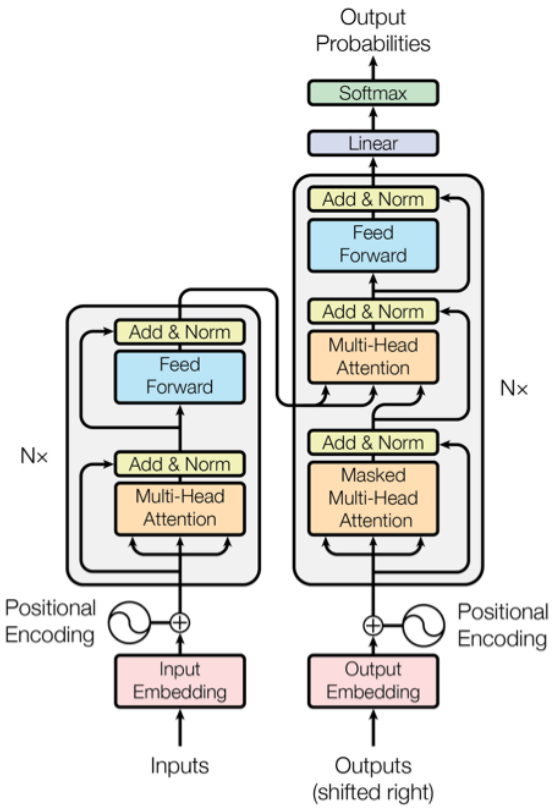

Figure 2.4: The Transformer Architecture. Taken from Vaswani et al. [1]

The transformer block has 3 types of elements: multi-head attention layers, feed-forward layers and normalization layers. Figure 2.4 shows a multi-head attention layer, followed by a normalization layer with residual connections, after which the output is passed to a feed-forward layer, also followed by a normalization layer with residual connections. The term residual connection comes from the fact that information is passed through a connection between a lower and higher layer, without requiring an

intermediate layer. In the case of the transformer, this happens before passing the output forward to the normalization layers, by first adding the layer's input to the output vector.

Self-Attention Layer

Although LSTMs networks appear to provide a solution to the problem of distant information, it still doesn't completely solve the problem of information loss in extended recurrent connection. A proposed solution was presented by Bahdanau et al. [9], where an attention layer is used to decided and retrieve relevant information from the source sequence during the decoding phase of a language model. This approach is very similar to a feed-forward neural network. This was the base for the Transformers architecture, which we will see in detail.

Self-Attention layers are very similar to attention layers, having the same principle and basic formulation. Instead of just focusing on the source sequence of the decoding phase, self-attention layers can be used in any network on the architecture of a model. The major advantage of this layer comes from the fact that it is able to retrieve large size contexts without passing through intermediate RNNs like LSTM networks.

The self-attention layer receives the input sequence x_n and returns a sequence of the same length y. When analyzing a certain item, the model only has access to all items previous to the current item. All computations for each item also have the peculiarity of being independent.

Comparisons with all items from a certain sequence are performed to reveal which ones are more relevant for the current context. These comparisons are performed using the dot product operation. For each item a "score" is calculated based on the result of the comparison.

$$score(x_i, x_j) = x_i \cdot x_j \tag{2.3}$$

We can perform the operation described by equation 2.3 to all items in our input sequence and normalize them with a softmax distribution to create a vector of weights $\alpha_{i,j}$. This vector can be generated with the following equation:

$$\alpha_{i,j} = \frac{exp(score(x_i, x_j))}{\sum_{k=1}^{i} score(x_i, x_j)} \forall j \leq i \tag{2.4}$$

This gives us the proportional relevance that each input item has in respect to the current one. With the values of α we can generate an output vector y_i .

$$y_i = \sum_{j \leq i} \alpha_{ij} x_j \tag{2.5}$$

Nevertheless, when using self-attention layers in transformers, each input element can play 3 different roles in terms of attention. When an input element is our current focus as we have seen before, all other remaining elements that have preceded it are compared to it. This role is called query. The same item can also be a preceding input item that is being compared to the current focus, which is called key. The third role is of value that is used to compute the current focus of attention.

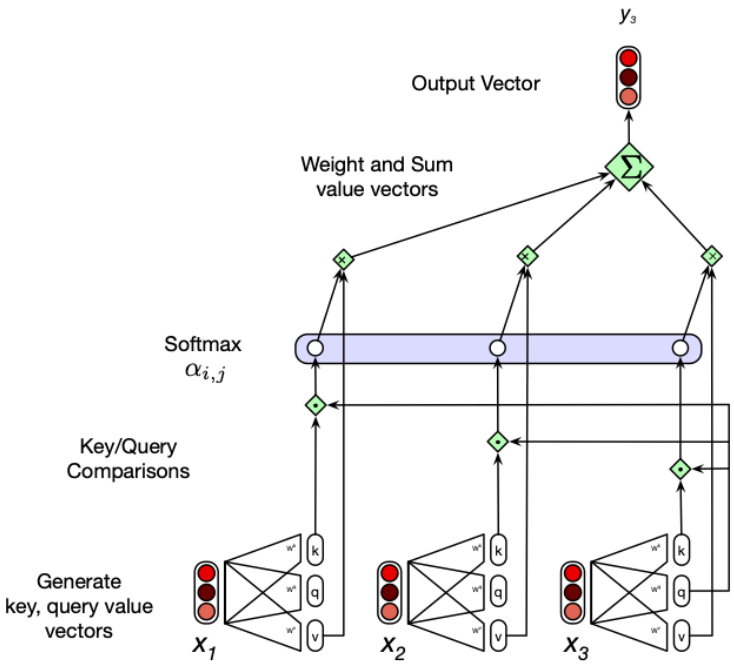

Figure 2.5: Example of the calculation for the third element of a sequence using causal (left-to-right) self-attention. Taken from Jurafsky et al. [6]

Three different weight matrices are needed to capture the roles discussed above:

$$q_i = W^Q x_i; k_i = W^K x_I; v_i = W^V x_i; \tag{2.6}$$

With x and y with dimensionality 1xd and $W^Q \epsilon \mathbb{R}^{d \times d}$, $W^Q \epsilon \mathbb{R}^{d \times d}$ $W^V \epsilon \mathbb{R}^{d \times d}$.

The equations 2.3, 2.4 and 2.5 can then be written as:

$$\begin{aligned} score(x_i, x_j) &= q_i \cdot k_j \\ y_i &= \sum_{j \leq i} \alpha_{ij} v_j \end{aligned} \tag{2.7}$$

The result obtained from the dot product can be arbitrarily large. To reduce the chances of problems appearing during the training phase of a model with its gradient values, the score values are divided by the dimensionality of the query and key vectors d_k.

$$score(x_i, x_j) = \frac{q_i \cdot k_j}{\sqrt{d_k}} \tag{2.8}$$

Add and Normalize Layer

The Add and Normalize layer has two functions. The first one was already described. It is responsible for adding to the output vector of a layer its original input. The second functions is to normalize the values of the output vector. This is particularly useful for the values of the hidden layers which can be converted to a different range, improving the gradient operations during training.

The values of the output vector are normalized through two operations, the subtraction of the mean value of the vector to each element and, dividing the result by the standard deviation.

$$\hat{x} = \frac{x - \mu}{\sigma} \tag{2.9}$$

The final output of this layer will depend on two parameters that can be learned during training, γ and β.

$$AddNorm = \gamma\hat{x} + \beta \tag{2.10}$$

Multi-Head Attention Layer

We already discussed self-attention layers, which are the real novelty aspect of the transformer architecture. These are responsible for capturing context and relations between the input sequence. Nevertheless, word relations are very complex to capture in a single context vector. The authors of Vaswani et al. [1] addressed this fact by proposing multi-head attention layers, which are nothing more than groups of self-attention layers, but with the difference of existing in several parallel layers at the same depth in a model. Each attention layer or "head", has its own parameters, which allows, for each head, to capture different relations of the input.

This implies that each head will have their own weight matrices W_i^Q, W_i^K and W_i^V.

The dimension space of these matrices differs from the original self-attention layer, with $W^K \epsilon \mathbb{R}^{d \times d_k}$, $W^Q \epsilon \mathbb{R}^{d \times d}$ $W^V \epsilon \mathbb{R}^{d \times d}$.

The final output consists on the individual output of each head being concatenated together. A final linear projection that is applied with a new matrix W^O with the task of reducing the dimensions of the output to the original dimension of $N \times d$, with N being the dimension of the input vector and d the model dimension.

The individual calculations of each head are identical to the ones previously described for the self-attention layer, but performed for matrices.

$$OutputMultiHead(X) = (head_1 \oplus head_2 ... \oplus head_h)W^O \tag{2.11}$$

Decoder

The decoder's structure is identical to the one of the encoder. Resembling the transformer block described above, with an extra attention layer described as encoder-decoder attention layer. This layer receives the output of the final encoder(the attention weight matrices from the encoder, W^K and W^V). The purpose of this layer is to help the decoder know which sections of the input sequence are most relevant.

Linear and Softmax Layer

The output of the decoder is then passed through a linear layer, a fully connected neural network, responsible for converting the float values to a vector of the size of the vocabulary of our model, called logits vector. This vector is forwarded to the softmax layer, which converts this logits vector of scores into probabilities. The item associated with the highest probability is outputted for the current step.

Positional Embeddings

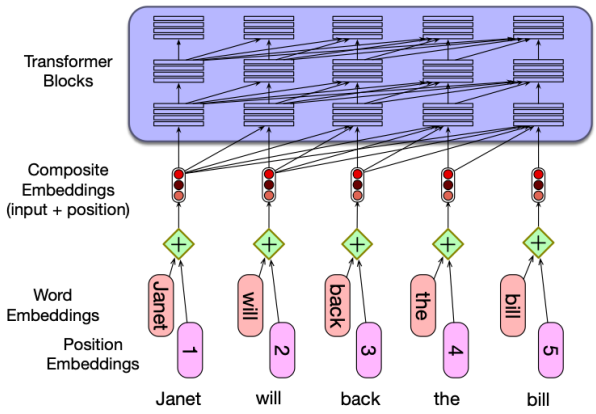

Figure 2.6: Input Embeddings of the Transformer. Taken from Jurafsky et al. [6]

Since the transformers architecture doesn't have any recurrence or convolution, it has no way of interpreting or using as context the positional distribution of elements in a sequence. To overcome this disadvantage, positional embeddings were creating. Just like we have embeddings for tokens, we can capture positional information of a position 5 in a sequence and store it in a vector of embeddings. These positional embeddings are them added to the input embeddings and passed to the first stack of the encoder and decoder. Similarly to normal embeddings, positional embeddings are learned during training. A combination of sine and cosine functions can be used to capture the inherent relations between different positions [1].

2.6 Text-to-Text Transfer Transformer Model (T5)

T5 or Text-Text Transfer Transformer model (Raffel et al. [10]) is based on the original Transformer architecture proposed by Vaswani et al. [1], with some minor differences, which is composed of an encoder-decoder structure. For context, we refer to section 2.5.

To pre-train, the original models, Raffel et al. [10] used the Colossal Clean Crawled Corpus (C4), i.e. a 750GB size corpus based on a clean version of Common Crawl's original web archive.The T5 models released were pre-trained using an unsupervised denoising task, that is based on masked language modeling and word dropout regularization. The original input tokens are randomly sampled and 15% are dropped. These tokens are later replaced by unique sentinel tokens. The model's objective is to predict the sentinel tokens that correspond to the original dropped out text. This model was them trained

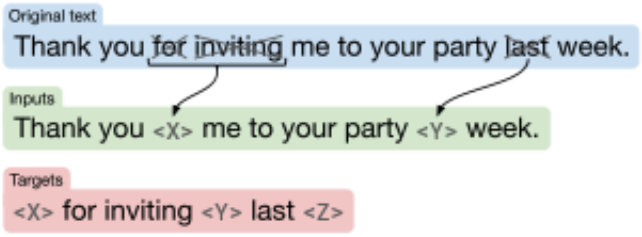

Figure 2.7: Training task described on the original article. Taken from Raffel et al. [10]

in several tasks, in a multi-task training approach, where a conditioned training strategy was applied. This is, the input was modify in an way that would condition the decoding step after. The authors used prefixes that would mark the specific task. Each task has an original and unique prefix(prompt) that the model later learns during training and uses as cue to detect the task and improve decoding.

2.7 Decoding Techniques for Text Generation

As we saw in section 2.5, the final step of a transformer model is passing through the Linear and Softmax layer. A vector is produced with the probabilities of each token in a given vocabulary. Choosing the token with the highest probability at each decoding step is called greedy search. The name is derived from the fact that the algorithm focus only on the local optimal, by doing this it tends to miss the most probable sequences to generate because, it is only focusing on the current token and not the overall sequence.

2.7.1 Beam Search

One of the most common decoding approaches is to use beam search. At the first decoding step, we choose multiple x potential candidates or hypothesis. These x candidates are kept and the generation sets continues for each candidate, in different decoders. Eventually, we will start diverging through different branches, just like a light beam of a flashlight. This beam reaches an end when the end-of-sentence token is produce. The sequences generated may have different lengths and the total generated sequences are normally the same number as the number chosen for the x potential candidates, given as a parameter for the generation step.

$$\begin{aligned} score(y) &= logP(y_i|x) \\ &= \sum_{i=1}^{t} \log P(y_i|y_1, ...y_{i-1}, x) \end{aligned} \tag{2.12}$$

The probability of each sequence is then compared to each other and, calculated by adding the log probability of generating the next token to the log probability of the sequence predicted so far.

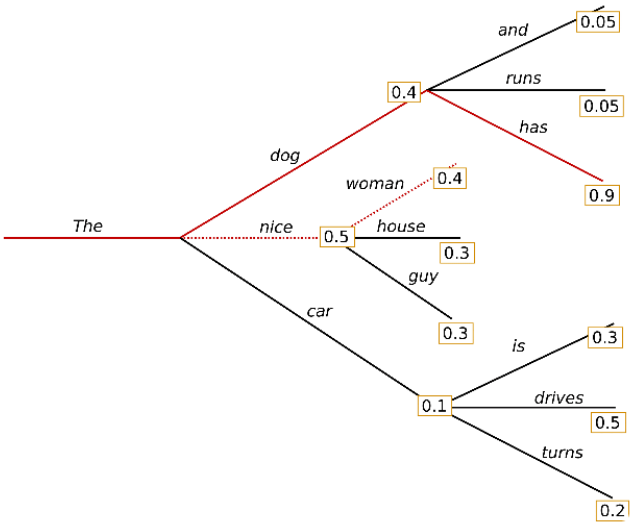

Figure 2.8: Illustration of beams produced by Beam Search. Taken from `https://tinyurl.com/79hzujkj`.

2.7.2 Sampling

Although beam search represents a good alternative to produce very likely sentences, recent literature indicates that, although it produces coherent sentences, beam search does not reproduce very human like writing. This is mainly due to the fact that human speech does not follow a distribution of high probability next words [11].

A common approach to overcome this problem is to use a decoding method called Sampling. Instead of choosing the next most likely word, the next word is sampled based on the conditioned probability distribution. This means that we can generate words that otherwise would be completely ignored.

This method alone is not enough, sometimes randomness can lead to the decoding of incoherent words, taken into account the previous decoded sentence until that step. To ensure that we obtain more coherent results, other parameters of the decoding need to be optimized. One example is to turn the probability distribution given by the softmax layer more skewed. This will redistribute the original probability through higher probability words and reduce the probability of less likely ones. By doing so, we decrease the likelihood of sampling a very unlikely word.

Chapter 3

Explored Tasks

In this chapter, a general overview of literature that influenced our work, and that showed us what strategies would be interesting to pursue, is given in detail.

Pretrained language models have become a hot research topic in the past half-decade. The promise of transfer learning by pre-training a model and later fine-tuning it is nowadays the most common approach to NLP problems [10]. Experimentation in several medical NLP tasks have been reported in the literature and here we provide a brief overview on previous methods, for different tasks.

3.1 Automatic Labeling

The current availability of medical datasets is very scarce. Annotation and labeling require experts in the field and it is very time-consuming, which creates a bottleneck in terms of the data available to train and evaluate machine learning. Attempts to create automatic labelers are not new but these in turn, also need large amounts of training data.

In the paper by Meng et al. [12], the authors focus on the problem of delays in the communication of urgent clinical findings in radiology exams. The authors trained and fine-tuned a model in a large radiology report dataset, to identify critical reports with time sensitive findings and to feed this information to an existing pipeline that delivers the information to physicians. The proposed self-supervised contextual language representation model is based on a pre-trained BERT [13]. The decision to use contextual embeddings resulted in better results than the state-of-the-art at the time, based in word2vec representations.

Another automatic labeler named ChexBERT was developed by Smit et al. [14]. ChexBERT is also a BERT-based model that was previously pre-trained in biomedical texts. The paper proposes a training scheme where the model is trained on radiology reports with annotations from a rule-based labeler named CheXpert [15], and later fine-tuned in expert annotations augmented with a back-translation strategy. The idea behind this training is to use the already learned information of the rule-based model and feed it to the new model in the form of the generated outputs. The results outperformed previous rube-based labelers on the MIMIC-CXR dataset.

3.1.1 Natural Language Inference (NLI)

Natural Language Inference (NLI) is a task designed to assess the inference relation between an hypothesis and a premise expressed in natural language. Three types of inferences can be verified, entailment, neutral or contradiction.

The release of MedNLI in 2019 [16] allowed anyone to test their models in the NLI task, specifically focused on the medical domain. Several models have been specifically trained and tested in the MEDNLI dataset. For example, Phan et al. [4] proposed SciFive, i.e. a T5 based model that is pre-trained in a biomedical corpus of PubMed abtracts, followed by a training phase using a multi-task learning strategy and later fine-tuned in five biomedical NLP tasks. The paper defined new state-of-the-art scores for several tasks including also the MedNLI dataset.

3.1.2 Semantic Text Similarity (STS)

Semantic text similarity is an NLP task designed to quantitatively assess the semantic similarity between two text snippets [17]. The release of public tasks in the last couple of years, focused on this topic has helped the exploration of new models that perform better in medical STS.

In Wang et al. [18], BERT-based models are tested in the STS task. In particular, the authors try different ways of extending training data, using unlabeled domain data which is assigned labels from a general model. This strategy produced new state-of-the-art results in one of the datasets used.

3.2 Summarization

The task of summarization of radiology reports, although technically very challenging, presents several incentives to promote and develop advancements. In particular, the most important factor is the direct impact on the efficiency of clinical communication pipelines and the acceleration of the radiology workflow [12].

The MEDIQA shared task focused specifically in this domain. In the 2021 edition, the overview article of the task from Abacha et al. [2] reports the results presented by the participants, as well as a brief overview of the type of models that were used. From the total of 7 teams, 6 presented pre-trained models based in BART [19] or PEGASUS [20].

3.2.1 Medical Paraphrase Detection

Paraphrases are defined by Bhagat and Hovy [21] as sentences that convey the same meaning using different wording. Unfortunately, very few datasets are available for this task when considering medical domain data. A particularly well-known dataset is the MSRP corpus released by Microsoft [22], nonetheless featuring general text.

Chapter 4

Approach

Pre-trained models allow us to reuse features from the general domain, learned from large corpora of unlabeled data, to generalize better when training and fine-tuning specific models. This approach tends to result in better performance when compared to training a model from scratch only in task-specific data. Here we propose to take advantage of the power of transfer learning, with the use of the text-to-text-transfer transformer (T5) proposed by Raffel et al. [10]. Text-to-text means that, for each input received, the model returns a string as an output, making it appropriate for question answering, summarization, and other text generation or classification tasks.

4.1 T5 Model and Architecture

The Text-to-Text Transfer Transformer model is a very powerful model, that as recently shown impressive results in several task from multiple domains. The original article also presented five size versions of the T5. For our work, we developed all the experiments using the original T5-Base model, which is the original baseline model with roughly 220 million parameters. No changes were performed to the original architecture, training objective, and vocabulary of the original model publicly available at the HuggingFace website[1]

The released model was already trained in several task, some of those very relevant for this work, such as summarization or natural language inference. A great advantage of the T5-Base model is the fact that it as already seen very large corpus and datasets. The frequent necessity of prompts or prefixes in the begining of the input sequence, makes it even more attractive for the type of task that are pretended to be explored.

4.2 Multi-Task Learning

The original paper by Raffel et al. [10], as well as other recent papers, suggest that multi-task learning can provide good results across several tasks. We use maximum likelihood as training objective,

[1] https://huggingface.co/t5-base

together with teacher forcing to fine-tune the T5-base already trained in multiple tasks, like questions-answering, summarization, or natural language inference.

Influenced by these results, we propose to follow a multi-task learning approach to achieve a medical model focused on radiology reports that is trained in multiple tasks simultaneously. By doing so we are not only reusing weights from previous trained models but we are also sharing weight's information learned from the different tasks in the same domain.

We selected summarization , natural language inference, semantic text similarity, paraphrase detection and artificial task ChexBERT, that generates labeling for radiology reports, as the tasks for our model to learn. The reason behind this choice comes from the fact that all these task have the objective of generating text.

A great advantage of using T5 is that the model was already trained in several tasks using a prompting approach (i.e. at the beginning of the input, a prompt dedicated to a specific task is added, serving as a clue for the model to know which task it is handling). In most of our tasks we were able to re-purpose some of the prompts used during pre-training of the T5 baseline, by pre-appending the prompt at the beginning of the examples. Table 4.1 summarizes the tasks that we considered, which are also explained next.

Table 4.1: Prompts appended per task.

Task	Prompt	Model Input
Summarization	summarize	summarize : ”sentence 1” sentence 2: ”sentence 2” , ”label”
STS	stsb	stsb sentence1: ”sentence 1” sentence 2: ”sentence 2” , ”label”
MEDNLI	mnli	mnli hypothesis: ”hypothesis” premise: ”premise” ,”label”
PD	mrpc	mrpc sentence1: ”sentence 1” sentence 2: ”sentence 2” , ”label”
Classification	chex	ChexBERT hypothesis: ïmpression” premise: öbservation ”label”

4.3 Training Scheme

In order to optimize the training of our models, the training scheme presented in figure 4.1 was developed. An important aspect of the T5 model is that it receives input and label sequences with a fixed size, defined in the data module. Nevertheless , sequences for our tasks have variable sizes. To compensate for this fact, padding is added to the input and label vectors until the required size is reached.

The tasks of NLI, STS, PD and ChexBERT generate relatively small sequences due to the nature of the corresponding labels. It becomes attractive to combine their training together and reduce the label size parameter that our model requires for training. This change in turn, will directly impact the computational costs of the training step, reducing the time required for training a model on these tasks.

Compared to the remaining tasks, summarization lables have a 90 percentile reaching 55 tokens per label, which requires a larger label size to be fed to the model, leading to higher training times.

The final strategy takes what was mentioned above into consideration. We start with the T5-Base model, pre-trained in the C4 Corpus and, we fine-tune it on the set of smaller tasks that generate small

labels, with very few tokens. This allows us to define the T5 label max length parameter to 20 tokens only, which represents a significant decrease compared to the summarization labels. A label max length parameter of 200 tokens was used for the summarization tasks, which is intended to include almost all labels on our training set whiteout performing truncation.

Our training scheme is divided in two phases. On the first phase we perform a multi-task training where we fine-tuned a T5-base model with data from the all tasks mentioned above with the exception of summarization (referred to as small tasks). The second phase uses the last checkpoint from the first phase and is followed by fine-tuning the model on the summarization task. After phase two, we proceed to fine-tuning the models for each individual small task, with the exception of the classification task. Figure 4.1 presents an illustrative representation of our training scheme.

A fine-tuned approach was performed on the following 5 biomedical tasks:

- Natural Language Inference (NLI);
- Semantic Text Similarity (STS);
- Paraphrase detection (PD);
- Summarization;
- Classification (CHEX);

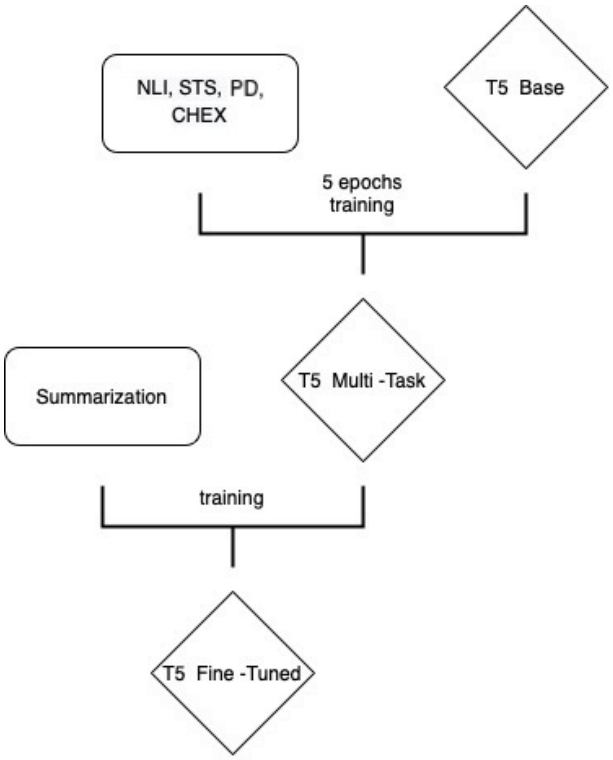

Figure 4.1: Training Scheme used to fine-tuned our models. NLI - Natural Language Inference, STS - Semantic Text Similarity, PD - Paraphrase Detection

4.4 Data Augmentation

As mentioned before, biomedical datasets are very scarce and limited. The original datasets for the small tasks had two disadvantages, namely the reduced size and not being related to radiology context. To surpass these limitations, we considered several common strategies to augment the available data with radiology data (MIMIC-CXR examples) and subsequently help the model learn with more examples.

Back Translation

A common strategy to increase the dataset size involves the use of back-translation, where an input sequence is translated to another language and translated back again. This approach tends to modify the original sentence with paraphrasing [23]. To achieve a higher diversity of paraphrasing, the examples of the datasets where this strategy was applied were translated to Portuguese, French, Spanish and German and later back to English. The example with the higher number of changes, compared to the original English input, was selected as the new example and added to the dataset. This allowed doubling the number of examples of the datasets, without introducing significant noise to the original and with very small computational costs. We took advantage of the google translator API, which is a very common API to perform translations with good quality and publicly available, with very few restrictions in terms of use.

Artificial Tasks

WWe created an artificial task(i.e., the classification task) to simulate the labels given by the ChexBert labeler [14] and add an additional task to our multi-task training. We then trained a T5-base model with the impressions from MIMIC-CXR and as label the outputs given by the original ChexBert labeler. The ChexBert model is based on a pre-trained biomedical model that was fine-tuned on radiology labels from another model and fine-tuned again on a small set of expert annotations augmented with back-translations. The task of report labelling is to extract the presence of one or more cliniclally relevant observations from free-text radiology reports. This particular model performs labelling in 14 different medical observations: Pneumonia, Fracture, Consolidation, Enlarged Cardiomediastinum, No Finding, Pleural Other, Cardiomegaly, Pneumothorax, Atelectasis, Support Devices, Edema, Pleural Effusion, Lung Lesion and Lung Opacity. The labels can be blank, positive, negative or uncertain, with the exception of No Findings which can only be either blank or positive.

For each example we propose an hypothesis, a premise and a label. The hypothesis corresponds to a category of observations that is predicted by the original CheXpert [15], the premise is an impression taken from a radiology report and, the label is one of the possible labels for that specific observation. The original MIMIC train split used has a total of 91545 examples, which produced over 1 million ChexBERT examples . To reduce the total number of examples due to our computational limitations, a random dropout per category was performed, reducing the total number of examples to 205959. We called this datasete the ChexData.

Data Generation

The scarcity of medical datasets for natural language inference and paraphrase detection presented a challenge in terms of the available data to train our models. To overcome this difficulties some creative approaches were use to take advantage of public datasets that may be augmented to generate data for other tasks.

The relation between findings and impressions tends to be very strong in terms of similarity of tokens and general content. We decided to explore this fact to increase the number of examples in our datasets. Using the ChexBert Labeler to generate labels for our MIMIC-CXR findings and impression in 14 different categories, we explored this information to identify relations between both sections of the radiology reports. In particular, the impressions as a premise should be sufficient to infer the overall summary, in which case say there is an entailment relation. If one cannot infer neither contradict the hypothesis (findings section), this represents a neutral relation. Lastly, if one can use the premise (impressions sections) to deny the findings, a contraction relation is found. A threshold of similarity is defined between the 14 labels to classified each pair of findings/impression has having a relation of entailment, neutral or contradiction.

We attributed the entailment label to a pair of findings, impressions that achieved a ratio superior to 0.7 of equal labels divided by the total labels different than blank and, with zero labels that where opposites i.e. positive and negative label for the same observation on the findings impressions pair. A contradiction label was assigned when the pair of findings impressions achieved a ratio superior to 0.2 of opposite labels divided by the total labels different than blank. Th neutral label was given when the pair of findings/impressions achieved a value less than 0.5 on the same ratio described for the entailment label and conditions.

A summary of findings in radiology reports is also meant to preserve key features of the impression section, basically being a shorter version but with the same overall meaning. In other words, there is a paraphrase relation between findings and impression, since the impression section should be descriptive enough to pass the same content with less words. Additional PD data was thus created exploring this relation.

This PD dataset has two possible labels: equivalent, which means that the sequences are equivalent to each other in meaning, and not equivalent. The following criteria was used to label the examples: If all labels of the ChexBert labeler were equal for both impressions and findings, we attribute the label equivalent. All other cases different from the previous were considered not equivalent.

Decoding Strategies

One of the most underestimated parts of producing a model tends to be its decoding phase. A range of approaches and parameters can be used and fine-tuned to optimize decoding results. We discussed on section 2.7 some strategies that are interesting to explore. For generation tasks of very small lengths, such as the NLI or STS task, beam search is a predictable and reliable decoding strategy. This was the standard decoding approach for the tasks of NLI, PD, ChexBERT and STS.

Generating longer sequences of text becomes a much harder challenge. We propose to explore the results of several decoding strategies and parameters to optimize this particular task. When producing summaries on the radiology domain, we must retain and not alter the information on the summary. This led us to take the decision for our experiments to apply only beam search and to not pursue a sampling approach.

Overview

On this chapter, we presented our approach, as well as the main reasons behind it. We started by given a brief explanation of the T5 model and architecture used in this work. The characteristics of this particular model make it advantageous for a multi-task learning approach, which allows for the reuse of prompts, already learned by the model during the pre-training step, for each task. We layout our training scheme, which leverages from the nature of the tasks chosen and maximizes the use of available computational resources. To finalize our approach, we define data augmentation strategies to apply to our datasets and by doing so, we intend to expand the original training data.

Chapter 5

Results

This section describes our experimental procedure and the obtained results. A full comparison with the current state-of-the-art results on the each task is provided. All experiments were run on Google Colab.

5.1 Datasets

We provide a brief description of the datasets and the pre-processing techniques used. These are comparable to the ones used in related works and state-of-the-art models that were used as comparison. An analysis of the distribution of our datasets was performed. A statistical approach allows for a better understanding of the datasets, and what bottlenecks these may introduce to our experimental approach. We performed a simple representation of the distribution of our train datasets using histograms to visualize the results.

MIMIC-CXR

The MIMIC-CXR dataset release by Johnson et al. [3], is one of the largest radiology datasets publicly available. It corresponds to a total of 227,835 radiographic studies, particularly chest studies. The original dataset was constructed based on three different modalities of information: electronic health records, radiography images and free-text reports. All three modalities suffered from an extensive anonymization process to ensure that none of the patients personal information could be trace back.

In our work we only focused on radiology reports, that were constructed by combining the information of electronic health records (EHR) and, free-text entries of radiologists. We highlight two sections of these reports that where crucial for our approach, the findings and impression sections. The findings section of radiology reports correspond to observations that the radiologist made during the overall observation of the chest radiography. The impressions section compose a summary of the major highlights of the findings.

The nature of these reports, taken on a health emergency department at the Beth Israel Deaconess Medical Center, is sometimes incomplete, inaccurate and error prone. To overcame this and to sample such one big corpus without introducing noise to our dataset, we used the script provided on the

MEDIQA 2021 Radiology Reports Summarization task [2] to generate a training set of 91,544 radiology reports clean of errors, useless sections, typos and other noise sources. As evaluation set we used a test subset also provided by the same task, that is a collection of 600 chest X-ray reports. No further pre-processing was done.

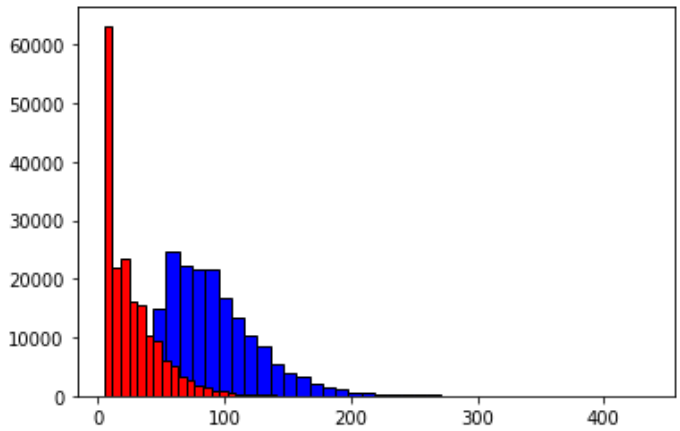

Figure 5.1: Distribution of token size of inputs(blue) and labels(red) for the MIMIC-CXR train split.

The distribution of the number of tokens per input and label of the training split can be observed in figure 5.1. The large size of both inputs and labels were taken in to consideration in our approach. To optimize training and decoding, the number of tokens per input was limited to 512 tokens and the labels to a max length of 200 tokens.

ClinicalSTS 2019 dataset

Released for the 2019 n2c2/OHNLP task [17], this dataset was derived from electronic health records of the Mayo Clinic, combining 1068 sentence pairs from the previous year task that used a subset of MedSTS with new 574 pairs. Each pair is given a score between 0 and 5, where the first is the lowest and the last the highest score possible. A total of 1642 pairs are used for the train subset and 412 for the testing subset. In order to reduce the number of possible scores, we converted all scores to even decimals by performing a rounding operation to the closest number. For example, if the score was 2.5, the rounded up score would be 2.6.

The STS dataset was the only dataset which distribution was not altered compared to the original dataset. The task of semantic text similarity differs from all the remaining tasks by being actually a regression task. The output of this tasks varies between a range of 0 and 5 with increments of 0.2. This is not an obstacle for the T5 model, which deals with this problem by converting the original task into a classification one, with a total of 30 possible labels. The 90 percentile of our distribution is around 121 tokens of length for the input.

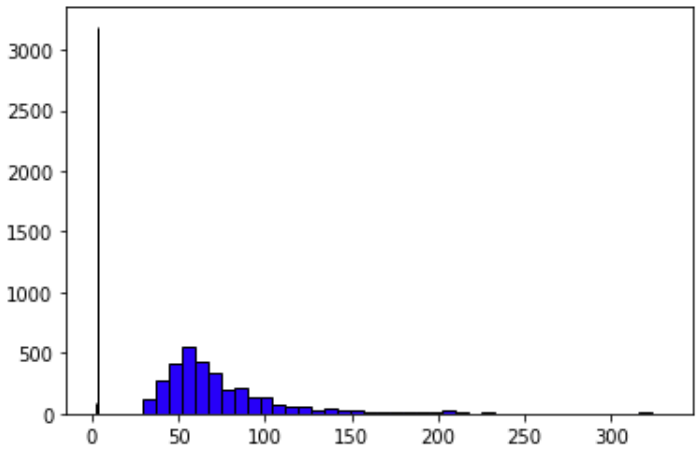

Figure 5.2: Distribution of token size of inputs(blue) and label(red) for the STS train split.

MEDNLI

Release in 2019 [16], the MEDNLI dataset is composed by pairs of premises and hypothesis. Each pair is associated a label of entailment, neutral or contradiction. The original premises were sampled from the MIMIC-III dataset [24] and the hypothesis were manually annotated by physicians. A total of 11,232 training examples, 1395 validation examples, and 1422 test examples are available.

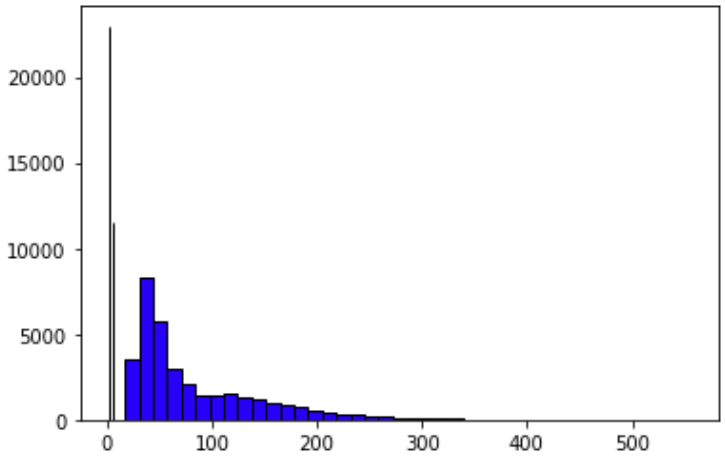

Figure 5.3: Distribution of the token size of inputs(blue) and label(red) for the NLI train split.

When we look at the distribution of the dataset on the NLI task, we see that we have most of the input lengths concentrated between 10 and 100 tokens. The 90 percentile of the distribution is 173 tokens which is a consequence of the data augmentation using the MIMIC-CXR generated examples. The lengths of the labels in this task on the other hand, are much smaller compared to the summarization labels.

MED PD

We used a small subset of medical paraphrases contains 150 examples for training and 60 validation examples, compiled by Aditya [25] and that was based on examples from the i2b2 (2018) cohort detection task dataset. [26].

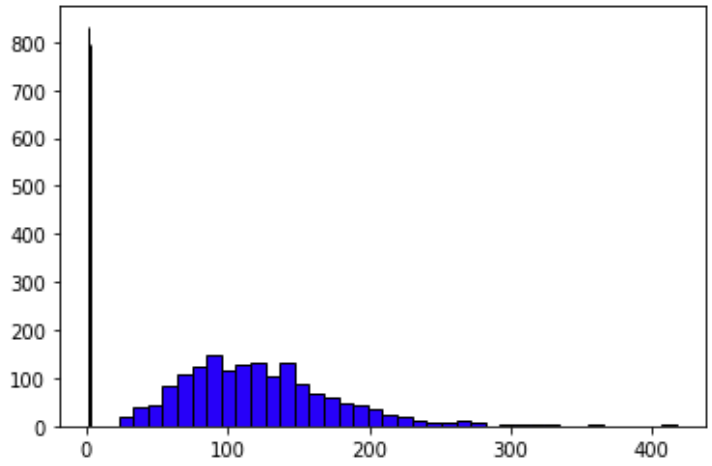

Figure 5.4: Distribution of token size of inputs(blue) and label(red) for the MSR train split.

The PD dataset is the smallest dataset incorporated in our work. Only with 150 examples, it highlights the reality of the scarcity of data in the medical domain. Similarly, after using the data augmentation techniques already described, the overall input size increased. The 90 percentile is located at a size length of 194 tokens.

Augmented Datasets

As explained under the data augmentation section, we increased the size of the original datasets with some simple techniques, like back-translation or generation of new examples through some heuristics. Table 5.1 presents a comparison between the original sizes of the datasets, and which augmentation strategies contribute to increase each dataset.

ChexData

The ChexData dataset is a dataset created for this work, specifically to train a T5 model on who to simulate the labels given by the original ChexBERT labeler. The examples are composed by MIMIC-CXR impressions and their respective observation labels given by the ChexBERT labeler. This dataset was only created and used for this chext X-ray labeling task. It represents a similar input distribution to the MIMIC-CXR dataset and label distribution similar to the remaining small tasks.

[1] This data was augmented using back-translation.

Table 5.1: Original train dataset sizes and changes due to data augmentation.

Dataset	Original Size	Back-translation	ChexBERT	Final Size
MRPC	150	-	1476	1626
NLI	11232	11232	12000[1]	34464
STS	2020	2020	-	4040
ChexData	205960	-	-	205960
MIMIC-CXR	91544	91544	-	183088

5.2 Metrics

Taking into consideration the current metrics being used for each task and that are widely reported in the literature, this next section provides a brief description on the metrics that were used to evaluate the different tasks.

ROUGE

Recall oriented understudy for gisting evaluation (ROUGE) is the standard metric used to evaluate automatic summarization [27]. The metric evaluates the quality of the produced summaries by comparing them with an human reference that is meant to serve as an example of a perfect summary. The original paper proposed 4 different ROUGE scores, but we opted to only report ROUGE-2, since it is the official metric for the MEDIQA task of radiology reports summarization [2]. ROUGE-2 measures the co-occurrence of bi-grams between the machine generated summary and the references.

$$\frac{\sum_S \sum_{ngram_n \in S} Count_{match}(ngram_n)}{\sum_S \sum_{ngram_n \in S} Count(ngram_n))} \tag{5.1}$$

In this equation S represents the set of summary references and $ngram_n$ is an ngram belonging to one of the references.

On the denominator section, we add all the n-grams presented in the reference summary, where we are performing a measure of the recall of the n-grams. The choice of ROUGE-1, this is, recall of unigrams, takes into account the fact that we are dealing with radiology reports, that belong to a radiology domain. Any deviation from what would be consider an ideal reference summary may affect significantly quality of the generated summary. Another reason to use this metric comes from the fact that a very similar summary to the reference must be a fluent text, since it is very similar to what would be expected from a human reference. The same logic was used for the ROUGE-2 metric.

ROUGUE-L computes a score based on the longest common sequence between the generated summary and the reference sequence. This is a good way to have a notion of the similarity of to summaries. The longest common sequence(LCS) is used in the typical recall and precision formula instead of the number of matched n-grams. The final score is computed as an F-Score that combines both recall and precision plus a $beta$ factor, that can be achieved by dividing precision by recall.

$$
\begin{aligned}
R_{LCS} &= \frac{LCS(X,Y)}{m} \\
P_{LCS} &= \frac{LCS(X,Y)}{n} \\
F_{LCS} &= \frac{(1+\beta)^2 R_{LCS} P_{LCS}}{R_{LCS} + \beta^2 P_{LCS}}
\end{aligned}
\tag{5.2}
$$

Accuracy

Several tasks that we explored correspond to classification problems, like medical inference or paraphrase detection. The common metric used to evaluate classification performance is accuracy.

$$
Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.3}
$$

We can simply define accuracy as being the number of right predictions, divided by the number of total predictions.

Pearson Correlation Coefficient

To make an accurate judgment of the performance of our model on the STS, we reported our results using the Pearson Correlation Coefficient. This metric is reported in the literature and public tasks as a good indicator of similarity between two scores [28].

The coefficient varies in a range between -1 and 1, being 1 when the two sequences to compare are identical. Value 0 when the two sequences are completely different and -1 when the values of the two sequences are complete opposites.

The following equation describes how to compute the coefficient value:

$$
Pearson = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2} \sqrt{\sum (y_i - \bar{y})^2}} \tag{5.4}
$$

For each position of both sequences we calculate their difference regarding the mean value of each sequence multiplied by each other. The result described before is them divided by the multiplied sum of both the quadratic difference of the two sequences.

F1-Score

Besides considering accuracy, the evaluation of the performance of our models on the paraphrase detection and medical inference tasks was also performed using an F1 metric, which performs a geometric mean of the per label recall (R) and precision (P). The outputted score was unweighted, which means that that we did not take into account the distribution of the labels in our test set, due to the fact that it is assumed that all present an evenly occurrence distribution.

$$
F1 = 2\frac{P \times R}{P + R} \tag{5.5}
$$

We compute the recall by dividing true positive labels by the sum of true positive and false negative labels. The precision is calculated by dividing true positive labels by the true positive and false positive labels.

5.3 Experimental Results

This section presents our experimental results. The following table is a description of the most relevant models trained during our work and that are worth reporting.

$Model$	$BaseModel$	$TrainingEpochs$	$TrainDatasets$
Base	-	-	C4
X1	T5-Base	1.5	ChexData
F1	T5-Base	2.5	MEDNLI,STS and PD
F2	T5-Base	5	MEDNLI,STS and PD
S1	F2	2.5	MIMIC-CXR
S2	F2	5	MIMIC-CXR
S3	F2	1	MIMIC-CXR

Table 5.2: Model's Description and Training Parameters

5.3.1 Training parameters

Regarding the training parameters, he number of epochs and conditions were adjusted, taken into account the difference in the examples between the multi-task datasets and the original single task datasets. A weight decay value of 0.01 and the value of 500 for the warming steps was used in all experiments. During the generation phase only beam search was used and the max length was adapted according to the tasks.

5.3.2 Classification

We trained a T5-base model(X1) on outputs from the ChexBERT labeler of the MIMIC-CXR dataset (ChexData). The idea was to have a model that could accurately generate predictions of labels when compared to the original dataset. The model was trained for 1.5 epochs in total on the ChexData dataset. We than compared the performance of X1 using accuracy when compared to references produced by the original ChexBERT, on a test set of 20000 examples, different from the examples used during the training phase. The overall results are documented on Table 5.3 . To reduce the size of the dataset we decided to not predict labels for the class "No Findings".

5.3.3 Multi-task Training

In order to validate our multi-task learning approach, we intended to verify that by training the model in several generation tasks we would see an improvement on the model's performance on those same

Observation	Accuracy
Edema	0.9925
Fracture	0.9945
Consolidation	0.9855
Enlarged Cardiom.	0.9855
Pneumonia	0.9945
No Finding	-
Pleural Other	0.9980
Cardiomegaly	0.9945
Pneumothorax	0.9995
Atelectasis	0.9920
Support Devices	0.9875
Pleural Effusion	0.9955
Lung Lesion	0.9945
Lung Opacity	0.9765

Table 5.3: Acccuracy of the T5-model, for each observation category, trained on the ChexData dataset and tested a test set of 20000 random ChexBERT labels from the MIMIC-CXR dataset

tasks. We compared training the model first individually in some tasks and than we compared the performance results on Table 5.4.

5.3.4 Natural Language Inference Task (NLI)

The Natural Language Inference task was used on the first phase of our training scheme to fine-tune the T5-base model, combined with the STS, PD and ChexBERT tasks. The performance of our model after 2.5 and 5 training epochs on this multi-task training scheme are reported on Table5.4. We see that the model's performance on this task improved with more significantly with more training epochs, noticeable by comparing model F1 and F2. Another interesting result was the T5-base zero-shot attempt, which performed relatively bad on the NLI task. This suggests that although being pre-trained on natural language inference [10], the performance of the model is very sensitive to different domains for this specific task.

We also fine-tune our model on the NLI dataset after the second phase of our training scheme, this time only with this specific task. If we observe the results on Table 5.4, we see that after 2 epochs of fine-tuning, the f1-score and accuracy on this tasks increases to values close to 0.80. This is an improvement compared to the performance of first phase model. Overall, the results obtained are not very far from current state-of-the-art models [4].

5.3.5 Semantic Text Similarity Task

Similarly to the previous tasks, the Semantic Text Similarity task was used on the first phase training scheme of our model, combine with the PD, NLI and ChexBERT tasks. A striking result that can be seen on the section of the STS task on Table 5.4 is the relatively high correlation score achieve by the T5-base model zero shot attempt. The T5 model was pre-trained on semantic text similarity tasks [10], nevertheless, it shows a relatively strong performance for a particular radiology and medical domain that was not previously anticipated. The performance of our models after 2.5 and 5 epochs, also present some interesting. One can notice that there a decrease of the correlation score when the model is trained for more epochs (model F2 compared to F1). The model's performance is still an improvement of 0.15 (F1 model) and 0.14 (F2 model) compared to the zero-shot attempt. A possible explanation for this decrease in overall performance is most likely due to the model having reached a point of overffiting, a typical phenomenon when the model gets stuck to the training examples and loses its capability of generalization, leading to poorer results when reaching a stage of inference to predict labels.

A second fine-tuning with only this task (FF1 and FF2), was performed after the second phase of our training scheme. The results presented on Table 5.4 show that we it was not possible to achieve better performance than the first training phase. This is most likely due to the nature of the classification task that perhaps requires more training data, so that it does not overfit so easily. Secondly, we conclude that this task doesn't appear to benefit from the second phase training.

5.3.6 Paraphrase Detection Task

Regarding the Paraphase Detection task, similarly to the previous tasks, it was also used on the first training phase of our scheme, combined with the STS, NLI and ChexBERT tasks. All results of the first training phase are present on Table 5.4. Similarly to the NLI tasks, we see that the zero-shot attempt of the T5-base model produces very poor results despite being pre-trained on paraphrase corpus [10]. The same phenomenon that occurred for the STS task happens for the PD performance, where we see a decrease on the F1-score when the number of training epochs is increased. For this specific dataset there is no available literature for comparison, but we still opted to include it in order to increase the available training data of our multi-task approach.

The final fine-tuning with this task only is displayed on Table 5.4 and reports a great improvement compared to the first training phase results.

5.3.7 Summarization Task

When it comes to the summarization task, the results are reported on Tables 5.5, 5.6 and 5.7, under the models S1, S2 and S3, and for different metrics. This task is only used on the second training phase, where we fine-tuned the T5-base model fine-tuned from the small tasks. The T5-model was already pre-trained on the summarization task [10], but still achieves poor results similar to the previous tasks reported when we perform a zero-shot attempt.

The results achieved after fine-tuning the previous multi-task model show great improvements compared to the T5-base zero shot approach. Surprisingly, we see that the best performing model is S3, that was fine-tuned only for 1 epoch. This means that the model is showing overfitting signs in a very early despite the high number of training examples. A comparison to recent literature, in particular to the 2021 MEDIQA winner, described on Abacha et al. [2], shows that although we do achieve a performance we are still distant from the state-of-the-art result.

5.3.8 Overview

In this chapter, we present findings and observations. A simple statistical analyses is performed on our datasets, to help understand better the nature of each task. We report major differences in sizes of inputs and labels on our datasets, which will directly impact the way we train our models. This is followed by a brief description of the metrics used to evaluate the performance of our models in the different tasks. Since we have several tasks with different objectives, we also report different metric results for each task. We finalize this chapter by comparing the performance results obtained for each task and model, and an explanation is given to justify the observed results.

	PD		STS	NLI	
	$F1Score$	$Accuracy$	$PearsonCor.$	$F1Score$	$Accuracy$
$Base$	0.39583	0.63333	0.69233	0.36050	0.41421
$F1$	0.79107	0.81667	0.84018	0.73117	0.73769
$F2$	0.77815	0.81667	0.83952	0.78746	0.78832
$FF1$	0.81667	0.77011	0.81587	0.78551	0.78443
$FF2$	0.86667	0.84596	0.81490	0.79958	0.79867
IBM [29]	-	-	0.90100	-	-
$SciFive$ [4]	-	-	-	-	0.86570

Table 5.4: Results - Small Tasks, with FF1 and FF2 correspond to the final fine-tuned model, trained on each task individually, for 1 and 2 epochs, respectively.

	$ROUGE-1$		
	$Precision$	$Recall$	$F1-Score$
$Base$	0.19412	0.25569	0.19623
$S1$	0.51951	0.38811	0.41816
$S2$	0.51781	0.40390	0.42788
$S3$	0.52231	0.39212	0.42263
L[2]	-	-	0.55730

Table 5.5: Results - Summarization Task ROUGE-1

	$ROUGE-2$		
	$Precision$	$Recall$	$F1-Score$
$Base$	0.08357	0.09791	0.07938
$S1$	0.34269	0.25308	0.27133
$S2$	0.33146	0.25800	0.27081
$S3$	0.35009	0.2611	0.27947
L[2]	-	-	0.43620

Table 5.6: Results - Summarization Task ROUGE-2

	$ROUGE-L$		
	$Precision$	$Recall$	$F1-Score$
$Base$	0.16932	0.23041	0.17435
$S1$	0.49906	0.37408	0.40278
$S2$	0.49644	0.38779	0.41069
$S3$	0.50317	0.37778	0.40746
L[2]	-	-	0.53660

Table 5.7: Results - Summarization Task ROUGE-L

Chapter 6

Conclusions

6.1 Summary of Contributions

Our work had the purpose of exploring the potential of new and powerful model architectures like the Text-to-Text Transfer Transformer when applied to the medical domain. We developed a multi-task medical model capable of dealing with 4 different tasks: summarization of radiology reports, medical semantic text similarity, medical natural language inference, and medical paraphrase detection. We also presented a training scheme that takes advantage of the differences in the nature of the datasets sizes and optimizes the time required to train a model. Our fine-tuning approach revealed significant results on the mentioned tasks, similar to state-of-the-art models, but with the ability to generalize for 4 different tasks. We also demonstrated the potential of data augmentation for the medical domain by applying a back translation strategy and by generating new examples based on the radiology report labelling task outputs of the ChexBERT Model. Our models suffered from one of the most common problems during training of medical models, overfitting of the training data, due to the problem of scarce datasets that were available. The results achieved with our last model, show significant improvement regarding all the three small tasks and summarization task. In particular, according to the official results table from the 2021 MEDIQA task, our model would be rated on the 10th place, without any modification to the model architecture, which indicates the robustness of our training scheme and potential for improvements which take advantage purely from training strategies and data augmentation.

6.2 Future Work

The application of models to the medical domain is still in a very early stage. Future work to improve models like T5 still needs to be done. Much of the research work is spent researching new architectures, which shifts the focus of experimenting and achieving the best results with the current ones. Efforts should pass by exploring simple ideas, like exploring different training strategies such as on how to take full advantage of multi-task learning. A possible modification in our models to be explore is to add different training objectives related with the tasks to explore. For example, instead of minimizing the

loss for summarization, to modify this objective to minimize a combination of loss and a metric such as ROUGUE or QAEval [30]. Due to restrictions of resources, our experiments on the decoding phase were incomplete and are not reported on this document. This particular step also showed great promising. Future work should explore the the use of re-ranking techniques to select the best output based on a metric of our interest, This allows us to explore different outputs of our models. Techniques like beam search and sampling with multiple returned sentences allow for a multitude methods that can applied as an aide to the inference of our model. Another relevant topic we manage to explore fully and it is not described here is conditional decoding, where we restrict the type of tokens the model was able to predict, taken into account the task. This strategy seemed particularly promising for tasks like STS, were a very short output was generated. Finally, we suggest further exploration on a hot topic in the NLP community nowadays, which is quality evaluation. This paradigm is particularly relevant in machine translation and could potentially by applied to generation as a way of measuring good or bad summaries.

Bibliography

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. 6 2017. URL http://arxiv.org/abs/1706.03762.

[2] A. B. Abacha, Y. M'rabet, Y. Zhang, C. Shivade, C. Langlotz, and D. Demner-Fushman. Overview of the mediqa 2021 shared task on summarization in the medical domain. pages 74–85, 6 2021. doi: 10.18653/V1/2021.BIONLP-1.8. URL https://aclanthology.org/2021.bionlp-1.8.

[3] A. E. Johnson, T. J. Pollard, S. J. Berkowitz, N. R. Greenbaum, M. P. Lungren, C. ying Deng, R. G. Mark, and S. Horng. Mimic-cxr, a de-identified publicly available database of chest radiographs with free-text reports. *Scientific Data*, 6, 12 2019. ISSN 20524463. doi: 10.1038/s41597-019-0322-0.

[4] L. N. Phan, J. T. Anibal, H. Tran, S. Chanana, E. B. Glu, A. Peltekian, and G. Altan-Bonnet. Scifive: a text-to-text transformer model for biomedical literature. 2021. doi: 10.1093/bioinformatics/xxxxxx. URL https://www.ncbi.nlm.nih.gov/pmc.

[5] W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics 1943 5:4*, 5:115–133, 12 1943. ISSN 1522-9602. doi: 10.1007/BF02478259. URL https://link.springer.com/article/10.1007/BF02478259.

[6] D. Jurafsky, J. H. Martin, and J. Austen. Speech and language processing deep learning architectures for sequence processing time will explain. 2021. URL https://web.stanford.edu/~jurafsky/slp3/.

[7] J. L. Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990. ISSN 0364-0213. doi: https://doi.org/10.1016/0364-0213(90)90002-E. URL https://www.sciencedirect.com/science/article/pii/036402139090002E.

[8] S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 11 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

[9] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate, 2016. URL https://arxiv.org/abs/1409.0473.

[10] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu.

Exploring the limits of transfer learning with a unified text-to-text transformer. 10 2019. URL http://arxiv.org/abs/1910.10683.

[11] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi. The curious case of neural text degeneration, 2020. URL https://arxiv.org/abs/1904.09751.

[12] X. Meng, C. H. Ganoe, R. T. Sieberg, Y. Y. Cheung, and S. Hassanpour. Self-supervised contextual language representation of radiology reports to improve the identification of communication urgency. 2019. URL https://arxiv.org/abs/1912.02703.

[13] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. pages 4171–4186, June 2019. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423.

[14] A. Smit, S. Jain, P. Rajpurkar, A. Pareek, A. Ng, and M. P. Lungren. Chexbert: Combining automatic labelers and expert annotations for accurate radiology report labeling using bert. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing,EMNLP2020*, 2020.

[15] J. Irvin, P. Rajpurkar, M. Ko, Y. Yu, S. Ciurea-Ilcus, C. Chute, H. Marklund, B. Haghgoo, R. Ball, K. Shpanskaya, J. Seekins, D. A. Mong, S. S. Halabi, J. K. Sandberg, R. Jones, D. B. Larson, C. P. Langlotz, B. N. Patel, M. P. Lungren, and A. Y. Ng. Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison. *33rd AAAI Conference on Artificial Intelligence, AAAI 2019, 31st Innovative Applications of Artificial Intelligence Conference, IAAI 2019 and the 9th AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019*, pages 590–597, 1 2019. URL https://arxiv.org/abs/1901.07031v1.

[16] A. Romanov and C. Shivade. Lessons from natural language inference in the clinical domain. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018*, pages 1586–1596, 2020.

[17] Yanshan, S. Fu, F. Shen, S. Henry, O. Uzuner, and H. Liu. The 2019 n2c2/ohnlp track on clinical semantic textual similarity: Overview. *JMIR Med Inform 2020;8(11):e23375 https://medinform.jmir.org/2020/11/e23375*, 8:e23375, 11 2020. doi: 10.2196/23375. URL https://medinform.jmir.org/2020/11/e23375.

[18] Y. Wang, K. Verspoor, and T. Baldwin. Learning from unlabelled data for clinical semantic textual similarity. *Proceedings of the 3rd Clinical Natural Language Processing Workshop*, pages 227–233, Nov. 2020. doi: 10.18653/v1/2020.clinicalnlp-1.25. URL https://aclanthology.org/2020.clinicalnlp-1.25.

[19] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, July 2020. URL https://aclanthology.org/2020.acl-main.703.

[20] J. Zhang, Y. Zhao, M. Saleh, and P. J. Liu. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization, 2020. URL https://arxiv.org/abs/1912.08777.

[21] R. Bhagat and E. Hovy. What is a paraphrase? *Computational Linguistics*, 39:463–472, 9 2013. ISSN 0891-2017. doi: 10.1162/COLI_A_00166. URL http://direct.mit.edu/coli/article-pdf/39/3/463/1801912/coli_a_00166.pdf.

[22] W. B. Dolan and C. Brockett. Automatically constructing a corpus of sentential paraphrases. 2005. URL https://aclanthology.org/I05-5002.

[23] J.-P. Corbeil, P. Montreal, and H. A. Ghadivel. Bet: A backtranslation approach for easy data augmentation in transformer-based paraphrase identification context a preprint. 2020. URL https://www.kaggle.com/c/quora-question-pairs.

[24] A. E. W. Johnson, T. J. Pollard, L. Shen, L. wei H. Lehman, M. Feng, M. M. Ghassemi, B. Moody, P. Szolovits, L. A. Celi, and R. G. Mark. Mimic-iii, a freely accessible critical care database. *Scientific Data*, 3, 2016.

[25] A. Aditya. Paraphrase detection using deep learning. 2018.

[26] A. Stubbs, M. Filannino, E. Soysal, S. Henry, and Uzuner. Cohort selection for clinical trials: n2c2 2018 shared task track 1. *Journal of the American Medical Informatics Association*, 26(11):1163–1171, 09 2019. ISSN 1527-974X. doi: 10.1093/jamia/ocz163. URL https://doi.org/10.1093/jamia/ocz163.

[27] C.-Y. Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-1013.

[28] X. Yang, X. He, H. Zhang, Y. Ma, J. Bian, and Y. Wu. Measurement of semantic textual similarity in clinical texts: Comparison of transformer-based models. *JMIR Medical Informatics*, 8, 11 2020. doi: 10.2196/19735. URL /pmc/articles/PMC7721552//pmc/articles/PMC7721552/?report=abstracthttps://www.ncbi.nlm.nih.gov/pmc/articles/PMC7721552/.

[29] X. Yang, X. He, H. Zhang, Y. Ma, J. Bian, and Y. Wu. Measurement of semantic textual similarity in clinical texts: Comparison of transformer-based models. doi: 10.2196/19735. URL http://medinform.jmir.org/2020/11/e19735/.

[30] D. Deutsch, T. Bedrax-Weiss, and D. Roth. Towards Question-Answering as an Automatic Metric for Evaluating the Content Quality of a Summary. *Transactions of the Association for Computational Linguistics*, 9:774–789, 08 2021. ISSN 2307-387X. doi: 10.1162/tacl_a_00397. URL https://doi.org/10.1162/tacl_a_00397.