**FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University**

MASTER THESIS

Bc. Lukáš Chaloupský

**Automatic generation of medical
reports from chest X-rays**

Institute of Formal and Applied Linguistics

Supervisor of the master thesis: Mgr. Rudolf Rosa, Ph.D.

Study programme: Computer Science

Study branch: Software and Data Engineering

Prague 2022

First of all, I would like to thank my supervisor Mgr. Rudolf Rosa, Ph.D. for all his time, guidance and valuable advices he gave me while working on this thesis. I would also like to thank my parents for their unlimited support and patience during my studies.

Title: Automatic generation of medical reports from chest X-rays

Author: Bc. Lukáš Chaloupský

Institute: Institute of Formal and Applied Linguistics

Supervisor: Mgr. Rudolf Rosa, Ph.D., Institute of Formal and Applied Linguistics

Abstract: This thesis deals with the problem of automatic generation of medical reports in the Czech langugage based on the input chest X-ray images using deep neural networks. The first part deals with the analysis of problem itself including comparison of existing solutions from several common points of view. In order to interpret medical images in the Czech language we present a fine-tuned a Czech GPT-2 model specialized on medical texts based on the original pre-trained English GPT-2 model along with its evaluation. In the second part the created Czech GPT-2 is used for training neural network model for generating medical reports. The training was conducted on freely available data along with data pre-processing and their adjustment for the Czech language. Furthermore the model results are discussed and evaluated using standard metrics for natural language processing to determine the performance.

Keywords: natural language processing, image captioning, x-ray, medical report generation

Contents

Introduction

In hospital, inspecting the X-rays and writing corresponding medical reports is a hard work that requires experienced specialized doctors, of which there are not many. A great deal of people visit hospitals daily and X-rays are taken for many of them. Automatic interpretation of X-ray image has a great potential to improve health care and it could be particulary helpful to doctors in order to distinguish serious cases from the ordinary ones and overall accelerate and improve their work.

Automatic generation of radiology reports is a subset of a general problem called Image Capitioning, i.e. generation of overall textual captions to input images. Image captioning is a combination of Natural Language Processing and Computer Vision areas, experiencing a lot of progress in the last years. Most often the Image Captioning problem is solved using Deep Learning techniques. The specificity of this subset is that we do not want to generate just a general caption of the image, but the exact description of all findings contained in the given medical image. There were done multiple studies for this task in other languages but none in the Czech language.

Deep learning by its very nature has wide range of uses in a medical sector as it can often capture complex relations in any kind of data with excellent performance results. Nevertheless, in the medical environment the accuracy of predictions is crucial in order to determine the final diagnosis. Therefore, we should not consider the models as such as something that is unmistakably true, but as an auxiliary tool that should help doctors to examine X-rays.

Inasmuch as it is not so challenging to detect fractures on the limbs, this area is less interesting than others which have a variety of diverse possible problems. One of these areas is chest for which there exists multiple freely accessible datasets containing full textual medical reports. However, all these available datasets have one common downside, they are not in the Czech language. The natural question arises, where do we obtain these much needed data? We have to face and solve this core problem in our thesis.

Goals

First of all, we will take a closer look at the problem itself. This includes breaking down the problem and analyzing all its parts individually together with presenting possible existing alternatives for each part.

The main target of our thesis is to train a neural network for the purpose of generating textual medical reports for X-ray images. An example of our problem can be seen in Figure 1.1 for illustration.

Our first goal is to fine-tune a language model directly for the Czech language. The language model will be specialized directly to medical texts in order to capture the essence of the problem. However, ahead of the medical specialization, we want to fine-tune a general Czech language model. Fine-tuning will be based on the original English GPT-2 model presented in Radford et al. [2019].

Finally, we want to utilize our fine-tuned language model for training a neural network model interpreting chest X-rays images and generating corresponding medical textual reports to them in the Czech language. This section also involves the overall data preparation directly for the Czech language. In addition, the training will be done in multiple setups. All possibilities will be evaluated with the purpose of determination of their final performance.

Thesis structure

In the very first chapter we present a detailed description of our problem. Every aspect of our problem is introduced and all existing solutions or possibilities are discussed with their pros and cons. Moreover we introduce there some of the important related works.

Following chapter is dealing with the design of solution to our problem, with all reasosings and decisions made. This includes not only the final neural network model, but also the language model fine-tuning and data preparation.

All experiments done with our models take their part in the third chapter, describing all used scripts and differents setups together with data variatns.

Whole fourth chapter is then dedicated to evaluation of experiments done in the preceding part. Furthermore, our models will be compared to the performance of other existing solutions.

Finally, in epilog we discuss what we have accomplished in the thesis, what the resulting consenquences are and what the future possibilities are.

1. Problem Analysis

This chapter deals with the overall analysis of the problem itself. In the very beginning, we present the definition of the problem. Every aspect of the problem is further discussed in detail along with a comparison of possible solutions. Moreover, the next section of the chapter describes data we work with and their alternatives. The final part of this chapter presents some of the important related works.

1.1 Problem definition

First of all, we need to define the problem we are solving. The problem is defined as follows. At the input we are given an X-ray image of the patient and our goal is then to generate a corresponding textual report that will describe in detail every important aspect of the given radiograph, in particular it will describe as much as possible all the diseases actually present in the radiograph. Figure 1.1 depicts an example of our problem with a chest X-ray image along with its ground truth and predicted report.
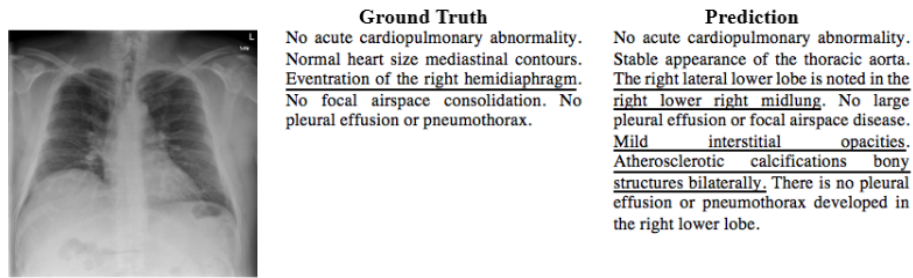
Figure 1.1: Example of an X-ray image along with its ground truth and predicted report. Jing et al. [2017].

1.2 Methods of generation

In the first part of the analysis, we will present posssible approaches for image caption generation. Inasmuch as almost all solutions for image captioning based on neural networks are using an encoder-decoder architecture as their core with further minor or major adjustments, we will discuss just this architecture. Encoder serves for the purposes of extracting visual features from the input images into intermediate vector representations for the decoder. The decoder is subsequently fed with these visual feature vectors and decodes them token by token into the natural language text.

Moreover, over the past years the encoder-decoder architecute has been enhanced using the attention mechanism. The problem of encoder-decoder architecture without any other mechanism is that all the information about the image have to be encoded globally inside a single vector. However, not all the information about the image is relevant for the caption generation. For this reason, the

attention mechanism was introduced. Instead of a single vector, the set of feature vectors representing the spatial information of the input image is used. This allows us to dynamically focus only on specific areas during the generation. The overall nature of the mechanism is derived from the way our brains concentrace on the images.

Attention mechanism for image captioning was first introduced in the Xu et al. [2015] work using the Bahdanau attention proposed in Bahdanau et al. [2014]. After extractions of visual feature vectors h_j from image, we want to assign a weight α_{ij} to each of these vectors indicating the relevance of the image position j when generating output at position i. This results in the context vector c_i, a dynamic weigthed combination of h_j features, that is presented as another input to the decoder. All attention calculations are defined as follows:

$$c_i = \sum_{j=1}^{t} \alpha_{ij} h_j \tag{1.1}$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{t} \exp(e_{ik})} \tag{1.2}$$

$$e_{ij} = a(s_{i-1}, h_j) \tag{1.3}$$

where s_{i-1} is a previous hidden state and $a(s_{i-1}, h_j)$ is an attention model.

1.2.1 Encoder

Encoder is responsible for extracting high-level visual features of the input image into one or more visual feature vectors. Moreover, the encoder can include also additional parts for other independent purposes, e.g. detection of objects in the image and their classification, which may be further passed to the decoder as a separate or an extra input. We will describe some of the most used neural network architectures utilized as an encoder.

1.2.1.1 Convolutional neural networks

Convolutional neural networks (CNN)[O'Shea and Nash, 2015] are the most leveraged type of neural networks for the purposes of image analysis. They apply 2D convolution filters for all positions in the image with the shared weights allowing to detect similar patterns independently on the position. Each layer of filters will downsize the dimensions of the previous layer and increases the number of filters. By gradually applying filters, the network is learning more complex patterns.

Instead of training convolutional neural networks from scratch, already pretrained CNN models are often used and possiblly further fine-tuned to a specific area. These models have been trained on a vast amount of data thanks to which they learned to recognize relevant features in the lower layers applicable to any kind of images. In contrast, training from scratch is costly and takes a much longer time. Among the used pre-trained CNN model architectures are for example ResNet[He et al., 2016], VGGnet[Simonyan and Zisserman, 2014], DenseNet[Huang et al., 2017] or EfficientNet[Tan and Le, 2019].

1.2.1.2 Transformers

With the work of Vaswani et al. [2017], a new neural network archtecture called tranformers was introduced. The transformers use multi-head self-attention mechanism to compute the relations between the elements in the sequences. Although the transformers were originally designated for NLP tasks, they can be also utilized for other domains like images due to their robustness. Vision transformers (ViT), presented in Dosovitskiy et al. [2020], is an adaptation of transformers encoder for image classification without any CNNs. The core of the encoder is identical to the one in the original transformer. Nevertheless, the input image is represented as a sequence of patches for which their embeddings are computed together with the positional encodings as the input, as we can see in the Figure 1.2.
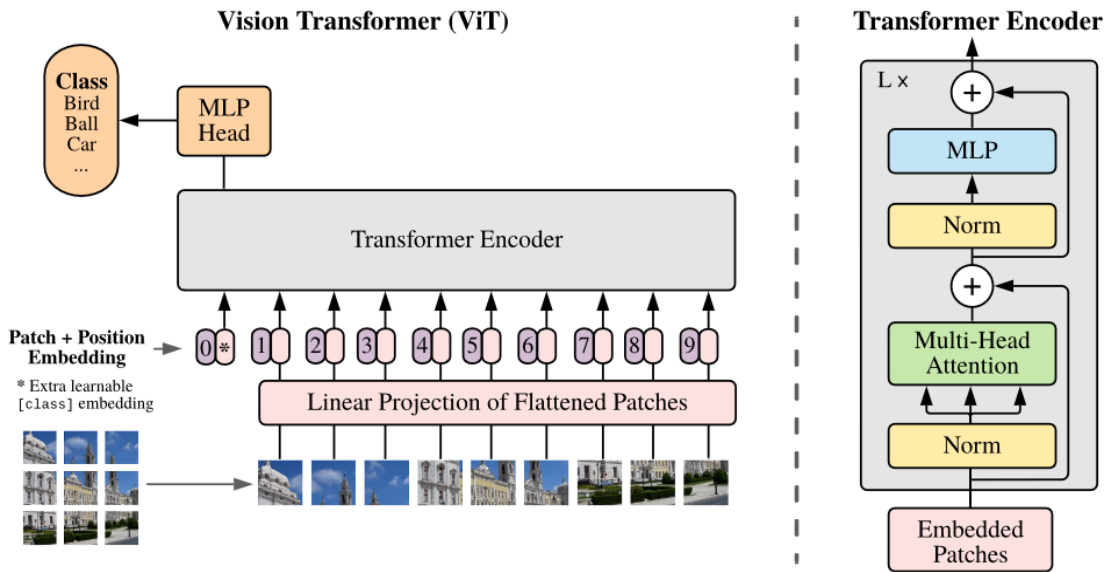
Figure 1.2: Visual Transformer architecture from Dosovitskiy et al. [2020].

1.2.2 Decoder

The second main part of the network is the decoder which serves as a language model for generating corresponding captions to input images. During the generation process the visual features of the image from the encoder and the already generated text are taken into account for the predtiction of the next token or word. In the process of generation, the attention mechanism (described above in the Chapter 1.2) is almost always incorporated in order to focus only on the substantial particular areas. The following subsections present common approaches used as the backbones for the decoder part of the network.

1.2.2.1 Recurrent neural networks

Recurrent neural networks (RNN)[Rumelhart et al., 1985] are a category of neural networks designated for sequence data processing. Context of the previous part of the sequence can influence the subsequent elements due to the RNN cell's internal memory (hidden state). Each time step RNN cell computes its activation

function from current input and hidden state producing updated hidden state as output. For language modeling tasks, the last generated token or word is given to the RNN as the next input until the entire text is produced. The autoregressiveness of the RNNs is the reason why they are ideal for use as a language model. The two most used types of RNN cells are LSTM[Hochreiter and Schmidhuber, 1997] and GRU[Cho et al., 2014], however the LSTM cells are more utilized as the decoder because they can remember longer sequences. Moreover, we can combine them in a hierarchical manner to capture more complex structures in the generated text. Neverthless, the downside of the RNNs is the training time due to their sequential nature. The whole sentence must pass through RNN token by token and cannot be parallelized. In addition, due to the sequential pass of the input, there is also another serious problem - the RNN may forget the information about earlier elements on long sequences because of the vanishing gradient.

1.2.2.2 GPT-2

Just as in the case of encoder, with the advent of transformers presented in the Vaswani et al. [2017] paper, the decoder part of transformers started to be used as the language model for image captioning task for their great results in the natural language processing (NLP) tasks. One of the advantages of the transformers against the previously used recurrent neural networks is the loss of the need for sequential processing during the training. Due to the fact that the processing of the whole caption can be done in parallel, the entire training process is significatnly accelerated. Moreover, the transformers can capture longer ranges dependencies in the text, as they process the sentence as a whole instead of sequentially by words.

One of the state-of-the-art autoregressive language models using transformers as their backbone is OpenAI GPT-2 model from the Radford et al. [2019] paper, which outperforms other language models on many NLP tasks. It was trained on a massive English dataset called WebText (introduced in the same article) containing a total of 40 GB of raw text. The resuling model is able to generate large coherent texts. Furthermore, it can be fine-tuned to a different domain or to a completely different language.

1.3 Data

In the previous part, we talked about possible methods of generation. Another crucial aspect we need to discuss are data, which are a basic building block of our thesis. This part focuses on the analysis of the data we used in our thesis, but also on their alternatives.

In order to solve our task and train neural network, we need to get dataset containing the X-rays images along with their textual descriptions and optionally some other attributes of the examined X-rays. Moreover, the fundamental feature we need is that the data must be in the Czech language.

1.3.1 Existing datasets

Medical environment provides a plenty of diverse potential problems, which can be researched. As already mentioned, in this thesis we focus specifically on the X-ray images. Because it is not so hard to detect fractures on the limbs, this area is not as interesting as others. One area that is rich in its diversity is the chest. As a result, this area is explored the most and therefore there exist multiple datasets with full textual medical reports. In the following section we describe some of them.

1.3.1.1 Indiana University chest X-ray

Indiana University chest X-ray dataset has become a standard in the field of medical report generation. It was presented in the Demner-Fushman et al. [2015] paper. This dataset is an open source collection of pairs of chest X-rays and their corresponding semi-stuctured textual radiology reports, which is freely available on the web[1] without any additional requirements. We have a choice if we want to download just reports or images and in either PNG or DICOM[2] format. The entire dataset consists of 7470 chest X-ray images that cover not only the frontal (PA[3]) view, but also the lateral (side) one. These images corresponds to a total of 3995 patients' medical text reports.

Figure 1.3 shows an example from the Indiana University chest X-ray dataset. Each dataset pair is carefully de-identified in order to remove any personal information. The text of the report is semi-structured in up to 5 sections. The most important sections are *impression*, where the overall diagnosis is stated, *findings* section describing the details of examination and *tags* which are of two types - manual and automatic. Manual tags were annotated manually using MeSH[4] and RadLex[5] codes, automatic were encoded from the reports using the MTI indexer.[6] The rest of the sections are *indication* and *comparison*.

The disadvantage of this dataset is that it is relatively small. On the other hand, it is a quite clean and manually checked dataset containing also additional information about images in a form of tags described above.

1.3.1.2 MIMIC-CXR v2.0.0

MIMIC-CXR v2.0.0 is another dataset consisting of full semi-structured medical textual reports against corresponding chest X-rays that was presented in the Johnson et al. [2019a] paper. As the previous dataset, it is openly available on the web[7]. In order to get access to the dataset, we have to go through registra-

---

[1] https://openi.nlm.nih.gov/faq#collection
[2] https://www.dicomstandard.org/
[3] Posterior-Anterior
[4] https://www.nlm.nih.gov/mesh/meshhome.html
[5] http://radlex.org/
[6] https://lhncbc.nlm.nih.gov/ii/tools/MTI.html
[7] https://physionet.org/content/mimic-cxr/2.0.0/

Figure 1.3: Sample from the Indiana University Chest X-ray dataset.

tion and verification steps. The verification phase includes completion of CITI[8] *Data or Specimens Only Research* course for *Human Subject Research*. Moreover we need somebody trustworthy as a reference to confirm the authenticity of our identity. After the verification we get access to all datasets in the same repository.

The dataset consists of 377,110 X-ray images in the DICOM format connected to a total of 227,835 radiology reports for 65,379 patients. Each report is structured into multiple different sections as we can see in the Figure 1.4. In order to satisfy legal requirements, entire dataset is automatically de-identified to remove any protected health information.[9] Similarly to the previous dataset, the essential two sections of each report are *impression* and *findings*. There also exists older MIMIC-CXR-JPG[10] dataset, presented in the Johnson et al. [2019b] paper. This is an older version of MIMIC-CXR v2.0.0 dataset consisting of the exactly same images, only in JPG format, but each image is assigned 14 labels indicating the presence of the category in the report instead of its textual form. Each category has assigned either a *1, 0* or *-1* label with the meaning *positively mentioned, negatively mentioned* or *uncertain*. The labels were determined from the reports utilizing the CheXpert[Irvin et al., 2019] and the NegBio[Peng et al., 2018] open-source labelers.

The advantage of this dataset is its vast number of samples. Moreover, as described above, we can get additional information in a form of categories to every image. Nevertheless, the textual reports carry some noise in them in the form of grammatical mistakes and incorrect formatting. We face these issues in the Chapter 2.3.2.

1.3.1.3   Other datasets

Apart from the datasets described above, other datasets with similar type are being used with the aim of solving our task. Amongst them belong datasets such as ImageCLEFmed Caption[Rückert et al., 2022], PadChest[Bustos et al., 2020], BCIDR[Zhang et al., 2017] and PEIR Gross[Jing et al., 2017]. Moreover,

---

[8]https://about.citiprogram.org/series/human-subjects-research-hsr/

[9]https://en.wikipedia.org/wiki/Protected_health_information

[10]https://physionet.org/content/mimic-cxr-jpg/2.0.0/
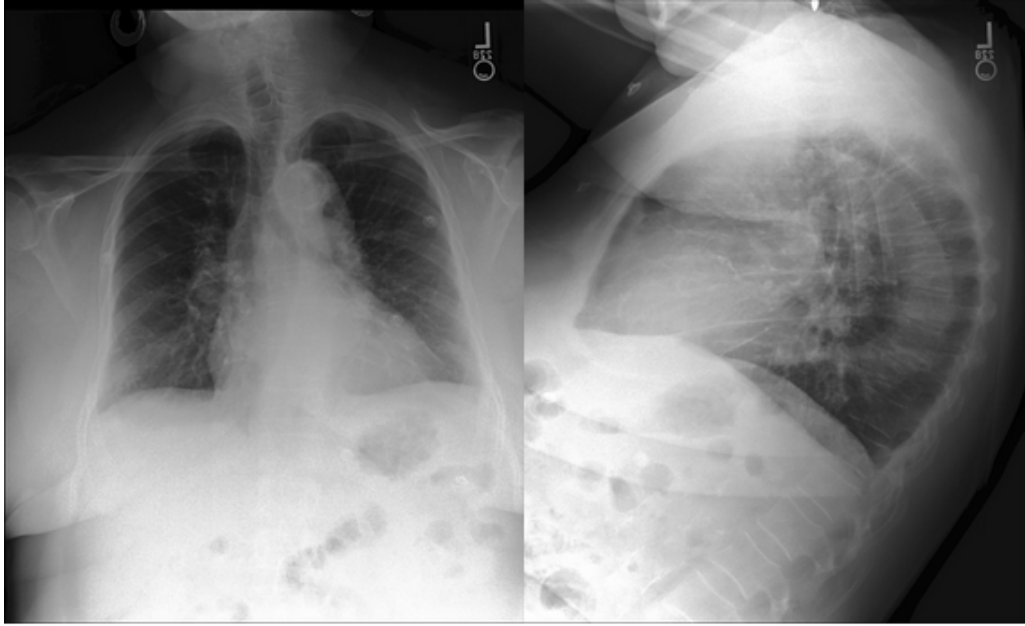
FINAL REPORT

EXAMINATION: CHEST (PA AND LAT)

INDICATION: History: ___F with dyspnea

TECHNIQUE: Chest PA and lateral

COMPARISON: ___

FINDINGS: Heart size remains mild to moderately enlarged. The aorta is tortuous and diffusely calcified. Mediastinal and hilar contours are otherwise unchanged. Previous pattern of mild pulmonary edema has essentially resolved. Mild atelectasis is seen in the lung bases without focal consolidation. Blunting of the costophrenic angles bilaterally suggests trace bilateral pleural effusions, not substantially changed in the interval. No pneumothorax is present.

IMPRESSION: Interval resolution of previously seen mild pulmonary edema with trace bilateral pleural effusions.

Figure 1.4: Sample from the MIMIC-CXR dataset.

except for datasets containing textual reports there exist a lot of other datasets worth mentioning containing different kind of information for each X-ray. These include, for example, CheXpert[Irvin et al., 2019], VinDr-CXR[Nguyen et al., 2020], ChestX-ray8[Wang et al., 2017] and its expanded version ChestX-ray14.

1.3.2 Czech data

All freely available datasets presented in the previous part have one common downside, namely they are not in the Czech language. As a part of elaboration of this thesis, an intesive communication with real Czech hospitals and other possible sources of real data took place. The goal of this communication was to create the very first open Czech dataset of this kind. Processing of this kind of data would mean not only preparing the data into suitable format but also it

would include proper anonymization of any personal information about the patients within the data.

However, inasmuch as the authentic patients data from hospitals are subject to strict privacy rules and we are not employees of any hospital, the institutions decided that they cannot provide the data in any way without the concious permission of patients given before the examination. With this result, we need to find a different way how to obtain this much needed Czech data.

1.3.3 Translators

In the previous sections, we discovered that there is no dataset in the Czech language for our problem and there is no easy way how to get acces to the real data in order to build one. The only thing left is to create a new artificial dataset using an automatic or manual translation. Manual translation would be expensive, time consuming and would require proper domain knowledge. Therefore, we will compare different freely accessible translators and choose the right one for our needs.

1.3.3.1 DeepL

At the moment, DeepL[11] translator provides the finest available translations beating even the ones from Google Translate. Moreover, it has freely usable web application and REST API. However, the main drawback of the DeepL translator is that its REST API is highly limited - only 500 000 characters per month can be translated for free. Furthermore, any translation above this limit is costly and thus this path is not appropriate for translating large textual datasets.

1.3.3.2 Google Translate

Google Translate[12] has become already de facto standard in the world of machine translation and it is the most used freely accessible language translation service in the world. In terms of quality, the translations are still great although little bit worse than those from DeepL. The web application is free of any charge and anybody can use it as much as he needs. Nevertheless, just as in the case of DeepL, their REST API services are limited and translation of anything above that limit is expensively charged. For these reasons, as in the previous case, we must find another way.

1.3.3.3 CUBBITT

Machine Translation[Akhbardeh et al., 2021] is an extensive area of research, as a result of which there exist many other projects and academic papers nowadays. One of them is CUBBITT[13] translator, which was developed at our faculty. The whole system is presented and described in detail in the Popel et al. [2020]

---

[11]https://www.deepl.com/translator
[12]https://translate.google.com/
[13]https://lindat.mff.cuni.cz/services/translation/

paper.

CUBBITT translator provides translations which are comparable to the ones from DeepL and Google Translate services. As other mentioned translators it provides an openly available web application for machine translation. Moreover and most importantly it provides REST API that is practically unlimited in text volume and free to use without any additional charges. These are the reasons why we will utilize CUBBITT in our thesis as a translator to create our artificial dataset.

On the other hand, CUBBITT has no support for auto-correcting input text compared to above mentioned services. Moreover, there are some patterns in the text which CUBBITT cannot translate at all or translates them incorrectly. These problems complicates our situation as the data from hospitals carry some natural noise in them. We face these complications in Chapter 2.3.2.

1.4   Related work

The last section of this chapter is dedicated to the description and comparison to some of the related works that solve identical or similar problem as we do.

The most significant related work is Alfarghaly et al. [2021] inasmuch as we base our thesis on it. This paper focuses on the identical problem as we do, only in the English language. The proposed solution uses a pre-trained and further fine-tuned Chexnet model, presented in Rajpurkar et al. [2017], as a visual features encoder and distilGPT2[14] language model[Sanh et al., 2019] as a decoder which is additionally conditioned on the visual features and predicted tag's word2vec[Mikolov et al., 2013] embeddings. For training the neural network the Indiana University chest X-ray dataset, described in more detail in Chapter 1.3.1.1, is used. Figure 1.5 shows examples of the model outputs.

Another paper making use of transformers is Chen et al. [2020]. The visual features of images are extracted using pre-trained convolutional neural network and they are further passed to the transformer encoder outputting hidden states, that are further presented to the transformer decoder for the report generation. However, the decoder architecture contains special memory module and also enhances the layer normalization. The memory module serves for memorization of text patterns which occur in the similar images inasmuch as they can further help for generating the report. We can see its effect in the Figure 1.6. Indiana University chest X-ray and MIMIC-CXR v2.0.0 datasets (see Chapter 1.3.1 for more details) were used as training data.

Yuan et al. [2019] paper proposes a hiearchical encoder-decoder architecture as we can see in the Figure 1.7 for the purpose of generating textual reports. Pairs of frontal and lateral X-ray images are used as an input to the network instead of single images, as the authors claim that the images should be complementary to
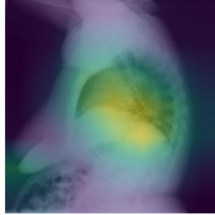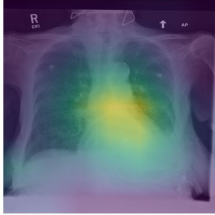
---

[14]https://huggingface.co/distilgpt2

	Normal	Accurate	Missing Details	False
Ground Truth	Negative. The cardiomediastinal silhouette is normal in size and contour. No focal consolidation, pneumothorax or large pleural effusion. normal.	Stable cardiomegaly with mild pulmonary interstitial edema. Unchanged cardiomegaly negative for pneumothorax or focal consolidation. No large effusion. Mildly prominent interstitial opacities.	Minimal left basilar atelectasis versus infiltrate. Low lung volumes. Normal cardiomediastinal contours. Low lung volumes with minimal left basilar opacities. No pneumothorax or pleural effusions.	Prominent hiatal hernia. left basilar opacity compatible pleural effusion and atelectasis. Right pleural effusion. no pulmonary edema / overt chf identified. stable senescent mediastinal contour.
Prediction	No evidence of active Disease. The heart size and pulmonary vascularity appear within normal limits. The lungs are free of focal airspace disease. No pleural effusion or pneumothorax is seen.	Cardiomegaly and mild interstitial pulmonary edema. Moderate cardiomegaly bibasilar and perihilar interstitial opacities. No pneumothorax	No acute cardiopulmonary abnormality The lungs are clear, and without focal air space opacity The cardiomediastinal silhouette is normal in size and contour. There is no mediastinal contour, large pleural effusion, pneumothorax, or focal airspace consolidation.	Interval left subclavian central venous catheter with tip approximating the high svc. No evidence of pneumothorax. Generalized heart size and mediastinal contours appear within normal limits. Atherosclerotic changes of the aorta. Moderate degenerative changes of the thoracic spinea.

Figure 1.5: Examples of generated medical reports from Alfarghaly et al. [2021].

each other instead of being processed independently. The RestNet-152 model pre-trained on the CheXpert[Irvin et al., 2019] dataset is utilized as the encoder with three outputs used later - the global and local features of the images, predicted observations and medical concepts. The decoder is hierarchical LSTM decoder comprising of the two parts: *sentence decoder* and *word decoder*. The sentence decoder takes visual features and generates hidden state for each sentence, which are along with the predicted medical concepts presented to the word decoder in order to generate the report. There are many other papers using a hierarchical architecture, e.g. Huang et al. [2019].

The last related project we mention in this work is CareBot,[15] which is being developed concurrently with our thesis. CareBot is a Czech startup founded in 2021 as a reaction to the then ongoing Covid-19 pandemic. As in our work, Care-Bot focuses its attention mainly on the processing of chest X-rays using neural networks. However, unlike us, it does not generate the textual reports for the doctors. Instead of textual reports, it focuses on finding and classifying a total of 15 different types of individual diseases on X-ray images and their subsequent spatial localization. Moreover, they already have a support from the medical environment. In their related papers [Kvak et al., 2022] and [Kvak and Kvaková, 2021] they as Resnet50V2 and DenseNet121 as the backbone atchitectures. However, their final architecture and methodlgy have not been publshed anywhere yet.

---

[15]https://www.carebot.com/
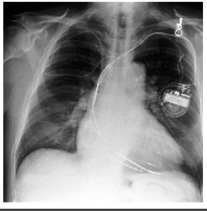
Ground-truth	BASE	BASE+RM+MCLN
In comparison with study of there is again enlargement *of the* cardiac silhouette *with a pacer device in place.* No *definite* vascular congestion *raising the possibility of underlying cardiomyopathy or* pleural effusion. *No acute focal pneumonia. The right picc line has been removed.*	*There is a left pectoral pacemaker with leads terminating in the right atrium and right ventricle. The heart is enlarged. There is* no *pneumothorax or* pleural effusion. *The lungs are clear.*	*In comparison with the study of there is little change in the appearance of the pacer leads which extend to the right atrium and apex of the right ventricle. Continued* enlargement *of the* cardiac silhouette *without* vascular congestion *or* pleural effusion. *No evidence of pneumothorax.*
Ground-truth *There are* low lung volumes. *Bibasilar* atelectasis *have minimally* improved. *Mild vascular congestion has minimally improved. There are* no *new lung abnormalities or* pneumothorax. *Bilateral* pleural effusions *are* small. *Right picc tip is at the cavoatrial junction.*	**BASE** *In comparison with the study of there is little overall change. Again there is some indistinctness of pulmonary vessels consistent with elevated pulmonary venous pressure.* No *evidence of acute focal* pneumothorax.	**BASE+RM+MCLN** *The* lung volumes *are* low. *There is a* small *left* pleural effusion with *associated* atelectasis. *The right lung is clear. There is* no pneumothorax. *The heart size is top normal. The hilar and mediastinal contours are normal. A right subclavian catheter terminate in the mid svc.*

Figure 1.6: Examples of generated medical reports from Chen et al. [2020].

Figure 1.7: Hierarchical architecture from Yuan et al. [2019].
E is encoder, D is sentence decoder and D' is word decoder.

2. Design

This chapter summarizes the overall design of our approach to generating textual medical reports for X-ray images. The problem consists of multiple independent parts we need to deal with - our apporach in Chapter 2.1, training Czech GPT-2 in Chapter 2.2 and translation of English data in Chapter 2.3. For each of them, we will present the fundamentals of our solution, along with description of related problematics and decisions made.
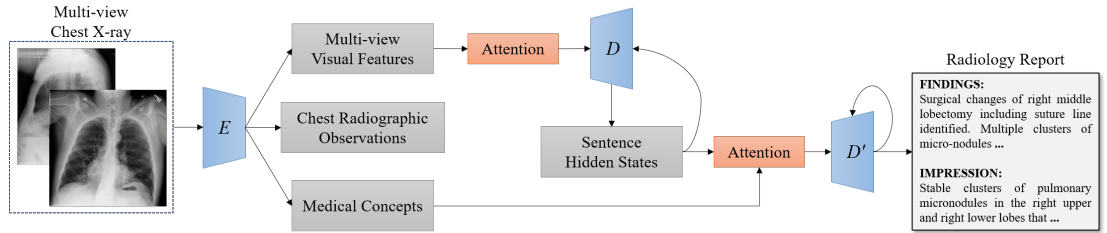
2.1 Our approach

As we already mentioned in the Chapter 1.4, the overall solution for the final medical report generation model is based on the Alfarghaly et al. [2021]. We have chosen this approach for multiple reasons. The main reasons to use this work as the backbone for our thesis are following:

1. In the work, the state-of-the-art GPT-2 model is utilized as the language model. This gave us a great opportunity as there was no Czech GPT-2 model available at the time this thesis began.

2. The encoder is already fine-tuned to extract visual features for specific dataset.

3. All solution source code is freely accessible on the github.[1]

As in most works for image captioning, the architecture is encoder-decoder based with an attention mechanism. The high-level solution architecture is depicted in Figure 2.1.

The encoder part utilizes fine-tuned Chexnet[Rajpurkar et al., 2017] as its visual backbone. The Chexnet is a CNN Densenet121 model trained specifically for medical enviroment on the ChestX-ray14[Wang et al., 2017] dataset predicting 14 distinct disease classes from the input chest X-ray images. Input images are resized to the 224×224 resolution as in the case of the original Chexnet model. Moreover, the encoder is further fine-tuned to predict 105 of the most common manual tags from the Indiana University chest X-ray dataset(see Chapter 1.3.1.1) for the purpose of wider range of possible semantic features. The model thus produces two types of outputs - visual features from the base CNN model and class scores from the multi-label tag prediction.

In order to obtain semantic features from the predicted classes, each predicted class score is multiplied by its corresponding word2vec[Mikolov et al., 2013] embedding, trained specifically on the biomedical texts, from the McDonald et al. [2018] and resulting in the weighted embedding matrix. Both the embedding matrix and visual features are further passed to the language model as the context vectors for the self-attention mechanism.

---

[1]https://github.com/omar-mohamed/GPT2-Chest-X-Ray-Report-Generation

Instead of the original English GPT-2, the original solution used its smaller version distilGPT-2.[2] The distilled version differs in the number of layers as it contains only a half of the layers compared to the original model and predicts only 512 tokens instead of 1024. Nevertheless, distilled versions of GPT-2 do not usually work well for languages other than English due to their reduced capacity. For these reasons, we use the original architecture of the GPT-2 for further fine-tuning. But, our fine-tuned model will predict 512 tokens as well as the distilGPT-2, so we can compare it to a similar setup, reduce the training time, and use a larger batch size. During the medical report generation training, the maximal sequence length is set to 200 due to the fact that the reports in the dataset are shorter.
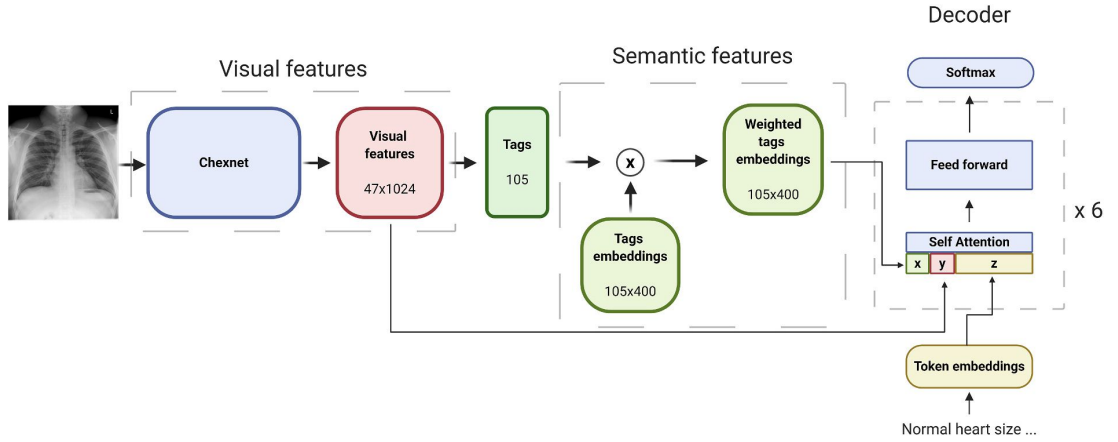
Figure 2.1: Overall architecture used in our solution proposed in Alfarghaly et al. [2021].

2.2 Czech GPT-2

The aim of this work is to generate medical reports in the Czech language. In the previous parts, we decided to use the GPT-2 as the language model. However at the time of the beginning of this work, no Czech GPT-2 model was freely available and thus it was essentially necessary to create one. The recently published GPT-2 is described in the Hájek and Horák [2022]. This section describes all the steps needed for fine-tuning the small English GPT-2 to the Czech language. Respectively, we train two versions of the Czech GPT-2 model. One trained on the general Czech textual data and one specialized specifically on Czech medical texts. The reason is that we have only small amount of medical data, that is insufficient to directly train a medical Czech GPT-2 model. Instead, we apply a gradual approach, where we train the general Czech model on large data and then only fine-tune it on the medical data.

---

[2] https://huggingface.co/distilgpt2

2.2.1 Data

In this section we will describe the possible data applicable for training of both the general and medical Czech GPT-2 model together with the decision made about the final data selection and data cleaning.

2.2.1.1 General

For the training of general Czech GPT-2 we have a plenty of data options we can choose from. However, there are important properties of the data we need to sastisfy. As we are transfer learning from English to Czech language, we need to have sufficiently large data, so we ensure the GPT-2 will learn properly syntactical and semantical information. Moreover, the data have to be also heterogeneous enough, so the model can capture different types of information and not just for example newspaper articles from a specific area. In order to create a good enough general model, we need to meet these criteria.

Several different datasets were investigated and tested for the training of the general Czech GPT-2 model.

Czech Wikipedia

The first data we used for training the model is the Czech Wikipedia dump[2]. After extraction, the total size of the dataset is approximately 800 MB of raw text. The advantages of this dataset are its easy accessibility and fairly clean data quality. On the other hand, the data are very homogeneous despite the various topics. Each article is written in the general descriptive style. Moreover the data themself are not large enough, the trained model made many both syntactical and semantical mistakes during the text generation.

Balanced Czech National Corpus

Another possibility was to use a balanced version of the Czech National Corpus[Křen et al., 2021] as the original is composed mainly of journalistic articles. The balanced version[3] tries to equalize the amount of data from each category. These categories include *journalism*, *poetry*, *prose*, *educational literature* etc. The major advantage of this dataset is its purity, the texts are syntacticaly correct without any undesirable non-Czech elements and written in the standard Czech language. The dataset does not have any significant downsides and the trained model understood Czech language without any significant ailments. In total, the dataset is comprised of 3,3 GB of raw text.

OSCAR

OSCAR, from the Ortiz Su'arez et al. [2020], is a huge deduplicated multilingual corpus created from the Common Crawl corpus[3] providing data for 166

---

[2]https://dumps.wikimedia.org/cswiki/latest/cswiki-latest-pages-articles.xml.bz2

[3]We would like to thank Tomáš Musil forthe preparation and balancing of the Czech National Corpus data.

[3]https://commoncrawl.org/

different languages and available directly in the huggingface datasets library.[4] It consists of the text scraped from websites of very different kinds and thus the data are heterogeneous enough. Moreover, its huge size, as the Czech part of the dataset occupies a total of 24 GB of raw text, is another major benefit. On the other hand, because the data are automatically scraped, they carry a noise in them. Besides that, not negligible part of the text are in the non-standard Czech language as the data come from diverse web sources such as forums etc. Nevertheless, the disadvantages are outweighed by the huge size of the corpus and together with following filtering of the text:

1. We take only texts that are at least 1200 characters long as these texts tend to be longer articles written in the standard Czech language instead of advertisements, incomplete texts etc.

2. Any texts containing control characters are filtered out, because the text contains generally undesirable content.

Conclusion

We analyzed various datasets along with their overall properties. Furthermore, the advantages and disadvantages of each were discussed. As a result, we have chosen the **OSCAR** dataset due to its size and heterogeneity. The **Wikipedia** is too small and homogeneous. On the other hand, the **Balanced Czech National Corpus** is heterogeneous enough, however it is almost an order of magnitude smaller than **OSCAR**.

2.2.1.2 Medical

Since we will have a trained general Czech GPT-2 model from the previous section that already understands Czech language, the final fine-tuning for medical environment does not require that much data. We need to specialize the model to understand the mecial environment inherently. For this purpose, we use a subset of the UFAL Medical Corpus v. 1.0.[5] These data are further filtered to remove any inappropriate characters, lines and redundant structures. As a result, the data contain a total of 100 MB of raw medical texts. The texts comprise of general medical descriptions, articles and package leaflets for medicines.

2.2.2 Training

In this part, we will describe the fine-tuning process of our Czech GPT-2 models. Both the general and the medical GPT-2 models are trained using the same process. The solution is based on the Guillou [2020] article using specific fastai[6] library providing with powerfull tools using best practices aiming for training and fine-tuning neural network models. The reason why we used this article is that it has shown great results for transfer learning the English GPT-2 to Portuguese in a short time againts traditional fine-tuning. The whole process comprises of several important parts, that we will describe in the subsequnet part of the text.

---

[4]https://huggingface.co/datasets/oscar
[5]https://ufal.mff.cuni.cz/ufal_medical_corpus
[6]https://www.fast.ai/

Detailed information about the insides and experiments done are described in the following Chapter 4.2.

Training tokenizer

First thing that needs to be done in the whole training process is to train a tokenizer specifically for the Czech language. As in the case of original GPT-2 model, we use the same byte-level byte-pair-encoding[Sennrich et al., 2015] tokenizer dividing input text into tokens (a word or its part). Original size of the vocabulary is kept and set to 50257, as well as the original special token $< |endoftext| >$ as indication of the beginning/end of the sequence token and the pad token. The entire prepared datasets from Chapter 2.2.1 are used for the tokenizers training.

Initialization

As we have indicated several times, for the GPT-2 model weights initialization we will use the weights from the original English GPT-2. However, we will do a modification of the initial word token embeddings.

For the word token embedding, we will compare which tokens have the English model tokenizer and our tokenizer have in common. The embeddings of these tokens will remain unchanged, as they have been already trained on a large text corpus and they already carry information, even though they come from a completely different language. The rest of the tokens will be initialized using the mean value of the English word token embeddings.

Data preparation

Another step in the process is to prepare the dataset into a suitable structure for the training. Entire dataset is loaded and pre-tokenized in advance in order to reduce the data transfer time between CPU and GPU needed. After the pre-tokenization process, all texts are further passed to a specific language modeling data object that is responsible for preparation and handling of training and validation data. All texts are concatenated, split by the defined sequence length and formed into the batches. We use bach size of 16 as it is the maximum size we were able to fit into the GPU and sequence length of 512, because we need to have model correponding to the one used in Alfarghaly et al. [2021] - distilGPT2 - as our main goal is to generate medical reports.

The batch size is one of the important hyper-parameters. In the following section we will discuss, how learning rates are determined and also that we will use higher learning rates. Nevertheless, Smith [2018] recommend to use batch size as large as possible, because higher learning rates are regularization themself and therefore other regularizations can be reduced.

Finding optimal learning rate

Fine-tuning is a very fragile process in terms of learning rates. Right choice of the learning rate is essential, if we choose a very small learning rate, the model

will train slowly and will tend to overfitting. On the other hand, if we choose too high learning rate, the whole process can diverge and all the progress will be lost.

Smith [2017] came with a solution to this problem. Before the training itself, we do a pre-training run in which we are trying a wide range of learnings rates for which we monitor their behaviour. Starting from a very low learning rates up to the very high learning rates, for every mini-batch we try a current learning rate, collect the resulting loss and move to the next iteration with a little higher learning rate. In the end, we plot collected losses against the corresponding learning rates.

An illustration of the result of this process can be seen in the Figure 2.2. The graph gives us an overview for which learning rates the model is still learning. General recommendation is to use a learning rate that is an order of magnitude less than the minimum as the minimum is very close to the moment of divergence. We can also see in the graph that the current implementation gives us several other significant points usable for training.
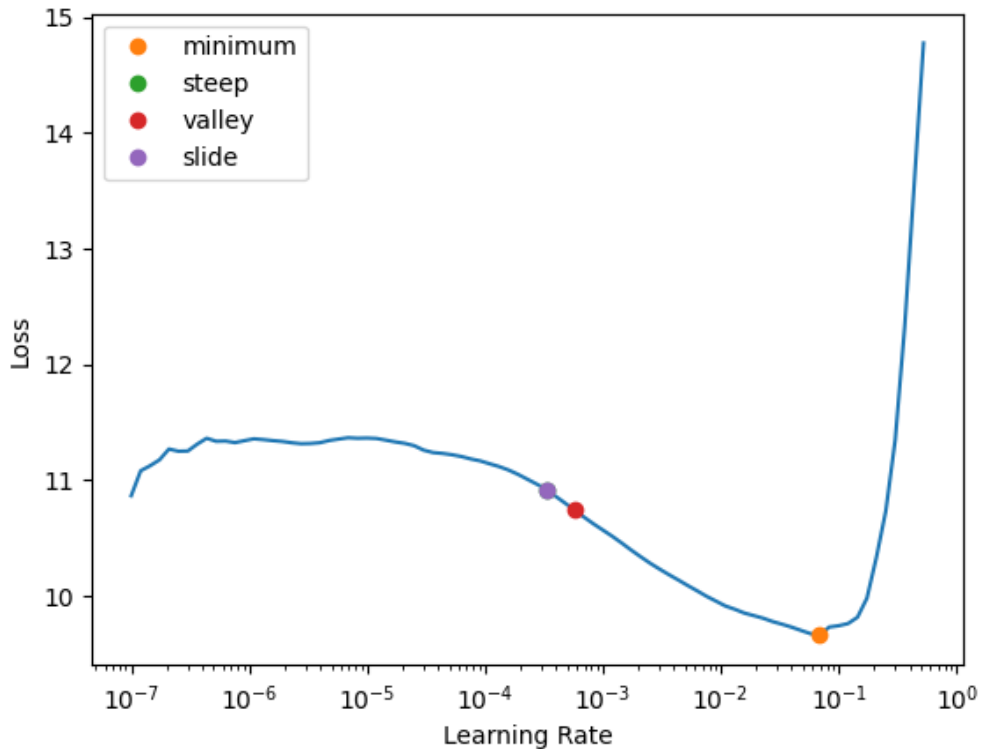
Figure 2.2: Learning finder output.
The *Learning rate* axis is in logarithmic scale.

Training process

We have prepared everything we need in the previous parts. However, we still have to describe the essence of our fine-tuning process. We will first describe the learning rate schedule. Fine-tuning or transfer learning is usually done using small learning rates with decay, so the general information in the already pre-trained weights are kept while the model is specialized to a selected task. The

disadvantage of this method is the long training time. Instead, we use a different approach from Smith [2018] called "1cycle" policy that is parametrized by *minimal* and *maximal* learning rates. In the beginning we start with a *minimal* learning rate and linearly increase it up to the *maximum*. The second phase is then the opposite direction going down with cosine annealing, but instead of stopping at the *minimum* the decrease continues down by several order of magnitudes lower as we can see in the Figure 2.3.

The intuition behind this policy is quite natural. In the beginning we start with a lower learning rate to find an optimal direction. As we are increasing the learning rate we are taking bigger steps this direction, skipping sharp local minima and preferring wide flat local minima area as shown in the Smith and Topin [2019]. The rest of the training is intented for improvement inside this area. This allows us to use higher learninng rates and thus overall accelerate the entire training process. The *maximal* learning rate should be chosen according to the previous section and the *minimal* is defined as $min = max/25$ by default.

Another major part of the training are discriminative learning rates. This method has been proposed in the Howard and Ruder [2018]. Instead of training all layers at once, we assign a different learning rate for each layer or a group of layers as each of them. This arises from the reasoning about what the different parts of the network are focusing on. Therefore, we split our model into 4 distinct groups of layers and train each of them with different learning rate.

The fine-tuning process itself is divided into two parts: training the new head for the Czech language only and training the whole model at once. In the original article, they suggest gradual unfreezing approach, where they unfreeze one more layer each time and train for one epoch. However after numerous experiments, we observed that unfreezing of the whole model and running multiple epochs at once gives better results.

2.3 Medical dataset translation

For machine learning in general and NLP tasks especially, the data quality is the alpha and omega of the performance of the final model. Since, as we have already mentioned in the Chapter 1.3.2, we do not have any Czech data directly for the medical examinations of X-rays, to obtain Czech data we must arrange ourselves in a different way. One potential way is to create a new artificial dataset using machine translation of existing datasets. This section discusses the required steps to build a quality dataset using translation.

2.3.1 Translator choice

In the previous part of the text, specifically in the Chapter 1.3.3, we already discussed several possibilities for automatic translation. Our final choice for the translator is the CUBBITT as it provides REST API unlimited in the number of requests and volume.
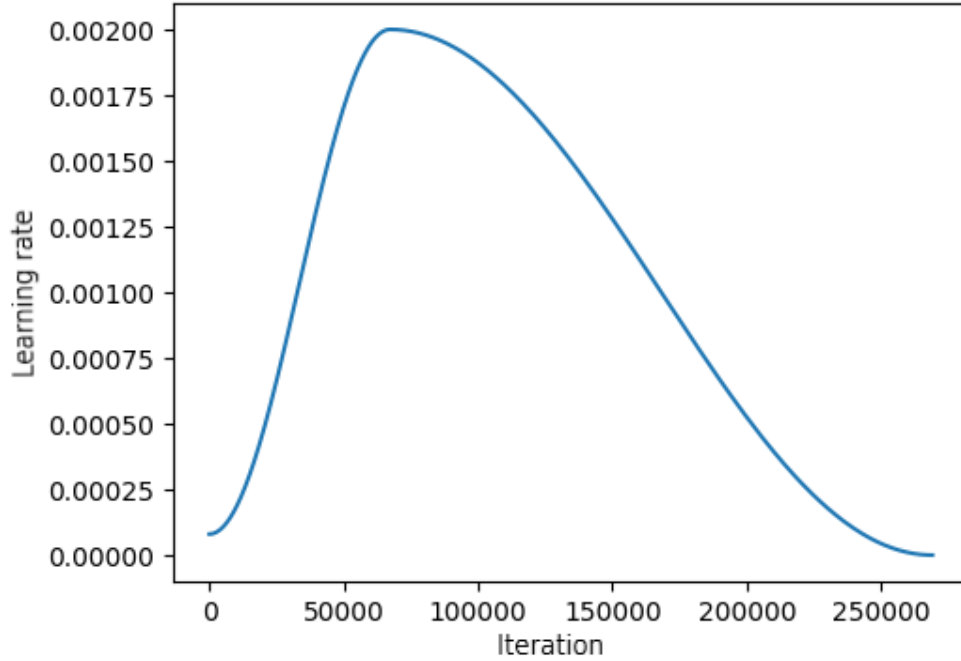
Figure 2.3: Learning rate schedule for 1cycle policy training.
The graph depicts the learning rate throughout the entire training process.

2.3.2 Preprocessing

The most important part of our machine translation process is the preprocessing of the input text. We already outlined in the Chapter 1.3.1 that the data contain some noise in them as we are dealing with reports in natural language. Moreover, we will use CUBBITT translator, that doest not perfom autocorrection itself and cannot translate some patterns at all as we described in the Chapter 1.3.3.3. For these purposes we incorporate preprocessing before the translation as it would be beneficial to have all texts in standardized form in order to firstly, help CUBBITT with translation to get the report correctly translated, and secondly to ensure that our model receives and process all the data in an indentical report format.

Our preprocessing pipeline encompasses of following procedures. Some of them are dealing with general CUBBITT issues and other with specifics of the medical data.

Line starts

First of all we start with a very simple procedure. We analyze all lines of the report and remove all whitespaces at the beginning and end of lines that are common to all lines. The purpose of this modification is to standardize the report format and get rid of unneccessary whitespaces while preserving its structure.

Anonymous sequences

Inasmuch as the whole datasets are anonymized due to the legal reasons and privacy protection, the reports contain "anonymous sequences", such as "XXXX"

or "___", denoting places with original private information about patients. However, these sequences can be attached to surrounding words or numbers forming undesirable words. As we already said, the CUBBITT does not auto-correct its input automatically, so these inputs will not be handled in any special way and therefore could be translated incorrectly. For this reason, we separate these sequences to form independent words.

- "old man with left ___rib" →
  "starý muž s levým ___rib" (not translated *rib*)

- "old man with left ___ rib" →
  "starý muž s levým ___ žebrem" (translated *rib*)

Units

Subsequent form of correction that we perform is the separation of numbers and units attached to them. In addition, this also includes general cases, where the number and subsequent word are glued together, while keeping specific medical terms with a similar structure. This is associated with the following step, as the units will not be true-cased properly without this procedure.

- 10cm → 10 cm

- 3VD → 3VD (medical abbreviation)

True-casing

The most imporant part of the whole preprocessing pipeline is true-casing of the input text. This adjustment is necessary for two essential reasons. CUBBITT has problems with translation of any uppercase texts in general. Capturing the true-case of a text is a complex problem requiring either a large statistical language dictionary or a trained model in order to properly determine the case.

As medical reports are a very specific area, the existing solutions for general text true-casing are inapplicable. Medical reports contain a lot of abbreviations and acronyms, which can be often confused with ordinary English words. Training the model for medical true-casing requires even more specific data for a certain domain, because in different contexts the common words can be treated differently. Moreover, obtaining flawless data to cover the entire specific domain is a challenging task. For these reasons, we chose the way of a statistical dictionary. Before the translation of the dataset begins, we create a statistical dictionary from the whole dataset containing the most often used form of every word. We also count with some exceptions, such as headings, that in some datasets can be in uppercase only.

Using the created dictionary, we deal with all uppercase words to assign them the proper form. The results of the true-casing preprocessing are demonstrated in the following examples, where *1a* and *2a* are translations without true-casing and *1b* and *2b* are translations with true-casing:

- (1a) "EXAMINATION: CHEST (PORTABLE AP)" →
  "PŘEZKOUŠENÍ: CHEST (PORTABLE AP)"
  RE-EXAMINATION: CHEST (PORTABLE AP)

- (1b) "Examination: Chest (portable AP)" →
  "Vyšetření: Hrudník (přenosný AP)"
  Examination: Chest (portable AP)

- (2a) "SMALL RIGHT PLEURAL ABNORMALITY" →
  "MALÉ PRÁVO PLEURÁLNÍ ABNORMALITY"
  SMALL LAW OF PLEURAL ABNORMALITY

- (2b) "Small right pleural abnormality" →
  "Malá pravá pleurální abnormalita"
  Small right pleural abnormality

Paragraphs structure

In some of the medical reports the section headings and corresponding texts do not begin on the same lines. We adjust these situations to a form where each heading and the text belonging to it always starts on the same line, for two reasons. Firstly, we want to normalize the report structure in general and secondly, we want to move the section content as close as possible to the heading, so the translator and even the final model have the context close to each other.

Capitalization

Another part of the preprocessing pipeline is a simple capitalization of each heading and each sentence in the report. This text capitalization process helps CUBBITT not only in the case of medical data, but in general during the translation process of some texts to better understand the boundaries between sentences.

Time

One of the patterns that CUBBITT doest not recognize nor auto-corrects itself in general are times. This applies both to the specification of hours and minutes, and to the part of the day specification. If any of these parts are incorrectly formatted, the time will we translated incorrectly or not translated at all, and thus we would lose some information or it could damage the fluency of the translated text. For these reasons, we apply preprocessing to normalize all times. We can see the difference in the following examples:

- "Chest radiograph at 1045PM" → "Rentgen hrudníku v 1045PM"

- "Chest radiograph at 10:45 PM" → "Rentgen hrudníku ve 22:45"

White spaces

After all previous procedures, we apply one last very simple final modification, namely, we squash all the whitespace characters inside each line into a single space, while maintaining the format from the very first preprocessing step described above. This step is performed only for normalization purposes.

Lowercasing

The last preprocessing procedure we will mention is lowercasing of the whole text followed by capitalizing the first word of each sentence. This is a separate procedure that was used in the earlier phases of elaboration of this work, because CUBBITT has problem with uppercased words as we already mentioned. This process has been replaced by better true-casing process and is not used in the final version.

3. Implementation

Entire implementation part of our thesis is divided into three independent parts. Each of them deals with different task desribed in the previous chapter.

The entire code is written in Python 3[1] with the use of tensorflow[2], pyTorch[3], huggingface[4] and fastai[5] libraries as they are the standard for working with neural networks. We want to emphasize that different parts of the project may require different dependencies. In order to solve this, virtual environments can be used. To install all the required dependencies, run the following command in the appropriate part of the project.

```
> pip3 install -r requirements.txt
```

3.1   Czech GPT-2

The implementation of the GPT-2 related part comprises of several scripts. This section describes all of them with the necessary level of detail. Scripts related to the GPT-2 training and subsequent work are prefixed with *gpt2*.

Training

The most important script is *gpt2_train.py* that takes care of the entire training process. The script is parametrized with numerous arguments such as base GPT-2 model, required sequence length, path to data etc. The data are expected to be inside the *storage/data* directory. In the beginning, the script creates all directories necessary for the training and its results - this includes the directories for the final model (*storage/models*), checkpoints (*storage/checkpoints*), intermediate training data (*storage/training_data*) and tokenizers (*storage/tokenizers*). Inside each of them, a specific directory is created for the model or the dataset so that different run settings do not overwrite each other's data . After the initialization, the script runs according to the description of the process in Chapter 2.2.2. Further, the script also allows to resume the training from the last saved checkpoint. Final trained model is saved in both tensorflow and pyTorch version at the end along with its corresponding tokenizer in the *storage/models* directory. Description of all arguments, that can be entered at the input, is directly inside the script. The script can be run as follows:

---

[1]Different parts may require different versions of Python

[2]https://www.tensorflow.org/

[3]https://pytorch.org/

[4]https://huggingface.co/

[5]https://www.fast.ai/

```
> python3 gpt2_train.py [-h] [--type {gradual,full}]
              [--model MODEL] [--max_len MAX_LEN]
              [--pretrained_weights PRETRAINED_WEIGHTS]
              [--dataset DATASET] [--data_path DATA_PATH]
              [--batch_size BATCH_SIZE]
              [--train_data_ratio TRAIN_DATA_RATIO]
              [--sequence_length SEQUENCE_LENGTH] [--debug DEBUG]
              [--resume_training RESUME_TRAINING]
              [--find_learning_rates FIND_LEARNING_RATES]
              [--find_learning_rates_epoch FIND_LEARNING_RATES_EPOCH]
              [--pretokenize_data PRETOKENIZE_DATA]
              [--save_checkpoints SAVE_CHECKPOINTS]
              [--learning_rates LEARNING_RATES [LEARNING_RATES ...]]
```

Testing

The *gpt2_generate.py* script serves for the text generation purposes. It is applicable to any huggingface GPT-2 model whose model name or path is taken as an argument. Moreover, all parameters that can be passed to the general huggingface GPT-2 *generate* method can be also passed as arguments of the script such as *top_p*, *top_k* etc. The ouput of the script is the texts generated by the given GPT-2 model. The script can be run as follows:

```
> python3 gpt2_generate.py [-h] [--max_len MAX_LEN]
                            [--model MODEL]
                            [--top_k TOP_K]
                            [--top_p TOP_P]
                            [--repetition_penalty REPETITION_PENALTY]
                            [--temperature TEMPERATURE]
                            [--do_sample DO_SAMPLE]
                            [--num_return_sequences NUM_RETURN_SEQUENCES]
                            [--num_beams NUM_BEAMS]
                            [--length_penalty LENGTH_PENALTY]
                            [--bos_token_id BOS_TOKEN_ID]
                            [--eos_token_id EOS_TOKEN_ID]
                            [--pad_token_id PAD_TOKEN_ID]
```

Data preparation

The last important script is *gpt2_data_utils.py*. As the *gpt2_train.py* script needs the data to be in a specific format, this script provides tools for their preparation. For the specified data location, it goes through all files with *.txt* or other user specified extensions. Each of the files can be further split after each number of lines or by a given delimiter. Finally, two files are created- one *.txt* file with all texts merged inside the file for the tokenizer training and one *.csv* file, where each split part corresponds to one row, for the GPT-2 fine-tuning itself. The script can be run as follows:

```
> python3 gpt2_data_utils.py [-h] [--folder FOLDER]
                             [--text_delim TEXT_DELIM]
                             [--regenerate REGENERATE]
                             [--line_split LINE_SPLIT]
                             [--extensions EXTENSIONS]
```

For the OSCAR dataset (and generally any other dataset from hugging-face), we implemented the *DatasetsWrapper* class encapsulating the huggingface datasets classes, that provides us with better data handling, direct entry text iteration, control characters filtering and offers us the ability to save data directly to a specified file while preserving all the original functionality of the huggingface classes. The class is implemented in the *datasets_wrapper.py* file.

3.2   Medical data translation

Implementation details of all related medical data translation source codes are described below.

To translate medical reports from English into Czech, the *dataset_translate.py* script was utilized. It is designed to be versatile and extensible for use on any dataset and any translation service. One can define its own *Translator* or *Extractor* class, described in the following parts of text, and run the script with different data or translator. Moreover, the preprocessing pipeline can also be customized to specific requirements. The script can be run as follows:

```
> python3 dataset_translate.py [-h] [--translator {cubbitt,deepl}]
                               [--dataset {mimic,openi}]
                               [--data DATA]
                               [--preprocess {lowercase,pipeline,none}]
                               [--preprocess_only PREPROCESS_ONLY]
                               [--anonymous_seq ANONYMOUS_SEQ]
```

It takes several arguments, such as the data path, and runs the translation. The script runs in multiple threads, so it can make efficient use of system resources and therefore speed up the overall translation. The final report translations are stored in the *translations* directory, where each run is uniquely identified by its start date, dataset name etc., while preserving the original dataset structure. If the user do not want to translate the dataset and only preprocess, the behaviour of the script can be changed with the *--preprocess_only* argument.

For each file, the script first loads the report text using the specified *Extractor*. Next, the text goes through each of the *Preprocessors* to process the text. After the preprocessing finishes, the report is passed to the selected *Translator* to get the translated text. Finally, the translated report is saved under the original file name. Throughout the entire translation process, the progress indication of how much data has been already translated is displayed.

All implemented preprocessors, described in Chapter 2.3.2, are located in the *preprocessors* directory. Each of them implements the *Preprocessor* interface with only one method *preprocess*, which makes it extensible for possible additional user defined preprocessors. This method takes the text as its only argument and returns the preprocessed text.

In our thesis we utilize CUBBITT as our translator. However, the user can implement its own translator class as the main script for dataset translation works with *Translator* interface. This interface defines two methods - *translate* that should return the response object from the translation service, and *get_text* that should get or create the translated text from the response. During the elaboration of this thesis we also implemented a translator for the DeepL, but it is unusable in practice due to its strict limits. All imlemented translators are located in the *translators* directory.

The last extensible part of the translation process is text extraction from the dataset files. Since not all datasets are just simple *.txt* files, we created the *Extractor* interface for this purpose. It defines only one method *extract_report* intended for full report text extraction. We implemented two extractors inside the *extractors* directory - for the Indiana University chest X-ray and MIMIC-CXR v2.0.0 datasets.

We also covered the important parts with the *pytest* test suites in the *tests* subdirectory. The tests can be invoked by:

```
> pytest tests/
```

3.3 Medical report generation

This section describes all implementation details for the final medical report generation task.

The dataset for training and testing the trained models must preserve the original format of *.csv* files. The script for the modification of the original *.csv* files can be run as follows:

```
> python3 modify_csv_to_translations.py [-h]
                                [--original_csv_dir ORIGINAL_CSV_DIR]
                                [--translations_dir TRANSLATIONS_DIR]
                                [--output_dir OUTPUT_DIR]
```

First, follow the setup steps described in the project *README*. Next, in order to run a training for our GPT-2 models and Czech data, several steps need to be carried out:

- In the *IU-XRay* directory, the English report files must be swapped with their Czech translations - *all_data.csv*, *testing_set.csv*, *training_set.csv*.

- The desired trained GPT-2 model should be placed inside the top level of the directory.

- Inside the *train.py*, *test.py*, *tokenizer_wrapper.py* files, the original "distil-gpt2" and "gpt2" model and tokenizer string identifiers should be changed to the name of our trained GPT-2 model.

- Change hyperparameters in *configs.py* to adjust training parameters as needed.

After all the mentioned steps, we can run the training or testing procedure using the following commands:

```
> python3 train.py
> python3 test.py
```

The checkpoints are gradually stored during the training process in the *checkpoints/CDGPT2* directory. After each epoch, the latest model checkpoint is stored. Moreover, the best training checkpoint is stored in the *best_ckpt* subdirectory. The testing process uses the model checkpoint stored in the *checkpoints/CDGPT2* directory.

For direct prediction on individual images, we implemented an additional script *predict_for_img.py*, which loads the checkpoint in the same way as in the testing process and the user is then prompted to enter the location of the chest X-ray image for which the prediction should be made. The corresponding medical report is then printed to the standard output. The script can be run in the following way:

```
> python3 predict_for_img.py
```

3.4   Evaluation

For the evaluation purposes, the NLG evaluation from Sharma et al. [2017] was utilized. Nevertheless, as some metrics are language specific, we needed to modify both the source code and data in order to use them. For METEOR metric, we had to change the source code and download the Czech paraphrases[6] file to the appropriate part of the package in order to use it on the Czech language. For embedding-based metrics, the original English GloVe vectors were replaced by the Czech word2vec vectors from Grave et al. [2018]. The word2vec file had to be further changed to the format used by the package. All neccessary information on how to modify the package is describeb within the corresponding *README* file.

---

[6]`https://github.com/cmu-mtlab/meteor/blob/master/data/paraphrase-cz.gz`

4. Experiments

This chapter describes all the experiments we conducted during the elaboration of our thesis and which will be further evaluated in the following chapter. Moreover, we will describe some technical details and environments we used for our experiments.

In the beginning we want to emphasize the fact that during the elaboration of this thesis we performed a lot of experiments with different setups, methods and data. Nertheless, we present only the experiments and setups we used for the final experiments and evaluation.

4.1 Environment

For training our neural networks, we used two different computational clusters, namely - AIC cluster[5] and IT4I cluster.[6]. Both of them provide a support for CPU and GPU computational tasks. However, since our models are large, we perform the trainings only on GPUs. For less time-consuming tasks, especially in the earlier phases of our thesis, we employed the AIC cluster. The IT4I cluster was used for the final training of all our models and all values reported in the subsequent parts of the text refer to the IT4I cluster hardware. Table 4.1 lists the hardware available in both clusters. This work was supported by the Ministry of Education, Youth and Sports of the Czech Republic through the e-INFRA CZ (ID:90140)

Cluster	GPU	Memory
AIC	NVIDIA GeForce GTX 1080 Ti	11 GB
IT4I	NVIDIA A100 SXM4 40GB	40 GB

Note: Clusters provide multiple types of GPUs. Only the types used are listed.

Table 4.1: Available GPU hardware on clusters.

4.2 Czech GPT-2

This section summarizes all experiments and results related to the training of Czech GPT-2 models. All experiments were performed according to the training procedure description in Chapter 2.2.2.

During the elaboration of this thesis, we trained several GPT-2 models on different datasets and with different approaches. Specifically, in addition to experiments described below, we performed training on many different learning rates given by the learning rate finding procedure. Experiments were done for all

---

[5]`https://aic.ufal.mff.cuni.cz/index.php/Main_Page`
[6]`https://www.it4i.cz/`

datasets described in Chapter 2.2.1. Furthermore, the original gradual unfreezing training procedure was another part that was experimented with. However, we present only the most important results as we are interested only in the best possible GPT-2 models.

The learning rates for both models were determined using the approach described in Chapter 2.2.2. For the general Czech GPT-2 model, the learning rates of $4e^{-3}$ for the first phase of learning only the new head and $2e^{-3}$ for the training of the entire unfrozen model were used on the OSCAR dataset. Subsequent fine-tuning of the medical model was performed using the learning rates of $8e^{-4}$ and $4e^{-4}$ respectively. Both of the models use batch size of 16 as it is the maximum that could fit into the GPU and sequence length of 512 for the reasons discussed in the referenced chapter. Entire training of each of the models took a total of 5 epochs. After some experimenting with the number of training epochs, we concluded that additional epochs do not bring any significant improvement. Overview of hyperparameters used for each model training are listed in Table 4.2.

Model	LR_{Head}	LR_{All}	Batch size	Seq length	Epochs
General GPT-2	$4e^{-3}$	$2e^{-3}$	16	512	5
Medical GPT-2	$8e^{-4}$	$4e^{-4}$	16	512	5

Note: The rationale for the hyperparameters is given above.

Table 4.2: Training hyperparameters of the Czech GPT-2 models. LR - Learning rate, *Head* - Learning rate used during the first epoch where only the new head is trained, *All* - Learning rate used during the rest of the epochs where the whole model is trained

4.2.1 Results

In this section, we present the training results for our fine-tuned Czech GPT-2 models. The training results can be seen in Table 4.3 and Table 4.4. During training, we measured accuracy and perplexity[7] metrics.

We can see that the general model achieved an accuracy of 40.02 and a perplexity of 30.35. These values are comparable to the original article[Guillou, 2020], where they trained a model for Portuguese on Wikipedia articles. However, there are two major differences between ours and theirs setup. First, our training was conducted on a total of a 21 GB of raw data, compared to the original 1.6 GB. Further, we train our model on sequences of length 512 compared to original 1024. The factors of sequence length and the heterogeneity of our data increase our measured perplexity values and thus the results cannot be compared directly. Nevertheless, even if we trained the model with an identical sequence length, our results would be worse than in the original paper even though we trained on an order of magnitude larger data. This is due to the fact that Portuguese and English are more similar languages than Czech and English

---

[7]https://en.wikipedia.org/wiki/Perplexity

and because the original article used only more homogenous data from Wikipedia.

As for the specialized medical GPT-2 model, the results are worse than for the general Czech GPT-2 model due to the specificity of the data. The fine-tuning was really fast as each epoch took less than 8 minutes. Nevertheless, we could get even better results, if we had better and more extensive Czech medical data to train the model for a longer period of time. Especially useful data would be directly from radiology.

Epoch	Loss$_{train}$	Loss$_{val}$	Accuracy (%)	Perplexity	Time (h)
1	4.42	4.28	30.40	72.10	26:56
2	3.75	3.76	35.99	42.85	29:07
3	3.74	3.65	37.20	38.32	29:07
4	3.61	3.54	38.48	34.38	28:58
5	3.52	3.41	40.02	30.35	28:57

Note: All values are rounded to 2 decimal places.

Table 4.3: General Czech GPT-2 model training results.
During the first epoch, only the new head is trained.

Epoch	Loss$_{train}$	Loss$_{val}$	Accuracy (%)	Perplexity	Time (m)
1	4.14	4.02	32.87	55.45	6:45
2	3.96	3.96	33.39	52.29	7:17
3	3.78	3.83	34.78	46.16	7:17
4	3.53	3.76	35.82	43.06	7:16
5	3.35	3.76	36.06	42.84	7:19

Note: All values are rounded to 2 decimal places.

Table 4.4: Medical Czech GPT-2 model training results.
During the first epoch, only the new head is trained.

4.2.2 Examples

In the last part of the GPT-2 experiments section, examples of texts generated by both of the trained models are shown and further discussed.

General Czech GPT-2

Firstly, we want to discuss outputs from our general GPT-2 model trained on OSCAR dataset. The generated sample from the empty input is shown in Figure 4.1. We can see, that the model can quite well understand the context and the Czech language in general. Moreover, we can notice the overall nice text fluency.

Nevertheless, the generated texts may have some imperfections, mistakes or defects. This includes for example incorrect interpunction, wrong word forms, grammatical errors, repetition or changes of the subject inside the text. All of these mistakes are the result of the data we trained on. The OSCAR dataset consists of crawled texts from all over the internet, comprising from various texts from news and articles to forums or advertisements. Some of these areas introduce noise into the data. Furthermore, all texts were concatenated and then split by the defined number of tokens before the training. For this reason, the model can change the subject during the text generation.

In the end, however, it is not such a problem for us because our objective was to capture the overall essence of the Czech language in our model for further fine-tuning and we accomplished that.

Medical Czech GPT-2

Similar properties to the general model are produced in the output generated by the medical model, as we can see in Figure 4.2. Nevertheless, there are more frequent changes of the subject. Again, this is the result of the training data. The medical texts we trained on, are mainly brief texts much shorter than the defined model sequence length, thus the training sequences may contain several different subjects due to the style of training data preparation. Also, some medical terms may not correspond directly because of this.

On the other hand, we can notice that the output does not suffer so much from general grammatical or interpunction errors compared to the general model even though we trained on smaller, but more formal data. Nertheless, some unexpected and undesirable elements may occur.

Despite the mentioned imperfections, we have trained a model understanding the medical environment that is suitable for further fine-tuning to generate medical reports.

4.3   Dataset translation

Medical dataset translation is a large part of the thesis. This section describes all necessary details of the data translation. As our final data source we chose the Indiana University chest X-ray dataset and we utilized CUBBITT as the translation service. All texts were preprocessed according to Chapter 2.3.2. The maximum number of concurrently running threads is set to 64 to avoid overloading the CUBBITT service. Translation of the whole dataset (3955 files) took a total of 32 minutes. Figure 4.3 shows an example of the original and its corresponding translated report. All translations can be found in Attachment TODO.

4.4 Medical report generation model

Tha last section of this chapter describes all the experiments conducted for the medical report generation. Furthermore, all the necessary technical information about running models and data preparation is summarized.

4.4.1 Setup

As we already mentioned in the previous parts of text, we use the solution from Alfarghaly et al. [2021] paper that is freely available on the github.[8] The code uses older tensorflow 2.3.0 with some deprecated features, thus some minor adjustments had to be done in order to run on a newer 2.5.0 version. This version was chosen because of the available support for the corresponding CUDA and cuDNN version on the IT4I cluster.

For the training, the Inidiana Univesity chest X-ray dataset was utilized as in the original paper. The key reason is that the solution uses a pre-trained CNN backbone, which is directly fine-tuned for the Inidiana Univesity chest X-ray dataset tags. The MIMIC-CXR v2.0.0 does not provide these tags, but only 14 labels which are different from the original ones. Thus the original Chexnet model would have to be fine-tuned again for this huge dataset. Moreover, the whole training time would be considerably prolonged and the code would have to be modified more extensively.

As we need to have the Czech translated reports in the same format as in the original case, we implemented a script to do so - described in Chapter 3.3. The reports contain several sections, however for the final report training and prediction, only the *Impression* and *Findings* sections are taken as they contain key diagnostic information.

4.4.2 Experiments

Final thing that needs to be perfomed is to run experiments for the medical report generation. We conducted our experiments with multiple different setups of the neural network. All of them were performed on the IT4I cluster with the batch size of 8 as it was the maximum size we were able to fit into the GPU. In the original article, they trained the model with the constant learning rate of $1e^{-3}$ and Adam optimizer. Nevertheless, we trained our models with a smaller learning rate of $1e^{-4}$, because we find it better to train already pre-trained models with a lower value. The max sequence length is set to be 200 tokens, as the reports are generally of a shorter nature. The entire training took place in a total of 100 epochs and the model is evaluated every third epoch on the test set.

---

[8]`https://github.com/omar-mohamed/GPT2-Chest-X-Ray-Report-Generation`

We run the final training on both of our trained Czech GPT-2 models - general Czech model and specialized Czech medical model. On top of the original paper where they trained only the decoder part of the network with the frozen CNN backbone, we also train with setups in which we have left the entire network unfrozen. In the end, we have a total of 4 different setups listed in Table 4.5 for training. For each of them, the best and the last model will be eveluted.

In addition to the aforementioned setups, we conducted multiple trainings also with a smaller batch size, the original learning rate or with unpreprocessed training data, where the data were only lowercased before their translation - see Chapter 2.3.2 for more details.

Model	Training scope	Identifier
General Czech GPT-2	Decoder only	GEN-dec
General Czech GPT-2	Entire network	GEN-all
Medical Czech GPT-2	Decoder only	MED-dec
Medical Czech GPT-2	Entire network	MED-all

Note: Tags' embeddings are frozen during all setups.

Table 4.5: Medical report generation experiments' setups.

Při výběru hotelu je třeba brát v úvahu také jeho polohu ve městě i okolí – město může být vzdálené od hotelu až 10 minut pěší chůzí (na kole). V letních měsících se zde nachází množství možností zábavy pro děti nebo dospělé s dětmi či dospělými za poplatek 5 €/den. Pro náročnější klienty jsou k dispozici tenisové kurty, hřiště na plážový volejbal a stolní tenis. Za příplatek lze využít možnost pronájmu sportovních zařízení jako fitness centrum, masáže, sauna a parní lázeň. Hotel nabízí širokou škálu aktivit během pobytu: golf, potápění, vodní sporty, rybaření, cyklistika, jízda na koni, rybolov, windsurfing, paragliding, plachtění, šnorchlování, jízdy na banánu, lyžování, minigolf, jízdu na koni a další aktivity dle přání klienta. K návštěvě láká také oblast Flat Islands, která má mnoho zajímavostí spojených zejména s turistikou po ostrově. Navštívit můžete například přírodní rezervaci Galeries of the Cave, kde si vychutnáte místní kuchyni z místních zdrojů. Na své dovolené určitě zavítáte do oblasti Mt Railway, což leží přímo u moře. Další místa spojená s turistikou patří např. národní park Ngorong Forest, který zahrnuje více než 20 km pobřeží. Dále pak rezervace Treefoldland, které nabízejí celou řadu turistických tras vedoucích okolo ostrova a národního parku Taurus, jenž obsahuje přes 200 kilometrů značených stezek různých obtížností vhodných jak začátečníkům tak pokročilým turistům. Pokud budete chtít navštívit některý ze zdejších ostrovů doporučujeme Vám návštěvu některého místního muzea s průvodcem. Po návratu domů nezapomeňte ještě zajít do jedné restaurace nebo baru. S nabídkou výletů Vás jistě mile překvapí ostrov San Andreas, nacházející se přibližně 7 km jižně od města Beverly Hills. Mezi hlavní atrakce tohoto ostrůvku patří Národní muzeum s expozicí o historii USA a historie Austrálie. Je známo především díky svým krásným scenériím a historickým památkám ale stejně dobře jej poznáte při procházce po jeho březích. Dostanete-li chuť poznat trochu jiná místa Vašeho života navštivte ostrov Cookington, známý pod názvem „Cookingtonská riviéra". Tento kraj tvoří 4 ostrovy a 6 hlavních ostrůvků. Většina těchto míst byla osídlená již před tisíci lety. Nacházíme tady malebnou krajinu plnou lesů a hornatých údolí lemovaných borovicovými lesy. Nejznámějšími místy této země bývají městečka Townsend a Oakland. Tyto tři vesnice mají velice zajímavou minulost spojenou právě s těžbou zlata a stříbra.

Figure 4.1: Sample output of the general Czech GPT-2 model.
Note: the first and last uncomplete sentences were omitted. Text was generated for empty input.

Ve všech případech bylo nezbytné zajistit stabilitu kloubu a to buď přímou fixací ke kostnímu krytu, nebo pomocí kleští. Při fixaci K-dráty jsme využili úhlově stabilní LCP dlahy, které jsou velmi tenké jak u proximálního tak distálního konce humeru bez použití jiných augmentací šroubů než např. FBM či SDS šrouby.

V literatuře se uvádí jako možný mechanizmus vzniku tohoto syndromu postižení CNS především syndrom mnohočetné endokrinní neoplázie. Jedná se o dědičné onemocnění s postižením řady genů spojených zejména s poruchou regulace transkripce genu pro růstový hormon. V současnosti má NF1 mutaci ve více než 60 % případů: v oblasti limbického systému, nervového systému včetně kosterního svalstva; také gonády, ledviny, lymfatické uzliny, ale i štítnou žlázu, srdce aj., což vede k častější manifestaci těchto chorob.

Proto byla vyvinuta řada dalších medikamentů - tricyklických antidepresiv, která mají snížit počet nežádoucích účinků léků užívaných při depresi. Patří mezi ně především antidepresiva ze skupiny selektivních alfa2-adrenergních agonistů, tricyklická antidepresiva (amitriptylin), inhibitory zpětného vychytávání serotoninu (SSRI); mirtazapin, milnacipran, venlafaxin, duloxetin, klomipramin, imipramin, venlafaxin, nortriptylin, flumipramin, levodopa. Nejnovějšími antidepresivy z této třídy jso u bupropion, citalopram a sertralin. Další možnou skupinou farmak používaných při léčbě deprese jsou antagonisté serotoninových receptorů typu SSRI. Jsou zde psychofarmaka prvé generace (selektivní beta-blokátory, blokátory receptoru N-metyl-Daspartátového kanálu) a některá další léčiva používaná po roce 2000.

Vzhledem k tomu není možné stanovit přesné věkové rozložení pacientů zařazených do jednotlivých studií. Klíčová slova: diabetes mellitus 2. typu, hypertenze, dyslipidemie, obezita, metabolický rozvrat.

U starších nemocných lze využít různých forem nutriční podpory formou sippingu anebo sondové výživy. Ve stáří již nejsou doporučovány žádné speciální léčebné postupy ani dietní opatření ovlivňující průběh nemoci, proto by neměly být tyto přípravky podávány nemocným mladším 45 let věku. Literatura 1.

Orientačně byl prokázán vzestup TK až po 3 měsících léčby.

Tato studie měla za cíl sledovat účinek sorafenibu v kombinaci s dalšími cytostatiky ovlivňujícími VEGFR3b/IIIa receptory a ověřit jejich bezpečnost oproti placebu. Studie rovněž prokázala pozitivní efekt temsirolimu v první linii terapie mRCC po selhání imunoterapie interferon α. Léčba sunitinibem v druhé linii nebyla spojena signifikantně s nárůstem celkové mortality.

Je nutné mít však před aplikací léku anamnézu infekce kůže celého těla a počítat s tím, že může dojít k poškození sliznice dýchacích cest (např. erysipel nebo otitida).

Figure 4.2: Sample output of the medical Czech GPT-2 model.
Note: the first and last incomplete sentences were omitted. Text was generated for empty input.

Comparison: XXXX

Indication: Palpitation

Findings: PA and lateral views the chest were obtained. The cardio-mediastinal silhouette is normal in size and configuration. The lungs are well aerated. No pneumothorax, pleural effusion, or lobar air space consolidation. XXXX right middle lobe collapse appears less distinct than on prior study.

Impression: No acute cardiopulmonary disease.

(a) Original report.

Porovnání: XXXX

Indikace: Palpitace

Nálezy: Byly získány PA a boční pohledy na hrudník. Kardiomediastinální silueta má normální velikost a konfiguraci. Plíce jsou dobře provzdušněny. Žádný pneumotorax, pleurální výpotek ani konsolidace lobárního vzdušného prostoru. XXXX kolaps pravého středního laloku se zdá být méně zřetelný než v předchozí studii.

Dojem: Žádné akutní kardiopulmonální onemocnění.

(b) Translated report.

Figure 4.3: Example of original and corresponding translated report. The example is taken from the Indiana University Chest X-ray dataset.

5. Evaluation

The last chapter of our thesis presents an evaluation results, included in Attachement TODO, of all conducted experiments, described in detail in Chapter 4.4.2, for medical reports generation task. Each performed experiment was evaluated automatically with the common machine translation metrics. Furthermore, the outputs of the best model were manually evaluated by an experienced radiologist.

5.1 Automatic evaluation

First part of this chapter reports the results of the automatic evaluation of our trained models. Important thing to mention is that these metrics give us only an indication of which models could be theoretically perfoming good and not their true performance. Each image can be described in multiple different ways. The metrics only compare words between generated and reference texts with minimal logic to capture meaning. Therefore they measure whether the texts are similar rather than having the same meaning. This fact is further intensified by the nature of our area, where we need to capture all findings appearing in the image and the overall diagnosis.

The evaluation was performed on a subset of 500 X-ray images. We subjected all models to standard automatic evaluation metrics used in image captioning and machine translation areas using NLG evaluation from Sharma et al. [2017].

5.1.1 Metrics

This section outlines all NLG evaluation measured metrics applicable for the Czech language supplemented with GLEU metric. For the embedding NLG evaluation metrics, the NLG package had to be modified in order to use Czech word2vec trained vectors from Grave et al. [2018].

BLEU

BLEU metric has become a standard metric in the field of image captioning. It was first introduced in the Papineni et al. [2002] paper. This metric computes the number of common unigrams up to n-grams between the reference text and the generated one. Commonly used are unigrams up to tetragrams as they capture the similarities between texts the best. BLEU-4 thus uses unigrams up to tetragrams. The final BLEU score is the mean of the BLEU scores of all evaluated texts.

GLEU

GLEU (Google BLEU) metric, presented in Wu et al. [2016], is Google's alternative instead of BLEU for machine translation evaluation. The difference is that it does not have the downsides of BLEU for single sentences. For sentence GLEU score, the n-gram precision and recall are calulated and the minimum of these values are taken as the results. All unigrams up to n-grams are taken for

the calculation. The final corpus-level GLEU score is computed as the ratio of the sum of all matching n-grams and the sum of all n-grams used for precision or recall in the sentence-level case for all texts, instead of the mean of the sentence-level GLEU scores.

METEOR

METEOR is a unigram based F-score metric first defined in Banerjee and Lavie [2005]. In the beginning, it tries to match all unigrams from the generated text to zero or one unigram from the reference text. This could also include matching paraphrases, synonyms, stems or others. The mapping with the minimum number of chunks is taken for subsequent calculation of precision and recall. The final score is the harmonic mean, where the recall is many times more significant than the precision, multiplied by a fragmentation penalty. In order to calculate this metric for Czech data, we had to modify the original NLG code and download the paraphrases for the Czech language.[1]

ROUGE-L

ROUGE-L metric, from Lin [2004] paper, works with the word longest common subsequence (LCS) between the generated and reference text. The found subsequence does not have to be consecutive. The reasoning behind this metric is that longer common subsequences results in more similarities between the compared texts. ROUGE-L is an F-score metric and is therefore computed using both precision and recall based on the LCS.

CIDEr

CIDEr[Vedantam et al., 2015] is another word-overlap metric. It first determines the TF-IDF weight vector representation of unigrams up to tetragrams for both texts. Thus for each generated and reference text, we obtain four vector representations. A cosine similarity is calculated for each of these vectors. The mean of these cosine similarities is then taken as the final score.

Embedding Average cosine similarity

For both of the texts - generated and the reference, the embeddings of all words are averaged. The final score is then computed as the cosine similarity between these averaged embedding vectors.

Vector Extrema cosine similarity

This metric was presented in Forgues et al. [2014]. As in the previous case it computes the cosine similarity between the generated and reference text. However, instead of the mean embeddings, the most extreme values (the largest in terms of absolute value) in each dimension are taken.

---

[1] `https://github.com/cmu-mtlab/meteor/blob/master/data/paraphrase-cz.gz`

Greedy Matching score

The final used metric is Greedy Matching score from Rus and Lintean [2012]. For each word embedding in the generated text, the maximum cosine similarity is calculated against the words' embeddngs of the reference text. These score are then averaged by the total number of words in the generated text. The identical procedure is then used the other way around with the reference text. As the final score, the mean of these scores is taken.

5.1.2 Results

The results of the automatic evaluation are reported in Table 5.1, Table 5.2 and Table 5.3 for the standard NLP metrics and in Table 5.4 for the embedding-based metrics. For metrics that are evaluated for different n-gram bases (BLEU and GLEU), we further refer to those working with up to tetragrams if not specified otherwise. Since there are no works solving our problem in the Czech language, we do not have a direct baseline to compare our results with and thus we will only compare them to each other.

For the BLEU and GLEU metrics, we can see in Tables 5.1 and 5.2 that the best results have the models, where the entire network was unfrozen during the training. This is naturally expected as the network has more space and freedom for the learning. Moreover, the *GEN-all-best*, with 0.0751 and 0.0928 for BLEU and GLEU respectively, surpassed the *MED-all-best*, which utilized the medical GPT-2 model as the language model, by 0.3 on both BLEU and GLEU. The other *best* models also show nice results and they are good candidates for further comparison. In general, the *last* models are worse. The only exception is *MED-dec-last*, where BLEU was higher than for the corresponding *best* model. However, $BLEU_1$ - $BLEU_3$ are better for the *MED-dec-best* model.

The same trend continues in the rest of the word-overlap metrics, presented in Table 5.3. However, the *MED-all-best* is worse than other *best* models on most metrics. We can also notice that the *MED-dec-last* surprisingly outperformed all models on CIDEr metric with 0.2978, whilst underperforming in others. The oppsosite phenomenon occured for the *MED-dec-best* model, which was the worst on CIDEr with 0.1798, but comparable to other *best* models on the rest.

Regarding the embedding metrics evaluation listed in Table 5.3, the *GEN-dec-best* model beat the rest of the models, but the otherwise best *GEN-all-best* model's results are very close to it. Suprisingly, the *MED-dec-best* has better results on all of these metrics than the *MED-all-best* model, where the whole model was trainable.

We can see that *GEN-all-best*, which was trained on a entirely unfrozen network, model outperformed the other models in almost all metrics. Nevertheless, the model comes from the earlier phases of the training. All other *best* models were performing well and outdid each other in different examined metrics. As we mentioned earlier, the automatic evaluation only suggests which models could be performing well, and their final performance will be determined by manual evaluation in the next section.

Model	Epoch	$BLEU_1$	$BLEU_2$	$BLEU_3$	$BLEU_4$
GEN-dec-best	28	0.2451	0.1489	0.0951	0.0655
GEN-dec-last	100	0.2080	0.1207	0.0764	0.0519
GEN-all-best	13	0.2487	**0.1550**	**0.1044**	**0.0751**
GEN-all-last	100	0.2161	0.1281	0.0809	0.0551
MED-dec-best	13	**0.2526**	0.1511	0.0951	0.0617
MED-dec-last	100	0.2207	0.1358	0.0918	0.0683
MED-all-best	76	0.2336	0.1447	0.0987	0.0726
MED-all-last	100	0.2145	0.1322	0.0877	0.0630

Note: All values are rounded to 4 decimal places.

Table 5.1: BLEU evaluation results comparison.
Models are described in Chapter 4.4.2, *best* suffix denotes the best trained models, *last* denotes the last trained models, $BLEU_n$ index states the number of n-grams used

Model	Epoch	$GLEU_1$	$GLEU_2$	$GLEU_3$	$GLEU_4$
GEN-dec-best	28	0.2122	0.1460	0.1094	0.0875
GEN-dec-last	100	0.1900	0.1275	0.0951	0.0758
GEN-all-best	13	0.2154	**0.1502**	**0.1146**	**0.0928**
GEN-all-last	100	0.1902	0.1290	0.0962	0.0767
MED-dec-best	13	**0.2170**	0.1480	0.1104	0.0873
MED-dec-last	100	0.1966	0.1360	0.1038	0.0848
MED-all-best	76	0.2083	0.1447	0.1108	0.0903
MED-all-last	100	0.1968	0.1362	0.1033	0.0835

Note: All values are rounded to 4 decimal places.

Table 5.2: GLEU evaluation results comparison.
Models are described in Chapter 4.4.2, *best* suffix denotes the best trained models, *last* denotes the last trained models, $GLEU_n$ index states the number of n-grams used

Model	Epoch	METEOR	ROUGE-L	CIDEr
GEN-dec-best	28	0.1103	0.2087	0.2290
GEN-dec-last	100	0.0952	0.1878	0.2038
GEN-all-best	13	**0.1138**	**0.2125**	0.2964
GEN-all-last	100	0.0988	0.1958	0.2240
MED-dec-best	13	0.1133	0.2074	0.1798
MED-dec-last	100	0.1013	0.1953	**0.2978**
MED-all-best	76	0.1072	0.2027	0.2658
MED-all-last	100	0.1003	0.1988	0.2598

Note: All values are rounded to 4 decimal places.

Table 5.3: Word-overlap metrics evaluation results.
Models are described in Chapter 4.4.2, *best* suffix denotes the best trained models, *last* denotes the last trained models

Model	Epoch	Embedding Average	Vector Extrema	Greedy Matching
GEN-dec-best	28	**0.8023**	**0.4092**	**0.6340**
GEN-dec-last	100	0.7778	0.3765	0.6169
GEN-all-best	13	0.7974	0.4050	0.6327
GEN-all-last	100	0.7854	0.3950	0.6192
MED-dec-best	13	0.7998	0.4089	0.6265
MED-dec-last	100	0.7758	0.3738	0.6160
MED-all-best	76	0.7902	0.4019	0.6241
MED-all-last	100	0.7754	0.3796	0.6175

Note: All values are rounded to 4 decimal places.

Table 5.4: Embeding metrics evaluation results.
Models are described in Chapter 4.4.2, *best* suffix denotes the best trained models, *last* denotes the last trained models

5.2 Manual evaluation

This section presents the results of the manual evaluation. We would like to express our gratitude to MUDr. Martin Hyršl from the University Hospital Hradec Králové for the execution of manual evaluation on our generated reports. The manual evaluation was performed on a total of 40 X-ray images and their corresponding reports for all four *best* models from the previous section. The X-rays are divided into two equally sized groups for evaluation - with and without the presence of any findings.

5.2.1 Method

All images were evaluated using the following method. Each report is classified into one of three categories. First category reflects the case where the report contains all corresponding findings and overall diagnosis without any false infor-

mation. The next category decribes cases in which the report is missing some important points without any false information. To the last category fall reports that contain untrue facts.

5.2.2 Results

TODO po obdržení výsledků

Group	Accurate	Incomplete	Incorrect
GEN-dec-best	—	—	—
GEN-all-best	—	—	—
MED-dec-best	—	—	—
MED-all-best	—	—	—

Note: All values are rounded to 4 decimal places.

Table 5.5: Manual evaluation results - without findings. Evaluation categories are described above

Group	Accurate	Incomplete	Incorrect
GEN-dec-best	—	—	—
GEN-all-best	—	—	—
MED-dec-best	—	—	—
MED-all-best	—	—	—

Note: All values are rounded to 4 decimal places.

Table 5.6: Manual evaluation results - with findings. Evaluation categories are described above

5.3 Examples

TODO Demostration examples of generated reports by our best model are depicted in Figures TODO, TODO? and compared to the ground truth reports in mutiple different situtations.

5.4 Conclusion

TODO Celkový zhodnocení, až po tom co budeme mít metriky a výstupy

Conclusion

In this thesis, we have been dealing with a problem of automatic generation of medical reports from chest X-rays in the Czech language. Many studies have been conducted for English, but none for Czech, to the best of our knowledge. The first part has presented a detailed analysis of the overall problem. We have introduced neural network architectures and data alternatives alongs with their properties that are commonly used for this problem and a survey of related works.

We have designed our approach based on the Alfarghaly et al. [2021] paper. Nevertheless, as the original solution is intended for English, we have gradually solved all associated problems. Firstly, we have described a fine-tuning process for the two Czech GPT-2 models on general Czech and medical data with reasoning about different possibilities of data. Furhermore, since there are no Czech medical datasets comprising of X-ray images and their corresponding reports, we proposed a solution for an automatic translation of English datasets.

We have then described the implementation details and all important source codes of our solution separately for each independent part.

In the next part, the final experiments took their place. We have performed two GPT-2 model trainings according to our design along and presented their results. Next, we have translated the Indianta University chest X-ray dataset into the Czech language for our final task. Afterwards, we have conducted experiments for the medical report generation in multiple different setups of the neural network. Individual setups differed in the used language model or the extent to which the network has been trained.

Finally, all ran experiments have been evaluated both automatically and manually. The automatic evaluation consisted of numerous stadard NLP metrics and it have given us an indication of which models could be performing good. These assumptions have been further verified by manual evaluation by a trained psysician. We found out that the medical report generation for the Czech language is (not)possible and has (not) great future potential.

Future work

There are several aspects that could be further worked on in the future. The Czech GPT-2 models could be improved by training on cleaner and more extensive data with data preparation so that within one sequence there is always only one source of text. Moreover, instead of fine-tuning from the original English model to Czech, we could train the general Czech GPT-2 directly in order to get better results and more coherent texts. For the final chest X-ray report generation, we could modify the model or even create our own with newer architecture in order to train on a larger dataset. Another major improvement would be to obtain real Czech hospital data without translation noise on which we could train. Based on the data we would have, we might even train additional objectives, such as bounding boxes to focus on the right parts of the X-ray.

Bibliography

Farhad Akhbardeh, Arkady Arkhangorodsky, Magdalena Biesialska, Ondřej Bojar, Rajen Chatterjee, Vishrav Chaudhary, Marta R Costa-jussa, Cristina España-Bonet, Angela Fan, Christian Federmann, et al. Findings of the 2021 conference on machine translation (wmt21). In *Proceedings of the Sixth Conference on Machine Translation*, pages 1–88, 2021.

Omar Alfarghaly, Rana Khaled, Abeer Elkorany, Maha Helal, and Aly Fahmy. Automated radiology report generation using conditioned transformers. *Informatics in Medicine Unlocked*, 24:100557, 2021.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72, 2005.

Aurelia Bustos, Antonio Pertusa, Jose-Maria Salinas, and Maria de la Iglesia-Vayá. Padchest: A large chest x-ray image dataset with multi-label annotated reports. *Medical image analysis*, 66:101797, 2020.

Zhihong Chen, Yan Song, Tsung-Hui Chang, and Xiang Wan. Generating radiology reports via memory-driven transformer. *arXiv preprint arXiv:2010.16056*, 2020.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Dina Demner-Fushman, Marc D. Kohli, Marc B. Rosenman, Sonya E. Shooshan, Laritza Rodriguez, Sameer Antani, George R. Thoma, and Clement J. McDonald. Preparing a collection of radiology examinations for distribution and retrieval. *Journal of the American Medical Informatics Association*, 23 (2):304–310, 07 2015. ISSN 1067-5027. doi: 10.1093/jamia/ocv080. URL https://doi.org/10.1093/jamia/ocv080.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Gabriel Forgues, Joelle Pineau, Jean-Marie Larchevêque, and Réal Tremblay. Bootstrapping dialog systems with word embeddings. In *Nips, modern machine learning and natural language processing workshop*, volume 2, page 168, 2014.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. Learning word vectors for 157 languages. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

P Guillou. Faster than training from scratch—fine-tuning the english gpt-2 in any language with hugging face and fastai v2 (practical case with portuguese), 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*, 2018.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

Xin Huang, Fengqi Yan, Wei Xu, and Maozhen Li. Multi-attention and incorporating background information model for chest x-ray image report generation. *IEEE Access*, 7:154808–154817, 2019.

Adam Hájek and Aleš Horák. Czegpt-2 – new model for czech summarization task. preprint available at `https://openreview.net/forum?id=H43eQtxZefq`, 3 2022.

Jeremy Irvin, Pranav Rajpurkar, Michael Ko, Yifan Yu, Silviana Ciurea-Ilcus, Chris Chute, Henrik Marklund, Behzad Haghgoo, Robyn Ball, Katie Shpanskaya, et al. Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 590–597, 2019.

Baoyu Jing, Pengtao Xie, and Eric Xing. On the automatic generation of medical imaging reports. *arXiv preprint arXiv:1711.08195*, 2017.

Alistair EW Johnson, Tom J Pollard, Seth J Berkowitz, Nathaniel R Greenbaum, Matthew P Lungren, Chih-ying Deng, Roger G Mark, and Steven Horng. Mimic-cxr, a de-identified publicly available database of chest radiographs with free-text reports. *Scientific data*, 6(1):1–8, 2019a.

Alistair EW Johnson, Tom J Pollard, Nathaniel R Greenbaum, Matthew P Lungren, Chih-ying Deng, Yifan Peng, Zhiyong Lu, Roger G Mark, Seth J Berkowitz, and Steven Horng. Mimic-cxr-jpg, a large publicly available database of labeled chest radiographs. *arXiv preprint arXiv:1901.07042*, 2019b.

Michal Křen, Václav Cvrček, Jan Henyš, Milena Hnátková, Tomáš Jelínek, Jan Kocek, Dominika Kováříková, Jan Křivan, Jiří Milička, Vladimír Petkevič,

Pavel Procházka, Hana Skoumalová, Jana Šindlerová, and Michal Škrabal. SYN v9: large corpus of written czech, 2021. URL http://hdl.handle.net/11234/1-4635. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Daniel Kvak and Karolína Kvaková. Carebot: asistenční systém s podporou umělé inteligence pro klasifikaci a prostorovou lokalizaci pneumonie ve skiagrafickỳch snímcích hrudníku. 2021.

Daniel Kvak, Marian Bendik, and Anna Chromcova. Towards clinical practice: Design and implementation of convolutional neural network-based assistive diagnosis system for covid-19 case detection from chest x-ray images. *arXiv preprint arXiv:2203.10596*, 2022.

Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

Ryan McDonald, Georgios-Ioannis Brokos, and Ion Androutsopoulos. Deep relevance ranking using enhanced document-query interactions. *arXiv preprint arXiv:1809.01682*, 2018.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

Ha Q Nguyen, Khanh Lam, Linh T Le, Hieu H Pham, Dat Q Tran, Dung B Nguyen, Dung D Le, Chi M Pham, Hang TT Tong, Diep H Dinh, et al. Vindrcxr: An open dataset of chest x-rays with radiologist's annotations. *arXiv preprint arXiv:2012.15029*, 2020.

Pedro Javier Ortiz Su'arez, Laurent Romary, and Benoit Sagot. A monolingual approach to contextualized word embeddings for mid-resource languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1703–1714, Online, July 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.acl-main.156.

Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, 2015.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

Yifan Peng, Xiaosong Wang, Le Lu, Mohammadhadi Bagheri, Ronald Summers, and Zhiyong Lu. Negbio: a high-performance tool for negation and uncertainty detection in radiology reports. *AMIA Summits on Translational Science Proceedings*, 2018:188, 2018.

Martin Popel, Marketa Tomkova, Jakub Tomek, Łukasz Kaiser, Jakob Uszkoreit, Ondřej Bojar, and Zdeněk Žabokrtský. Transforming machine translation: a deep learning system reaches news translation quality comparable to human professionals. *Nature Communications*, 11(4381):1–15, 2020. ISSN 2041-1723. doi: 10.1038/s41467-020-18073-9. URL https://www.nature.com/articles/s41467-020-18073-9.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

Pranav Rajpurkar, Jeremy Irvin, Kaylie Zhu, Brandon Yang, Hershel Mehta, Tony Duan, Daisy Ding, Aarti Bagul, Curtis Langlotz, Katie Shpanskaya, et al. Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning. *arXiv preprint arXiv:1711.05225*, 2017.

Johannes Rückert, Asma Ben Abacha, Alba García Seco de Herrera, Louise Bloch, Raphael Brüngel, Ahmad Idrissi-Yaghir, Henning Schäfer, Henning Müller, and Christoph M. Friedrich. Overview of ImageCLEFmedical 2022 – caption prediction and concept detection. In *CLEF2022 Working Notes*, CEUR Workshop Proceedings, Bologna, Italy, September 5-8 2022. CEUR-WS.org.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

Vasile Rus and Mihai Lintean. An optimal assessment of natural language student input using word-to-word similarity metrics. In *International Conference on Intelligent Tutoring Systems*, pages 675–676. Springer, 2012.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015.

Shikhar Sharma, Layla El Asri, Hannes Schulz, and Jeremie Zumer. Relevance of unsupervised metrics in task-oriented dialogue for evaluating natural language generation. *CoRR*, abs/1706.09799, 2017. URL http://arxiv.org/abs/1706.09799.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Leslie N Smith. Cyclical learning rates for training neural networks. In *2017 IEEE winter conference on applications of computer vision (WACV)*, pages 464–472. IEEE, 2017.

Leslie N Smith. A disciplined approach to neural network hyper-parameters: Part 1–learning rate, batch size, momentum, and weight decay. *arXiv preprint arXiv:1803.09820*, 2018.

Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial intelligence and machine learning for multi-domain operations applications*, volume 11006, pages 369–386. SPIE, 2019.

Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575, 2015.

Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, Mohammadhadi Bagheri, and Ronald M Summers. Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2097–2106, 2017.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057. PMLR, 2015.

Jianbo Yuan, Haofu Liao, Rui Luo, and Jiebo Luo. Automatic radiology report generation based on multi-view image fusion and medical concept enrichment. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 721–729. Springer, 2019.

Zizhao Zhang, Yuanpu Xie, Fuyong Xing, Mason McGough, and Lin Yang. Mdnet: A semantically and visually interpretable medical image diagnosis network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6428–6436, 2017.

List of Figures

List of Tables

List of Abbreviations

GPT-2 – Generative Pre-trained Transformer 2
CITI – Collaborative Institutional Training Initiative
REST API – Representational State Transfer Application Programming Interface
CNN – Convolutional Neural Network
RNN – Recurrent Neural Network
LSTM – Long Short-Term Memory
GRU – Gated Recurrent Unit
ViT – Vision Transformer
NLP – Natural Language Processing
PA – Posterior-Anterior
AP – Anterior-Posterior
DICOM – Digital Imaging and Communications in Medicine
MTI – Medical Text Indexer
RadLex – Radiology Lexicon
MeSH – Medical Subject Headings
MIMIC – Medical Information Mart for Intensive Care
MIMIC-CXR – MIMIC Chest X-Ray
CLEF – Cross Language Evaluation Forum
PadChest – PAthology Detection in Chest radiographs
BCIDR – Bladder Cancer Image and Diagnostic Report
PEIR – Pathology Education Informational Resource
CUBBITT – Charles University Block-Backtranslation-Improved Transformer Translation
OSCAR – Open Super-large Crawled Aggregated coRpus
UFAL – Institute of Formal and Applied Linguistics
distilGPT2 - Distilled-GPT2
AIC – Artificial Intelligence Cluster
IT4I – IT4Innovations
LR – Learning Rate
CUDA – Compute Unified Device Architecture
cuDNN – CUDA Deep Neural Network library
NLG – Natural Language Generation
BLEU – Bilingual Evaluation Understudy
GLEU – Google BLEU
METEOR – Metric for Evaluation of Translation with Explicit ORdering
ROUGE-L – Recall-Oriented Understudy for Gisting Evaluation LCS
CIDEr – Consensus-based Image Description Evaluation
LCS – Longest Common Subsequence
TF-IDF – Term Frequency - Inversed Document Frequency
GloVe – Global Vectors for Word Representation

A. Attachments

A.1 First Attachment