Chapter 1

Methodology

In this chapter we will discuss how we created the dataset used in our research, what the dataset contains and the purpose for which we created it.

1.1 Data collection

As discussed in Section **??** there is number of publicly available datasets online, and some which are restricted. Some of these datasets are difficult to access, and there is a need for more publicly available datasets which are collected for the purpose of deep learning. To assist the under-explored field of research within medical computer assisted analysis tools the datasets need to be large and well annotated. Some of the mentioned datasets lack adequately documented, annotated samples from a good source and is not well suited for our research. Thus, as a vital part of our research, we aim to produce a collection of well annotated and adequately big dataset that can be used not only in this study, but also contribute to the research community and have a impact on the research comparability in future. We achieve this by collecting medical data, sorting and annotating it and making the dataset publicly available and free for non-commercial, educational and research purposes.

1.1.1 Privacy, Legal and Ethics Issues

To obtain medical videos from a hospital in Norway is very difficult and not straight forward. All medical data is considered personal and is therefore strongly protected from unauthorized use and distribution by the **Pasientjournalloven??** legislation. A medical study conducted at 2 academic hospitals from May 2017 to September 2018 found that most patients are willing to share their data and biospecimens for research purposes **PatientPerspectives19**. Regardless of the patient opting in to share their data and biospecimens it is difficult for researchers to get their hands on it. We solved this problem by collaborating with a number of Norwegian hospitals and research teams working there. One of the research teams we collaborated with is Augere Medical (See Section **??** for more info), and through them we got in contact with Vestre Viken Hospital Trust, allowing our research team to download anonymous data from hospital systems and transfer it using secure media to our facility. Upon downloading the data we further stripped the metadata files for potential information regarding patients like time stamps, dates and camera equipment used.

1.1.2 Kvasir-PillCam

The dataset we used in our experiments consist of endoscopic videos collected from Bærum Hospital, a hospital in Vestre Viken Hospital Trust. Unlike Kvasir and Hyper-Kvasir datasets we have made the Kvasir-PillCam dataset for the purpose of this thesis. In total we have 44 videos which have gone through some re encoding to reduce the file sizes, and also because the original encoding is proprietary Sony technology. After that the videos are uploaded to Augere Medical[1] tagging tool. The data export from Bærum contains a number of findings for each video and they are annotated, extracted, converted to frame number and that helped us a great deal with tagging

[1] `https://augere.md/`

Class name	Samples
Angiectasia	908
Blood	658
Erosion	525
Erythematous	251
Foreign Bodies	776
Hematin	12
Ileo-cecal valve	4704
Normal	33,129
Polyp	583
Pylorus	1410
Ulcer	759
Unknown	190

Table 1.1: Hyper-PillCam class names and corresponding amount of samples.

the videos. When all 44 videos have been precisely labeled the dataset is exported from Augere Medical tagging tool and split into folders for each class. The number of images per class are given in Table **??**. In total we have 43,905 labeled images in 12 classes. The sample distribution across the classes is skewed depending on how many findings there are in the videos. Some findings occur often and some very rarely. The dataset also contain one class for 'normal' images, which there is quite a bit more of than findings.

Imbalanced dataset pose a challenge for predictive algorithms as most learning algorithms are based on the assumption of an equal number of samples for each class. This results in models that have poor predictive performance, especially for minority class or classes. This is a great problem because in many medical datasets the minority class is the most important and therefore more sensitive for classification errors.

In addition to labeling the images the dataset also contain a JSON format file which stores coordinates for where in the frame the finding is located. The Kvasir-PillCam dataset will be an open-source dataset available for others scientists, and will later be grown to include more PillCam videos, both labeled and unlabeled samples.

1.2 Data process

1.2.1 Data preprocessing

- When splitting the dataset it is important to ensure minority classes are represented in both training, validation and test dataset. Found that opencv resizes images a bit faster (1.2s vs 1.6 for pillow)
- What is the necessity of rotating the samples. Will they not always be 'square'?
- Normalization; any benefit of using L2 normalization in last dense layer when using a pre-defined model?

Splitting dataset

Discovered non-optimal implementation of dataset splitting, which left some of the most vulnerable minority classes with no samples in the test data. To mitigate this issue I created a new script for pre-splitting dataset into sub folders for each, training, testing and validation dataset. The outline of this script is seen in Figure XX.

We split the data into XX. To make it as reproducible as can be I have first sorted the data then used seeded random to shuffle before splitting. In this step we also reduce the image dimensions to make the data easier to use during the next preprocessing steps and to reduce file size.

1.2.2 Data pipeline

The preprocessing pipeline is implemented by using tensorflows data.Dataset library. This was chosen over datagenerator due to the easy of use, but later became an issue due to the complexity.

The main benefits by using data.Dataset is that all data is handled in tensors and computed by the GPU which enhance the load distribution of processing power.

The pipeline outline can be seen in Figure XX.

Caching

We call this consumer / producer overlap, where the consumer is the GPU and the producer is the CPU.

1.3 System implementation

1.3.1 Generating new pseudo labels

When we use the trained teacher model to generate new pseudo labels from the dataset we set a threshold for which the predicted image is saved if it is above the set value. Here we have two options, either to set the threshold low and extract a large dataset with high uncertainty, but get the most of our minority classes. Or we can set the threshold high, and gather a dataset with lower uncertainty, but might miss more samples for the minority classes.

After subtracting samples from the unlabeled dataset through predicting and saving images with a probability score higher than the threshold, we sort the images based on the probability scores. This is done so for majority classes with a large amount of samples can be squeezed to fit the minority classes and we can select the samples with the highest probability scores.

Inspecting the pseudo labels

Since the unlabeled dataset don't contain labels, it is difficult to fully comprehend the results of the extracted data.

1.3.2 Learning rate

Many models train better if you gradually reduce the learning rate during training. We are using learning rate which follows a inverse time decay which means that the learning rate is decreased to 1/2 of the base rate at 10 epochs, 1/3 at 20 epochs and so on.

1.3.3 Hyper-parameter tuning

Class weighting

Found that when weighting the classes the model learns slower. Which means decay rate of learning rate must be lowered for it not to drop too soon. Also increase the buffer for early stopping of training.

1.4 Summary