Danmarks Tekniske Universitet

Katrine Bay, s183910
Lukas Leindals, s183920
Matilde de Place, s183960
Sunniva Olsrud Punsvik, s183924

02463 - Aktiv machine learning og agency F20

Project 2
Detecting pneumonia in X-ray images

9th of April 2020
Danmarks Tekniske Universitet

Introduction

New techniques in computer science is developing rapidly and constantly, one new and intriguing method is *active learning*. Active learning can be applied in many contexts, in computer science the basic idea is that "a learning agent can ask meaningful questions about the world which acts within in order to learn what it does not already know"[1]. This mini-project will investigate the effects of implementing different active learning strategies in a pool-based scenario using an open source dataset of thoracic X-ray images from both healthy and pneumonia patients. By implementing these strategies: random sampling, entropy sampling, margin sampling and query-by-committee (QBC). Here the random sampling serves as a baseline model that the query strategies can be compared against. How well does the different query strategies perform opposed to the random sampling strategy and how little data does it take to train a model that performs as well as a model trained on all the data?

By applying active learning in this setting, one can obtain alternative methods for diagnosing and/or test for distinct diseases and infections which affects specific areas of the body and is detectable through some producible image. Reflecting on what is happening today, when the world is affected by a highly contagious virus, a pandemic that has resulted in pressured medical professionals, limited protection gear and test kits, etc., then active learning methods may help alleviate the pressure and help monitoring the spread of the virus.

Description of data

The data is acquired from Kaggle inspired from an article published in Cell Press[2] which is an open source dataset consisting of 5,863 X-ray images (JPEG) of a chest belonging to either a healthy person or a pneumonia patient, where the disease is caused by either bacterial or viral infection. The images are labelled with 3 parameters; disease, randomised patient ID, image number by this patient, and split into the 2 categories consisting of normal and diseased.

Method & Analysis

The model used to classify which X-ray images belongs to a pneumonia and healthy person is a convolutional neural network (CNN) with 2 convolutional layers, a 2D max pooling layer and 3 fully connected layers. The CNN uses a ReLU activation function and an ADAM optimiser with a learning rate of 0.001. This is trained for 3 epochs each time new training samples is queried to the model. This results in some very heavy computations and the training is therefore done on a Tesla V100-PCIE-16GB GPU via DTU clusters.

All active learning strategies are applied in the context of pool-based sampling, where a

[1] Active Learning Notes (12.03.2020)

[2] `https://data.mendeley.com/datasets/rscbjbr9sj/2`

subset of a certain pool i.e. a dataset of unlabelled data points is determined to query labels for. Here the model is first trained with 716 samples leaving a pool of 4500 unlabelled data points. From here, each of the strategies chose 300 samples to query for each of the 15 rounds.

The query strategies used is random sampling, uncertainty sampling, margin sampling and QBC sampling. When using random sampling 300 random training samples are queried to the model. When using using the uncertainty sampling strategy, it must be specified that the uncertainty is measured and computed as the entropy, which takes the uncertainty of all labels into consideration. When using margin sampling, the learner looks at the difference between the first and second most likely labels, e.g. the labels with highest calculated probability. The data points with the smallest difference is then selected for query. When using QBC sampling, 10 different models give their prediction of the label for each of the samples in the pool. From this margin sampling is used to find the samples that the 10 models disagree the most on. The 10 different models are obtained by bootstrapping 10 different training datasets from the labelled training data.

Results

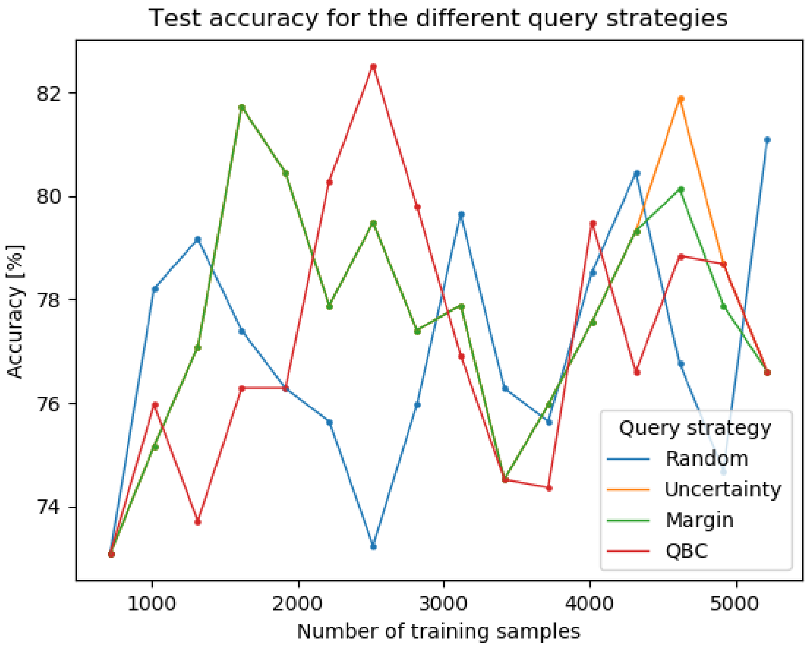

Figure 1: Comparison of test accuracy of the 4 different query strategies with seed 4200

Discussion & Conclusion (learning outcome)

Ideally all the query strategies should have the same accuracy when getting to 5217 training samples as this is all the data. However, when looking at figure 1 this does not seem to be entirely the case. A reason for this is that the structuring of the code for the different methods leads to drawing a different amount of numbers from the seed used and even though measures were taken in order to reset the seed at the right time, this did not seem to turn out perfectly.

When looking at figure 1 is seen that there does not seem to be a significant difference between the performance of the query strategies and random sampling would in this case be ideal due to less computational complex choice of query. This does however seem strange as the query strategies should perform at least as good as random sampling as this was the baseline. For future work, one might look into if there was a problem with the code not doing what was intended.

When looking at how little data should be used to obtain a model that performs as well as a model trained on all the data, figure 1 does not seem to provide any great answer as the query strategies does not seem to have any pattern when increasing the amount of training samples.

In conclusion random sampling seems to be the best of the 4 strategies tested as it is less computational heavy. However, there might be a problem in the code as all models does not perform evenly when using the same data in the end, and the query strategies perform worse than random sampling in some cases. Furthermore, the results does not seem to indicate any amount of training samples performing better than others.

From this project we have learned something about how to implement different query strategies for a convolutional neural network, we must however look into whether or not there was a problem with the querying of data, as our results did not come out to be as we expected. Furthermore, we gained some experience in how to use PyTorch and the DTU clusters with an arbitrary dataset.

Appendix

Link to code in GitHub

Link for Github repository