Convolution Neural Network for Polyp Detection in Colonoscopy

Ramachandra C Nanjegowda
School of Computing
University of Utah
Email: cn.ramachandra@gmail.com

Mohammad Rehan Ghori
School of Computing
University of Utah
Email: rghori7@gmail.com

Abstract—CNNs in biomedical image analysis is widespread, but its success is impeded by the lack of such large annotated datasets in biomedical imaging. The Polyp dataset [5] used within Automatic Polyp Detection challenge is one of the openly available annotated dataset of colonoscopy images. We evaluate the convolutional neural networks (CNNs) for polyp detection on these set of colonoscopy images.

I. INTRODUCTION

Colorectal cancer (CRC) is the third largest cause of cancer deaths in United States among men and women. Just in 2016, it resulted in 49,196 deaths [2]. And the numbers continue to grow due to the aging population. Colonoscopy is the gold standard test for colon cancer screening. The analysis of the colonoscopy results requires extensive analysis by radiologist and gastroenterologist. Even with the subject matter experts, it is possible to miss polyps in the colonoscopy images. To date, several computation systems have been proposed for polyp detection during colonoscopy but so far, the results have been inconsistent.

Recently different deep learning architectures involving different CNN architectures have demonstrated detection results better than other computational methods. Convolutional neural network is the state of the art technique in machine learning and is more successful [1] than other computational methods. We are motivated to research the literature and use CNN architectures as recommended in [1] and apply them on the colonoscopy images dataset.

II. RELATED WORKS

The Automatic Polyp Detection sub-challenge, conducted as part of the Endoscopic Vision Challenge (http://endovis.grand-challenge.org) at the international conference on Medical Image Computing and Computer Assisted Intervention (MICCAI) in 2015, was an effort to make the polyp dataset available for public. This dataset was used to report the results of a comparative evaluation of different polyp detection methods [1]. Tajbakhsh et al. [3] systematically investigated the capabilities of transfer learning in several medical imaging applications. And then proposed a novel AIFT method [4] to integrate active learning into fine-tuning CNNs in a continuous fashion to make CNNs more amicable for biomedical image analysis with an aim to cut annotation cost dramatically.

III. OUR METHOD

Convolutional neural network (CNN) is the state of the art technique in machine learning and is more successful [1] than other computational methods. CNNs are special type of feed forward neural networks. Although simpler than traditional feed forward networks, they perform very well on the visual recognition tasks. We are using CNNs for polyp detection in the colonoscopy images. We are using Caffe, which is a deep learning framework developed by Berkley Vision and Learning Center (BVLC). It is written in C++ and has python and Matlab bindings. We used CNN for binary classification of the data, i.e. frames with polyp and framer without polyp.

Following are the high level steps to get the preliminary results:

- Data Preparation: Caffe requires data in certain format. It doesn’t work with video files. Our data preparation is described in detail in the next section.
- Model Definition: We used the existing CNN architecture from the example and only updated some of the parameters to work with the polyp and no polyp images in our dataset.
- Solver Definition: Solver is responsible for model optimization. We reused the solver parameters from the example and defined them in the configuration file.
- Model Training: We trained the model by executing Caffe command from the terminal.
This produced the trained model that was used to make predictions for the new dataset. We also spent some time on understanding and defining the performance benchmarks for polyp detection as described in [1]

IV. DATA PREPARATION

Our input data consisted of 20 videos in .wmv format. We used OpenCV to convert them into .mp4 format and then extracted colored frames from each video file. On average, each video produced just under 2000 colored images. We also appended ‘wp’ for with poly and ‘np’ for no poly to the file names. We then separated and set aside 20% of the images as test data and used 80% of the images for the training. The the contrast of all images is adjusted by running histogram equalization. Finally all images are converted to 227 x 227 based on some of the examples online. We ended up having more than 14000+ images for the training data and around 4000+ images for the test data.

V. CNN ARCHITECTURE

We are using the bvlc reference caffenet architecture which is a replication of the AlexNet[6] with some modifications. Our architecture consists of the following layers:

- **Input Layer:**
  - Uses images of size 227 x 227 with 3 channels
- **Convolution Layer 1:**
  - Kernel size 11, Stride 4, Filter type Gaussian, Number of Outputs 96
- **ReLU Layer 1:**
- **Pooling Layer 1:**
  - Kernel size 3, Stride 2
- **LRN Layer 1:**
  - alpha 0.0001, beta 0.75
- **Convolution Layer 2:**
  - Kernel size 5, Padding 2, Filter type Gaussian, Number of Outputs 256
- **ReLU Layer 2:**
- **Pooling Layer 2:**
  - Kernel size 3, Stride 2
- **LRN Layer 2:**
  - alpha 0.0001, beta 0.75
- **Convolution Layer 3:**
  - Kernel size 3, Padding 1, Filter type Gaussian, Number of Outputs 384
- **ReLU Layer 3:**
- **Convolution Layer 4:**
  - Kernel size 3, Padding 1, Filter type Gaussian, Number of Outputs 384
- **ReLU Layer 4:**
- **Convolution Layer 5:**
  - Kernel size 3, Padding 1, Filter type Gaussian, Number of Outputs 384
- **ReLU Layer 5:**
- **Pooling Layer 5:**
- **Fully Connected Layer 6:**
- **ReLU Layer 6:**
- **Fully Connected Layer 7:**
- **ReLU Layer 7:**
- **Dropout Layer 7:**
- **Fully Connected Layer 8:**
- **Softmax Layer:**
- **Final layer:** Label

We think going to the details of each layer is outside of the scope of this report, but the above listing of layers can be used to get an idea of the architecture of the training model.

VI. EXPERIMENTS

We used a sample Caffe example and adopted to process the polyp image dataset using the architecture described in the previous section. Below are the steps involved in the experiments.

- Creating the LMDB database: Divide the training data into 2 sets: One for training and the other for validation. The training set is used to train the model, and the validation set is used to calculate the accuracy of the model. Store the training and validation in 2 LMDB databases, train-lmdb for training the model and validation-lmdb for model evaluation. Using the ‘wp’ or ‘np’ postfix in the image filename, we assigned the label 1 or 0 to the database entries.
- Computing the image mean: Generate the mean image from each input image to ensure every feature pixel has zero mean. This is a common preprocessing step in supervised machine learning.
- Model and Solver definition and training: The caffe framework provides .prototxt to define the model (section V) and solver which helps to specify different parameters and hyper parameters of training.
- Prediction: Now that we have a trained model, we can use it to make predictions on new unseen data.

VII. ANALYSIS

The learning curve plot in figure 2 shows the training and test losses as a function of the number of iterations. These plots are very useful to visualize the train/validation losses and validation accuracy. We can see from the learning curve that the model achieved a validation accuracy of 90 percent, and it stopped improving after 1000 iterations. The accuracy was pretty good, but there are false positives, when the model is classifying the image without polyp as polyp.

After debugging this we found out that the images in the training set were incorrectly labeled. Some of the images which did not have polyp were incorrectly labeled as with polyp, and hence the model suffered false positives. After fixing this bug the learning plot showed 100 percent accuracy after 2000 iterations.

However having accuracy of 100 percent seemed too perfect. On closely examining the whole process, we found out two issues with our setup that contributed to the 100 percent accuracy

- We did not shuffle the input data. A single video produced around 2000 images and based on the label of the video,
all the images belonging to the video were assigned the polyp or no polyp label. Without shuffling the data, we ended up having series of 1000+ examples with the same label, followed by another series of 1000+ examples with the opposite label. While researching the results, we came to a conclusion that CNNs are quite powerful and data with sequence of same labels would contribute to an accuracy of 100 percent.

- A single video with positive for polyp can have few images with no polyp since a colonoscopy video doesn’t start with polyp immediately. And we found one example of this behavior in the ground truth data. But there is no easy way to compare our colored images (produced with OpenCV) with the ground truth images of the same video. For a given video, if we extracted 1500+ images for a given video, the ground truth data would only have 200+ images. We concluded that a better method to produce the input data from a video would have been to first find out how many ground truth images belonged to a single video, record the count, then extract colored images by sampling the video at the interval corresponding to the count of the ground truth images.

VIII. CONCLUSION AND FUTURE WORK

Overall, we learned a lot about neural networks specially CNNs by taking on this project. Along the way, we also learned the intricacies of taking a real world example and using highly advanced Machine Learning technique to solve the problem. We were also able to understand the problem with the accuracy of our model by going back to some of the principles and techniques we learned in our machine learning class assignments.

In the future, we want to continue working on this problem, fix some of the problems identified in the data preparation phase and also focus on fine tuning the model.

REFERENCES