Liver Lesion Localization using Deep Convolutional Neural Networks

Sheun Aluko
Department of Medicine
Stanford University
oluwa@stanford.edu

Surya Narayanan Hari
Department of Management Science and Engineering
surya21@stanford.edu

Vadim Piccini Yakubenko
Department of Statistics
Stanford Univeristy
vadimp@stanford.edu

Abstract

Within the field of medicine, automated annotation of Computed Tomography (CT) scans is poised to revolutionize the practice of radiology and to improve patient care through increased efficiency and accuracy of diagnosis. In this work, we develop a deep learning algorithm designed for the automated detection of liver lesions on CT scans. We explore three distinct neural network architectures for the object detection task, with increasing complexity respectively. These consist of a baseline convolutional network, a transfer learning approach which incorporates VGG-16 features into the baseline model, and finally the Faster R-CNN network. We find that Faster R-CNN greatly outperforms the baseline and can achieve a sensitivity of 0.5 with one average false positive per image even when trained and evaluated on noisy labels.

1 Introduction

Computed Tomography (CT) scans are a widely used imaging modality in global healthcare systems owing to their fast acquisition time and high resolution 3D rendering of the human body. These scans are read by radiologists who spend hours each day visually inspecting and interpreting the images, to provide a report that will be used to guide clinical care. Initially, lesion detectors will likely serve as an aid to the radiologist to highlight potential lesions that they may have missed. This will augment the radiologists ability to process large volumes of images and scale to the volumes required for our growing healthcare system, without a simultaneous decrease in accuracy.

This work specifically explores the automated detection of liver lesions in CT images. The liver performs essential filtration of blood from the digestive tract as well as detoxifies and metabolizes harmful chemicals in the body. Because the liver is the largest internal organ by mass and in a healthy state appears relatively homogenous on CT Imaging, it is an appropriate baseline target for lesion detection. Lesions in the liver consist of benign cysts, collections of blood (hematoma), and hepatocellular carcinoma among other pathologies. Our detector does not aim to classify these lesions, but instead to localize any abnormality that is present.

Specifically, we explore three neural network architectures for the task of lesion detection. The input to each model consists of 3 slices of CT image data of the liver (the slice with the lesion of interest and its two adjacent slices which are provided for volumetric context). The output of the model consists of 4 bounding box coordinates, which attempt to localize the lesion. The first model architecture consists of a baseline model with 4 convolutional and one fully connected layer. Next, we develop a model that computes VGG-16 features from the input and concatenates these with the input before it is passed into the baseline convolutional model. Finally, we implement the Faster R-CNN architecture. We discuss the performance of these models and compare them to a reference publication which utilizes the same dataset.

CS230: Deep Learning, Fall 2019, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

2 Related work

Our project is based on the DeepLesion paper published by Yan, Lu and Summers [3]. Yan et al produced a 220GB CT lesion dataset by mining a hospital imaging system, and released it for public use under the name of "DeepLesion". Using this dataset, they created a deep learning model for automated localization of CT Scan lesions which aimed to detect any type of lesion in any part of the body, and they report a sensitivity of 81.1 percent with five false positives per image. The researchers chose VGG16 as an ImageNet backbone and implemented a modified RCNN architecture for lesions localization. We instead focus only on liver lesions, and thus use a subset of the DeepLesion dataset, with the goal of achieving a higher sensitivity and better false positive rate than that reported by Yan et al. Additional prior work on liver lesion detection in CT scans can be divided into deep learning (convolutional) approaches and geometric approaches, and are described below.

Convolutional Methods: Chris et al [2] explore liver segmention and lesion localization using FCNs and conditional random fields (CRFs). They achieve a Dice score of 94%. Li et al [4] also perform liver tumor segmentation but compare results to tradional ML models, and report recall of 84.34%. These studies differ to our study in that they performed segmentation rather than bounding box regression, and focused exclusively on liver tumors rather than all type of liver abnormalities.

Geometric Methods: Rusko et al [5] perform liver lesion detection using algorithmic detection of basic geometric features (such as assymetry, compactness, and volume) rather than a deep learning method, and are able to achieve a detection rate of 92%. Additionally, Ben-Cohen et al [1] perform detection using adaptive region growing and mean shift clustering, and are able to achieve detection rate of 90%.

3 Dataset and Pre-processing

We used the publicly available DeepLesion dataset with over 10,000 Computed Tomography (CT) studies and over 4,000 patients [6]. This data set includes approximately 32,000 2D slices of CT scans that are annotated with lesion type, bounding box and metadata. The image files are available as 512x512 pixel 2D slices stored in png format for compression. We modified the download script and successfully downloaded the 220GB dataset to a local machine for pre-processing.

This work required extensive pre-processing and understanding of the CT medical imaging format. The raw values in CT data consist of Hounsfield units, which are units of xray attentuation. When radiologists view CT images on the computer the images are "windowed" such that a specific range of Hounsfield units is mapped to gray scale and values outside the range are rendered as black or white. This process aids in the viewing of specific organs. We wrote the appropriate python scripts for parsing the raw png files and applying the appropriate windowing to each. For each slice we selected the neighbor slices to produce a three channel input, which results in a model input shape of 512x512x3. All pixel values were re-scaled to [0,255]. Of these, the liver lesions were selected and moved to an AWS instance for model training. This resulted in a final 10GB dataset with 1989 training set images, 179 validation set images, and 203 test set images. An example image is shown in Figure 1.

4 Methods

We divide the methods and results sections into two subsections: a baseline convolutional model with and without VGG-16 transfer learning and a final model which uses the Faster R-CNN architecture.

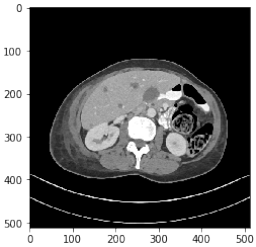


Figure 1: Example CT Image

4.1 Baseline and VGG Transfer Learning Models

4.1.1 Shared Model Features

Both the baseline model and the VGG transfer learning model were convolutional neural network models, and both utilized the mean squared error loss function, given by $MSE = \frac{1}{n}\sum_i^n (Y_i - \hat{Y}_i)^2$, (where Y_i and $\hat{Y}_i$ represent 4d bounding box ground truth and pre-

diction respectively). Both were trained on the training set with Adam optimization (with β_1 = 0.9 and $\beta_2 = 0.999$), and results were analyzed using the validation set during iteration. The test set was not used until the end of the study for model accuracy calculation.

4.1.2 Baseline Convolutional Model

Figure 2 describes the baseline convolutional model architecture, which consisted of 2 convolutional layers and a pooling layer, followed by 2 convolutional layers, a pooling layer, and a dense layer with 4 outputs for the bounding box. The dimensions of these filters are shown in Figure 2. The Relu function was used for all Activation layers. No form of regularization was applied to the baseline model.

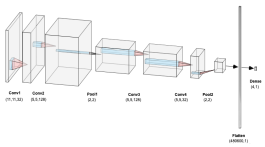


Figure 2: Baseline Model Architecture

4.2 Baseline with Transfer Learning

For the baseline with transfer learning model, we employed two strategies for incorporating VGG-16 features into the baseline model. First, we added a VGG-16 frontend to the baseline model by inputting the image through VGG-16 and extracting the feature matrix at various VGG-16 convolutional layers (Convolutional blocks 2,3, and 4) and then passing the resulting feature matrix as input to the baseline model.

Second, we utilized a modified VGG16 architecture (which removed the block 4 and 5 pooling layers) in order to generate a feature matrix which had enhanced resolution (output size 64x64x512). This feature matrix was then upsampled to (512x512x512) and concatenated with the original input image. This modified input served as the input to the original baseline model.

Our experimental analysis of these methods revealed that the latter architecture (concatenation of VGG features with the original input image) was more robust and thus the remainder of the paper focuses on this method.

4.3 Faster RCNN

The Faster RCNN model uses a convolutional neural network to determine region proposal. This CNN, called a Region Proposal Network(RPN), predicts object bounds and class scores. The RPN shares convolutional features with the detection network, allowing for low-computational-cost region proposals. The bounding boxes predicted by the RPN are fed into a Region of Interest (ROI) pooling layer which performs max pooling on inputs of different shape and outputs feature maps of the same size. The feature maps are then fed into a classifier network that outputs class probabilities as well as into a regressor network that outputs the final bounding boxes. The RPN and the final classifier and regressor can be trained alternatively or at the same time, the latter being faster.

The RPN loss for an image is defined as:

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*) \tag{1}$$

Where p_i is the predicted probability of the anchor being an object and p_i^* is the ground truth label, which is 1 when the IoU of the anchor box i and any ground truth box is > 0.7 and zero otherwise. t_i and t_i^* are the vectors representing the parameterized bounding boxes coordinates of the predicted and ground truth box respectively. L_{cls} is the log loss over the two classes (foreground and background) and L_{reg} is defined as $R(t_i - t_i^*)$, where R is smooth L1 loss. λ is a weight parameter.

5 Experiments and Results

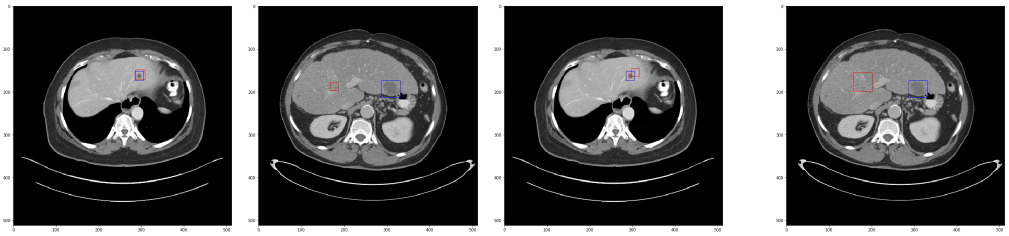
5.1 Baseline and VGG Transfer Learning Models

5.1.1 Hyperparameter Selection

For the baseline model, the main hyperparameters we tuned were the batch size and learning rate. For the batch size, we trained the model for 1 epoch on a small subset of data over a range of batch

Model	MSE	Sensitivity
Baseline Train	0.00077	0.96
Baseline Test	0.065	0.069
Baseline+VGG Train	0.0025	0.79
Baseline+VGG Test	0.067	0.079

Figure 3: Baseline and Baseline+VGG Results



(a) Baseline Train (b) Baseline Test (c) Baseline+VGG Train (d) Baseline+VGG Test

Figure 4: Example Visualizations of Baseline and Baseline+VGG Model Performance on Train and Test Sets

sizes, including 1,2,4,8, and 16. We found that the epoch trained fastest with a batch size of 8. Next, we trained the baseline model for 20 epochs using a logarithmic range of learning rates, and found that 0.003 produced a promising initial loss curve whereas 0.01 quickly failed to converge. A similar analysis was performed for the transfer learning model which led to the selection of a batch size of 1 and learning rate 0.0001. For the transfer learning model we also chose a dropout rate of 0.05.

5.1.2 Metrics

Both the baseline and VGG transfer learning models utilized the intersection over union (IoU) metric, given by $IoU = \frac{Area_{Overlap}}{Area_{Union}}$. For the train and test set, we considered a prediction to be accurate if the IoU was greater than 0.1. Because both of these methods produced single bounding box outputs, the accuracy and sensitivity are equal. When discussing the Faster RCNN network, however, and comparing to existing literature, sensitivity becomes more important as multiple regions may be proposed.

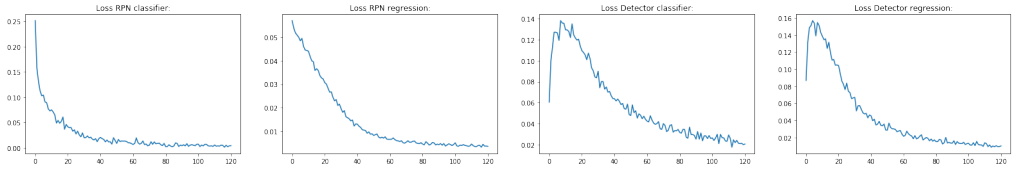
5.1.3 Results

Results for final loss and sensitivity are reported in Figure 3. The baseline model was trained for 150 epochs and was able to achieve 96% sensitivity on the training set, however failed to generalize and only achieved a 6.9% sensitivity on the test set. The transfer learning model (baseline with concatenated features from VGG16) was trained for 70 epochs and achieved reasonable sensitivity on the training set (79%), and very slightly increased sensitivity on the test set (7.9%) compared to baseline. Clearly, both of these models were overfitting the training data. However, the transfer learning model was able to slightly increase test set sensitivity, likely because of a combination of additional information from VGG features and from mild regularization via dropout.

5.2 Faster R-CNN

We set Faster R-CNN to run with anchor scales 16, 24, 32, 48 and 96 and three anchor ratios (1:2, 1:1 and 2:1) in line with the anchor configuration in the DeepLesion paper [6]. We used ResNet50 as the feature network. The train/validation split used was 76/15/9 (1660/329/203). After 150 epochs the validation losses for the RPN as well as the final regressors and classifiers converged.

The metrics tracked were the RPN loss of both the RPN classifier and regressor as well as the final classifier and regressor on a validation set, shown in Figure 6. We also computed the false positive rate and sensitivity on the previously unseen test set. We found a test set false positive rate of 67% (meaning 2/3 of predictions made were false positives) and a sensitivity of 50%. An example accurate Faster RCNN test set prediction is shown in Figure 5.



(a) RPN Classifier (b) RPN Regression (c) Detector Classifier (d) Detector Regression

Figure 6: Faster RCNN RPN and Detector Loss Curves

6 Discussion

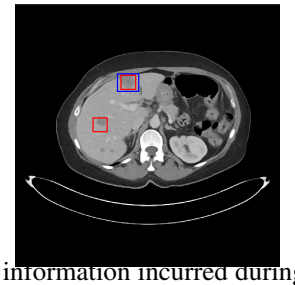


Figure 5: Faster RCNN Prediction

As the information content in CT scans is highly complex and variable, we did not expect our baseline model to achieve high accuracy. Although both the CNN baseline and the transfer learning model (with concatenated features from VGG-16) achieved better than random-chance performance on a test-set, they clearly could be significantly improved upon. During the experimentation for the transfer learning model, we originally attempted to pass the input through a VGG frontend rather than concatenating VGG features to the input. It was interesting to note that this former method prevented our model from overfitting on the training set as much as the baseline was able to. We believe this is because of the loss of spatial information incurred during the VGG front end extraction. Our final transfer learning model, where the VGG16 features were concatenated to the input and then fed into the baseline model, was able to get comparable loss to the baseline and improved the test performance marginally (0.06 vs 0.07 test set sensitivity). Using Faster R-CNN, as expected, improved our baseline results dramatically and we got results comparable with the existing literature, when taking into account additional context. Prior work referenced at the beginning of this paper largely achieved sensitivity measures ranging from 80 to 90% for liver lesion detection, compared to our reported 50% with Faster RCNN. However, upon closer inspection we find that Yan et al [3] report a range of sensitivities for liver lesion detection as a function of false positives per image. They are able to achieve approximately 90% sensitivity but with 10 false positives per image on average, versus 65-70% sensitivity with 1 false positive per image. In this work we achieve 50% sensitivity with approximately 1.1 false positives per image, and thus our results are comparable when considering the operating point of the FROC curve. Tuning our Faster RCNN architecture to explore a range of FROC operating points is discussed in future work below.

A challenge that we faced was working with noisy labels. The labels from our dataset only identify the main lesion in the CT scan and leave out other lesions that are present (a problem that is inherent to the DeepLesion dataset and publication [3]). This both adds noise to the training process as well as inflating the false positive rate.

7 Conclusion and Future Work

In this work we explored three model architectures for automated liver lesion localization in CT scans. Our custom built baseline model and transfer learning model with VGG feature extraction both were able to overfit to the training dataset, but failed to generalize on the test set. The Faster RCNN architecure performed comparably to reference literature when taking into account false positive rate. In the future we would like to explore the trade off between Faster RCNN sensitivity and higher false positive rate. We would also explore the effect of using different feature networks for Faster RCNN such as VGG-16. Finally, we would expand our analysis to include lesions from all parts of the human body, not just the liver.

8 Contributions

Vadim Piccini Yakubenko set up and trained Faster-RCNN, made most of the poster and contributed to loading the VGG-16 layers to the baseline model. Surya Narayanan worked on setting up a

baseline model, and contributed to the proposal, report and this paper. Sheun Aluko contributed to data pre-processing, baseline and transfer learning architectures, and to this paper.

References

[1] A. Ben-Cohen, E. Klang, I. Diamant, N. Rozendorn, M. M. Amitai, and H. Greenspan. Automated method for detection and segmentation of liver metastatic lesions in follow-up CT examinations. *Journal of Medical Imaging*, 2(3):1 – 12, 2015.

[2] P. F. Christ, M. E. A. Elshaer, F. Ettlinger, S. Tatavarty, M. Bickel, P. Bilic, M. Rempfler, M. Armbruster, F. Hofmann, M. D'Anastasi, W. H. Sommer, S.-A. Ahmadi, and B. H. Menze. Automatic liver and lesion segmentation in ct using cascaded fully convolutional neural networks and 3d conditional random fields. In S. Ourselin, L. Joskowicz, M. R. Sabuncu, G. Unal, and W. Wells, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016*, pages 415–423, Cham, 2016. Springer International Publishing.

[3] C. H. Lee and H.-J. Yoon. Medical big data: promise and challenges. 36(1):3–11.

[4] W. Li, F. Jia, and Q. Hu. Automatic segmentation of liver tumor in ct images with deep convolutional neural networks. 2015.

[5] L. Ruskó and Á. Perényi. Automated liver lesion detection in ct images based on multi-level geometric features. *International Journal of Computer Assisted Radiology and Surgery*, 9(4):577–593, Jul 2014.

[6] K. Yan, X. Wang, L. Lu, and R. M. Summers. Deeplesion: Automated deep mining, categorization and detection of significant radiology image findings using large-scale clinical lesion annotations, 2017.

[7] Faster R-CNN implementation: https://github.com/kbardool/keras-frcnn