Udaikaran Singh and Wesley Kwan

Gary Cottrell

CSE 190

11/11/2018

**PA3: Deep Convolutional Network for Thorax Disease Detection**

**Udaikaran Singh Wesley Kwan**

Department of Computer Science Department of Computer Science

University of California, San Diego. University of California, San Diego

La Jolla, CA 92093. La Jolla, CA 92093

**Abstract**

**(…)**

**Introduction**

[explain the problem]

[explain why we are using convolutional neural networks]

[explain different model architectures]

[explain Xavier weight initialization]

[explain experiments]

[explain issue with class imbalance & weighted loss function]

[explaining testing stats and confusion table]

**Related Works**

**(…)**

**Methods**

(i) Baseline Architecture

Baseline Architecture

|  |  |
| --- | --- |
| Layer (from input to output) | Description of Layer: |
| Input Layer | The input image is 1024 x 1024 x 1.  The image is in greyscale. |
| Convolutional Layer 1 | In-channel = 1  Out-channel = 12  Kernel size = 8  Zero-Padding = 0  Stride = 1  Activation Function = relU  Batch Normalization Applied |
| Convolutional Layer 2 | In-channel = 12  Out-channel = 10  Kernel size = 8  Zero-Padding = 0  Stride = 1  Activation Function = relU  Batch Normalization Applied |
| Convolutional Layer 3 | In-channel = 10  Out-channel = 8  Kernel size = 6  Zero-Padding = 0  Stride = 1  Activation Function = relU  Batch Normalization Applied |
| Max-Pool Layer | Kernel size = 4  Stride = 4 |
| Fully Connected Layer 1 | In-features = 121032  Out-features = 128  Activation Function = relU  Batch Normalization Applied |
| Fully Connected Layer 2 (output) | In-features = 128  Out-features = 14  Activation Function = Sigmoid |

The loss criterion used is a binomial cross entropy function. The weight parameters were initialized using Xavier weight initialization. The gradient descent optimization used was the adam optimizer. No regularization was added to the model. We dealt with the class imbalance issue by implementing a weights loss function that punishes false negatives. The motivation of this is that this motivates the model to learn rare cases. We implemented cross validation by leaving out 10% of the training set and testing against it to determine whether the model is overtraining.

(ii) Experimental Architecture

Experimental Architecture 1

|  |  |
| --- | --- |
| Layer (from input to output) | Description of Layer: |
| Input Layer | The input image is 1024 x 1024 x 1.  The image is in greyscale. |
| Convolutional Layer 1 | In-channel = 1  Out-channel = 4  Kernel size = 8  Zero-Padding = 0  Stride = 1  Activation Function = relU  Batch Normalization Applied |
| Convolutional Layer 2 | In-channel = 4  Out-channel = 8  Kernel size = 8  Zero-Padding = 0  Stride = 1  Activation Function = relU  Batch Normalization Applied |
| Convolutional Layer 3 | In-channel = 8  Out-channel = 12  Kernel size = 6  Zero-Padding = 0  Stride = 1  Activation Function = relU  Batch Normalization Applied |
| Max-Pool Layer | Kernel Size = 4  Stride = 4 |
| Fully Connected Layer 1 | In-features = 178608  Out-features = 512  Activation Function = relU  Batch Normalization Applied |
| Fully Connected Layer 2 | In-feature = 512  Out-feature = 128  Activation Function = relU  Batch Normalization Applied |
| Fully Connected Layer 3 (output) | In-features = 128  Out-features = 14  Activation Function = Sigmoid |

Experimental Architecture 2

|  |  |
| --- | --- |
| Layer (from input to output) | Description of Layer: |
| Input Layer | The input image is 1024 x 1024 x 1.  The image is in greyscale. |
| Convolutional Layer 1 | In-channel = 1  Out-channel = 16  Kernel Size = 8  Zero-Padding = 0  Stride = 1  Activation Function = relU  Batch Normalization Applied |
| Convolutional Layer 2 | In-channel = 16  Out-channel = 14  Kernel Size = 8  Zero-Padding = 0  Stride = 1  Activation Function = relU  Batch Normalization Applied |
| Convolutional Layer 3 | In-channel = 14  Out-channel = 12  Kernel Size = 8  Zero-Padding = 0  Stride = 1  Activation Function = relU  Batch Normalization Applied |
| Max-Pool Layer | Kernel Size = 3  Stride = 3 |
| Convolutional Layer 4 | In-channel = 12  Out-channel = 10  Kernel Size = 6  Zero-Padding = 0  Stride = 1  Activation Function = relU  Batch Normalization Applied |
| Convolutional Layer 5 | In-channel = 10  Out-channel = 8  Kernel Size = 6  Zero-Padding = 0  Stride = 1  Activation Function = relU  Batch Normalization Applied |
| Max-Pool Layer | Kernel Size = 3  Stride = 3 |
| Fully Connected Layer 1 | In-feature = 20808  Out-feature = 128  Activation Function = relU  Batch Normalization Applied |
| Fully Connected Layer 2 (output) | In-feature = 128  Out-feature = 14  Activation Function = sigmoid |

For both of these architectures, we used the adam gradient descent optimizer, no regularization function, and Xavier weight initialization. The class imbalance was addressed best testing a weighted loss function that punishes false negatives heavily. The idea is that the model will learn to guess negative for each disease, so by increasing the punishment for this, this motivates the better learning of rare classes. We implemented cross validation by leaving out 10% of the training set and testing against it to determine whether the model is overtraining.

**Results**

[describe implementation of results]

Experiment 1: Baseline Architecture

(i) Loss Curves

(ii) Accuracy Curves

(iii) Visualization of filter maps

(iv) Model Results

Experiment 2: Architecture 1 (w/o addressing class imbalance)

(i) Loss Curves

(ii) Accuracy Curves

(iii) Visualization of filter maps

(iv) Model Results

Experiment 3: Architecture 2 (w/o addressing class imbalance)

(i) Loss Curves

(ii) Accuracy Curves

(iii) Visualization of filter maps

(iv) Model Results

Experiment 4: Architecture 2 with Normalized Inputs

(i) Loss Curves

(ii) Accuracy Curves

(iii) Visualization of filter maps

(iv) Model Results

Experiment 5: Architecture 2 with Scaled Down Image (by ½)

(i) Loss Curves

(ii) Accuracy Curves

(iii) Visualization of filter maps

(iv) Model Results

Experiment 6: Architecture 2 with weighted objective function

(i) Loss Curves

(ii) Accuracy Curves

(iii) Visualization of filter maps

(iv) Model Results

Experiment 7: Architecture 2 with weighted objective function & Normalized Inputs

(i) Loss Curves

(ii) Accuracy Curves

(iii) Visualization of filter maps

(iv) Model Results

**Discussions**

**(…)**

**References**

**(…)**

**Authors’ Contributions**

**(…)**