Classifying Arrhythmia using Artificial Neural Networks

Siddarth Sampangi
College of Information and Computer Sciences
University of Massachusetts, Amherst
Amherst, MA 01003
ssampangi@cs.umass.edu

Abstract

Cardiac arrythmia refers to any abnormality in the sequence of electrical signals that govern heartbeats. There are many types of cardiac arrhythmia ranging in severity, including premature beats, atrial fibrillation, and ventricular fibrillation. While arrhythmia classification has been well researched, this study focuses on the use of recent techniques in deep learning to classify arrhythmia with minimal possible data pre-processing.

1 Introduction

2 Related work

3 Network architectures

In this section, we shall provide a brief introduction to convolutional neural networks (CNN) and Long Short-Term Memory (LSTM) networks. These two types of networks are themselves variants of a general class of discriminative classifier called artificial neural networks (ANN). The atomic unit of the ANN is called a neuron. A neuron can receive inputs from any number of real-valued signals, and it produces a real-valued output by taking the inner product of its inputs with a weight matrix W, adding a bias b, and then feeding the result to a nonlinear function. Multiple neurons that each receive identical copies of the input form a layer, and a basic ANN is formed by stacking layers such that the output of the i^{th} layer is the input of the $(i+1)^{th}$ layer.

An ANN is trained via stochastic gradient descent (SGD) over a loss function (usually some form of $truelabel - prediction$) with respect to the W matrix and bias b of each neuron. To accomplish this, we use the backpropagation algorithm to localize dependencies so that, given a loss function L, we can easily calculate the gradient $\frac{\partial L}{\partial W_i}$ of the weights of a neuron i given the gradients $\frac{\partial L}{\partial W_j}$ of the weights of all neurons j that i outputs to. Thus, the backpropagation algorithm calculates the gradients with respect to the final layer of the network, and then propagates the gradient to each preceding layer. For this to work, it is very important for the nonlinearity used by each neuron to be differentiable.

3.1 Convolutional neural networks

Convolutional networks are designed to exploit the spatial dependencies between units of an input. For example, an image of a flower is recognized as an image of a flower not only because of the RGB values of each pixel, but also because of the position of each pixel in relation to the surrounding pixels. Randomly scrambling the pixel locations in an image of a flower would result in an image that can no longer be recognized. As such, valuable information could exist in the order that an input is presented to the network. Currently, a vanilla ANN would output the same value regardless of the order of its inputs ($W_1x + W_2y = W_2y + W_1x$).

A one dimensional convolution operation involves a filter of size F, an input of size N, a stride of size S, and a padding of size P. A convolution is performed by mirroring the filter across its axis, and using it as a sliding window over the input, at each step taking the inner product of the filter with the units of the input that the filter's window resides over at that step. The stride S dictates how many input units to skip at each step, and the padding P dictates the number of zero-padding units that are appended to both ends of the input. Since convolving a filter of size x with an input of size y results in an output of size $y - x + 1$, padding is used to ensure a fixed output size. Thus, the overall formula for calculating output size is $\frac{N-F+2P}{S} + 1$. 1D convolution can be trivially extended to 2D.

Intuitively, a convolution of a filter over an image can be understood as an operation where the filter checks for the presence or response of a specific signal G in the image. This response will be maximal when the filter's window lies directly over the signal it is trained to respond to. Thus, the output of a convolution can be seen as a sort of heatmap of the likelihood of the presence of G at each point in the image.

A convolutional layer is specified by the number of filters it contains, and the size, stride, and padding (usually common across all filters in a layer) of each filter. The input to the layer is supplied to every filter, and the output of each filter is stacked and outputted to the next layer. A CNN is comprised of one or more convolutional layers, followed by one or more dense layers. A dense layer contains neurons that each receive the entire input of the layer.

3.2 LSTM networks

4 Data preprocessing

5 Results

6 Future work

7 Conclusion