INFERRING PROPERTIES OF NEURAL NETWORKS WITH INTELLIGENT DESIGNS

Draft of March 14, 2018 at 18 : 40

BY

KARAN GANJU

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2018

Urbana, Illinois

Adviser:

Carl Gunter

ABSTRACT

To my parents, for their love and support.

ACKNOWLEDGMENTS

Input your text here

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CHAPTER 1

INTRODUCTION

Neural Networks have gained great popularity both in academia as well as industry due to their proven ability to solve difficult tasks with reassuring capability. They are already being employed as part of conversational agents, autonomous vehicles and facial recognition software in the industry, just to name a few applications. Likewise, they have also shown great promise for sensitive operations such as facial recognition, speech recognition, precision medicine and other fields. In fields like malware analysis and network anomaly detection, there is a growing drive to share learned models within the community, which can help in serving quicker and more standardized responses to such issues.

However, it is important to draw the distinction between sharing hard-coded rules, such as antivirus policies and signatures, and learned anomaly classifiers. While the former can be easily interpreted and understood, the latter is often treated as a black box when attempting to derive the logic it serves. Additionally, the logic for the latter is a function derived from the training data it sees. As a simple example, consider the case of Support Vector Machines, a linear model often used for classification or regression. The classifier learned by the SVM, a hyperplane which divides the domain space into 2, can be represented as a weighted sum of it's training data points. Hence, while sharing classifiers, one must keep in mind that there is inevitably some leakage of data, and consequently privacy, as well.

Property Inference is the problem of inferring properties about the training dataset using only the parameters of the trained classifier as prior. These properties could range from bias in the dataset to existence of certain types of samples detec

CHAPTER 2

BACKGROUND

2.1 Neural Networks

Neural networks are machine learning models that are typically built of several layers of computational units that process or transform the output of the previous layer and produce input for the next layer. The type of transformation often depends on the type of the task at hand. For example, convolutional layers are often employed for images while recurrent layers have been traditionally employed for text. The most commonly used layers are linear layers built up of multiple perceptrons. Each perceptron has an associated weight for each each of it's inputs and an additive bias. Hence, for the input to the layer, x_{i-1}, the output of the perceptron x_{ij} with weight row vector $w_{ij} \in \mathbb{R}^{|x_{i-1}|}$ and bias $b_{ij} \in \mathbb{R}^1$ is

$$x_{ij} = w_{ij}^T \cdot x_{i-1} + b_{ij}$$

and the layer output, x_i is given as

$$x_i = [x_{i0}, x_{i1}, \cdots x_{iN}]$$

For each layer, the choice of number of perceptrons is a design choice and is treated as a hyperparameter to be tuned. The number of perceptrons in the final layer could range from 1 for regression and 2-way classification tasks to k for k-class/label classification.

2.2 Property Inference

Property inference attacks explore the general problem

CHAPTER 3

USING GRAPH CONVOLUTIONAL NETWORKS

While sorting and normalizing are processing steps that may partially aid in inferring attributes about classifiers, they are not able to fundamentally understand the architecture of the classifier. Need experiment here. In order to exploit the additional information we have about the architecture, we need a more intelligent design.

One way to address this issue is to change the representation of the classifier from a flat vector to that of a graph, with different neurons being the nodes, connected to each other with directed weighted edges and having an inherent bias feature. However, we also need a machine learning construct which will be able to deal with graphs and Graph Convolutional Networks do just that.