UNIVERSITY OF WATERLOO
Software Engineering

Application of Machine Learning Models as a Solution to Hyperbola Identification in Ground Penetrating Radar Data

Sensors & Software Inc.
Mississauga, ON

Prepared by
Xiang Yi (Irene) Chen
Student ID: 20707344
User ID: xy29chen
1B Software Engineering

August 17, 2018

Xiang Yi (Irene) Chen
277 Lester Street
Waterloo, ON N2L 3W6

August 17, 2018

Dr. D. Rayside, Director
Software Engineering
University of Waterloo
Waterloo, ON N2L 3G1

Dear Dr. D. Rayside:

The following work term report, entitled "Application of Machine Learning Models as a Solution to Hyperbola Identification in Ground Penetrating Radar Data", has been prepared for Sensors & Software Inc. as my first work term report for my 1B term. The objective of this work term report is to explore the feasibility of different machine learning models to the problem of hyperbola recognition and classification, in ground penetrating radar data.

Sensors & Software Inc. is a major provider in ground penetrating radars, both software and hardware.

I would like to thank my supervisor, Adam Fazzari for providing support and guidance along the way, as well as my co-workers for furthering my understanding of ground penetrating radars, from usage to data interpretation. Finally I would like to thank Sensors & Software Inc. for providing for the data set and other resources needed to retrain the models.

I hereby confirm that I have received no further help, other than what is mentioned above, in writing this report. I also confirm that this report has not been previously submitted for academic credit at this or any other academic institution.

Sincerely,

Xiang Yi (Irene) Chen
Student ID: 20707344

Executive Summary

The following report investigates the suitability of a small sample of pre-trained machine learning models, in response to the challenge of identifying hyperbolas in ground penetrating radar (GPR). The problem stems from the fact that hyperbola artefacts in GPR data are highly contextual, and present themselves in many different forms. As such, streamlining the interpretation process with software aids for GPR users proves itself to be a challenge, due to this visual variety in the data.

However, the identification of hyperbola may be solved with the use of machine learning training. Hence, the Inception V3 model, as well as MobileNet V2 model, and VGG 19 model will be compared to determine their suitability given restrictions to this identification problem, and ultimately proven by a proof-of-concept application to hyperbola identification.

Disclaimer

The pre-trained models and their retraining modules are publicly available on GitHub. They've been applied with proprietary GPR data provided by Sensors & Software Inc.

Table of Contents

List of Figures

List of Tables

1 Introduction

Ground Penetrating Radar (GPR) solutions serve a wide range of uses, from underground utility locating, to pavement or ice thickness mapping, to forensics as well as archaeology. However, because of this varied usage—combined with pre-processing and post-processing—GPR data presents itself in many various ways. Hence, interpretation of GPR data is one of the most error-prone aspects of using GPR solutions.

1.1 Problem Statement

One of the biggest challenges of GPR data interpretation is to avoid false positives, such as extreme ringing caused by metal, as well as interference caused by rebounding air waves off the data collection site.

Sensors & Software Inc.'s software is able to apply velocity fitting and other transformations to GPR data to improve and simplify data interpretation, such as identifying false-positive air waves in GPR data through a calculation of its velocity. However, the first and most limiting step to this process is to allow the software to locate hyperbola—evidence of a disturbance in the soil caused by an object.

1.2 Proposal of Solution

Since machine learning has been making advances in the field of computer vision, it seems to be a fit tool to solve this bottleneck in this interpretation process.

Machine learning, in the broadest of terms, is a subset of computer science which allows programs to improve and ”learn” without explicitly being told what exactly to do—the construction of algorithms to make predictions based on recognition of patterns on a data set and to make changes to itself so that its future performance is improved.

Specifically, object detection and localization problems generally involve training feature extraction: associating certain characteristics of the image with different weights of importance based on their reoccurrence. In this problem, a hyperbola feature might consist of its curved nature. In this case, the presence—or lack thereof—of hyperbolas in GPR data will be explored with machine learning solutions, as well as their differentiation from other forms of features in GPR data, such as metal ringing and boundaries in the soil.

The follow section will be analysing the steps in a small sample of machine learning models, as well as the transferability and possibility of integration into existing GPR systems.

The comparison process will be taking the following criteria into account:

- Correctness of learning, based on the core transformations of the architecture of the model
- Ease of being transferred for real-time detection, based on the model's cost of memory and processing on ARM systems

For the sake of quickly creating a proof-of-concept model, transfer learning will be applied to an existing open-source pre-trained model. Transfer learning entails modifying layers of existing mature models to customize the model while retaining its generic base layers. Specifically, the following pre-trained models will be explored, chosen for their current popularity and abundant availability in terms of support and resources:

- Inception V3
- MobileNet V2
- VGG 19

2 Comparison & Analysis

For the comparison of potential machine learning models, the general idea is to apply supervised learning, giving labelled datasets to distinguish hyperbolas in the image, then test the model with sets of unlabelled data of various degrees of resemblance to the training dataset.

2.1 General Procedure

2.1.1 Retraining Neural Networks

In the most intuitive sense, neural networks is an imitation of how the human brain works. Input features are weighted positively or negatively, forming and reinforcing connections between common features similar to neurons forming connections between each other.

Each layer performs a transformation on a single vector input, and the learning process is achieved by chaining these layers together, either cyclically or linearly. Deep learning refers to models with a great number of layers of transformations. The penultimate layer—the bottleneck—summarizes the extracted features, to allow the last layer to perform the classification task.

TODO Weights used as indentification model

2.1.2 Convolutional Neural Networks (CNN)

Since for this hyperbola identification problem, GPR data can be easily converted to images, thus CNN will be used for their suitability with image inputs. A CNN is a subset of multi-layer neural networks, characterized by their convolution layer process–the union of integrals, or how much two functions overlap as one passes over the other–usually used for image classification. This overlap is the main idea behind how common features between images is reinforced and extracted. Processing is achieved by obtaining the union of the product of input images as matrices.

The common components of CNN model architecture as as follows: the convolutional layer, pooling layer, and fully-connected layer.

Pooling is a transformation specific to convolutional neural networks: it combines clusters into a single entity to be used in the next layer, to summarize and condense previous operations. This step assumes an image input and condenses the input into a 3-dimensional

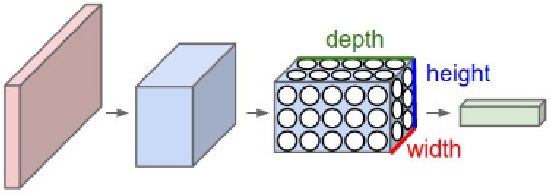

Figure 2-1: A neuron in a convolutional network condensing image inputs into a 3D entity. **ref:**

array, a convolutional neuron as seen in Figure 2-1. Max pooling is a downsampling function which takes the largest value from the prior cluster, effectively reducing the volume of data to process.

Concatenation step, ReLU layer TODO

During the fully connected layer, the softmax function normalizes the output to be between 0 and 1, as a sigmoid function, similar to a categorical probability distribution. In this case, it should be executed in the end to bring the model closer to indicating the probability and likelihood of hyperbolas.

2.2 Inception V3

Inception V3 is a CNN with five convolutional layers, max pooling, and softmax. However, it differs from the typical design of CNN models by it's successive stacking scattered throughout between the convolutional layers, hence recognized as a deep training network.

Moreover, inception extracts features at multiple levels, computing 1×1, 3×3, as well as 5×5 convolutional layers. These outputs are then concatenated before

As such, the precision of this model can reach TODO.

The trade-off of such deep training is in its training speed: TODO. Although pre-trained Inception model is used, it contains TODO more layers per iteration of image training compared to Mobilenet, and TODO more than VGG.

As for the result of the model, the weights for Inception V3 are small, coming in at 96MB.

2.3 MobileNet V2

2.4 VGG 19

VGG uses $3 \times 3 convolutionallayersstackedontopofeachotherinincreasingdepth.Reducingvolumesiz$
$connectedlayers, eachwith4,096nodesarethenfollowedbyasoftmaxclassifier(above)$.

”19” stand for the number of weight layers in the network.

The smaller networks converged and were then used as initializations for the larger, deeper networks âĂŤ this process is called pre-training.

It is painfully slow to train. 1. The network architecture weights themselves are quite large (in terms of disk/bandwidth). Due to its depth and number of fully-connected nodes, VGG is over 533MB for VGG16 and 574MB for VGG19. This makes deploying VGG a tiresome task. We still use VGG in many deep learning image classification problems; however, smaller network architectures are often more desirable (such as SqueezeNet, GoogLeNet, etc.).

Hence, the biggest drawbacks of VGG, despite its good depth and accuracy, would be hard to deploy onto Sensors and Software IncâĂŹs GPR systems, and would only be something usable in EKKO Project, a post-processing software product.

3 Conclusions

4 Recommendations