Hybrid Model for Interpretable Time Series Analysis

A THESIS

Presented to the Department of Computer Science and Computer Engineering

California State University, Long Beach

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Computer Science

Option in Computer Science

Committee Members:

By Ruben Rosales

May 2020

Abstract

Contents

Illustrations

Figures

Tables

Chapter 1

Introduction

[explain what black box and white box is] In this thesis, we study interpretable machine learning as applied to complex-valued time series. Scientists have studied the use of several machine learning methods such as Convolutional Neural Networks, Recurrent Neural Networks, and Support Vector Machines for time series classification.

[add more reasons] [add how most time series are classified] These methods, however, fall short of allowing users to visualize patterns within their dataset and ().

To address this issue, we propose an interpretable hybrid model that can be extended to any time series dataset. Our architecture is composed of two models, an ensemble method of classifiers that functions as a black box method, and a white-box model that encodes time series as images, attemps to classify them through a CNN, and outputs all images correctly classified in both black-box and white-box method.

In our black-box model, we analyze deep learning models such as CNN's which have become increasingly popular and widely used in the last decade but their complex nature makes it difficult for users to understand what is going on. So, as black-box models get more popular the need for an interpretable accessory? is needed so we propose a hybrid model consisting of a black box for classification and an interpretable white-box model which highlights characteristics of a time series for users to understand.

The need for interpretable models in management problems

The growth of mobile devices and demand for wireless data has created a need for igh quality spectrum sensing and adaptation to improve spectral allocation and interference mitigation is an important route by which we may achieve this. however have been constrained to relatively specialized solutions which lack the generality needed to deal with a complex and growing number emitter types, interference types and propagation environments. This is a significant challenge in the community as expert systems designed to perform well on specialized tasks often lack flexibility and can be expensive and tedious to develop analytically.

Contributions

[fill in]

Data

We analyzed our model on three radio signal datasets generated using Spatial Modulation. Spatial Modulation is a transmission technique that uses multiple antennas. It maps a block of information bits to two units, one is a symbol chosen from a constellation diagram, and the second one is a unique transmit antenna number that is chosen from a set of transmit antennas. The method in which transmit antennas send and receive radio waves can be described through polarization. Two popular basis polarizations are horizontal linear, H, and vertical linear, V. Our datasets contain data horizontally transmitted and vertically received (HV) and data vertically transmitted and vertically received.

The two properties HV and VV, are given to us in raw discrete format. Both properties are consist of time and amplitude values but vary in length and in how they are processed. The raw signal consists of over 31,000 data points with the exact time it was received. The size of the discrete data can vary from 22 to 44 data points, and is a processed representation of the raw signal that has equally spaced values.

[What is raw signal and discrete, compare them,] [explain hv, vv]

Chapter 2

Preprocessing

Radio signal data is presented in a complex-valued format that is unusable in a typical neural network, so in an attempt to make our model robust and reduce any complexity, we focused on using real values as features.

We tried numerous methods to extract real-valued features, including Fourier Transform, Short Time Fourier Transform, a custom sliding window method, and polar coordinates taken directly from the signals amplitude/phase value. Before applying any of these methods, we normalized all data using L2 normalization.

Transformations

STFT/ FT

[what is it]

[fft formula] The Fourier Transform (FT) is a mathematical tool that decomposes any function into a sum of sinusoidal basis functions. Each basis function is a complex value of a frequency, so it allows us to view our data in the frequency domain as opposed to amplitude.

One of the shortcomings of the FT is rooted in the Heisenberg Uncertainty Principle (HUP) [quote?]. The HUP states that the position and velocity of an object cannot be simultaneously measured, which can be applied to the time-frequency information of a signal. This means we cannot know which spectral components exist at any given time. The closest we can get is sampling at different ranges of time and finding a range of frequencies within that time frame. This method is described as the Short-Time Fourier Transform.

Shortcoming stft

[stft] The Short-Time Fourier transform (STFT) is a Fourier related transform used to determine the sinusoidal frequency and phase content of local sections of a signal as it changes over time.[Ref] Computing STF requires the signal to be divided into segments of equal length and then have the FT applied to that segment. This allows us to view the Fourier spectrum at a more granular level, which could potentially reveal different patterns amongst signals of different classes.

One of the issues with this method is that we can create very narrow window sizes, which gives us a better understanding of the data with respect to time, but we lose understanding of the frequency domain. Additionally, selecting an appropriate window size for segmenting the signal can be an arduous task that would require fine-tuning as well as increasing the dimensionality of our dataset.

Polar Coordinates

Transformation 2: Amplitude / Phase The representation of a complex number as a sum of a real and imaginary number, z = x + iy, is called its Cartesian representation. For every cartesian point, we calculate its radius and angular distance which are real values and use those as features.

[purpose and motivation? Behind making it ang/abs]

[definition plus method]

Transformation 2: Sliding window

Transformation 2: Sliding window We propose a method similar to STFT but to reduce the size of the data we take the mean at each window which would turn our size into size N where N is equal to the number of windows we use.

Wavelets

Wavelets are similar to the FT in which they deconstruct a signal using representations of other signals. The key difference is that wavelet signals are finite in time and frequency as opposed to sin and cos signals, which can carry on forever. This allows us to extract information from a signal with respect to time and location.

The use of wavelets is called the wavelet transform, which is a technique in which a signal is analyzed using different versions of a dilated and translated basis functions called the mother wavelet. There are two types of wavelet transformations, discrete and continuous. In this thesis, we focus on discrete. Discrete Wavelet Transform uses a discrete set of wavelet scales and translations which decompose the signal into a mutually orthogonal set of wavelets.

We take advantage of wavelets by applying discrete wavelet transforms as a filter-bank, which means it is composed of cascading high-pass and low-pass filters. This gives us the advantage of splitting a signal into several frequency sub-bands. Wavelet decompositions give us the advantage of gaining features that take into account frequency and time domains.

Data Shape

A neural network requires all data to be the same shape. However, because we downsample in wavelet decomposition, each data size is halved until making it impossible to place all levels of decomposition into the same dataset. In order to circumvent this, we tried two different methods, resampling the data and treating each level of decomposition as its own dataset.

Resampling

We use spline interpolation in order to resize each level of decomposition into a single array. We tested different sizes of interpolation from N to N/3, where N is the size of the largest discrete signal size and found no reduction in performance.

Each level

Figure X depicts our model in which each level of decomposition is treated as its own dataset. We found no difference in accuracy compared to the resampling method.

Chapter 3

Models

Definition

Machine learning algorithms can be seen as learning a target function (f) that maps input data (X) to an output (Y). There are several techniques to make this work, but we focus on nonparametric algorithms, namely Convolutional Neural Networks (CNN) and Long Short Term Memory networks (LSTM). Nonparametric algorithms are algorithms that attempt to make minimal assumptions about the form of the function so they can learn any form from data provided to it. An example would be a neural network because it has no prior knowledge about what it is classifying and attempts to generalize any new data points that it has not seen before.

CNN

A Convolutional Neural Network is a type of deep neural network that can be applied to different domains such as computer vision or time series analysis. It consists of an input and output layer as well as hidden layers. A convolution is an operation between a vector of weights w against an input x. It consists of taking the dot product between m and x were in steps of a filter size defined by n.

Convolutional neural networks perform feature learning via non-linear transformations implemented as a series of layers. The input data is a multidimensional array, called a tensor. The input data is passed through an input layer, followed by a series of hidden layers to extract features, and finally, an output layer, which in the case of classification, gives a probability for each class.

Hidden layers are crucial to neural networks because it is how a model can determine which data representations are useful for explaining the relationships in the given data. Each layer consists of several kernels that perform a convolution over the input; therefore, they referred to as convolutional layers. Kernels are feature detectors that convolve over the input and produce a transformed version of the data at the output.

LSTM

Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture[1] used in the field of deep learning. Unlike standard feedforward neural networks, LSTM has feedback connections. It can not only process single data points (such as images), but also entire sequences of data (such as speech or video). For example, LSTM is applicable to tasks such as unsegmented, connected handwriting recognition[2], speech recognition[3][4] and anomaly detection in network traffic or IDS's (intrusion detection systems).

A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The cell remembers values over arbitrary time intervals and the three gates regulate the flow of information into and out of the cell.

LSTM networks are well-suited to classifying, processing and making predictions based on time series data, since there can be lags of unknown duration between important events in a time series. LSTMs were developed to deal with the exploding and vanishing gradient problems that can be encountered when training traditional RNNs. Relative insensitivity to gap length is an advantage of LSTM over RNNs, hidden Markov models and other sequence learning methods in numerous applications.

multimodal

[multimodal] Due to the superior performance and computationally tractable representation capability (in vector spaces) in multiple domains such as visual, audio, and text, deep neural networks have gained tremendous popularity in multimodal learning tasks [38, 39, 44]. Typically, domain-specific neural networks are used on different modalities to generate their representations and the individual representations are merged or aggregated. Finally, the prediction is made on top of aggregated representation usually with another a neural network to capture the interactions between modalities and learn complex function mapping between input and output. Addition (or average) and concatenation are two common aggregation methods, i.e., The network structure is illustrated in Figure 1(b) . The arrows are function mappings or computing operations. The dotted boxes are representations of single and combined modality features. We call them additive combinations because their critical step is to add modality hidden vectors (although often in a nonlinear way).

[explain architectures]

Architectures Used

CNN

Our CNN architecture consists of 3 convolutional layers each followed by a batch normalization and dropout layer then finally connected to two dense layers (Figure X). We attempted to make it deeper but found inconsistent performance across all of our datasets. [add figure]

LSTM

We wanted to keep our LSTM network as small as possible, for X purposes, so we went with two LSTM layers of 128 hidden states and a dropout rate of 4/10 followed by a dense layer of 128 connections. We attempted different state sizes but found 128 to be the smallest number with the best consistent performance.

[add figure]

Final method

We propose a multimodal learning architecture for the purpose of time series classification which is illustrated in Figure (X). The multimodal architecture allows our data to vary in size and given that all four levels of decomposition have varying length due to downsampling, we propose each model to learn the representation of each distinct feature. For this architecture we propose two models, a 1D CNN and an LSTM, in which we employ two CNN's and 1 LSTM. They are all concatenated after their respective dense layer which performs the final We tried numerous variations and saw no impact on performance. [add figure]

mtf

We approach our MTF is slightly different than because instead of performing it over all samples in the dataset we perform it for every signal.

The three attributes we use are values are the absolute value of HV SBR, the absolute value of VV SBR, and a mathematical combination of the two (Figure X).

[add figure]

Notation

Layout

Chapter 4

Related Work

Chapter 5

Markov Transition Fields

We propose a framework similar to [] for encoding dynamical transition statistics, but we continue their work by considering ith order Markov transition probabilities. Given a time series X, we decompose its magnitude axis into two separate properties by representing it as a polar coordinate, Xangular and Xradial. We then identify the Q quantile bins for both properties and the temporal values Xtime and assign each x in Xangular, Xradial, Xtime to its corresponding bin qj (j in [1, Q]). Thus we construct three QxQ adjacency transition matrices, Wangular, Wradial, Wtime, by counting transitions among quantile bins in the manner of a ith order Markov chain along the time axis.

W does not take into account the temporal axis so to prevent any information loss we construct a Markov Transition Field (MTF) for each W. The MTF denotes the probability of transitioning from qi to qj for each x in X. This, in turn, allows us to consider the transition probability on the magnitude and temporal axis.

As described in [] the MTF encodes the multi-span transition probabilities of the time series, but given that we have three different M's we modify their approach and consider each M to be a separate color channel of RGB where Mangular is red, Mtime is blue, Mradial is green.

Chapter 6

Main Results

black box

Our black-box model is constructed of four different architectures, 1 LSTM and 2 CNN's, that are concatenated at the dense layer. The four features we use are amplitude/phase of each level of decomposition for HV and VV. We chose four because we wanted to take into consideration both HV / VV in the event that one had more descriptive features than the other. Figure X depicts our model.

To test our model we split our data into training and test sets by randomly selecting 70% of our data as test and left the other 30% as our test set. We then performed 3 fold cross validation and took the average of our results as our final outcome.

10k data

Given that this is much larger than our other data we split our data into 10% training and 90% test. We found no drop in performance nor in accuracy.

[add table for results]

white box

Our white box is composed of a model that takes in raw data as an input and converts each signal into an image through the MTF method. We then pass those images through a CNN and those that are correctly classified are shown to the user.

In order to classify those images we created a CNN for it specifically and avoided using our existing CNN in order to make it deeper and increase performance when images aren't that classifiable. We explored several architectures, such as, ResNet, Inception-v4, and AlexNet but we decided to create an architecture similar to AlexNet due to the variation and size of datasets we used we couldn't create a model deeper or more complex than it otherwise performance would drop because we didn't have enough data to properly train it.

[add results]

Chapter 7

Conclusion