Warsaw University of Technology

FACULTY OF
MATHEMATICS AND INFORMATION SCIENCE

Master's diploma thesis

in the field of study Computer Science
and specialisation Data Science

Transfer learning for time series classification

Paulina Pacyna

student record book number 290600

thesis supervisor
Agnieszka Jastrzębska, PhD

WARSAW 2023

Abstract

Transfer learning for time series classification

Loremmmm ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumyeirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diamvoluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumyeirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diamvoluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Keywords: keyword1, keyword2, ...

Streszczenie

Zastosowanie techniki transfer learning wzadaniuklasyfikacjiszeregów czasowych

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumyeirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diamvoluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumyeirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diamvoluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Słowa kluczowe: slowo1, slowo2, ...

Contents

Introduction

What is the thesis about? What is the content of it? What is the Author's contribution to it?

WARNING! In a diploma thesis which is a team project: Description of the work division in the team, including the scope of each co-author's contribution to the practical part (Team Programming Project) and the descriptive part of the diploma thesis.

1. Related works

In this chapter we would like to describe several algorithms used in time series classification. We will also recall theoretical definitions and distinctions used to describe transfer learning.

1.1. Time series classification

Time series is an ordered collection of observations indexed by time.

$$X = (x_t)_{t \in T} = (x_1, ..., x_T),\ \ x_t \in \mathbb{R}$$

The time index T can represent any collection with natural order. It can relate to point in time when the measurement was observed, or it can represent a point in space measured along X axis. We assume that indices are spaced evenly in the set T. The realisation or observation x_t in the times series is a numerical value describing the phenomena we observe, for example amplitude of a sound, stock price or y-coordinate. Time series classification is a problem of finding the optimal mapping between a set of time series and corresponding classes.

1.1.1. Dynamic Time Warping with k-Nearest Neighbour

The Dynamic Time Warping [1] with k-Nearest Neighbour classifier uses a distance based algorithm with a specific distance measure. A DWT distance between time series X^1, X^2 of equal lengths is:

$$DTW(X^1, X^2) = \min\{\sum_{i=1}^{S} dist(x^1_{e_i}, x^2_{f_i}) : (e_i)^S_{i=1}, (f_i)^S_{i=1} \in 2^T\}$$

subject to:

- $e_1 = 1, f_1 = 1, e_S = N, f_S = N$
- $|e_{i+1} - e_i| \leq 1,\ |f_{i+1} - f_i| \leq 1$

The measure defined above, used in k-Nearest Neighbour classifier is often used as a benchmark classifier.

1.1.2. Multi Layer Perceptron

The Multi Layer Perceptron (MLP) is the first artificial neural network architecture proposed in [2] and can be used for time series classification task. The MLP network can be formally defined as a composition of *layer* functions. The output of the function is a vector that usually models the probability distribution over the set of classes.

$$MLP(X; \theta_1, \ldots, \theta_M, \beta_1, \ldots, \beta_M) = L_M(\ldots L_2(L_1(X; \theta_1, \beta_1); \theta_2, \beta_2); \theta_M, \beta_M)$$

Each layer $L_i : \mathbb{R}^M \to \mathbb{R}^N$ is a function that depends on the parameters $\theta \in r^{M \times N}, \beta in \mathbb{R}^N$

$$L_i(X, ; \theta_i, \beta_i) = f_i(X\theta_i + \beta_i)$$

Function $f_i : \mathbb{R}^N \to \mathbb{R}^N$ is an arbitrary chosen non-linear function. The number of layers and dimensions of weights are also an arbitrary choice, except for the weights is first and last later. The weights in first and last layer have to match the dimensionality of input data (e.g. the length of time series) and number of classes. The output of the last layer is interpreted as a probability distribution over the set of classes.

The disadvantage of using Multi Layer Perceptrons for time series classification is that the input size is fixed. All time series is the training data must have the same length. In transfer learning, this means that if we want to reuse the source network (or a set of first layers from the network), the target dataset must consists of time series of the same length.

The MLP architecture fails at understanding the temporal dependencies [2]. Each input values in the time series is treated separately, because it is multiplied by a separate row in the weight matrix.

1.1.3. Convolutional Neural Networks

Convolutional Neural Networks are widely used in image recognition. A convolution applied for a time series can be interpreted as sliding a filter over the time series. A convolutional layer is a set of functions called convolutions or filters. The filter is applied at a given point, taking into account values that surrounds the point.

To define the convolution operation, let's assume the input is a matrix $X \in \mathbb{R}^{(N_1, \ldots, N_K)}$. In case of images, number of dimensions K is often equal to 3 (height, width, channels), for univariate time series we can assume just one dimension, and for multivariate time series wee need two dimensions - (feature, time). The filter consist of a matrix of weights $M \in \mathbb{R}^{(P_1, \ldots, P_K)}$. Usually, P_l are odd numbers, so that we can index the matrix with symmetrical numbers: $(\frac{-P_l+1}{2}, \frac{-P_l+3}{2}, \ldots, 0, \ldots, \frac{P_l-1}{2})$. The 0 index marks the center of the matrix.

Finally the convolution $*$ is defined as follows:

$$(X * M)_{i_1,\dots,i_K} = \sum_{l_1=\frac{-P_1+1}{2}}^{\frac{P_1-1}{2}} \cdots \sum_{l_K=\frac{-P_K+1}{2}}^{\frac{P_K-1}{2}} M_{l_1,\dots,l_K} X_{i_1+l_1,\dots,i_K+l_K}$$

The result of the convolution is passed elementwise to a nonlinear function. The nonlinear function together with the convolution operation will be called a filter.

In case of univariate time series the first layer of convolutional neural network is one-dimensional. The output of the first layer has dimensions (length of time series - the length of the filter - 1, number of filters). Below we define the value of the output for filter i

$$y_{t,i} = f_i([\theta^i_{\frac{-M+1}{2}}, \dots, \theta^i_{\frac{M-1}{2}}] \cdot [X_{t+\frac{-M+1}{2}}, \dots, X_{t+\frac{M-1}{2}}]),$$

where $\cdot$ is a dot product The weights θ^i are different for each filter. The same filter is applied over the whole length of time series. This is called *weight sharing* and it enables the patterns regardless of the position in the time series.

The architecture of the convolutional layer is not dependent of the size of the input data. Regardless the size of input data, number of filters and size of filters remain the same, only the output sizes depends on the input size. Therefore, if the convolutional layer is succeeded by layers with the same property, like other convolutional layers or Global Pooling with Dense Layer (see section 1.1.4), the whole network may be invariant to the input sizes [2]. Such networks may be interesting in terms of transfer learning, as the sizes of time series in the source task and in the target task do not have to match.

1.1.4. Fully Connected Networks

Fully Convolutional Networks are convolutional network used in time series classification [2]. FCN networks consists of blocks of convolutional layers proceeded by global pooling over the time axis and a dense layer. Because the architecture convolutional layers does not depend on the size of input data and the convolutional layer are followed by pooling over the time axis, the whole networks is capable of processing data of variable lengths.

1.2. Transfer learning

Transfer learning is a technique that attempts to apply knowledge learned while solving one task to enhance the learning process for another task. Formally, the problem can be described using the notions of tasks and domains [4, 5]. A *Domain* is a pair $\mathcal{D} = (\mathcal{X}, P(\mathcal{X}))$, where $\mathcal{X}$

is the feature space (e.g. the time series observations, and $P(\mathcal{X})$ is the probability distribution over the feature space. A *Task* is a pair of label space $\mathcal{Y}$ and the decision function f, $\mathcal{T} = (\mathcal{Y}, f)$. The decision function f is learned from $\mathcal{X}, P(\mathcal{X}), \mathcal{Y}$ in the learning process.

Transfer learning attempts to utilize knowledge domain/domains and task/tasks. Formally, given $S \in \mathbb{N}$ source domains and source tasks $(\{(\mathcal{D}_i^S, \mathcal{T}_i^S : ß = 1, \ldots, S\})$ and $T \in \mathbb{N}$ target domains and target tasks $(\{(\mathcal{D}_i^T, \mathcal{T}_i^T : ß = 1, \ldots, S\})$ transfer learning utilizes knowledge learned from source domains and tasks to improve the learning process of decision functions in target tasks $\mathcal{T}_i^T$

1.2.1. Types of transfer learning

1.2.2. Characteristics of a good source domain

In the field of image processing, it is very common to use convolutional neural networks pre-trained with the ImageNet dataset [3]. ImageNet is a large dataset of human-annotated images. It contains 1 milion labeled images of 1000 classes. The label space consists of fine grained classes such as breeds of dogs and cats, but also coarse-grained classes like *red wine* and *traffic light*. As transfer learning based on this dataset became more popular and successful, a question arisen: Which features of this dataset makes it so good for this task?.

A study conducted in [3] attempts to answer this question. The first hypothesis is that the volume of the dataset is relevant to train accurate, general classfiers. The authors compared models pretrained on the original dataset and models based on sampled subsets (reduced 2, 4 8 and 20 times).The results shown that the more training examples, the better results. The accuracy of the initial classifier occured to be more dependent on the size of dataset than the accuracy of classifiers fine-tuned from the former classifier.

Next experiments answer considerations on the label space. The authors examine if the granularity of the label space is essential for the problem. To compare the results, the label space is clustered and 127 classes are derived from the initial 1000 classes. Pre-training with the reduced label space has a minimal negative impact on accuracy of classifiers fine-tuned from this classifier. This suggest that such a fine division may not be needed.

Finally, the last question is if we train the classfier on the reduced label space with 127 classes, will it be able to distinguish between the fine-grained classes. To examine that, the authors extracted features from the first layers of the networks trained on recuded label space. Then, the authors performed classfication with 1-NN and 5-NN models on the extracted feature space, but with 1000 classes. The fidings are that the k-NN classifier performs 15% worse on reduced dataset vs normal dataset. This shows that CNNs are capable of implicitly learning representative features.

Bibliography

[1] Anthony Bagnall, Aaron Bostrom, James Large, and Jason Lines. *The Great Time Series Classification Bake Off: An Experimental Evaluation of Recently Proposed Algorithms. Extended Version.* arXiv, 2016.

[2] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, 33(4):917–963, mar 2019.

[3] Minyoung Huh, Pulkit Agrawal, and Alexei A. Efros. *What makes ImageNet good for transfer learning?* arXiv, 2016.

[4] Karl Weiss, Taghi M. Khoshgoftaar, and DingDing Wang. *A survey of transfer learning.* Journal of Big Data, 2016.

[5] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. *A Comprehensive Survey on Transfer Learning.* arXiv, 2019.