RGB GAF image: A possible solution to one weak point of Gramian Angular Field Imaging

Why this time series imagining method is not perfect and a suggestion to fix the issue

Shuyang Xiang · Follow
Published in Towards Data Science · 5 min read · Mar 11, 2022

19 1

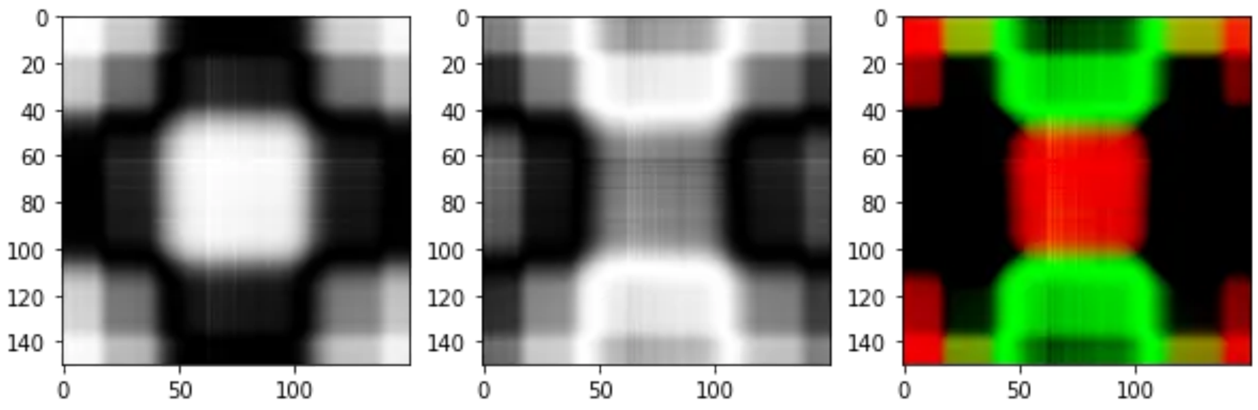


Image by author: left and middle are simple GAF images, right is the corresponding RGB GAF image

Data scientists, me included, have come across a huge amount of time series at work. Despite the success in natural language processing, speech recognition and computer vision, techniques developed from deep learning unfortunately fail to have many rivalling developments for time series. Time series imaging, a concept that leverages the techniques and the insights brought by the recent developments of computer vision, has therefore caught attention because it allows machines to "visually recognize time series.

Gramian Angular Field (GAF) Imaging turns out one of the most popular time series imaging algorithms. First proposed by team of faculty and students from the Department of Mathematics and Computer Science of the University of Cagliari in Italy, the approach transforms time-series into images and uses hence CNN to identify visual patterns for prediction.

However, when utilising GAF, I encountered one possible weak point, which motivates me to write this article. In the following, you will be reading:

1. A brief introduction of GAF and how the method might have a little concern.
2. My suggestion as a possible solution to this issue.

An overview of GAF

Gramian Angular Field (GAF) represents time series in a polar coordinate system instead of the typical Cartesian coordinates.

Suppose that we have a rescaled time series ranging in [-1,1] with the form (x_1, x_2, …x_n). After transforming it into the polar coordinate system, i.e. θ_i=arccos(x_i), we can easily exploit the angular perspective by considering the trigonometric sum/difference between each point to identify the temporal correlation within different time intervals: In the n*n GAF matrix, the i-j element is defined as either the cosine or the sine of the summation of the i-th and j-th angle. The figure above shows a time series in cartesian coordinated transformed in a GAF image with the intermediate polar coordinate system step. The implementation can now be easily realized by the corresponding functionality in pyts, a python package combining multiple methods for time series classification.

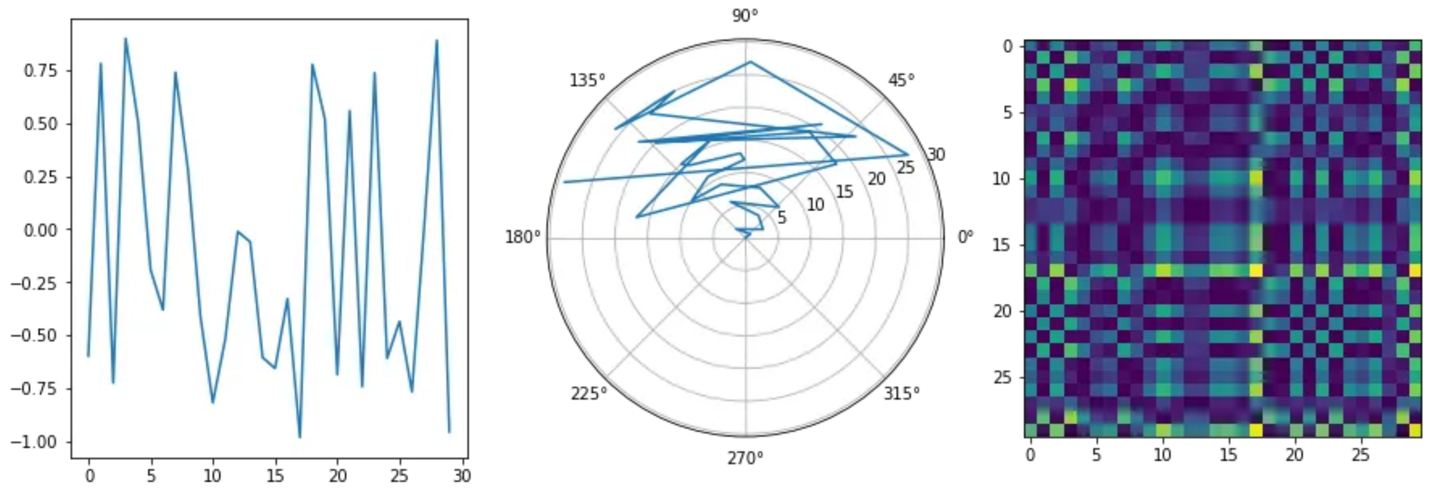


Image by author: from time series to GAF

Converting the i-j element of the matrix back to the Cartesian coordinates: cos(θ_i+θ_j)=cos(θ_i)cos(θ_j)+sin(θ_i)sin(θ_j)=x_i x_j+√(1-x_i^2)√(1-x_j^2) and we can immediately see the following properties of the definition:

1. If and only if x_i=x_j, the element equals -1.
2. If and only if x_i=x_j=0, the element equals 1.
3. If x_i=-x_j while the absolute value of both is close to -1. This means that the value of the element cannot capture the sign of the correlation.

A quick remark here is that I am not 100% agree with the name of "Gramian" Angular Field because the above definition does not give a REAL Gramian matrix whose entries are given by the inner product of the given vector according to the theory of linear algebra.

A weak point

Let us take a normalized time series (x_1, x_2, …x_n) and use the GAF method to imagine it. Despite the presence of both sine and cosine functions in the initial work, now the majority of examples I reviewed use only the cosine function to get the GAF image. As is said in the last section, the i-j element of such a matrix is obtained by cos(θ_i+θ_j) with both θ_i, θ_j varying in [0,π) where θ_i=arccos(x_i) and θ_j=arccos(x_j). However, I want to point out that such a transformation is not an injection.
Note that for any θ in [0,2π), we have also (2π-θ) in [0,2π). Recall that the cosine function in [0,2π) is symmetric with respect to θ=2π so that cos(θ) equals cos(2π-θ). Back to the GAF matrix, we see that cos(2π-θ_i-θ_j)=cos(θ_i+θ_j) and therefore π-θ_i and π-θ_j give us the same value as the i-j element of the GAF matrix. For all θ in [0,π), we know that π-θ=arccos(-x) and we can therefore get the fact that the time series (-x_1,-x_2, …-x_n) can

give us the same matrix. To resume, if we reverse the sign of every point on a time series, the transformation of GAF results in the same image.
The figure below shows the same image obtained by two time series with point-wise opposite signs.

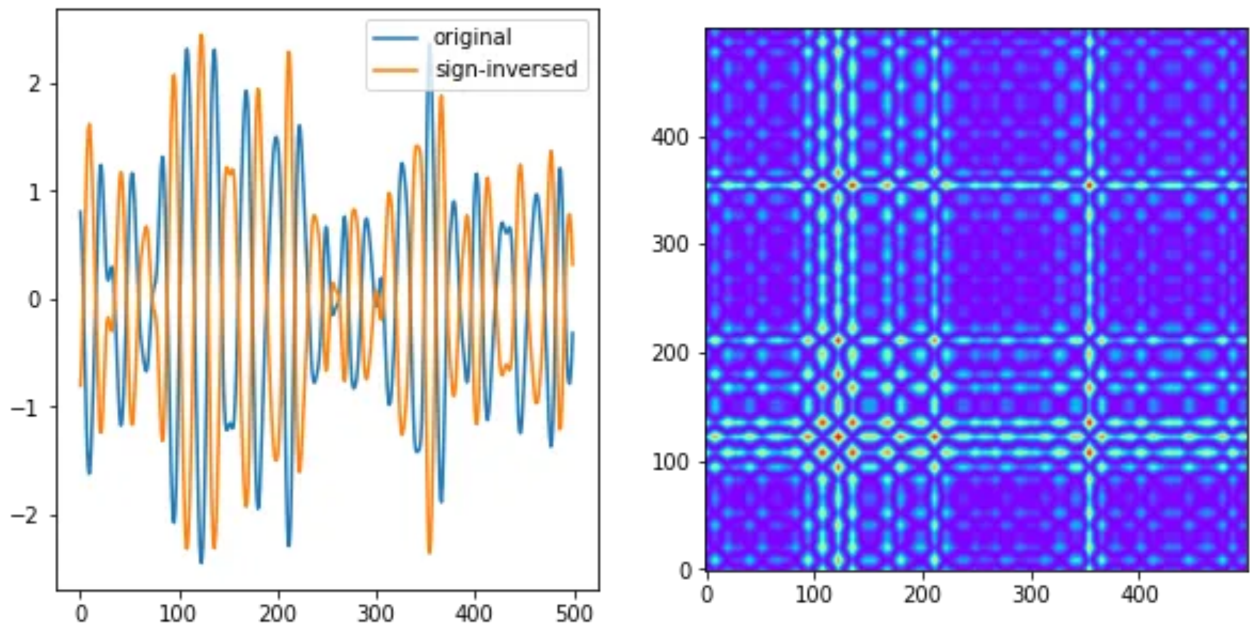


Image by author: the same GAF image transformed from two time series

Taking financial forecasting as an example, the ignorance of signs might be a concern because the absolute value of the performance of a financial product is not enough to describe the temporal pattern. Imagine that a stock price is increasing during a period and we will get a decreasing time series if we reverse the sign. Even though the two transformed GAF images are equivalent, the two time series correspond to complete opposite performances.

Fortunately, the issue is very easy to solve because there are indeed two types of GAF images. A single cosine has the problem of uniqueness but a pair of cosine and sine will remove the issue because the two are enough to determine the angular value.
Instead of encoding time series into a gray scaled image, I would like to create an RGB image from the time series. Consider the following code that creates such a GAF image:

```
from pyts.image import GramianAngularField
import numpy as np

gasf = GramianAngularField(method='summation')
x_train_gasf = gasf.transform(x_train)
gadf = GramianAngularField(method='difference')
x_train_gadf = gadf.transform(x_train)
x_train_gaf=np.concatenate((x_train_gasf,x_train_gadf,np.zeros(x_tra
in_gadf.shape)),axis=-1)
```

The below shows an RGB image created from the same time series as the former section and you can see that we take into account both sine and cosine values in the same image.

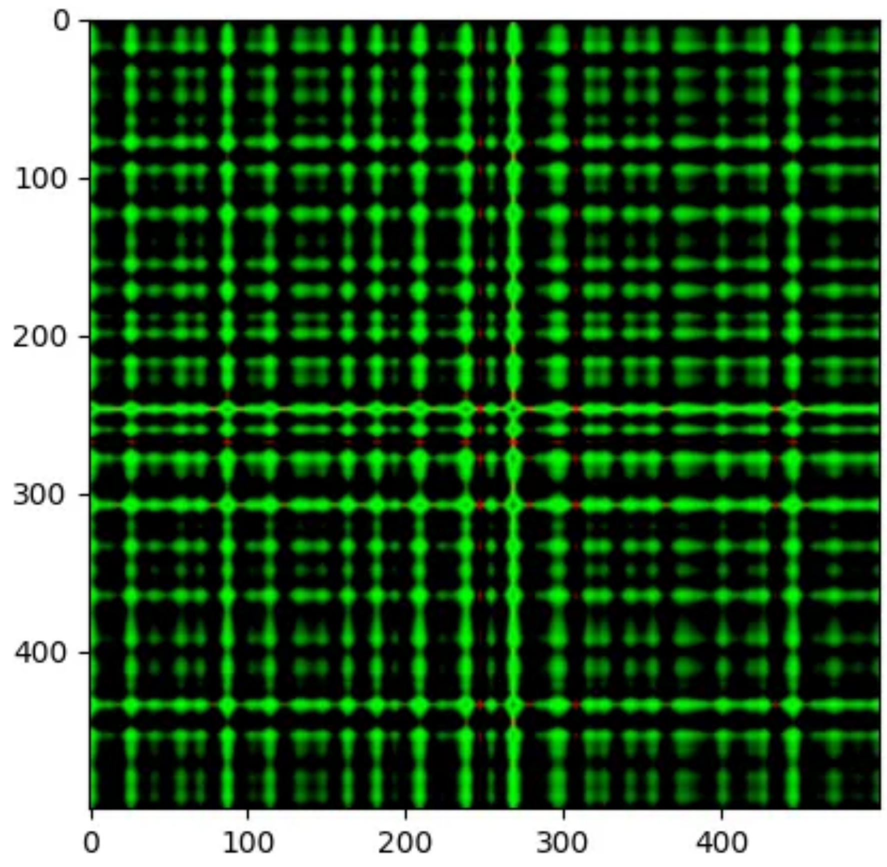


Image by author: RGB GAF image

Conclusion

In this article, I review briefly the GAF method for time series imagining and point out that there might be a weak point of this method. Then I suggest using the RGB GAF image encoding both the sine and cosine function to solve the non-uniqueness issue of the method.

Time Series Analysis Image Processing

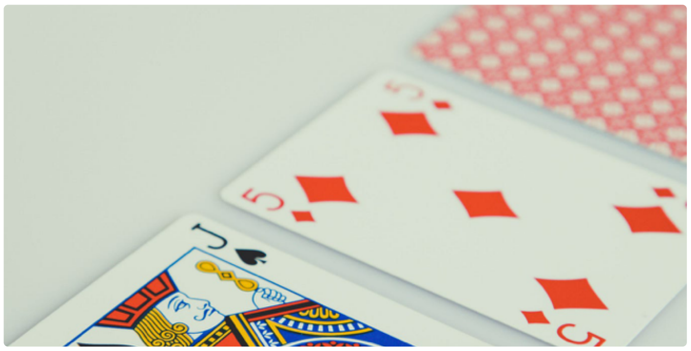
Written by Shuyang Xiang

Follow

217 Followers · Writer for Towards Data Science

I am a ph.D in maths, now working as a data scientist.

More from Shuyang Xiang and Towards Data Science



Shuyang Xiang in Towards Data Science

Aleatoric and Epistemic

Miriam Santos in Towards Data Science

Pandas 2.0: A Game-Changer for

Uncertainty in Deep Learning

How they differ from each other and how to deal with them with TensorFlow Probability

4 min read · Jul 19, 2022

23

Data Scientists?

The Top 5 Features for Efficient Data Manipulation

7 min read · Jun 27

1.8K 19

Dominik Polzer in Towards Data Science

All You Need to Know to Build Your First LLM App

A step-by-step tutorial to document loaders, embeddings, vector stores and prompt...

· 26 min read · Jun 22

2.6K 22

Shuyang Xiang in Towards Data Science

How to make clustering explainable

In this article, I will explain how to use SHAP values to have a better understanding of...

5 min read · Aug 31, 2021

151 7

See all from Shuyang Xiang

See all from Towards Data Science

Recommended from Medium

Shawhin Talebi in Towards Data Science

The Wavelet Transform

An Introduction and Example

· 6 min read · Dec 21, 2020

471 4

Nikos Kafritsas in Towards Data Science

Temporal Fusion Transformer: Time Series Forecasting with De...

Create accurate & interpretable predictions

· 12 min read · Nov 5, 2022

1.6K 20

Lists

Predictive Modeling w/ Python

18 stories · 138 saves

Practical Guides to Machine Learning

10 stories · 152 saves

ChatGPT prompts

22 stories · 125 saves

New_Reading_List

174 stories · 27 saves

Andy McDonald in Towards Data Science

Creating Scientific Plots the Easy Way With scienceplots and…

Instantly Transform Your Matplotlib Figures With a Few Lines of Python Code

· 9 min read · 4 days ago

165 2

Marco Peixeiro in Towards Data Science

PatchTST: A Breakthrough in Time Series Forecasting

From theory to practice, understand the PatchTST algorithm and apply it in Python…

· 10 min read · Jun 20

778 9

Dominik Polzer in Towards Data Science

All You Need to Know to Build Your First LLM App

A step-by-step tutorial to document loaders, embeddings, vector stores and prompt…

Thomas Smith in The Generator

Google Bard's New Visual Feature is a Game Changer

Chatbots can officially see the world

· 26 min read · Jun 22

2.6K 22

· 6 min read · Jul 14

5.9K 118

See more recommendations