

# Forecasting with Time Series Imaging



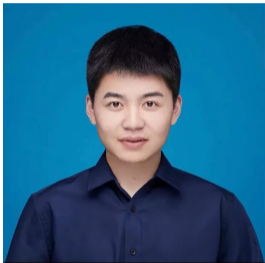
光华管理学院

Guanghua School of Management

**Feng Li**

[feng.li@gsm.pku.edu.cn](mailto:feng.li@gsm.pku.edu.cn)

<http://feng.li/>



Xixi Li (Tsinghua University, PostDoc)

<https://lixixibj.github.io/>



Yanfei Kang (Beihang University)

<https://yanfei.site/>

Xixi Li, Yanfei Kang and Feng Li (2020). Forecasting with Time Series Imaging. Expert Systems with Applications, Vol. 160pp. 113680.

Python package available at <https://github.com/feng-li/forecasting-with-time-series-imaging>



- 1 Time series features
- 2 Encoding time series to images
- 3 Image-based time series feature extraction
- 4 Forecast-model averaging



Transforming a given time series  $\{x_1, x_2, \dots, x_n\}$  to a feature vector  $F = (F_1, F_2, \dots, F_p)'$  (Kang et al., 2017)

A feature  $F_k$  can be any kind of function computed from a time series:

- 1 A simple mean
  - 2 The parameter of a fitted model
  - 3 Some statistic intended to highlight an attribute of the data
  - 4 ...
- with a nice R package: `tsfeatures` (Hyndman et al., 2023)



- Depends on both the **nature** of the time series being analysed, and the **purpose** of the analysis.
  - With unit roots, the mean is not a meaningful feature without some constraints on the initial values.
  - CPU usage every minute for large number of servers: we observe a daily seasonality. The mean may provide useful comparative information despite the time series not being stationary.
- **Bad News:**
  - There does not exist the best feature representation of a time series ([Fulcher, 2018](#)).
  - The manual features are typically **global**.



## Encoding time series to images

- Let  $R(i, j)$  be the element of the **time series image matrix** where  $i$  indexes time on the x-axis of the recurrence plot and  $j$  indexes time on the y-axis;
- **Recurrence Plot Encoding**

$$R(i, j) = \begin{cases} S, & \text{if } \|\vec{x}(i) - \vec{x}(j)\| / \epsilon > S, \\ \|\vec{x}(i) - \vec{x}(j)\| / \epsilon, & \text{otherwise,} \end{cases}$$

where  $S$  is the threshold distance and  $\epsilon$  is some small number.

- **Gramian Angular Field Encoding (Wang & Oates, 2015)**: Given a time series  $X = x_1, x_2, \dots, x_n$ , we scale the series  $X$  into  $[-1, 1]$ .

$$x_i = \frac{(x_i - \max(X)) + (x_i - \min(X))}{\max(X) - \min(X)}$$

Then, we convert the scaled time series  $X$  into “polar coordinates”.

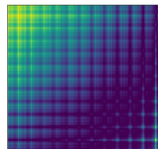
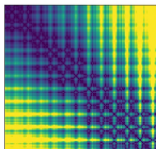
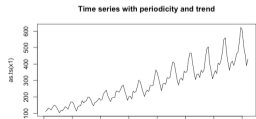
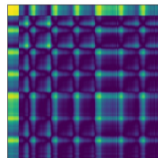
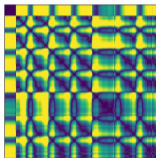
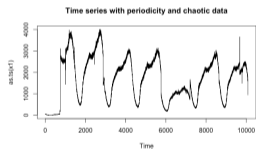
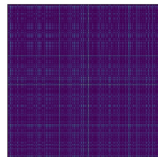
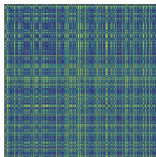
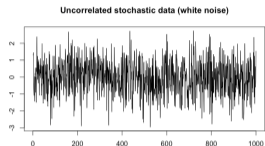
- We could even use a statistic for time series as the encoder (e.g. ACF, PACF).



# Why time series imaging?









- The original Bag of Features (BoF) model, which extracts features from one-dimensional signal segments, has achieved a great success in time series classification ([Baydogan et al., 2013](#); [Wang et al., 2013](#)).
- [Hatami et al. \(2017\)](#) transform time-series into two-dimensional recurrence images with recurrence plot ([Eckmann et al., 1987](#)) and then applies the BoF model.
- ([Razavian et al., 2014](#)) use the features acquired by the convolutional neural network as the input of the classifier, which significantly improves the accuracy of image classification.



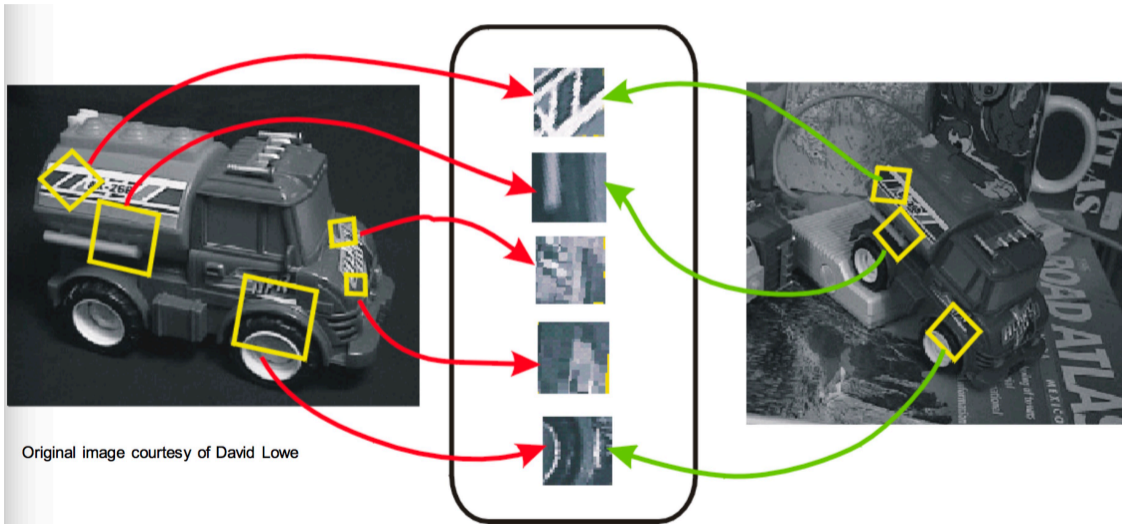
## Image-based time series feature extraction

### ↳ Scale-invariant feature transform (SIFT)

- The scale space of an image is defined as the original image is convoluted with a variable-scale 2-dimensional Gaussian function.
- Key points are then taken as maxima/minima of the difference of Gaussians that occur at multiple scales.
- In our study, we use a 128-elements vector to characterize the key descriptors.
- Firstly, we establish an 8-direction histogram in each  $4 \times 4$  sub-region, and a total of 16 sub-regions in the  $16 \times 16$  region around the key point are calculated. Then we calculate the magnitude and direction of each pixel's gradient magnitude and add to the sub-region.
- In the end, a total of 128-dimensional image data based on histograms are generated.
- The sift method calculates the distribution characteristics of feature points in the whole image, and then generates a global histogram, so the spatial distribution information of the image is lost, and the image may not be accurately identified.
- A spatial pyramid method statistically distributes image feature points at different resolutions to obtain spatial information of images.

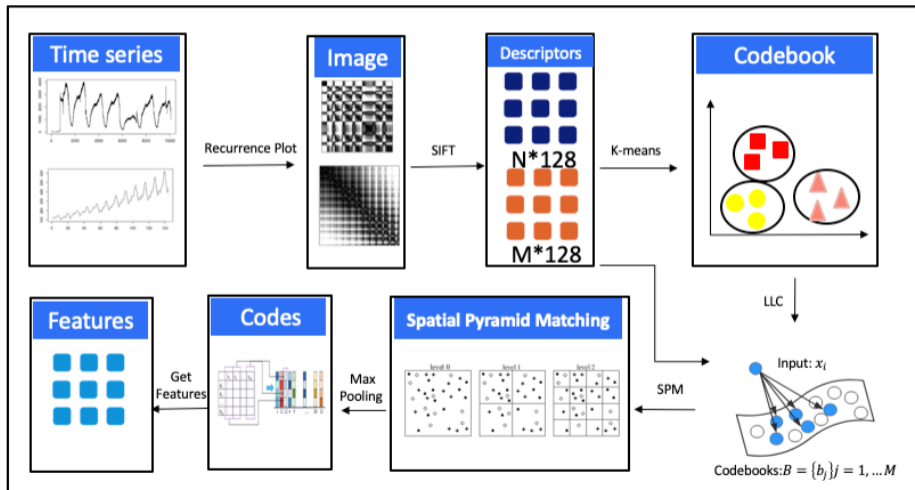
# Image-based time series feature extraction

↳ Scale-invariant feature transform (SIFT)



# Image-based time series feature extraction

## ↪ Scale-invariant feature transform (SIFT)





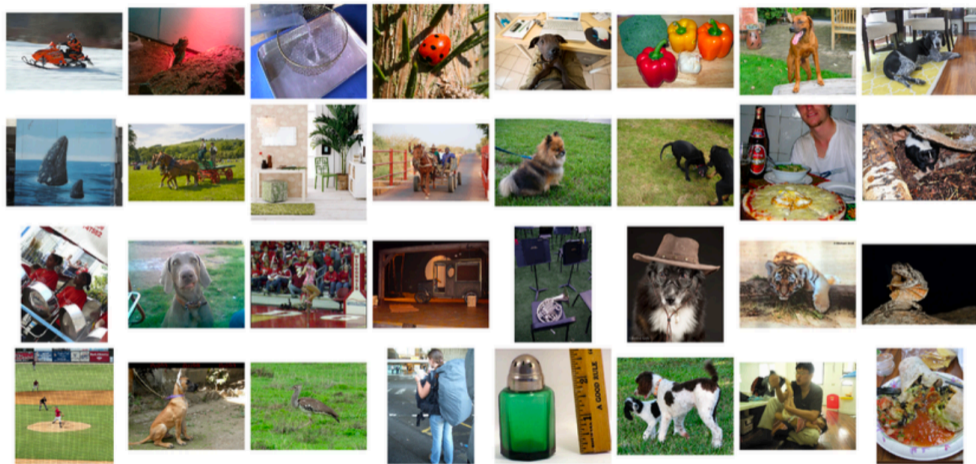
# Image-based time series feature extraction I

## ↳ Transfer learning with fine-tuning

- The deep convolutional neural networks has greater advantages in accuracy compared with the traditional image features.
- But building models from scratch is complex and time consuming.
- One could used a pretrained trained neural network model and make adjustments to her own task – **Transfer learning**.
- The pretrained model is based on **ImageNet** competition ([Deng et al., 2009](#)).

# Image-based time series feature extraction II

↳ Transfer learning with fine-tuning





# Image-based time series feature extraction III

## ↳ Transfer learning with fine-tuning

- The deep neural network models used for transfer learning
  - ResNet(Szegedy et al., 2016a), Deep Residual Learning for Image Recognition
  - Inception(Szegedy et al., 2016b), Evolved From GoogLeNet, merged with ResNet idea for image classification
  - VGG(Simonyan & Zisserman, 2014), Very Deep Convolutional Networks for Large-Scale Image Recognition
- With the pre trained model, we fix the parameters of the previous layers, and fine-tune the next few layers for our task. In this way, the speed of network training will be greatly accelerated.



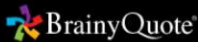


# Why Transfer Learning from ImageNet?



**If I have seen further than others,  
it is by standing upon the  
shoulders of giants.**

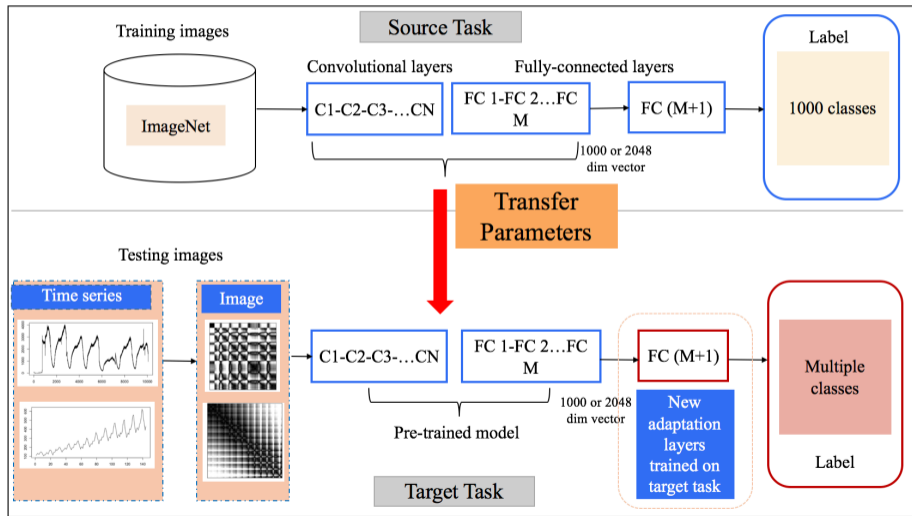
Isaac Newton



- Test top-5 errors in the ImageNet Competition: from 16.42% (2012) to 2.25% (2017)

# Image-based time series feature extraction

## ↪ Transfer learning with fine-tuning





Forecasting method	Description
ARIMA	The autoregressive integrated moving average model automatically estimated in the forecast package for <b>R</b> (Hyndman & Khandakar, 2008).
ETS	The exponential smoothing state space model (Hyndman et al., 2002).
NNET-AR	A feed-forward neural network using autoregressive inputs.
TBATS	The exponential smoothing state space model with a Box-Cox transformation, ARMA errors, trend and seasonal components.
STLM-AR	The STL decomposition (Cleveland et al., 1990) with AR modeling of the seasonally adjusted series.
RW-DRIFT	The random walk model with drift.
THETA	The decomposition forecasting model by modifying the local curvature of the time-series through a coefficient 'Theta' that is applied directly to the second differences of the data (Assimakopoulos & Nikolopoulos, 2000).
NAIVE	The naïve method, which takes the last observation as the forecasts of all the forecast horizons.
SNAIVE	The seasonal naïve method, which forecasts using the most recent values of the same season.



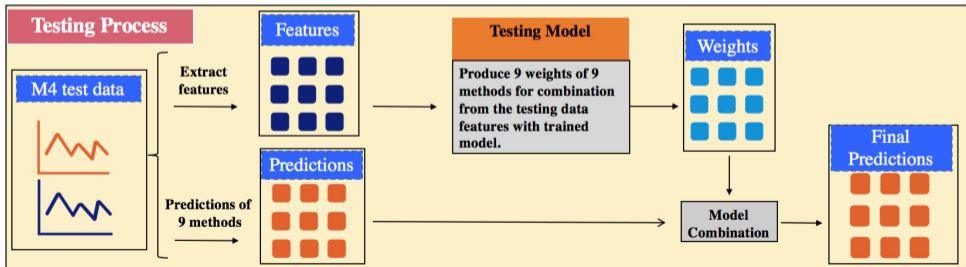
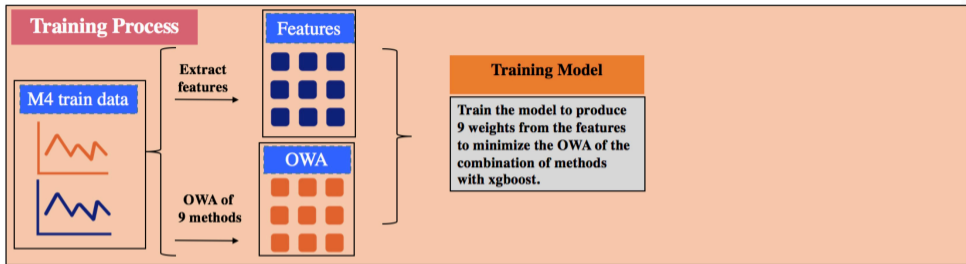
- **Forecast loss measurement:** Overall Weighted Average (OWA) is an indicator of two accuracy measures: the Mean Absolute Scaled Error (MASE) and the symmetric Mean Absolute Percentage (sMAPE).

$$\text{sMAPE} = \frac{1}{h} \sum_{t=1}^h \frac{2 | Y_t - \hat{Y}_t |}{| Y_t | + | \hat{Y}_t |},$$

$$\text{MASE} = \frac{1}{h} \frac{\sum_{t=1}^h | Y_t - \hat{Y}_t |}{\frac{1}{n-m} \sum_{t=m+1}^n | Y_t - Y_{t-m} |},$$

$$\text{OWA} = \frac{\text{sMAPE}/\text{sMPAE} + \text{MASE}/\text{MASE}}{2},$$

- Train a high dimensional regression model (Lasso) with  $X_{train}$  and  $MASE$
- Calculate predicted  $MASE$  with  $X_{test}$ .





- In order to get the weight  $w(f_n)_m$  for every method, softmax transform is carried on the output  $p(f_n)_m$ .

$$w(f_n)_m = \frac{e^{p(f_n)_m}}{\sum_m e^{p(f_n)_m}}$$

- The weighted average loss function is minimized:

$$\operatorname{argmin}_w \bar{L}_n = \sum_{m=1}^M w(f_n)_m O_{nm}$$



# Forecast-model averaging

## ↪ OWA performance against the M4 competition results

	Yearly	Quarterly	Monthly	Weekly	Daily	Hourly	Total
Ranking	M4 competition						
1	0.778	0.847	0.836	0.851	1.046	0.440	0.821
2	0.799	0.847	0.858	0.796	1.019	0.484	0.838
3	0.820	0.855	0.867	0.766	0.806	0.444	0.841
4	0.813	0.859	0.854	0.795	0.996	0.474	0.842
5	0.802	0.855	0.868	0.897	0.977	0.674	0.843
6	0.806	0.853	0.876	0.751	0.984	0.663	0.848
7	0.801	0.908	0.882	0.957	1.060	0.653	0.860
8	0.788	0.898	0.905	0.968	0.996	1.012	0.861
9	0.836	0.878	0.881	0.782	1.002	0.410	0.865
10	0.824	0.883	0.899	0.939	0.990	0.485	0.869
Method	Forecasting with time series imaging						
<b>SIFT</b>	0.820	0.858	0.863	0.839	1.009	0.498	0.848
<b>CNN</b>							
<i>Inception-v1+XGBoost</i>	0.814	0.867	0.885	0.895	1.002	0.552	0.854
<i>ResNet-v1-101+XGBoost</i>	0.816	0.872	0.877	0.916	1.025	0.542	0.855
<i>ResNet-v1-50+XGBoost</i>	0.816	0.869	0.873	0.881	1.025	0.538	0.853
<i>VGG-19+XGBoost</i>	0.814	0.863	0.876	0.877	0.994	0.549	0.850





## Forecast-model averaging

### ↪ Results compared to the Tourism competition

Forecasting method	MAPE				MASE			
	Yearly	Quarterly	Monthly	Total	Yearly	Quarterly	Monthly	Total
ARIMA	30.639	16.172	21.746	23.444	3.197	1.595	1.495	2.200
ETS	25.065	15.316	20.965	20.745	3.000	1.592	1.526	2.130
THETA	23.409	15.927	22.390	20.688	2.730	1.661	1.649	2.080
SNAIVE	23.610	16.459	22.562	20.988	3.007	1.699	1.631	2.197
DAMPED	27.975	35.830	47.192	35.898	3.061	3.221	3.404	3.209
Forecasting with time series imaging								
<b>SIFT</b>	24.164	<b>15.236</b>	<b>19.984</b>	<b>20.089</b>	2.760	<b>1.570</b>	<b>1.444</b>	<b>2.005</b>
<b>CNN</b>								
<i>Inception-v1+XGBoost</i>	24.633	15.333	<b>20.261</b>	<b>20.383</b>	2.834	<b>1.560</b>	<b>1.467</b>	<b>2.037</b>
<i>ResNet-v1-101+XGBoost</i>	24.288	<b>15.047</b>	<b>20.221</b>	<b>20.142</b>	2.779	<b>1.555</b>	<b>1.468</b>	<b>2.014</b>
<i>ResNet-v1-50+XGBoost</i>	24.347	<b>15.101</b>	<b>19.981</b>	<b>20.117</b>	2.750	<b>1.563</b>	<b>1.454</b>	<b>2.002</b>
<i>VGG-19+XGBoost</i>	23.616	15.599	<b>20.055</b>	<b>20.010</b>	<b>2.689</b>	1.638	<b>1.476</b>	<b>2.008</b>












# Thank you!

`feng.li@gsm.pku.edu.cn`

My web: <http://feng.li/>

Our lab: <http://kllab.org/>



-  Hyndman, R. J., Kang, Y., Montero-Manso, P., Talagala, T. S., Wang, E., Yang, Y. & O'Hara-Wild, M. (2023). *Tsfeatures: Time Series Feature Extraction*. Version 1.1.
-  Fulcher, B. D. (2018). "Feature-Based Time-Series Analysis". In: *Feature Engineering for Machine Learning and Data Analytics*. CRC Press, pp. 87–116.
-  Hatami, N., Gavet, Y. & Debayle, J. (2017). "Bag of Recurrence Patterns Representation for Time-Series Classification". *Pattern Analysis and Applications*, pp. 1–11.
-  Kang, Y., Hyndman, R. J. & Smith-Miles, K. (2017). "Visualising Forecasting Algorithm Performance Using Time Series Instance Spaces". *International Journal of Forecasting* 33.(2), pp. 345–358.
-  Szegedy, C., Ioffe, S. & Vanhoucke, V. (2016a). "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning".
-  Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. & Wojna, Z. (2016b). "Rethinking the Inception Architecture for Computer Vision". In: *Computer Vision and Pattern Recognition*.
-  Wang, Z. & Oates, T. (2015). "Imaging Time-Series to Improve Classification and Imputation". In: *Proceedings of the 24th International Conference on Artificial Intelligence*. AAAI Press. AAAI Press, pp. 3939–3945.
-  Razavian, A. S., Azizpour, H., Sullivan, J. & Carlsson, S. (2014). "CNN Features Off-the-Shelf: An Astounding Baseline for Recognition".
-  Simonyan, K. & Zisserman, A. (2014). "Very Deep Convolutional Networks for Large-Scale Image Recognition". *Computer Science*.



-  Baydogan, M. G., Runger, G. & Tuv, E. (2013). “A Bag-of-Features Framework to Classify Time Series”. *IEEE transactions on pattern analysis and machine intelligence* 35.(11), pp. 2796–2802.
-  Wang, J., Liu, P., She, M. F., Nahavandi, S. & Kouzani, A. (2013). “Bag-of-Words Representation for Biomedical Time Series Classification”. *Biomedical Signal Processing and Control* 8.(6), pp. 634–644.
-  Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. & Fei-Fei, L. (2009). “ImageNet: A Large-Scale Hierarchical Image Database”. *IEEE conference on computer vision and pattern recognition* 9, p. 8.
-  Hyndman, R. J. & Khandakar, Y. (2008). “Automatic Time Series Forecasting: The Forecast Package for R”. *Journal of Statistical Software* 27.(3), pp. 1–22.
-  Hyndman, R. J., Koehler, A. B., Snyder, R. D. & Grose, S. (2002). “A State Space Framework for Automatic Forecasting Using Exponential Smoothing Methods”. *International Journal of Forecasting* 18.(3), pp. 439–454.
-  Assimakopoulos, V. & Nikolopoulos, K. (2000). “The Theta Model: A Decomposition Approach to Forecasting”. *International Journal of Forecasting* 16.(4), pp. 521–530.
-  Cleveland, R. B., Cleveland, W. S., McRae, J. E. & Terpenning, I. (1990). “STL: A Seasonal-Trend Decomposition Procedure Based on Loess”. *Journal of Official Statistics* 6.(1), pp. 3–73.
-  Eckmann, J.-P., Kamphorst, S. O. & Ruelle, D. (1987). “Recurrence Plots of Dynamical Systems”. *EPL (Europhysics Letters)* 4.(9), p. 973.