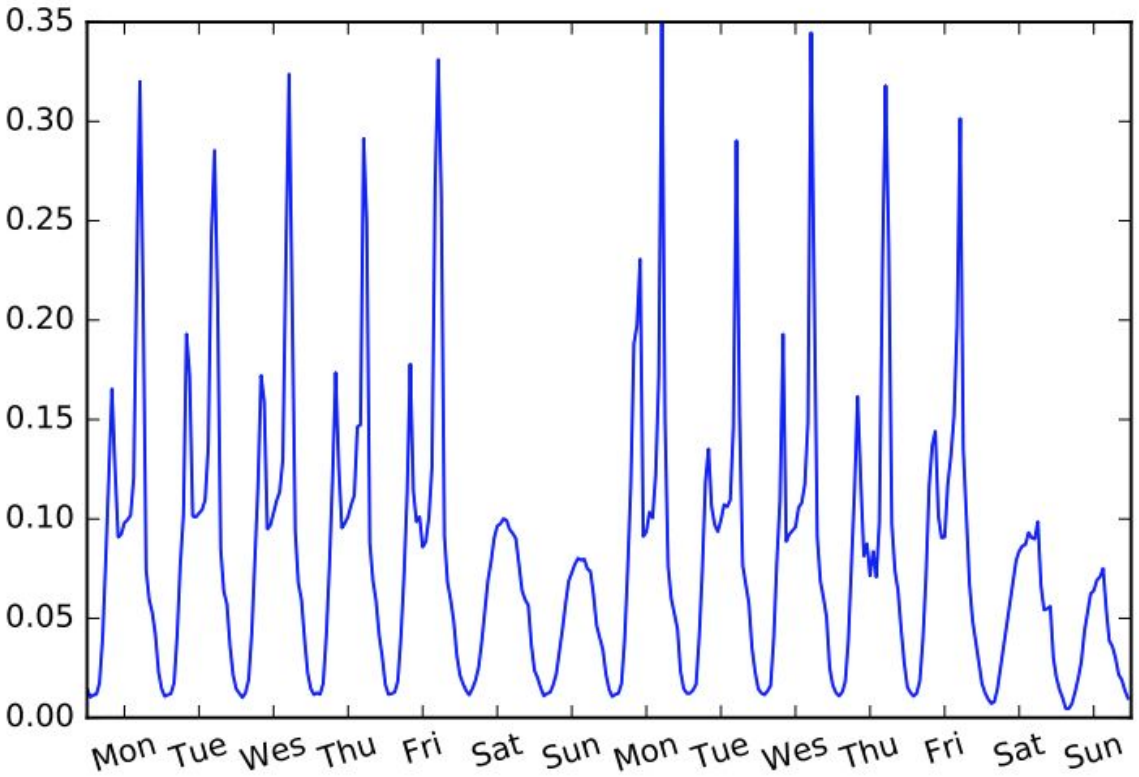
#ml-papers June 2019

Modeling Long- and Short-Term Temporal Patterns with Deep Neural Networks

1 - Intro



one of several factors that makes **time series analysis** tricky is **seasonality**

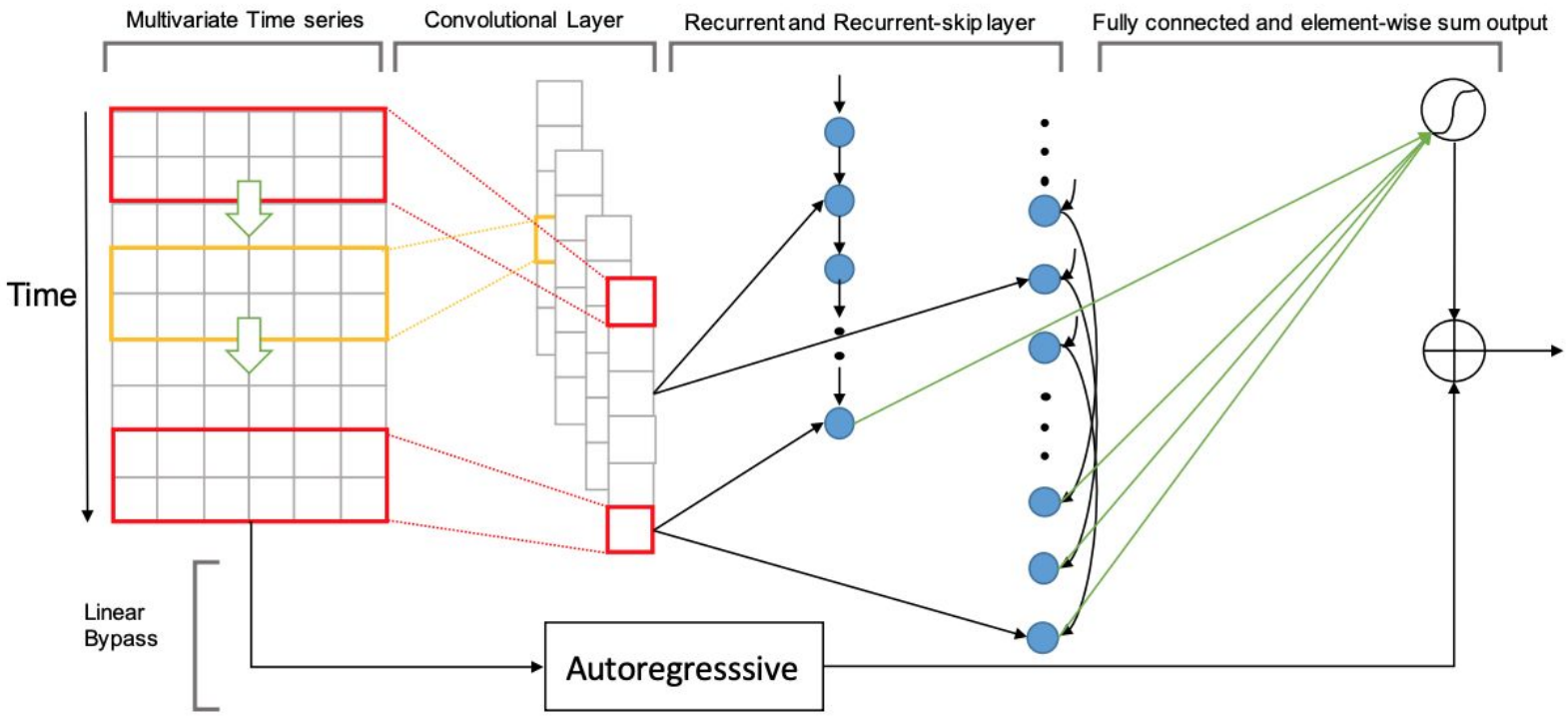
the plot at left displays daily percent usage for a particular highway over a two week period

as you can see, the plot exhibits seasonality at **multiple superimposed frequencies** (daily & weekly)

shallow approaches for treating seasonality include ARIMA or VAR models (traditional time series analysis), frequency-space models (signal processing), state space or structural models (you already know about these!), etc

these authors suggest a novel **deep** architecture they've used to generate performance improvements on data with multiple seasonal effects

1 - Intro



model components

1) **CNN** → high-frequency seasonality
2) **RNN+** → low-frequency seasonality
   - note the "recurrent-skip" layer special sauce; this is novel
3) **AR** → non-seasonal variance
   - parallel linear model makes predictions more robust

2 - Related Background

- ARIMA
  - PROS:
    - adaptive to various exponential smoothing techniques
    - flexible to subsume other ts models (AR, MA, ARMA)
  - CONS:
    - high computation cost (hence rarely used in high dimensional multivariate ts)
- VAR (vector autoregression)
  - PROS:
    - AR extension in multivariate setting
  - CONS:
    - overfitting for long-term temporal patterns
    - fix: dimension reduction + regularization

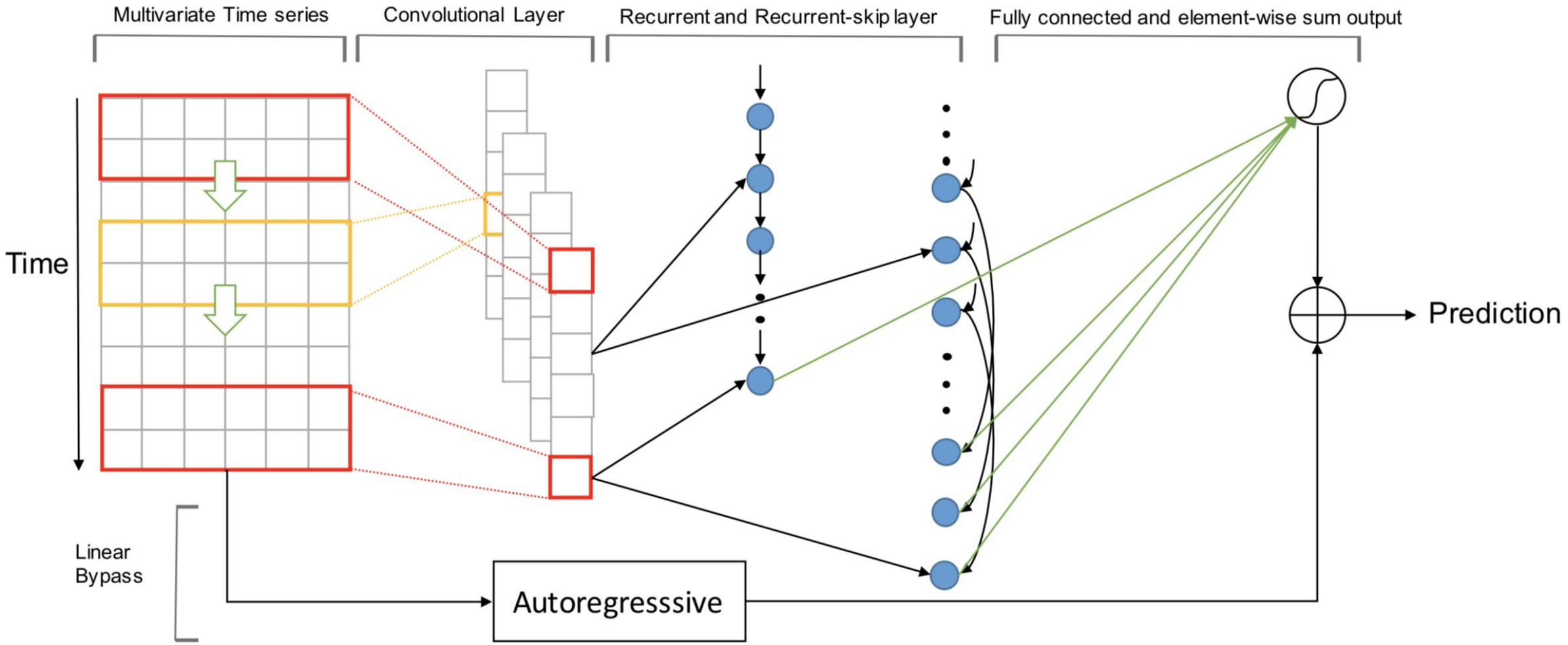
2 - Related Background

- Regularization Examples
  - SVR:
    - max margin hyperplane w. restriction of prediction errors
  - LASSO:
    - encourage sparsity

2 - Related Background

- GP (Gaussian Process)
  - PROS:
    - non-parametric method: capable of complex dynamical phenomena
    - can be used as a prior in Bayesian inference
  - CONS:
    - high computation complexity (O(n^3) due to matrix inversion)

3 - Framework


Multivariate Time series
Convolutional Layer
Recurrent and Recurrent-skip layer
Fully connected and element-wise sum output
Time
Prediction
Linear
Bypass
Autoregresssive

Convolutional Component

We formulate the input matrix at time stamp as $X_T = \{y_1, y_2, ..., y_T\} \in R^{n \times T}$.

The first layer of LSTNet is a convolutional network without pooling, which aims to extract short-term patterns in the time dimension as well as local dependencies between variables. The k-th filter sweeps through the input matrix X and produces

$$h_k = \text{RELU}(W_k * X + b_k) \qquad (1)$$

$$\text{RELU}(x) = \max(0, x)$$

Recurrent Component

$$
\begin{aligned}
r_t &= \sigma(x_t W_{xr} + h_{t-1} W_{hr} + b_r) \\
u_t &= \sigma(x_t W_{xu} + h_{t-1} W_{hu} + b_u) \\
c_t &= RELU(x_t W_{xc} + r_t \odot (h_{t-1} W_{hc}) + b_c) \\
h_t &= (1 - u_t) \odot h_{t-1} + u_t \odot c_t
\end{aligned}
\tag{2}
$$

Recurrent-skip Component

$$
\begin{aligned}
r_t &= \sigma(x_t W_{xr} + h_{t-p} W_{hr} + b_r) \\
u_t &= \sigma(x_t W_{xu} + h_{t-p} W_{hu} + b_u) \\
c_t &= RELU(x_t W_{xc} + r_t \odot (h_{t-p} W_{hc}) + b_c) \\
h_t &= (1 - u_t) \odot h_{t-p} + u_t \odot c_t
\end{aligned}
\tag{3}
$$

$$
h_t^D = W^R h_t^R + \sum_{i=0}^{p-1} W_i^S h_{t-i}^S + b \tag{4}
$$

Temporal Attention Layer

If more complex long term dependencies are desired, an optional temporary attention layer can be used to learn what distances in the past are most relevant to the current prediction.

Learns which part to pay attention too. In this case, it is only temporal attention not spatial attention.

Autoregressive Component

The AR model is formulated as follows,

$$h_{t,i}^{L} = \sum_{k=0}^{q^{ar}-1} W_k^{ar} \boldsymbol{y}_{t-k,i} + b^{ar} \tag{5}$$

Link of notes of the 3 part of the paper:
https://docs.google.com/document/d/1aKrnJyRzu459cEbB_m1m3dw6SqdJB0ECVupdmfDj5CU/edit

4 - Evaluation

9 methods were used to conduct the experiment on 4 datasets

- **LSTNet-skip** is our proposed LSTNet model with skip-RNN layer.
- **LSTNet-Attn** is our proposed LSTNet model with temporal attention layer

training : one model for each of the n output variables

Metrics

- Root Relative Squared Error (RSE):

$$RSE = \frac{\sqrt{\sum_{(i,t)\in\Omega_{Test}} (Y_{it} - \hat{Y}_{it})^2}}{\sqrt{\sum_{(i,t)\in\Omega_{Test}} (Y_{it} - mean(Y))^2}} \quad (10)$$

- Empirical Correlation Coefficient (CORR)

$$CORR = \frac{1}{n}\sum_{i=1}^{n} \frac{\sum_t \left(Y_{it} - mean(Y_i)\right)\left(\hat{Y}_{it} - mean(\hat{Y}_i)\right)}{\sqrt{\sum_t \left(Y_{it} - mean(Y_i)\right)^2 \left(\hat{Y}_{it} - mean(\hat{Y}_i)\right)^2}} \quad (11)$$

where $Y, \hat{Y} \in \mathbb{R}^{n\times T}$ are ground true signals and system prediction signals, respectively. The RSE are the scaled version of the widely used Root Mean Square Error(RMSE), which is design to make more readable evaluation, regardless the data scale. For RSE lower value is better, while for CORR higher value is better.

4 - Evaluation

1. **Traffic** : 48 months hourly which describes the the road occupancy rates (between 0 and 1)
2. **Solar-Energy** : sampled every 10 minutes from 137 PV plants
3. **Electricity** : recorded every 15 minutes from 2012 to 2014, for n = 321 clients (converted the data to hourly consumption)
4. **Exchange-rate** : collection of the daily exchange rates of eight foreign

training set (60%)
validation set (20%)
test set (20%)

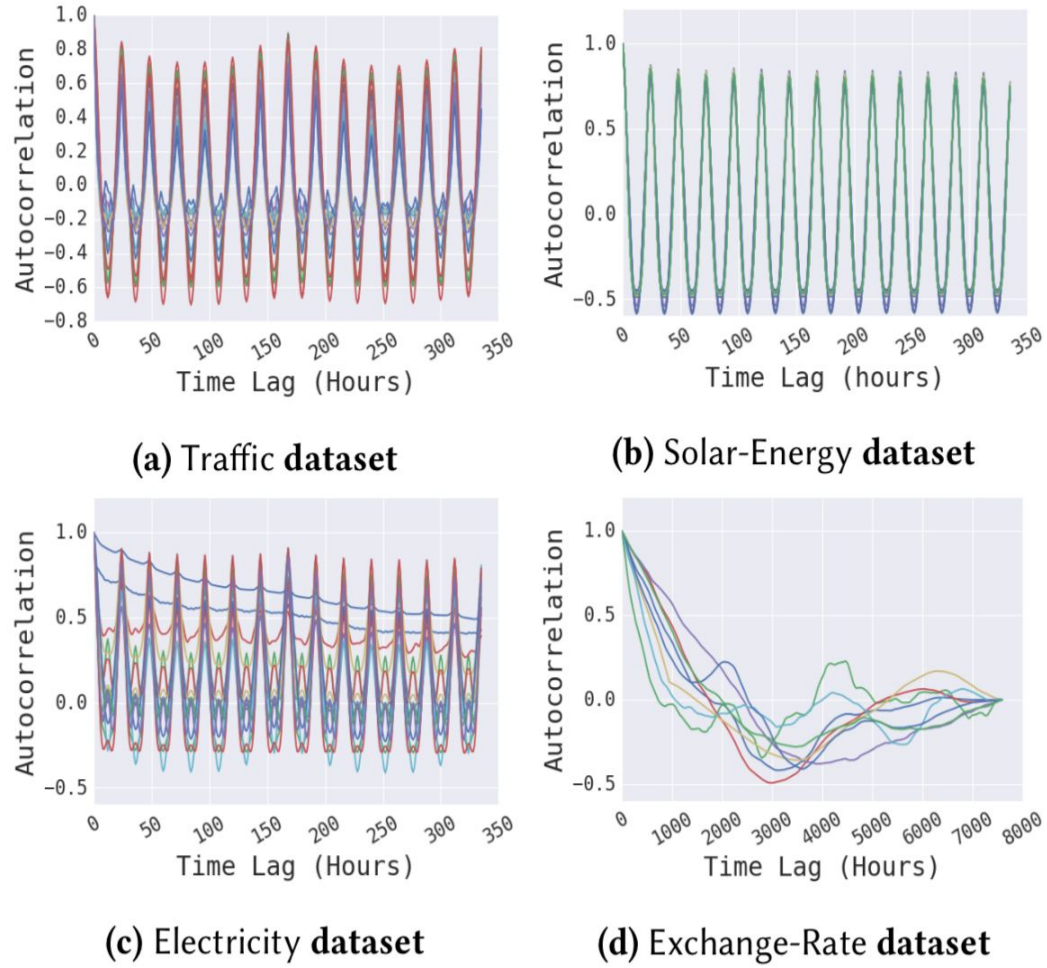
4 - Evaluation

Autocorrelation graph :

- to examine the existence of long-term and/or short- term repetitive patterns in time series data
- correlation of a signal with a delayed copy of itself as a function of delay

Observations

1. **Traffic** : short-term daily pattern (in every 24 hours)
2. **Electricity** : long-term weekly pattern (in every 7 days)
3. **Exchange-Rate** : hardly see any repetitive long-term patterns, expect some short-term local continuity



(a) Traffic **dataset** (b) Solar-Energy **dataset**

(c) Electricity **dataset** (d) Exchange-Rate **dataset**

4.4 Experimental deets

Dataset		Solar-Energy				Traffic				Electricity				Exchange-Rate			
		Horizon				Horizon				Horizon				Horizon			
Methods	Metrics	3	6	12	24	3	6	12	24	3	6	12	24	3	6	12	24
AR	RSE	0.2435	0.3790	0.5911	0.8699	0.5991	0.6218	0.6252	0.6293	0.0995	0.1035	0.1050	0.1054	0.0228	0.0279	**0.0353**	**0.0445**
(3)	CORR	0.9710	0.9263	0.8107	0.5314	0.7752	0.7568	0.7544	0.7519	0.8845	0.8632	0.8591	0.8595	0.9734	0.9656	0.9526	**0.9357**
LRidge	RSE	0.2019	0.2954	0.4832	0.7287	0.5833	0.5920	0.6148	0.6025	0.1467	0.1419	0.2129	0.1280	**0.0184**	0.0274	0.0419	0.0675
(3)	CORR	0.9807	0.9568	0.8765	0.6803	0.8038	0.8051	0.7879	0.7862	0.8890	0.8594	0.8003	0.8806	**0.9788**	**0.9722**	0.9543	0.9305
LSVR	RSE	0.2021	0.2999	0.4846	0.7300	0.5740	0.6580	0.7714	0.5909	0.1523	0.1372	0.1333	0.1180	0.0189	0.0284	0.0425	0.0662
(1)	CORR	0.9807	0.9562	0.8764	0.6789	0.7993	0.7267	0.6711	0.7850	0.8888	0.8861	0.8961	0.8891	0.9782	0.9697	**0.9546**	0.9370
TRMF	RSE	0.2473	0.3470	0.5597	0.9005	0.6708	0.6261	0.5956	0.6442	0.1802	0.2039	0.2186	0.3656	0.0351	0.0875	0.0494	0.0563
(0)	CORR	0.9703	0.9418	0.8475	0.5598	0.6964	0.7430	0.7748	0.7278	0.8538	0.8424	0.8304	0.7471	0.9142	0.8123	0.8993	0.8678
GP	RSE	0.2259	0.3286	0.5200	0.7973	0.6082	0.6772	0.6406	0.5995	0.1500	0.1907	0.1621	0.1273	0.0239	**0.0272**	0.0394	0.0580
(1)	CORR	0.9751	0.9448	0.8518	0.5971	0.7831	0.7406	0.7671	0.7909	0.8670	0.8334	0.8394	0.8818	0.8713	0.8193	0.8484	0.8278
VARMLP	RSE	0.1922	0.2679	0.4244	0.6841	0.5582	0.6579	0.6023	0.6146	0.1393	0.1620	0.1557	0.1274	0.0265	0.0304	0.0407	0.0578
(0)	CORR	0.9829	0.9655	0.9058	0.7149	0.8245	0.7695	0.7929	0.7891	0.8708	0.8389	0.8192	0.8679	0.8609	0.8725	0.8280	0.7675
RNN-GRU	RSE	0.1932	0.2628	0.4163	0.4852	0.5358	0.5522	0.5562	0.5633	0.1102	0.1144	0.1183	0.1295	0.0192	0.0264	0.0408	0.0626
(0)	CORR	0.9823	0.9675	0.9150	0.8823	0.8511	0.8405	0.8345	0.8300	0.8597	0.8623	0.8472	0.8651	0.9786	0.9712	0.9531	0.9223
LST-Skip	RSE	0.1843	0.2559	**0.3254**	0.4643	**0.4777**	**0.4893**	**0.4950**	**0.4973**	**0.0864**	**0.0931**	0.1007	**0.1007**	0.0226	0.0280	0.0356	0.0449
(17)	CORR	0.9843	0.9690	**0.9467**	0.8870	**0.8721**	**0.8690**	**0.8614**	**0.8588**	**0.9283**	**0.9135**	**0.9077**	**0.9119**	0.9735	0.9658	0.9511	0.9354
LST-Attn	RSE	**0.1816**	**0.2538**	0.3466	**0.4403**	0.4897	0.4973	0.5173	0.5300	0.0868	0.0953	**0.0984**	0.1059	0.0276	0.0321	0.0448	0.0590
(7)	CORR	**0.9848**	**0.9696**	0.9397	**0.8995**	0.8704	0.8669	0.8540	0.8429	0.9243	0.9095	0.9030	0.9025	0.9717	0.9656	0.9499	0.9339

bold => best result for that column, LST-S has 17, LST-A has 7, rest 0-3

They did a grid search over all HPPs and tested all models on all data sets with different **horizons** (3, 6, 12, 34)

LST-Skip outperformed on data-sets with strong temporal patterns (everything except E-R)

LST-attn came in 2nd -> authors suggest it as a good alternative if there's no strong **p** (temporal effect period) values

Both LSTN models performed **better on experiments with long time horizons**

Vanilla AR outperforms on Exchange-Rate dataset, because there is little seasonal effect

4.6 Ablation Study

- **LSTw/oskip**: The LSTNet models without the Recurrent-skip component and attention component.
- **LSTw/oCNN**: The LSTNet-skip models without the Convolutional component.
- **LSTw/oAR**: The LSTNet-skip models without the AR component.

Authors conduct an ablation study to demonstrate the efficiency of the framework.. ie no component of the model is redundant. Note: they try to keep # of trainable parameters equal across sub-models.

LSTw/oskip has the best performance

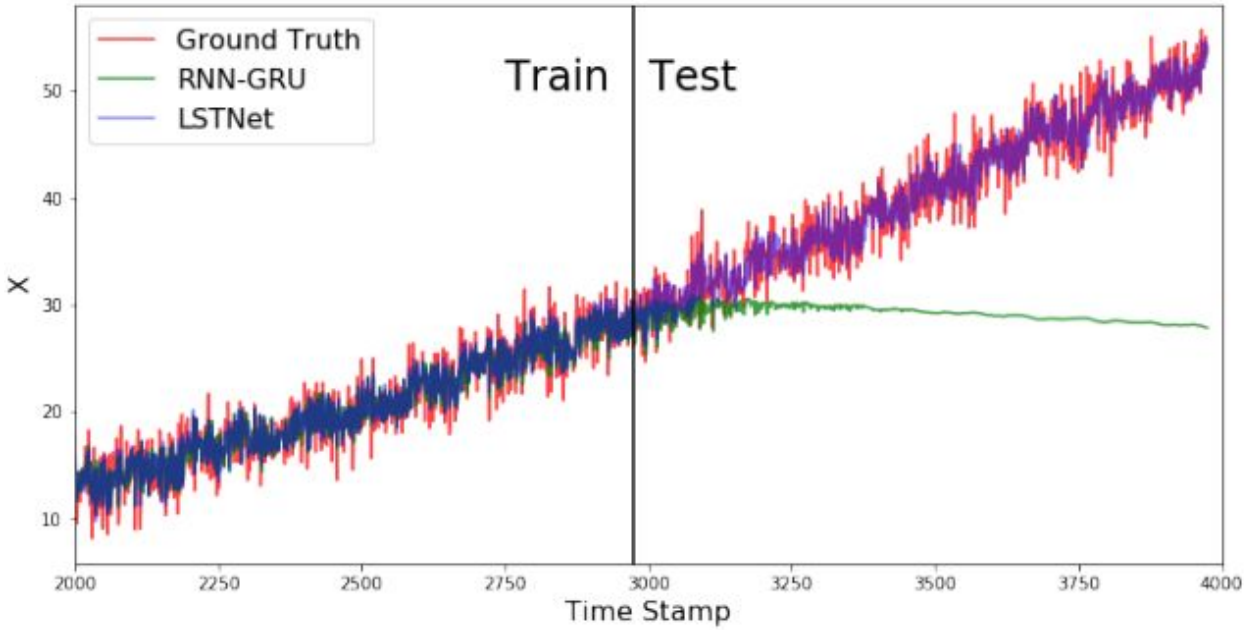
LSTw/oAR has the worst

LSTw/oskip && **LSTw/oCNN** have high performance variability depending on the dataset used

Takeaway: AR is necessary b/c it is robust to scale changing in data..

4.6 Ablation Study

- **LSTw/oskip**: The LSTNet models without the Recurrent-skip component and attention component.
- **LSTw/oCNN**: The LSTNet-skip models without the Convolutional component.
- **LSTw/oAR**: The LSTNet-skip models without the AR component.



Authors conduct an ablation study to demonstrate the efficiency of the framework.. ie no component of the model is redundant. Note: they try to keep # of trainable parameters equal across sub-models.

LSTw/oskip has the best performance

LSTw/oAR has the worst

LSTw/oskip && **LSTw/oCNN** have high performance variability depending on the dataset used

Takeaway: AR is necessary b/c it is robust to scale changing in data.. They prove this by fitting RNN-GRU and LSTN models to simulated data that is growing in magnitude over time

4.7 Long & Short term patterns

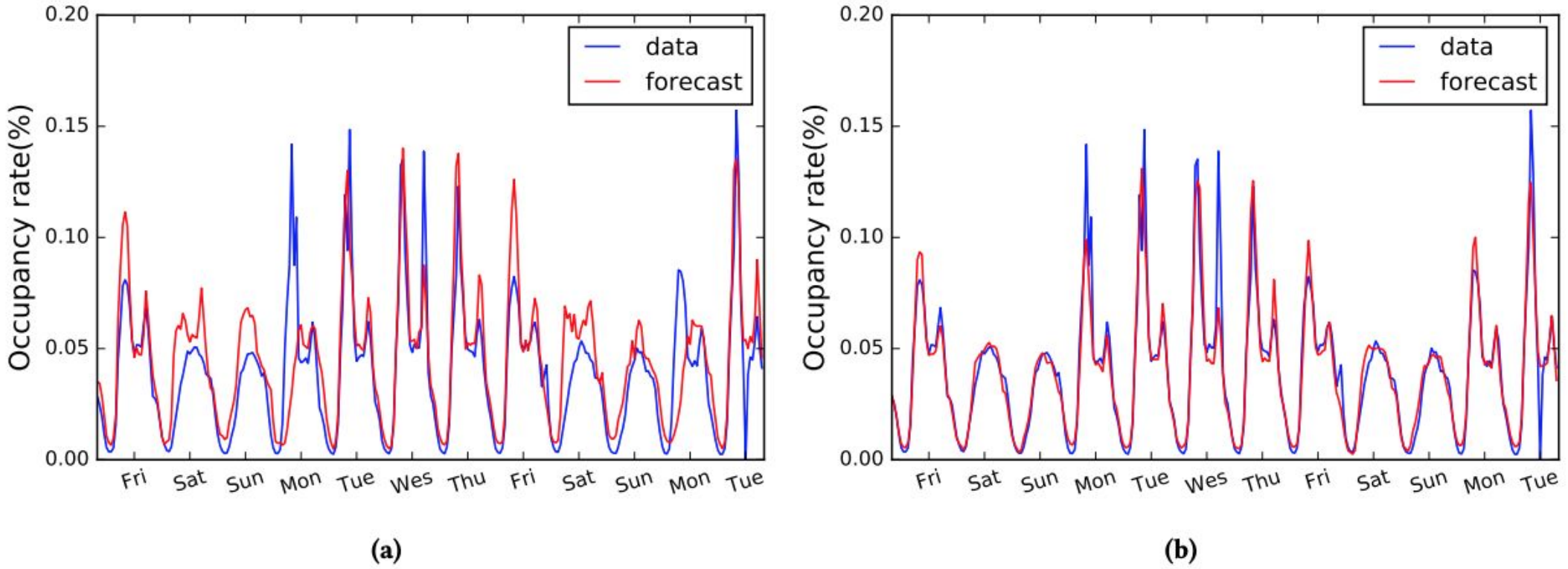


Figure 7: The true time series (blue) and the predicted ones (red) by VAR (a) and by LSTNet (b) for one variable in the Traffic **occupation dataset. The X axis indicates the week days and the forecasting** *horizon* $= 24$**. VAR inadequately predicts similar patterns for Fridays and Saturdays, and ones for Sundays and Mondays, while LSTNet successfully captures both the daily and weekly repeating patterns.**