State-Frequency Memory Recurrent Neural Networks

Hao Hu[1] Guo-Jun Qi[1]

[1]University of Central Florida

ICML, 2017
Presenter: Xueying Bai

Outline

Outline

Motivation

Task: Modeling temporal sequences (dynamics of time series).

- **RNN is able to capture data dependency:** RNN has shown great efficiency in modeling sequence data as well as temporal data.
- **Some cases hard to be handled by RNN:** RNN can capture long-range dependency in the time domain, but doesn't explicitly model the pattern occurrences in the frequency domain. So RNN fails in tasks like predicting investment strategy for the high frequency trading.

Outline

Overview

In this paper, authors combines the capacity of multi-frequency analysis with the modeling of long-range dependency to capture the temporal context of input sequences.

- **Decompose the input sequence into a set of frequency components.** Fourier basis.
- **Memory gates select a suitable set of state-frequency components.** These components are selected to predict and generate the target outputs.
- **Automatically adapt the frequency components.** Adaptive SFM, change fourier basis.

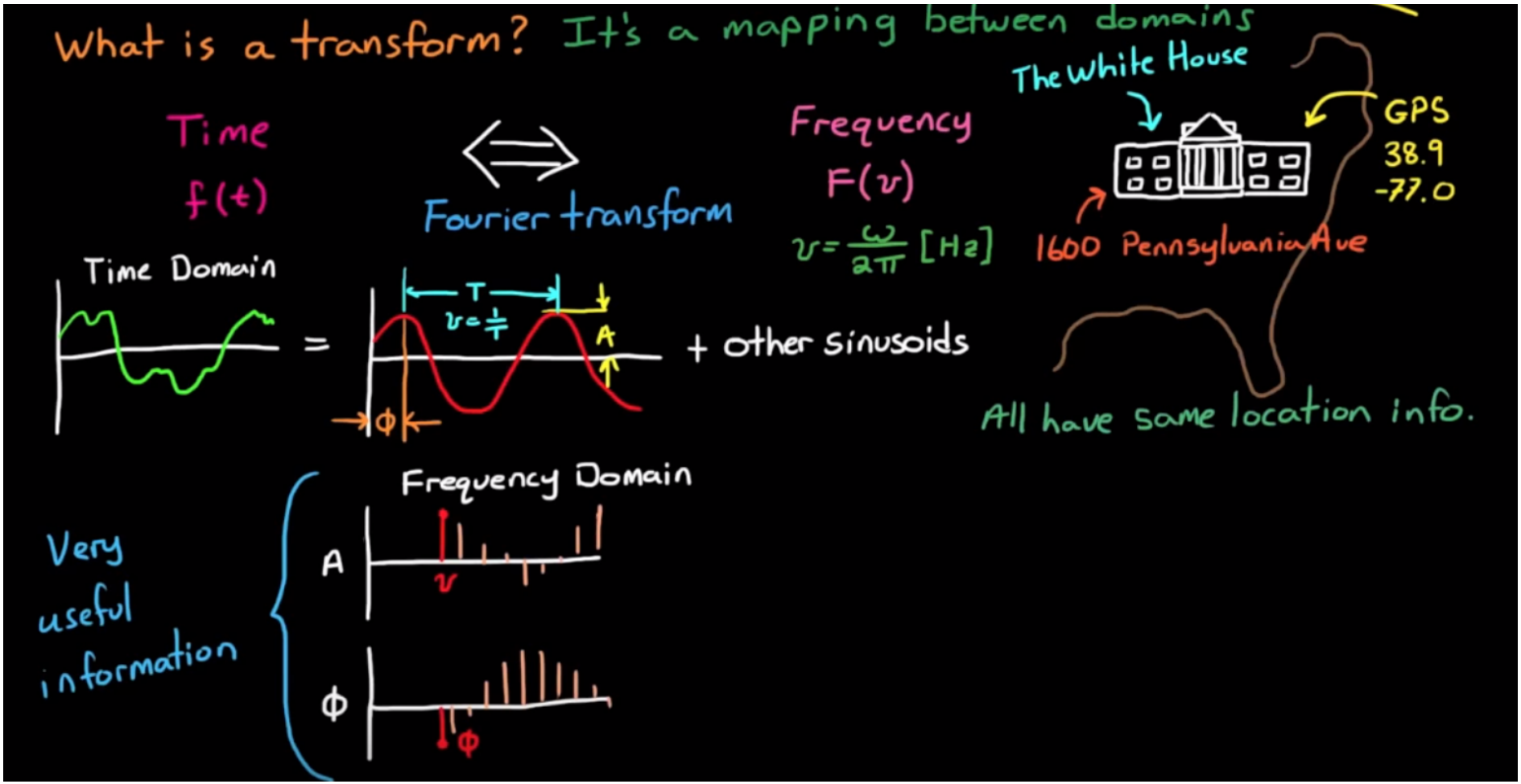
Outline

Recurrent Neural Network

- **RNN is good at modeling sequence which is highly related in time.** However, RNN has drawback in capturing the long term dependencies due to gradients vanishing.
- **Efforts to overcome this problem:** developing better learning algorithm; designing sophisticated structures (LSTM).
- **RNN has been applied to different areas.**

Fourier Transformation

Fourier Transformation: $F(\omega) = \int_{-\infty}^{+\infty} f(t) e^{j\omega t} dt$



Outline

Updating Rule

- **The updating rule is:**

$$S_t = f_t \circ S_{t-1} + (g_t \circ i_t) \begin{bmatrix} e^{j\omega_1 t} \\ \dots \\ e^{j\omega_K t} \end{bmatrix}^T \in C^{D \times K}$$

- **Notations:**
  S_t: State frequency matrix $\in C^{D \times K}$. Each memory cell contains D-dimensional memory states. Then decompose memory states into a set of frequency components (K components here).
  $[\cos \omega_1 t + j \sin \omega_1 t, \cos \omega_2 t + j \sin \omega_2 t, \dots \cos \omega_K t + j \sin \omega_K t]$: Fourier basis.
  f_t: Forget gate, $\in R^{D \times K}$.
  g_t: Input Gate, $\in R^D$.
  i_t: Input Modulation, $\in R^D$, collect current inputs.

Getting Amplitude

- **Decompose S_t into real and imaginary parts:**

$$\begin{aligned}\mathrm{Re}S_t &= f_t \circ \mathrm{Re}S_{t-1} + (g_t \circ i_t)[\cos\omega_1 t, \cos\omega_2 t, \ldots \cos\omega_K t] \\ \mathrm{Im}S_t &= f_t \circ \mathrm{Im}S_{t-1} + (g_t \circ i_t)[\sin\omega_1 t, \sin\omega_2 t, \ldots \sin\omega_K t]\end{aligned}$$

- **Amplitude:**

$$A_t = |S_t| = \sqrt{(\mathrm{Re}S_t)^2 + (\mathrm{Im}S_t)^2} \in R^{D\times K}$$

- The phase is ignored because experiments show that it doesn't effect the result.

The Joint State-Frequency Forget Gate

The input sequence $x = [x_1, x_2, ..., x_T], x_t \in R^N$.
The forget gate is designed to consider both state and frequency information.

- **State forget gate:** Which state information is allowed to update.

$$f_t^{ste} = \sigma(W^{ste} z_{t-1} + V^{ste} x_t + b^{ste}) \in R^D$$

- **Frequency forget gate:** Which frequency information is allowed to update.

$$f_t^{fre} = \sigma(W^{fre} z_{t-1} + V^{fre} x_t + b^{fre}) \in R^K$$

- **Joint forget gate:**

$$f_t = f_t^{ste} \otimes f_t^{fre}$$

$\otimes$ denotes the outer product. σ denotes an element-wise sigmoid function.

Input Gates and Modulations

Similar with the forget gate, the input gate is defined as:

- **Input controlling gate:**

$$g_t = \sigma(W_g z_{t-1} + V_g x_t + b_g) \in R^D$$

- **Input modulation gate:**

$$i_t = \tanh(W_i z_{t-1} + V_i x_t + b_i) \in R^D$$

All gates combine x_t and z_{t-1}

Multi-Frequency Outputs and Modulations

z_t is the output for time t. z_t is a combination of outputs from each frequency component.

- **Output from kth frequency component:**

$$z_t^k = o_t^k \circ f_0(W_t^k A_t^k + b_t^k)$$

- **Output gate for kth frequency component:**

$$o_t^k = \sigma(U_o^k A_t^k + W_o^k z_{t-1} + V_o^k x_t + b_o^k) \in R^M$$

- **Aggregated output:**

$$z_t = \sum_{k=1}^{K} z_t^k \in R^M$$

Outline

Fourier Analysis of SFM Matrices

- **Expanding the update rule, can get**:

$$S_t = (f_t \circ f_{t-1} \circ ... \circ f_1) \circ S_0 + (g_t \circ i_t) \begin{bmatrix} e^{j\omega_1 t} \\ ... \\ e^{j\omega_K t} \end{bmatrix}^T +$$

$$\sum_{t'=2}^{t} f_t \circ f_{t'} \circ g_{t'-1} \circ i_{t'-1} \begin{bmatrix} e^{j\omega_1 (t'-1)} \\ ... \\ e^{j\omega_K (t'-1)} \end{bmatrix}^T$$

- **Fourier transform denoting by the following sequence**:

$$\{(f_t \circ f_{t-1} \circ ... \circ f_1) \circ S_0\} \cup \{(g_t \circ i_t)\} \cup \{f_t \circ f_{t'} \circ g_{t'-1} \circ i_{t'-1} | t'\}$$

Gates determine the time window size.

Outline

Adaptive SFM

- For fixed frequency components, parameters $\{\omega_1, \omega_2, ..., \omega_K\}$ canbe calculated by

$$\omega_k = \frac{2\pi k}{K}$$

- However, the frequency bases will change overtime. Fixed base can't capture dynamic patterns of changing frequencies. So introduce a dynamic frequency component construction.

$$\omega = W_{\omega x} x_t + W_{\omega z} z_{t-1} + b_w$$

Outline

Baselines

	Task 1	Task 2	Task 3	Note
# of Params	$\approx$ 1k	$\approx$ 139k	$\approx$ 80k	-
GRU	18	164	141	-
LSTM	15	139	122	-
CW-RNN	30	295	245	$T_n \in \mathcal{T}^1$
ACT-RNN	18	164	141	-
RHN	9	94	76	$d = 4^1$
A-LSTM	15	139	122	$n = 4^1$
P-LSTM	15	139	122	$r_{on} = 0.05^1$
SFM	50×4	50×4	30×8	-

Outline

Sequence Data Generation

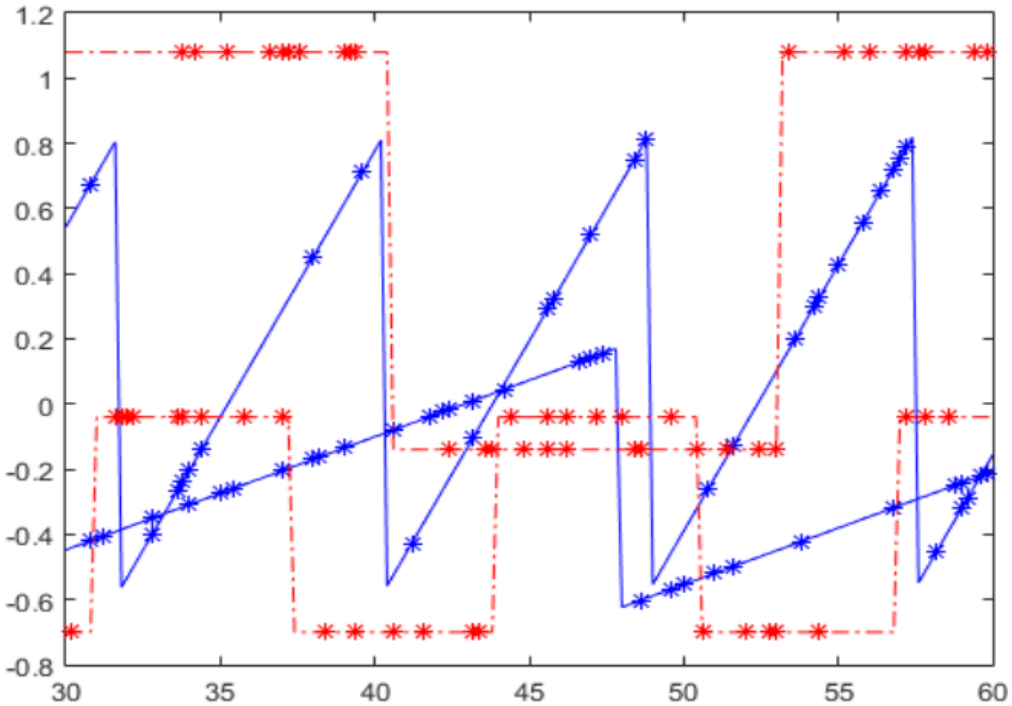


Figure 1. Several examples of the generated waves on the interval $[30, 60]$ with different periods, amplitudes, and phases. The red dash lines represent the square waves while the blue solid lines represent square waves. The '$*$' markers indicate the sampled data points that are used for training and testing.

Accuracy

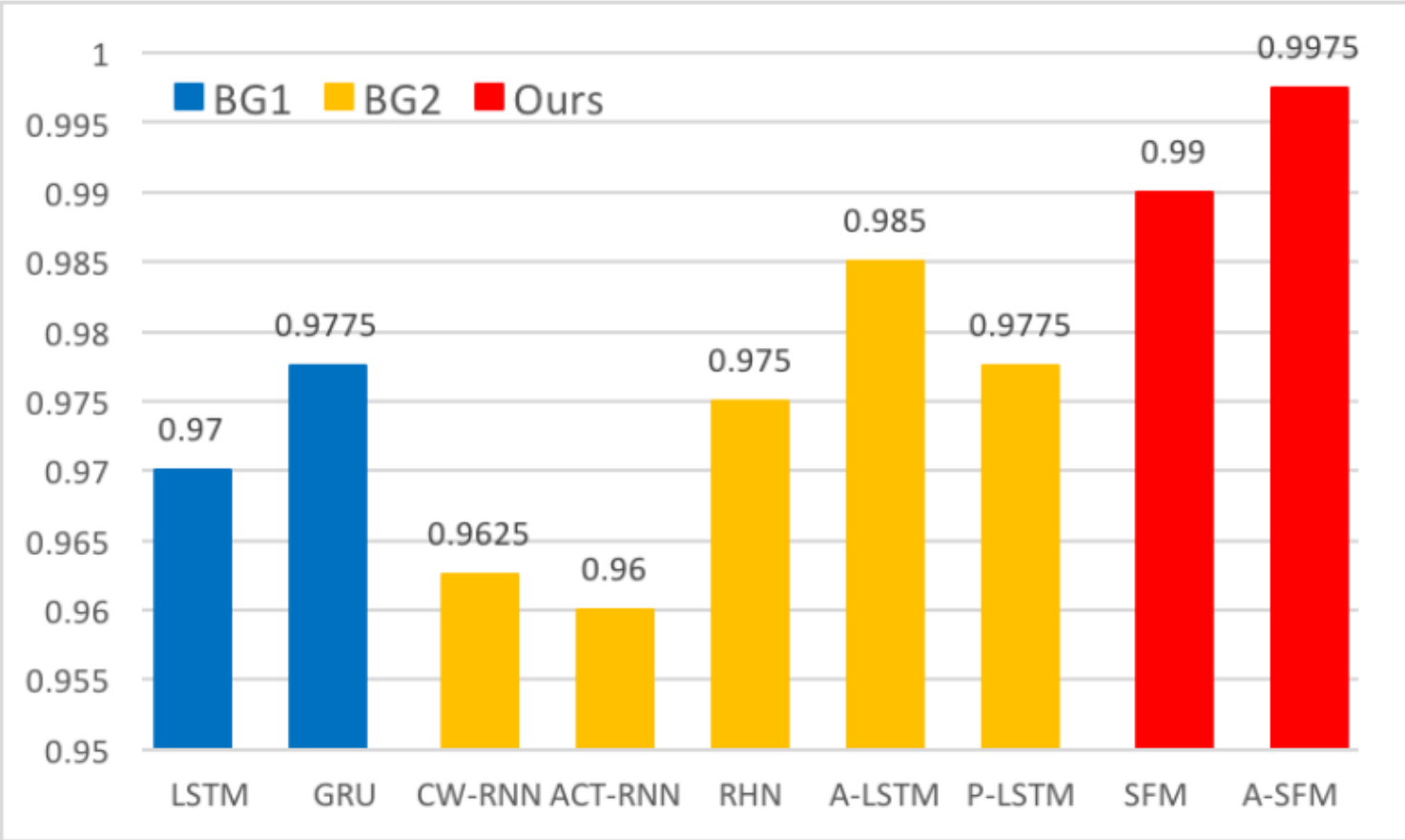


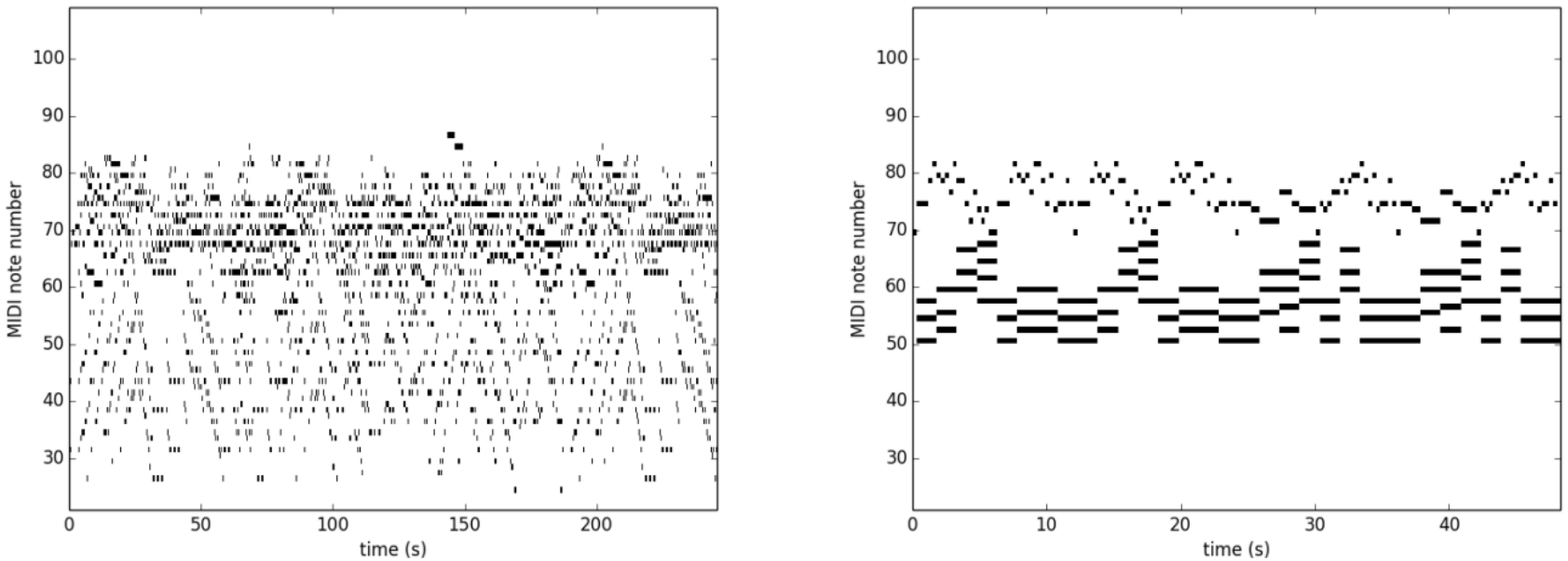
Figure 2. Signal type prediction accuracy of each model.

Outline

Log Likelihood

Dataset	LSTM	GRU	CW-RNN	ACT-RNN	RHN	A-LSTM	P-LSTM	RNN-RBM	RNN-NADE-HF	SFM	A-SFM
MuseData	−5.44	−5.36	−5.35	−5.20	−5.79	−5.03	−5.09	−6.01	−5.60	−4.81	**−4.80**
JSB chorales	−6.24	−6.14	−6.04	−5.89	−5.74	−5.63	−5.65	−6.27	−5.56	−5.47	**−5.45**
Piano-midi.de	−7.27	−7.14	−7.83	−7.41	−7.58	−6.96	−7.02	−7.09	−7.05	**−6.76**	−6.80
Nottingham	−5.60	−5.63	−5.90	−5.82	−5.60	−5.64	−5.70	−2.39	**−2.31**	−5.67	−5.63

Result Analysis



(a) MuseData (b) Nottingham

Figure 3. Piano rolls of the exemplar music clips from the MuseData and Nottingham dataset. Classical musics from MuseData are presented by complex, high frequently-changed sequences, while folk tunes from Nottingham contains simpler, lower-frequency sequences.

Outline

Accuracy

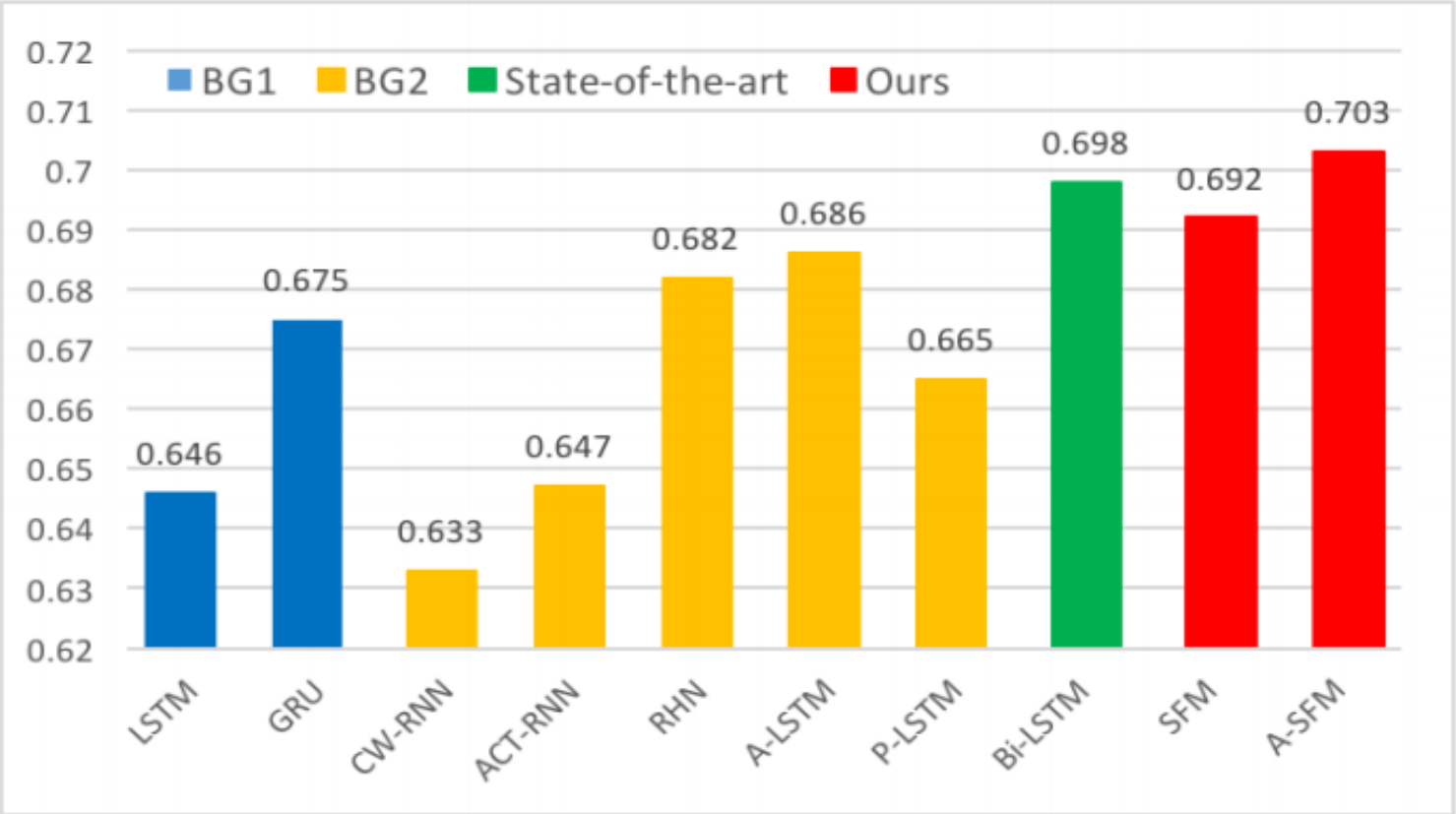


Figure 4. Accuracy for frame-level phoneme classification on TIMIT dataset.

Frequencies of SFM and A-SFM

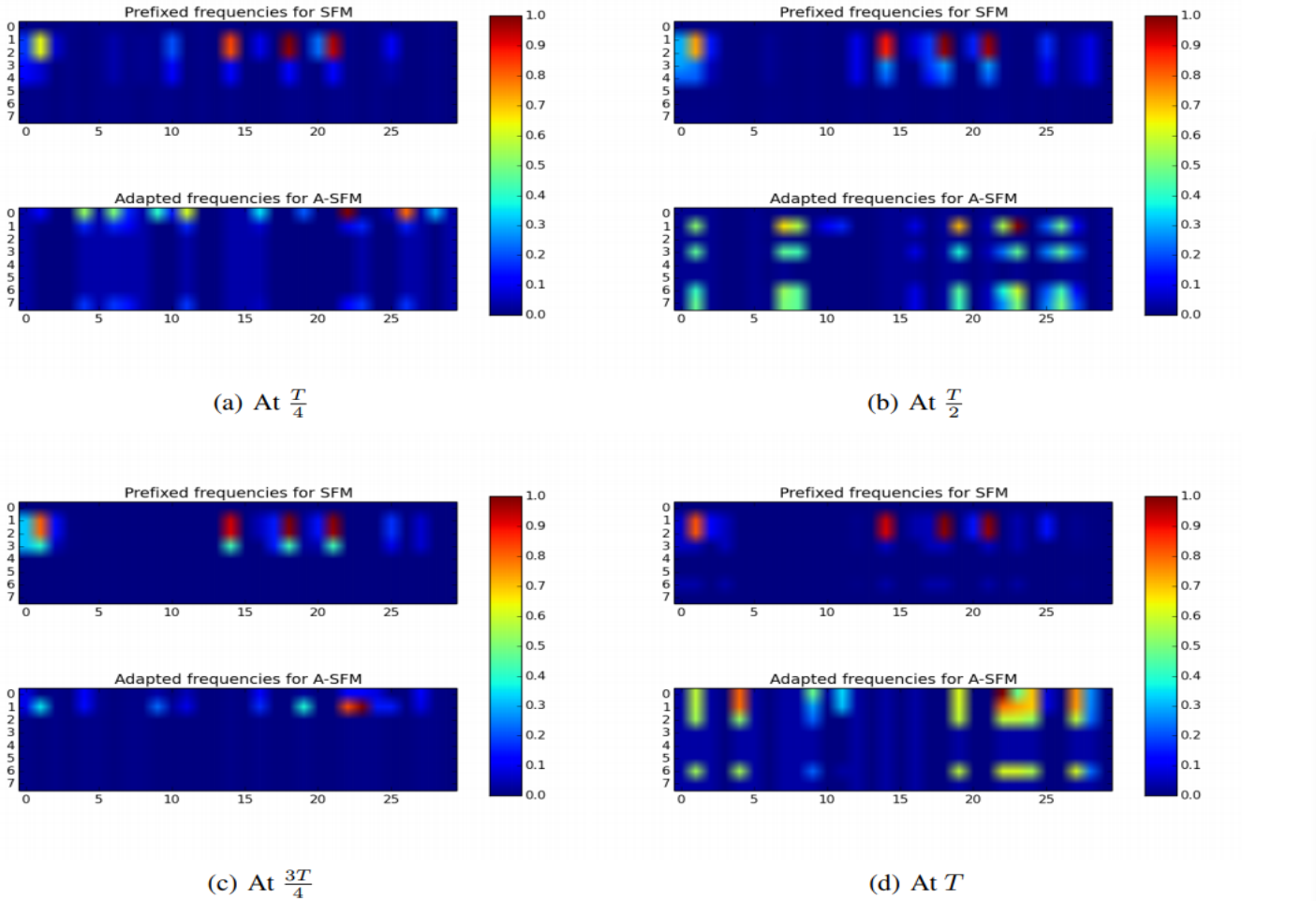


Figure 5. The amplitudes of SFM matrices for both the prefixed (SFM) and adaptive (A-SFM) frequencies. For all subfigures, each row represents a frequency component.

Conclusions

- The key idea of the SFM is to decompose the memory states into different frequency states such that they can explicitly learn the dependencies of both the low and high frequency patterns.
- The proposed SFM is powerful in discovering different frequency occurrences, which are important to predict or track the temporal sequences at various frequencies.