COMPACT RECURRENT NEURAL NETWORK BASED ON TENSOR-TRAIN FOR POLYPHONIC MUSIC MODELING

Andros Tjandra, Sakriani Sakti, Satoshi Nakamura

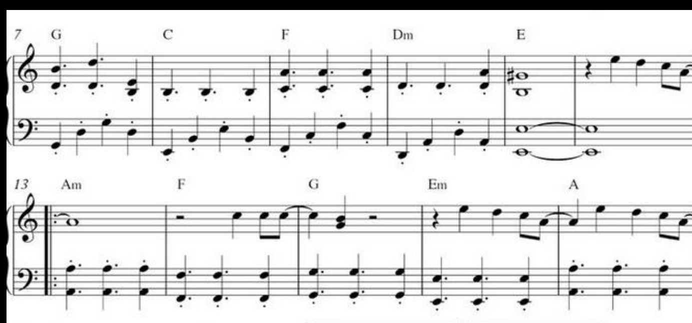
Nara Institute of Science and Technology, Japan

OUTLINE

- Motivation
- Tensor Train
- Recurrent Neural Network
- TT-based RNN
    - Representing Linear Transformation on TT-format
    - Simple RNN & Gated Recurrent Unit (GRU) RNN with TT-format
    - Initialization tricks for TTRNN
- Experiments & Results
- Conclusion

MOTIVATION

- RNNs architecture has become a popular choice for modelling temporal and sequential tasks



- However, most of current RNN models are computationally expensive and have a huge number of parameters

THIS PAPER PROPOSED …

- A new architecture TT-RNN : RNN with TT-format
- We applied TT-format to reparameterize two different RNNs
  - Simple RNN
  - Gated Recurrent Unit (GRU) RNN
- Goal : Reducing the number of parameters, meanwhile keep the expressiveness from RNN

TENSOR TRAIN (TT) FORMAT

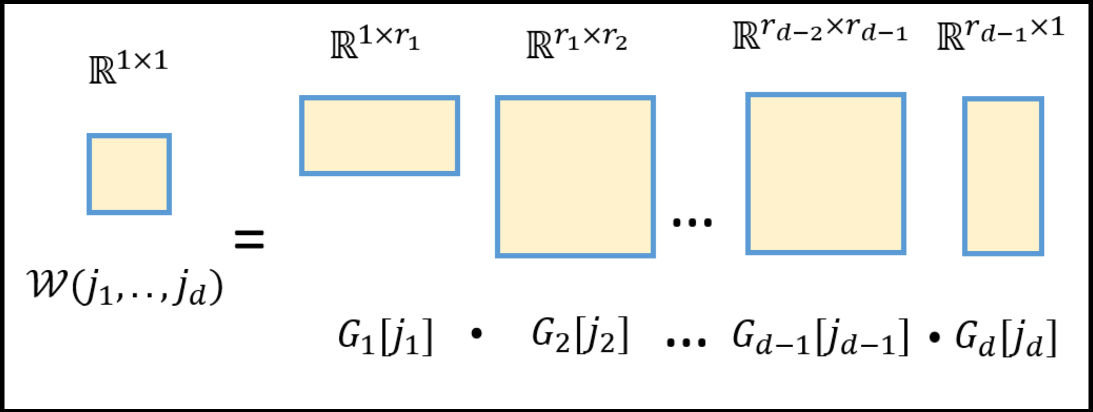
- Tensor Train decomposition (Oseledets, 2011):
  a factorization method for tensors (n-dimensional arrays)

- Efficient and simple representation for tensor through combination of multiple low-rank matrices

TT-FORMAT REPRESENTATION

Formulation:

$$\mathcal{W}(j_1, j_2 \dots j_{d-1}, j_d) = G_1[j_1] \cdot \dots \cdot G_d[j_d]$$



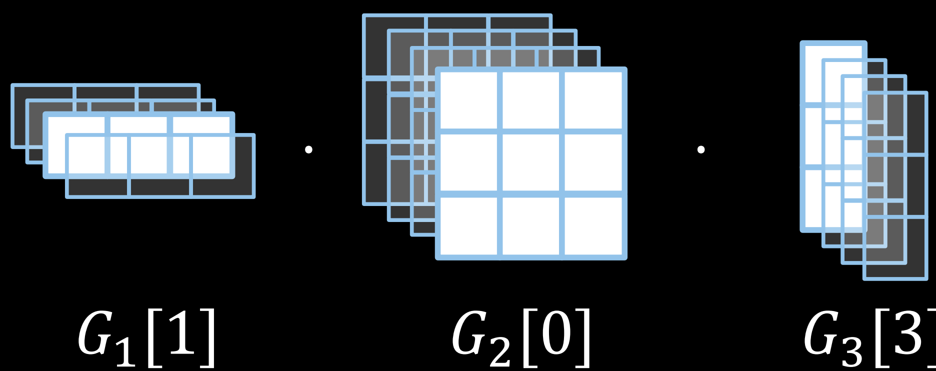
Terminology:

- G_d = TT-cores
- r_k = TT-ranks $(r_0, r_d = 1)$

EXAMPLE

Tensor $\mathcal{W} \in \mathbb{R}^{4\times4\times4}$
Can be represented by:

$\mathcal{W}(1,0,3) =$



REPRESENTING MATRIX USING TT-FORMAT

- We have matrix $W \in \mathbb{R}^{512 \times 512}$
- Represent them into tensor $\mathcal{W} \in \mathbb{R}^{(8 \times 8 \times 8) \times (8 \times 8 \times 8)}$
- Define bijective mapping :

matrix index → tensor index → TT-cores index

$$W(0,0) \rightarrow \mathcal{W}\big((0,0,0),(0,0,0)\big) \rightarrow G_1[0,0]G_2[0,0]G_3[0,0]$$
$$W(0,1) \rightarrow \mathcal{W}\big((0,0,0),(0,0,1)\big) \rightarrow G_1[0,0]G_2[0,0]G_3[0,1]$$

....

$$W(88,19) \rightarrow \mathcal{W}\big((1,3,0),(0,2,3)\big) \rightarrow G_1[1,0]G_2[3,2]G_3[0,3]$$
$$W(511,511) \rightarrow \mathcal{W}\big((7,7,7),(7,7,7)\big) \rightarrow G_1[7,7]G_2[7,7]G_3[7,7]$$

Assume we use TT-rank 3, we reduce the parameters from 512×512

$$\rightarrow (8^2 \times 1 \times 3) + (8^2 \times 3 \times 3) + (8^2 \times 3 \times 1)$$

$$262144 \rightarrow 960 \text{ !!!}$$

FORMAL DEFINITION

- Reparameterize matrix $W \in \mathbb{R}^{M \times N}$
  - where $M = \prod_{k=1}^{d} m_k$ and $N = \prod_{k=1}^{d} n_k$
  - as tensor $\mathcal{W} \in \mathbb{R}^{(m_1 \times \cdots \times m_d) \times (n_1 \times \cdots \times n_d)}$
  - by defining bijective function

$$f_i : \mathbb{Z}_{0+} \rightarrow \mathbb{Z}_{0+}^{d} \text{ and } f_j : \mathbb{Z}_{0+} \rightarrow \mathbb{Z}_{0+}^{d}$$

$$W(p,q) = \mathcal{W}\left(f_i(p), f_j(q)\right)$$

$$= \mathcal{W}([i_1(p), \dots, i_d(p)], [j_1(q), \dots, j_d(q)])$$

$$= G_1[i_1(p), j_1(q)] \cdot \dots \cdot G_j[i_d(p), j_d(q)]$$

- Number of parameters from :

$$M \times N \;\rightarrow\; \sum_{k=1}^{d} (m_k * n_k * r_{k-1} * r_k)$$

RECURRENT NEURAL NETWORK

- Simple RNN
  - Input : $\boldsymbol{x} = (x_1, \ldots, x_T)$
  - Hidden states : $\boldsymbol{h} = (h_1, \ldots, h_T)$
  - Equation for predict $h_t = f(W_{xh}x_t + W_{hh}h_{t-1} + b_h)$
  - Parameters :
    - W_{xh} = weight between input and hidden states
    - W_{hh} = weight between prev. and curr. hidden states

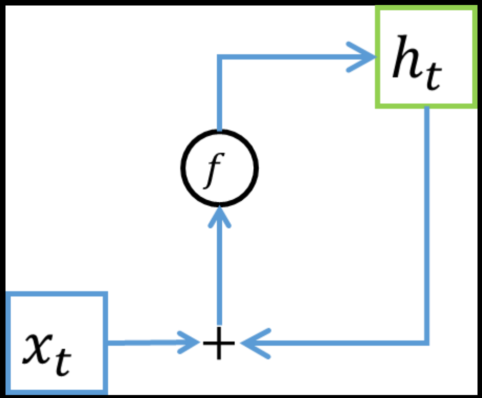


Fig. 1 : Simple RNN

RECURRENT NEURAL NETWORK (2)

- Limitations : vanishing gradient, harder to train compared to feedforward NN
- Gating mechanism was introduced to address these problems
  - Gated Recurrent Unit (GRU)
  - LSTM RNN

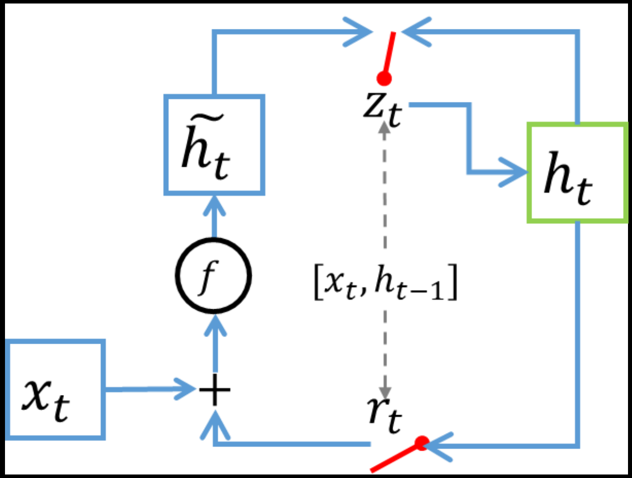


Fig. 2 : GRU

REPRESENTING LINEAR PROJECTION

- Most of RNN composed by linear transformation :

$$y = Wx + b$$

where $W \in \mathbb{R}^{M \times N}, b \in \mathbb{R}^{M}$

- Reformulate a linear projection

$$y(p) = W(p, :)x + b$$

$$y(p) = \mathcal{Y}\big(i_1(p) \ldots i_d(p)\big)$$
$$= \sum_{j_1, \ldots, j_d} G_1[i_1(p), j_1] \ldots G_d[i_d(p), j_d] \cdot \mathcal{X}(j_1 \ldots j_d)$$
$$+ \mathcal{B}\big(i_1(p) \ldots i_d(p)\big)$$

COMPRESSING RNN PARAMETERS

- Simple RNN has 2 parameters involved in linear proj. : $\{W_{xh}, W_{hh}\}$
- GRURNN has 6 parameters involved in linear proj. :
  - Reset gate : $\{W_{xr}, W_{hr}\}$
  - Update gate : $\{W_{xz}, W_{hz}\}$
  - Candidate state : $\{W_{xh}, W_{hh}\}$
- We replace all of them with TT-format

INITIALIZATION TRICKS FOR TT-CORES PARAMETERS

- Avoid saturation by using Glorot initialization (Glorot et al, 2010) independently for each TT-cores

$$\forall k \in \{1, .., d\}, \mathcal{G}_k \sim \mathcal{N}(0, \sigma_k)$$

$$\text{where } \sigma_k = \sqrt{\frac{2}{(n_k * r_k) + (m_k * r_{k-1})}}$$

EXPERIMENT

- Polyphonic Sequence prediction
  - Music dataset:
    - Piano-midi
    - Nottingham
    - MuseData
    - JSB Chorales

SEQUENCE MODELLING RESULT

PianoMidi Dataset

Model	RNN Params	Compr. Ratio	Test NLL
RNN-H512	393728	1	7.67 ± 0.02
TT-SRNN-H8x4x8x4-R3	2560	153.80	7.72 ± 0.04
TT-SRNN-H8x4x8x4-R5	**4864**	**80.95**	**7.68 ± 0.03**
GRU-H512	1181184	1	7.56 ± 0.01
TT-GRU-H8x4x8x4-R3	7680	153.80	7.61 ± 0.01
TT-GRU-H8x4x8x4-R5	**14592**	**80.95**	**7.59 ± 0.01**

More experiment results are in the paper …

CONCLUSION & FUTURE WORK

- Using TT-format to represent RNN, we are able to compress the number of parameters up to 80x and maintain the model expressiveness

- Future work : Compressing more complex NN architecture such as encoder-decoder

(｡◕‿◕｡) QUESTION ? (｡◕‿◕｡)

Link to full paper :
https://goo.gl/e9FEhB