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:

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

Terminology:

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

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

\[
\mathcal{W}(1,0,3) = G_1[1] \cdot G_2[0] \cdot G_3[3]
\]
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 $\rightarrow$ tensor index $\rightarrow$ TT-cores index

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

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

  ....

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

  $W(511,511) \rightarrow \mathcal{W}((7,7,7), (7,7,7)) \rightarrow G_1[7,7]G_2[7,7]G_3[7,7]$

Assume we use TT-rank 3, we reduce the parameters from $512 \times 512$

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

$262144 \rightarrow 960$ !!!
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

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

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

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

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

• Number of parameters from:

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

- **Simple RNN**
  - Input: \( x = (x_1, ..., x_T) \)
  - Hidden states: \( h = (h_1, ..., h_T) \)
  - 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

---

© Andros Tjandra - NAIST - 2017

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

© Andros Tjandra - NAIST - 2017
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

\[ y(p) = W(p,:)x + b \]

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

\[
\forall k \in \{1, \ldots, d\}, \mathcal{G}_k \sim \mathcal{N}(0, \sigma_k)
\]

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

<table>
<thead>
<tr>
<th>Model</th>
<th>RNN Params</th>
<th>Compr. Ratio</th>
<th>Test NLL</th>
</tr>
</thead>
<tbody>
<tr>
<td>RNN-H512</td>
<td>393728</td>
<td>1</td>
<td>7.67 ± 0.02</td>
</tr>
<tr>
<td>TT-SRNN-H8x4x8x4-R3</td>
<td>2560</td>
<td>153.80</td>
<td>7.72 ± 0.04</td>
</tr>
<tr>
<td><strong>TT-SRNN-H8x4x8x4-R5</strong></td>
<td><strong>4864</strong></td>
<td><strong>80.95</strong></td>
<td><strong>7.68 ± 0.03</strong></td>
</tr>
<tr>
<td>GRU-H512</td>
<td>1181184</td>
<td>1</td>
<td>7.56 ± 0.01</td>
</tr>
<tr>
<td>TT-GRU-H8x4x8x4-R3</td>
<td>7680</td>
<td>153.80</td>
<td>7.61 ± 0.01</td>
</tr>
<tr>
<td><strong>TT-GRU-H8x4x8x4-R5</strong></td>
<td><strong>14592</strong></td>
<td><strong>80.95</strong></td>
<td><strong>7.59 ± 0.01</strong></td>
</tr>
</tbody>
</table>

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
(°◡° plethora) QUESTION ? (°◡° plethora)

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