Introduction

RNNs are powerful sequence modeling tools. We introduce a compact architecture with Block-Term tensor decomposition to address the redundancy problem.

Challenges: 1) traditional RNNs suffer from an excess of parameters; and 2) Tensor-Train RNN has limited representation ability and flexibility due to the difficulty of searching the optimal setting of ranks.

Contributions: 1) we introduce a new sparsely connected BN architecture with Block-Term tensor decomposition; and 2) achieves better performance while maintaining fewer parameters.

Analysis

Comparison of complexity and memory usage of vanilla RNN, Tensor-Train RNN (TT-RNN) and our BT-RNN. In this table, the weight matrix's shape is \( I \times J \). Here, \( d_{max} = \max_{i,j,k} d \). [1, d].

Method

Background 1: Tensor Product

Given \( A = \mathbb{R}^{I \times J \times K} \) and \( b = \mathbb{R}^{I \times J} \), to simplify, \( i_{jk} \) denotes indices \((i,j,k)\), while \( i_{kj} \) denotes \((k,j,i)\):

\[
(A, b) \rightarrow i_{jk} = \sum_{r=1}^{R} a_{k,j}^{(r)} b_{j,i}^{(r)}
\]

(3)

Background 2: Block-Term Decomposition

\[ X = \sum_{i=1}^{N} X_i^{(1)} \cdots X_i^{(T)} W_d x_i + b \]

Implementation:

\[ W = \mathbb{R}^{I \times J \times K} \quad W_d = \mathbb{R}^{I \times J} \]

\[ J = \{ J_d \} \quad c = \mathbb{R}^{I \times J} \]

\( X_i^{(1)} \) denotes the core tensor, \( A_d^{(r)} \) denotes the factor tensor.

\[ X = \sum_{i=1}^{N} X_i^{(1)} \cdots X_i^{(T)} W_d x_i + b \]

(4)

Conclusion

We proposed a Block-Term RNN architecture to address the redundancy problem in RNNs. Experiment results on 3 challenge tasks show that our BTRNN architecture can not only consume several orders fewer parameters but also improve the model performance over standard traditional LSTM and the TT-LSTM.

References


Experiments


Datasets: 1. UCF11 YouTube Action dataset; 2. MNIST; 3. MEDICO.

Table: State-of-the-art results on UCF11 dataset reported in literature, in comparison with our best model.

Figure: Performance of different RNN models on the Action Recognition task trained with UCF11.

Task 1: We use a single LSTM cell as the model architecture to evaluate BT-LSTM against LSTM and TT-LSTM. The frames in video are directly input to the LSTM cell. The figure in left demonstrates the training loss of different models. From these experiments, we claim that our BT-LSTM has: 1) \( 8 \times 10^6 \) times parameter reductions; 2) faster convergence; 3) better model efficiency.

Task 2: In this experiment, we use an encoder-decoder architecture to generate images. We only substitute the encoder network to qualitatively evaluate the LSTM and BT-LSTM. The result shows that both LSTM and BT-LSTM can generate comparable images.

Task 3: We use the architecture described in [4], all the three models can generate proper sentences but with little improvement in BT-LSTM.