2017

Formal Languages and Computability CMPT440

Extracting Learned States From An LSTM

PROJECT MILESTONE

MICHAEL GUARINO

Abstract:

We purpose a method for extracting states from a Recurrent Neural Network architecture, Long Short Term Memory, to assemble rules that govern transitions between states in a Deterministic Finite Automata. The Long Short Term Memory (LSTM) deep learning architecture performs a language modeling task for word-level prediction. The LSTM is regularized using dropout between the layers not between the recurrent connections and experimentation has been done to determine the optimal number of layers and recurrent connections.[4] To perform the state extraction process LSTM states are clustered together using dimensionality reduction then states are clustered together based on a similarity metric and are then used to construct state transitions on the constructed Deterministic Finite Automata.

Introduction:

The Recurrent Neural Network architecture is quite popular for language modeling tasks. The LSTM variant of the Recurrent Neural Network family of deep learning architectures is specifically used for capturing long term dependencies within sequences.[3] Learning formal grammars via the Recurrent Neural Network architecture for the purpose of extracting the learned states have been explored by Giles 1992[1], Firoiu 1996[2], and Vahed 2004[5]; however, the LSTM architecture is not known to be used for this task. Once states are extracted from the LSTM architecture a Deterministic Finite Automata can be constructed using the extracted states that describe the transitions between states in the Deterministic Finite Automata.

System Description:

The LSTM Recurrent Neural Network will perform a language modeling task on a corpus dataset of text documents doing word-level prediction. The LSTM will learn states that allow it to properly perform word-level prediction task on the corpus. These learned states will then be extracted from the LSTM Recurrent Neural Network and be used to create state transitions between states in a Deterministic Finite Automata. This Deterministic Finite Automata can then accept a text input sequence to determine if the text input sequence was in the original corpus. The DFA essentially acts to validate that the text input sequence exists within the original dataset corpus which confirms that the LSTM state extraction process has been successful therefore valid input to the DFA is an text input sequence that exists in the dataset corpus.

The Recurrent Neural Network architecture that was used was heavily based on the LSTM architecture of Zaremba in "Recurrent Neural Network Regularization."[4] A two layer LSTM with 35 recurrent connections (steps) for every LSTM memory cell. The LSTM memory cell using a gating mechanism to update/forget 'memory' states as depicted in figure 1-1. The hidden states of the LSTM were initialized to zero. Several sizes were experimented with for the LSTM layers to determine the most effective architecture. We use a medium LSTM architecture with 650 units per layer with all parameters initialized uniformly between -0.5 and 0.5 with dropout applied to 50% of non-recurrent connections. A large LSTM architecture was also used with 1500 units per layer all parameters initialized uniformly between -0.5 and 0.5 with dropout applied to 65% of non-recurrent connections. With both of these architectures the gradients were normalized and clipped.

Once the LSTM Recurrent Neural Network has successfully learned the dataset a process to extract network states will proceed. Several authors have briefly eluded to how this is done Giles 1992[1], Firoiu 1996[2], and Vahed 2004[5]; however a concrete method has not yet been established. We hope to extract network states by performing a dimensionality reduction such as

principle component analysis on the network state then clustering them using some unsupervised machine learning algorithm to group related states together based on how they respond to input.

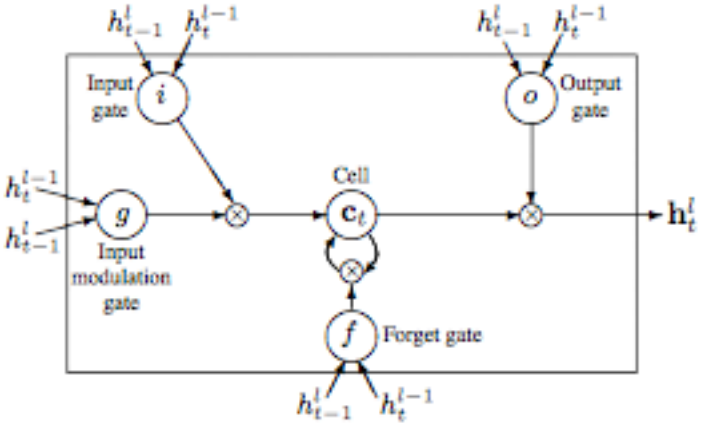

Figure 1-1 LSTM memory cell[4]

Requirements:

In order to train the LSTM Recurrent Neural Network we used a well known dataset for Natural Language Processing, Penn Tree Bank (https://catalog.ldc.upenn.edu/ldc99t42), which we determined to be small enough to train quickly on our computing resources but large enough to serve as an actual test to the method purposed in this document. The LSTM model was trained on a Nvidia 1080TI.

Literature Survey:

There has been some existing interest in using Recurrent Neural Network architectures to construct Deterministic Finite Automata in the area of natural language processing. Giles and Chen constructed a Recurrent Neural Network to learning an unknown grammar and used a dynamic clustering algorithm to extract the states learned by the Recurrent Neural Network in order to construct a Deterministic Finite Automata.[1] Firoiu, Oates, and Cohen also performed a similar task using a Elman Recurrent Neural Network to perform a word-level prediction task on a text dataset and used a state extraction process to extract the learned states of the neural network to construct a Deterministic Finite Automata.[2] There has not been any research done in this area using a Long Short Term Memory Recurrent Neural Network architecture. The Long Short Term Memory architecture has become very popular for problems involving learning long term dependencies within sequences with variants showing significant improvements in the area of speech recognition, polyphonic music modeling, and acoustic modeling.[3] These models area extremely powerful in their ability to learn time series/sequential data.

User Manual:

This system will validate that text input exists in the dataset corpus. The learned states of the LSTM Recurrent Neural Network will create a Deterministic Finite Automata after a state extraction process. The user will enter text input and the Deterministic Finite Automata will either accept the text input meaning that it exists in the dataset corpus or it will reject the text input meaning that it does not exist in the dataset corpus.

Conclusion:

There are currently no conclusion that can be made about this purposed process.

References/Bibliography:

1. Giles, C. I., C. B. Miller, D. Chen, G. Z. Sun, H. H. Chen, and Y. C. Lee. "Extracting and learning an unknown grammar with recurrent neural networks." *Neural Information Processing Systems* (1992): n. pag. Web.
2. Firoiu, Laura, Tim Oates, and Paul R. Cohen. "Learning deterministic finite automata with a recurrent neural network." (1996): n. pag. Web.
3. Greff, Klaus, Rupesh K. Srivastava, Jan Koutnik, Bas R. Steunebrink, and Jurgen Schmidhuber. "LSTM: A Search Space Odyssey." *IEEE Transactions on Neural Networks and Learning Systems* (2016): 1-11. Web.
4. Zaremba, Wojciech, Ilya Sutskever, and Oriol Vinyals. "Recurrent Neural Network Regularization." *International Conference on Learning Representations* (2015): n. pag. Web.
5. Vahed, A., and C. W. Omlin. "A Machine Learning Method for Extracting Symbolic Knowledge from Recurrent Neural Networks." *Neural Computation* 16.1 (2004): 59-71. Web.