One-shot Learning with Differentiable Neural Computer

A Proposal for Machine Learning Camp Jeju 2017

Xihan Li
xihanli@pku.edu.cn
https://snowkylin.github.io

School of Electronics Engineering and Computer Science
Peking University

April 20, 2017

1 Introduction

One-shot learning is a key feature of human intelligence, which means learning a new behavior or rapidly shifting away from an old behavior with comparatively little data. Inspired by [1], I'd like to build a differentiable neural computer (DNC)[2] and modify it to excel at one-shot learning. DNC is similar to neural Turing machine (NTM)[3] since both of which has external memory that provides the ability to quickly encode and retrieve information, but differed in the access mechanism.

2 Motivation and Background

Most of the time, we need massive amounts of data to train deep neural networks. It is not easy to get access to large amount of data, especially when you need to collect it by yourself (for example, taking pictures for a specific object). More importantly, it is not needed for intelligence life in many cases. For example, you needn't look through thousands of traffic signs before recognize them – only a handful of them is enough. That results in the concept of "one-shot learning". From a cognitive science perspective, one-shot learning is closer to the high-level learning process of humans.

This is why I'd like to choose DNC to perform one-shot learning. Neural Turing machine (the predecessor of DNC), a sequencial learning model with controller that has readwrite access to a memory matrix, is similar to the working memory of humans thus has a background of cognitive science. It is proved that a modified version of neural Turing machine can perform well in one-shot learning[1]. We have reason to believe that DNC, which overcomes several major drawbacks of NTM, can perform even better in one-step learning.

3 Previous Work

Last year, I have read dozens of ML/DL related papers, and implemented Alex Graves' RNN handwriting prediction & synthesis model[4] and Google Deepmind's DRAW model for image generation[5] with TensorFlow, which are available on snowkylin/rnn-handwriting-generation and snowkylin/rnn-vae respectively. With these papers and implementations, I have a comparetively deep understanding about RNN, generative models and attention & memory mechanism, which are essential to implement DNC and modify it to perform one-step learning.

4 Plan of Action

1. Implement NTM based on [4](have implemented) and [3]. (may bypass if can implement DNC directly) 1 week
2. Implement DNC based on [2]. 1 week
3. Modify DNC to perform one-step learning. 1-2 week

5 Resources

- One-shot learning using NTM:
  `https://github.com/hmishra2250/NTM-One-Shot-TF`
- Deepmind's DNC implementation:
  `https://github.com/deepmind/dnc`
  (Although Deepmind open-sourced their DNC implementation 2 days ago, I think it is needed to re-implemented the DNC model, since we need to make some major modifications to perform one-step learning)
- Alex Graves' RNN handwriting prediction & synthesis model:
  `https://github.com/snowkylin/rnn-handwriting-generation`
  (RNN with attention, implemented by myself)

References

[1] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. One-shot learning with memory-augmented neural networks. *arXiv preprint arXiv:1605.06065*, 2016.

[2] Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio Gómez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, et al. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538(7626):471–476, 2016.

[3] Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

[4] Alex Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

[5] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan Wierstra. Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*, 2015.