(May 30, 2018)

Project Title (Tentative): Toward developing a medical Natural Language Generation system using Deep Learning and Reinforcement Learning

Abstract

We aim to build a prototype framework for Natural Language Generation (NLG) for the medical field leveraging recent advances in Deep Learning and Reinforcement Learning. By defining the task of NLG as sequential decision-making process, we implement two baselines, one built upon Long Short-Term Memory (LSTM)-based Recurrent Neural Network (RNN) and the other with Monte Carlo Tree Search (MCTS). Later, other variants based on MCTS are further experimented for enhancing the performance, while focusing roles of evaluation metric, target corpus, and algorithmic benefits, etc.

I. References

* LSTM RNN
  + Karpathy’s blog for RNN : <http://karpathy.github.io/2015/05/21/rnn-effectiveness/>
  + Main Reference for LSTM-based sequence generation (baseline model I) : Graves, A. (2013). Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*.
  + (More to come)
* MCTS
* MCTS-based Natural Language Generation (baseline model II) : Kumagai, K., Kobayashi, I., Mochihashi, D., Asoh, H., Nakamura, T., & Nagai, T. (2016). Human-like Natural Language Generation Using Monte Carlo Tree Search. In *Proceedings of the INLG 2016 Workshop on Computational Creativity in Natural Language Generation* (pp. 11-18).
* (More to come)
* Deep Reinforcement learning
* Some sort of Introduction (Only for fun and your knowledge)
* <https://deepmind.com/>
* (Just one hit randomly) <https://www.wired.com/story/greedy-brittle-opaque-and-shallow-the-downsides-to-deep-learning/>
* (More to come)

II. Research Project

1. Introduction of Deep Learning

* Artificial Neural Network models (since 2006, emphasized in its deep architecture)
  + End-to-end training with backpropagation
  + Scalable for large datasets
* Great performance for human-like perception problems (vision, NLP, audio, speech, etc). \*biological sequences such as genomic and proteins. \*some interests on graph datasets
* Machine Learning framework with representation learning, offering effective methods for unsupervised learning and other advanced methods such as meta learning, transfer learning, active learning, multi-task learning, multi modal learning, etc
* Deep Reinforcement Learning
  + Deep Learning topics with Reinforcement Learning techniques
  + Reinforcement Learning combined with Deep Learning-based function approximation
* Types of Deep Learning
  + DNN, RNN, CNN, and my hybrid or complex architectures
  + RNN – Long Short-Term Memory (LSTM)

1. Introduction of Monte Carlo Tree Search and Reinforcement Learning
   * MDP vs. POMDP

(more to come)

* Monte-Carlo Tree Search

(more to come)

C. LSTM-based Sequence Generation

* Examples
  1. <https://pytorch.org/tutorials/intermediate/char_rnn_generation_tutorial.html> (Pytorch)
  2. <https://github.com/karpathy/char-rnn> (torch)

(Case l) Classification

<https://pytorch.org/tutorials/intermediate/char_rnn_classification_tutorial.html>

(Case ll) Sequence Generation

<https://pytorch.org/tutorials/intermediate/char_rnn_generation_tutorial.html>

Supp. Technical training

Pre-requisite

* PyTorch
* Spacy
* GitHub
* Jupyter
* Anadonda/PIP (Home Brew for Mac)
* Linux commands : ssh, etc
* Programming Development Environment Tool: ex) Microsoft Visual Studio Code

Note: There are numerous tutorials and youtube clips you can watch for more information. And, the most efficient way to master and to remember what you know is to repeat the same thing many times!

1. GitHub

<https://www.atlassian.com/git?utm_source=basic-git-commands&utm_medium=link&utm_campaign=git-microsite&_ga=2.75964245.1174987269.1527087707-1877757822.1527087707>

\* Our repository location : <http://github.com/sdaysmerrill/Summer-REU-Project/>

1. Python
   * Shell vs. Jupyter notebook
   * Package management with conda and pip

https://pip.pypa.io/en/stable/

* + Environment with anaconda

<https://conda.io/docs/_downloads/conda-cheatsheet.pdf>

1. Jupyter
2. Ssh
   * Ssh [reu@130.39.94.206](mailto:reu@130.39.94.206)
   * Password: your\_first\_name + “this year”

E. PyTorch

* Intro: <https://pytorch.org/tutorials/brginner/blitz/tensor_tutorial.html>
* Autograd: <https://pytorch.org/tutorials/beginner/blitz/autograd_tutorial.html>
* Neural Network: <https://pytorch.org/tutorials/beginner/blitz/neural_networks_tutorial.html>
* Training a classifier: <https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html>
* Other examples: <https://github.com/pytorch/examples> (ex: word\_language\_model, snli)

(Key Ideas)

* Forward calculation with Neural Network model
* Backward (Backpropagation) with automatic differentiation
* Training with Loss minimization (Regression and Classification)
* Stochastic Gradient Decent with batch