Parallelizing Legendre Memory Unit Training

Prallelizing legendre memory unit training leverage the linear time-invariant (LTI) memory component of the LMU to construct a simplified variant that can be parallelized during training (and yet executed as an RNN during inference), thus overcoming a well known limitation of training RNNs on GPUs.

**Main Idea:**

* simplifying the LMU such that recurrence exists only in the linear system.

Inspired by the success of self-attention. Self-attention based architectures have come to replace RNN based approaches for problems such as language modelling, machine translation, and a slew of other NLP tasks (Radford et al., 2018; Raffel et al., 2019). Three properties that make self-attention desirable over RNNs are:

(1) it is better at handling the challenging problem of long-range dependencies;

(2) it is purely feedforward

(3) when the sequence length is smaller than the dimension of representation

**Model:**

implement a general affine transformation followed by an element-wise nonlinearity



1. Parallel Training:

One of the motivations for the above mentioned architectural changes is that the model now has only one recurrent connection: mt’s dependence on itself from the past. But because this is an LTI system, standard control theory (Astrom & Murray , 2010) gives a non-iterative way of evaluating this equation as shown below

Text

Description automatically generated

It is also evident from the structure of the U matrix that although this reformulation turns the DN into a feedforward layer, it still respects causality. In other words, the state mt depends only on the inputs seen until that point of time



1. Complexity:

This can be made more efficient by employing the convolution theorem which gives an equivalent way of evaluating the convolution in the Fourier space as



It was argued in Vaswani et al. (2017) that a self-attention layer is cheaper than an RNN when the representation dimension of the input, dx, is much greater than the length of the sequence, n, which is seen in NLP applications.

1. Recurrent Inference:

Machine learning algorithms are usually optimized for training rather than deployment (Crankshaw, 2019), and because of that models need to be modified, sometimes non-trivially, to be more suitable for inference.

While this model can be trained in parallel, it can also be run in an iterative manner during inference, and hence can process data in an online or streaming fashion during inference.

**Experiments:**

In the following experiments, comparing the model against the LMU, LSTMs and transformers.

In psMNIST: as the name suggests, is constructed by permuting and then flattening the (28 × 28) MNIST images. The permutation is chosen randomly and is fixed for the duration of the task. It uses the standard 50k/10k/10k split.

1. Architecture:

Model uses 165k parameters, original LMU model, which uses 102k parameters, and the HiPPO-LegS model, which is reported to use 512 hidden dimensions

1. Results & Discussion:

Test scores of various models on this dataset are reported in the table below. Model not only surpasses the LSTM model, but also beats the current stateof-the result of 98.3% set by HiPPO-LegS (Gu et al., 2020) recently. Thus, Model sets a new state-of-the art result for RNNs of 98.49% on psMNIST. It is interesting that the model, despite being simpler than the original LMU, outperforms it on this dataset.

Table

Description automatically generated

1. Additional Experiment:

PyTorch implementations for Parallelizing Legendre Memory Unit Training on psMNIST

Chart

Description automatically generated with low confidence

**Diagram

Description automatically generated**

**Conclusion:**

Method of parallelization applies to all deep architectures with linear recurrent dependencies, and analysis highlights the utility of linear systems for the purposes of machine learning. In sum, we believe that linear systems offer a scalable solution to many time series problems without sacrificing accuracy.

**Citation:**

* [PyTorch LMU github](https://github.com/hrshtv/pytorch-lmu)
* [Parallelizing Legendre Memory Unit Training](https://arxiv.org/pdf/2102.11417v2.pdf)
* Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. Improving language understanding by generative pretraining. 2018.
* Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. arXiv preprint arXiv:1910.10683, 2019
* Astrom, K. J. and Murray, R. M. ¨ Feedback systems: an introduction for scientists and engineers. Princeton university press, 2010
* Crankshaw, D. The Design and Implementation of LowLatency Prediction Serving Systems. PhD thesis, UC Berkeley, 2019.
* Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In Advances in neural information processing systems, pp. 5998–6008, 2017.