MQP Daily Reports

Neil Kale

# 08/29/2023

I completed a Turing Account Request Form. My reason was “I am using Pytorch to train recurrent neural networks, and I need access to GPUs.”

Next, I read “RWKV: Reinventing RNNs for the Transformer Era” (Peng et al., 2023).

**ChatGPT Summary of RWKV Paper**

*RWKV, short for Receptance Weighted Key Value, is a novel neural network architecture designed for sequence processing tasks, particularly in the domain of natural language processing (NLP). It combines the strengths of both recurrent neural networks (RNNs) and transformer models while mitigating their limitations. Here's an explanation of how RWKV works:*

1. ***Basic Building Blocks****: RWKV consists of a series of stacked residual blocks. Each block has two main components: the time-mixing block and the channel-mixing block. These blocks are the key to RWKV's functionality.*
2. ***Time-Mixing Block****: This block captures the temporal relationships between elements in a sequence. It employs recurrence, which is a characteristic feature of RNNs. In this block, the input sequence is processed in a time-dependent manner.*
   * ***Receptance (R)****: This vector acts as a gate that determines the acceptance of past information. It helps control the flow of information from previous time steps.*
   * ***Weight (W)****: This is a positional weight decay vector, and it's a trainable parameter in the model. It's used to gradually diminish the influence of past information over time.*
   * ***Key (K), Value (V)****: These vectors are analogous to the keys and values used in traditional attention mechanisms.*
   * ***Time-Shift Mixing****: RWKV introduces a concept called time-shift mixing. It linearly interpolates between the current input and the input from the previous time step. This helps the model aggregate and gate information in the input channels effectively.*
   * ***Update of WKV****: The WKV computation in this block involves interactions between elements in a sequence and is carried out in a time-dependent manner, avoiding quadratic complexity.*
   * ***Output (O)****: The output of this block is a result of applying a linear transformation to the computed values, making it suitable for further processing.*
3. ***Channel-Mixing Block****: This block is designed to capture relationships between different features (channels) within each element of the sequence. It's somewhat analogous to self-attention in traditional transformers but is more efficient.*
   * ***Receptance (R), Weight (W)****: Similar to the time-mixing block, these components play a role in controlling the flow of information and maintaining efficiency.*
   * ***Key (K), Value (V)****: These components are used to compute interactions between channels in the input sequence.*
   * ***Squared ReLU Activation****: RWKV applies a squared rectified linear unit (ReLU) activation function in this block, enhancing its modeling capabilities.*
4. ***Parallelization and Efficiency****:*
   * ***Time-Parallel Mode****: RWKV can be efficiently parallelized during training, similar to traditional transformers. This parallelization reduces the computational complexity.*
   * ***Time-Sequential Mode****: During inference, RWKV can operate in a time-sequential mode, which is highly efficient for tasks that require autoregressive decoding, like text generation.*
5. ***Gradient Stability and Layer Stacking****:*
   * *RWKV avoids common gradient stability issues associated with traditional RNNs by using a combination of softmax operations and layer normalization.*
   * *This design also enables stacking multiple layers, allowing the model to capture complex patterns and dependencies across different levels of abstraction.*

*In summary, RWKV is a versatile architecture that combines the best of both RNNs and transformers. It efficiently processes sequential data, captures temporal and feature relationships, and offers stable gradients, making it competitive with state-of-the-art models in various NLP tasks. It achieves these capabilities through a combination of time-mixing and channel-mixing blocks, efficient parallelization, and careful weight management.*

I watched [(269) RWKV: Reinventing RNNs for the Transformer Era (Paper Explained) - YouTube](https://www.youtube.com/watch?v=x8pW19wKfXQ) to better understand the RWKV model. My one-sentence summary is that RWKV keeps a vector of weights as hidden input. Each weight is formed by taking a weighted sum of the past tokens, applying some non-linear function, and forgetting/erasing part of the information. I messaged Randy on Teams updating him about my progress with the paper.

# 08/30/2023

Randy responded to my message and asked me to implement RWKV to test my understanding. I followed [(270) RWKV from scratch Pytorch - YouTube](https://www.youtube.com/watch?v=0Ag83EhYD7k&t=1s) to write a 230-line Pytorch implementation of RWKV.

Here is some text generated by the model when using individual characters as tokens:

Tha, wation his iuglat rewis?

JULIY:

No do come will. i' no not moun you gar; who go is shounberse?

Clauton loblestgian is theing chown anyst beear the Sog,

Come of dishallowon'd what in play I hupose to for mustroye some ewhich.

Shithern, in nou upothen is, po of come otriwarest disperue,

Mushould for of with sir,

Thou my not: he goo, sha seecome whick, my don not one who heave for ohe acts would onour withsvain town, ba have which twer,

What don I will lought ou host we one is at is reso the wordon what sone, seer more with and givey, a ted to geout this exRie i as was him say

in't is in worddy tlies, abribeit

if she comet.

AUBE

Not quie.

Thee with is fighty

So no gueibs wolp 'people care's heried shall, belagt:

If of sen fall will at but sold mine somear:

Now of a hath: cready, wous try's glacher;

Bead.

By shy to reard.

Ord

MENE:

Go neath'teart dear

he sage vooy Pome.

Conet who I should welverief it your sourtue the hold chulkiname and Rome's act;

But not this is so hiveeds w

And here is some text generated by the model when using words as tokens. Training this model took 43 minutes:

O Romeo, Romeo, of his mother,

He dried thence.

Second sickness the king,

I'll of your friends':

God and misusest.

KING government,

Shall, to Vienna.

POMPEY:

Does law disdains fair Lodowick?

LUCIO:

My proclaim'd: doth not answer, will not hap? me to meddle with within! maw, good:

the fall, the crown.

YORK:

'Twas then divine men?

If as humane

And one: not to him: VINCENTIO:

There's partake must be the matter:--Nurse, No, loves OF get entreat no baited

With but same it Paulina,

Make a take

From OVERDONE:

What's well; aid: begot the earth to offends suspicion! upon this crown and both do; grows to her I side;

The I can Down, learn to take promise keep to I:

No, heart

To OF last,

Definitively and condition,

I also had a Zoom meeting with John Eismeier from IT. He taught me how to use the Turing cluster. To log in, you must either be on school wi-fi or Global Protect VPN. Use the following commands in Powershell.

ssh [nakale@turing.wpi.edu](mailto:nakale@turing.wpi.edu) : to login

cat run\_modules.sb : to view contents of the run\_modules.sb file

sbatch run\_modules.sb : to submit run\_modules.sb as a job

nano run\_modules.sb : to edit the job

squeue -u nakale : to view the current job

In a .sb file, there are commands to run Python code and load dependencies.

module load python py-numpy : to load Python and Numpy

python \_\_\_.py : to run \_\_\_.py

The output is stored in a .log text file. Determine which file by running `ls -altr` after the job completes to see which .log file was most recently altered.

To transfer files to and from my local machine, I can either use Github or scp.