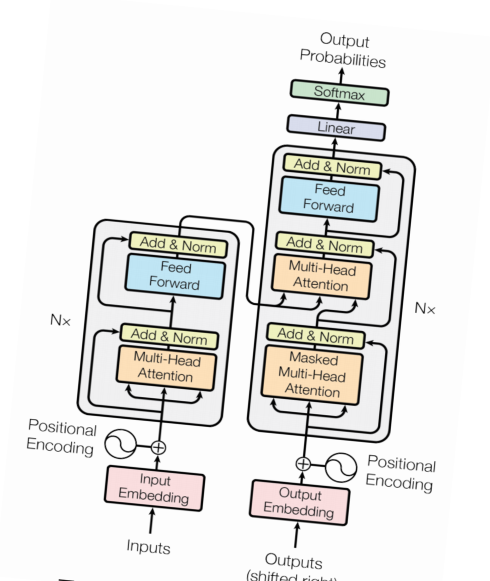
Informatics

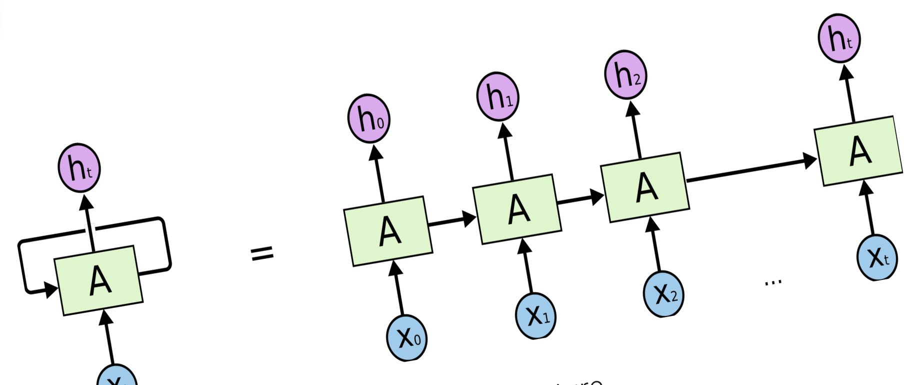
Neural Network Arena:
Investigating Long-Term Dependencies in Deep Models

Hannes Brantner
Computer Engineering

TU Wien Informatics
Institute of Computer Engineering
Cyber-Physical Systems Group
Supervisor: Univ.Prof. Dipl.-Ing. Dr.rer.nat. Radu Grosu
Contact: e01614466@student.tuwien.ac.at

GitHub repository


Transformer architecture

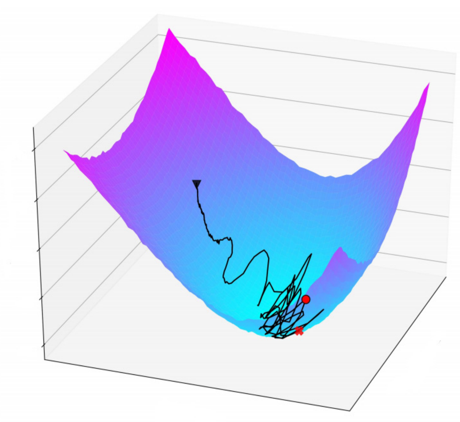

RNN architecture

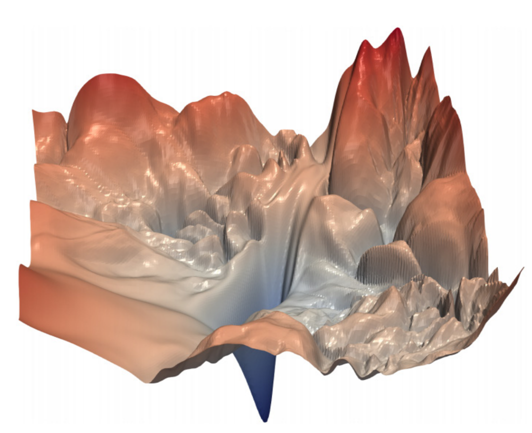
Implemented Models

- LSTM (Long Short-Term Memory)
- GRU (Gated Recurrent Unit)
- CT-RNN (Continuous-Time RNN)
- CT-GRU (Continuous-Time GRU)
- ODE-LSTM (Ordinary Differential Equation LSTM)
- NCP (Neural Circuit Policies)
- Unitary RNN
- Matrix Exponential Unitary RNN **new**
- Unitary NCP **new**
- Transformer
- Recurrent Network Augmented Transformer **new**
- Recurrent Network Attention Transformer **new**
- Memory Augmented Transformer **new**
- Differentiable Neural Computer
- Memory Cell **new**

Problem Statement

- implementation of a reusable benchmark suite to compare machine learning models used for sequence modeling
- benchmark suite should test the models for their capabilities to capture long-term dependencies and to model physical systems
- selection of state-of-the-art models should be implemented as well as possible outlined improvements
- thoroughful comparison of all implemented models using the benchmark suite
- all implemented models are Transformer or RNN (Recurrent Neural Network) architectures
- proof-of-concept design and implementation of a continuous-time memory cell architecture based on LTC Networks


loss surface and applied gradient descent procedure


rugged loss surface

Methodology

- extensive literature review in the domain of sequence modeling
- implementation of the benchmark suite and all the implemented models
- the benchmark suite was invoked three times on all models and the statistics of the invocation output were interpreted

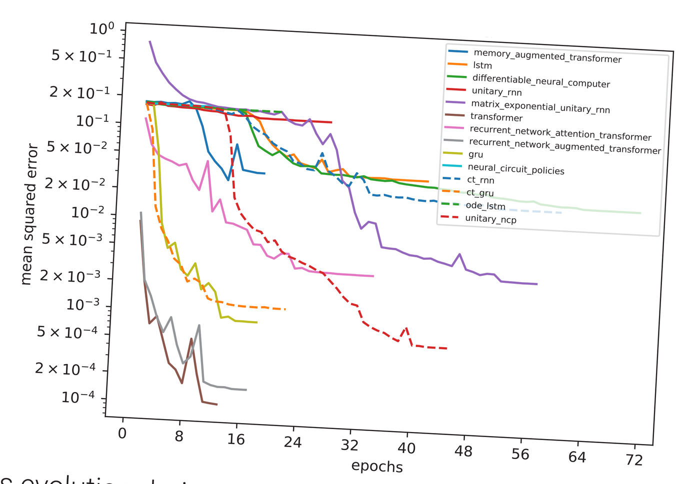
norm of W should be one to have a stable gradient if T-t is large, fulfilled by unitary matrices

$$\left\|\frac{\partial L}{\partial h_t}\right\|_2 \leq \left\|\frac{\partial L}{\partial h_T}\right\|_2 * \|W\|_{2,ind}^{T-t} * \prod_{k=t}^{T-1} \|diag(\sigma'(W*h_k + V*x_{k+1}))\|_{2,ind}$$

loss function gradient inequality for T>t and an RNN architecture

Motivation

- provide an objective comparison and overview of all implemented models on various sequence modeling tasks
- especially RNN architectures have difficulties of capturing long-term dependencies when being learned by gradient descent
- investigate which mechanism works best in RNN architectures to counteract this difficulty

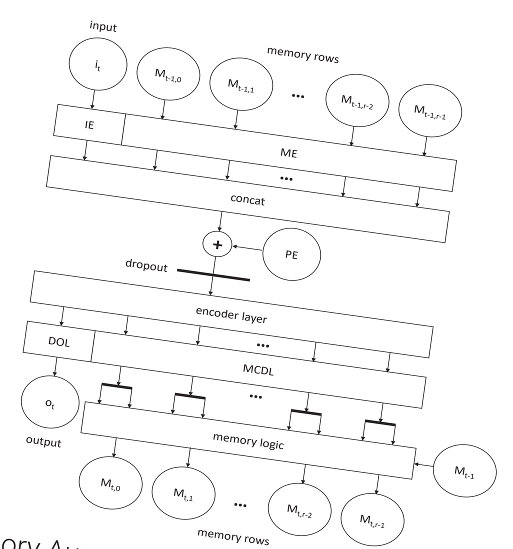


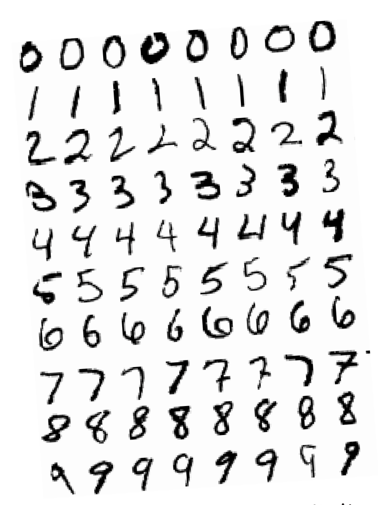
loss evolution during single training run for the Add Benchmark

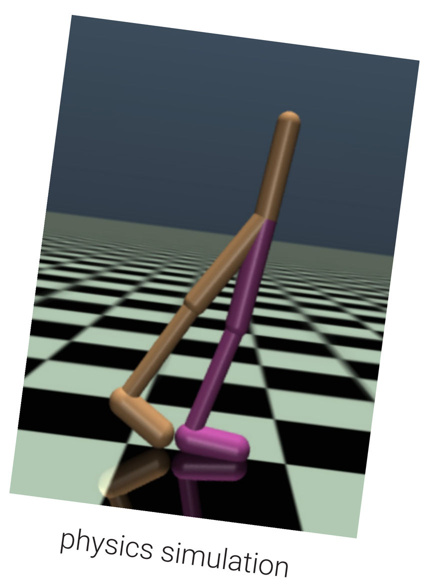
model	trainable parameters	training duration total [s]	training duration per epoch [s]	epochs	test mean squared error
transformer	21889	421.042 ± 62.125	33.175 ± 3.885	12.667 ± 0.577	0.000 ± 0.000
recurrent_network_augmented_transformer	3729	3369.947 ± 238.022	198.209 ± 6.384	17.000 ± 1.000	0.000 ± 0.000
gru	20241	423.332 ± 52.080	20.821 ± 0.322	20.333 ± 2.517	0.001 ± 0.000
ct_gru	19073	983.703 ± 123.086	48.464 ± 0.901	20.333 ± 2.887	0.001 ± 0.000
recurrent_network_attention_transformer	2349	10433.169 ± 3205.297	370.905 ± 13.507	28.000 ± 7.937	0.002 ± 0.002
differentiable_neural_computer	24774	6138.672 ± 4012.017	152.861 ± 2.424	40.333 ± 26.858	0.007 ± 0.011
ct_rnn	17025	9904.616 ± 1578.008	193.094 ± 1.600	51.333 ± 8.505	0.019 ± 0.003
matrix_exponential_unitary_rnn	17409	5956.370 ± 1220.646	115.400 ± 11.578	51.333 ± 6.028	0.022 ± 0.032
lstm	17217	589.247 ± 181.783	19.420 ± 0.162	30.333 ± 9.292	0.066 ± 0.085
unitary_rnn	2929	9694.727 ± 2483.450	278.442 ± 11.350	35.000 ± 9.849	0.094 ± 0.076
unitary_ncp	1885	3283.152 ± 1513.217	102.520 ± 0.625	32.000 ± 14.731	0.106 ± 0.092
memory_augmented_transformer	18066	3802.347 ± 1303.816	283.514 ± 8.116	13.333 ± 4.163	0.122 ± 0.083
neural_circuit_policies	1349	1007.356 ± 252.374	125.919 ± 0.414	8.000 ± 2.000	0.166 ± 0.001
ode_lstm	25537	748.046 ± 108.977	44.046 ± 0.415	17.000 ± 2.646	0.167 ± 0.002

Gradient Descent

- each model is just a parameterized function whose parameters are optimized by derivating a loss function that compares model output with expected output
- parameters are updated according to the loss function gradient which should not vanish or explode („walk down the hill")


Memory Augmented Transformer architecture


MNIST handwritten digits

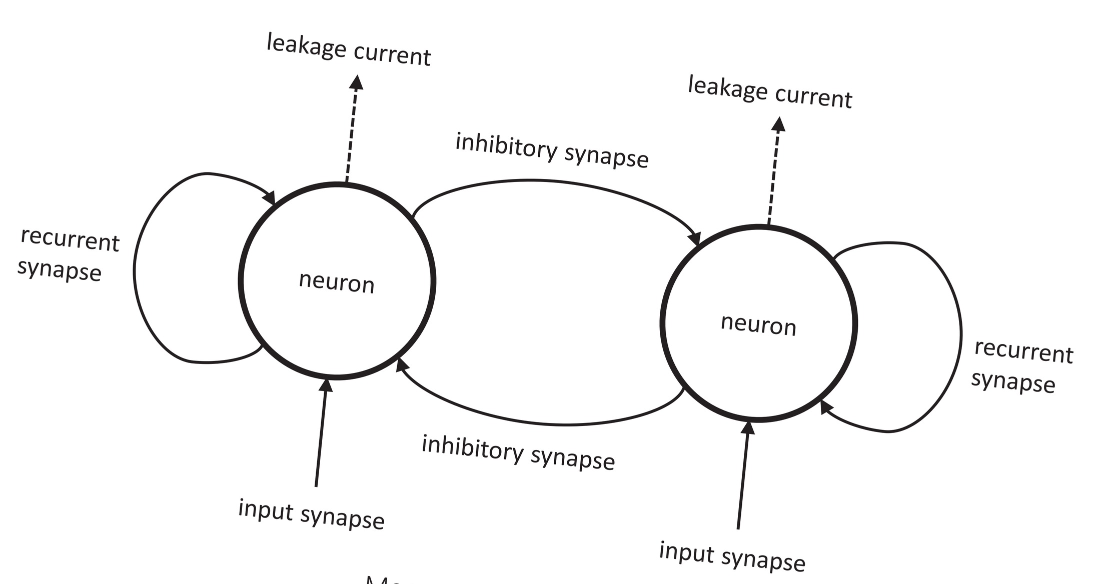

physics simulation

Results

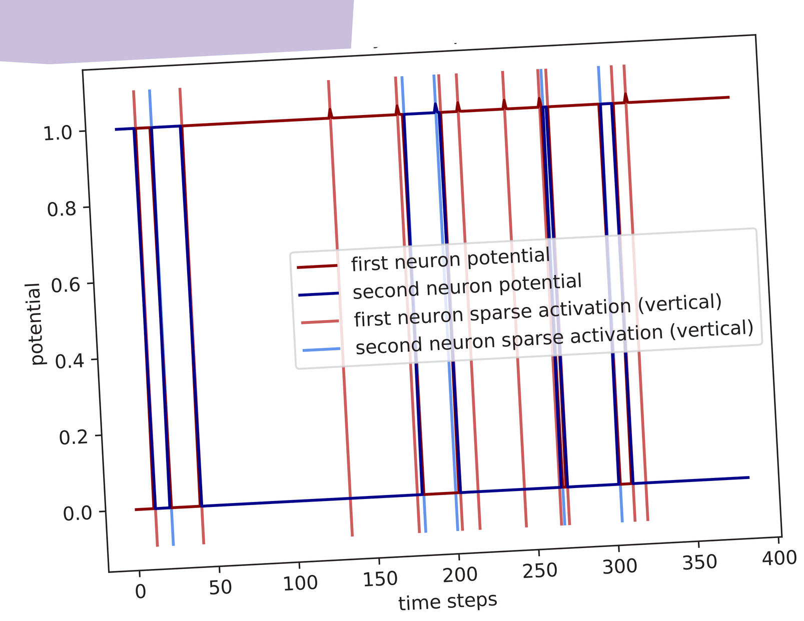
- state-of-the-art in efficient unitary matrix parameterization was improved by using an approximated matrix exponential
- continous-time memory cell architecture was successfully trained to store sparse activations
- positional encoding used in Transformer architectures shows deficiencies in tasks where exact positional information is required
- the newly introduced Memory Augmented Transformer architecture shows promising results in some tasks

Benchmark Suite Tasks

- Activity Benchmark - human activity classification of inertial sensor measurement data sequences
- Add Benchmark - adding up two marked numbers in a very long number sequence
- Walker Benchmark - predict the next state of a physics simulation given a sequence of previous simulation states
- Memory Benchmark - store a seen category exactly and recall it after seeing a sequence of irrelevant filler symbols
- MNIST Benchmark - digit classification using a sequence of MNIST handwritten digit image chunks
- Cell Benchmark - validates if sparse activations are correctly stored in the time-continuous memory cell architecture



Memory Cell architecture



successful Memory Cell operation