XLSTM

Extended Long short term memory by Sepp Hochreiter.

What is RNN?

A Recurrent Neural Network is a type of Neural Network that is designed for sequential data. RNN have connections that form directed cycles allowing them to maintain a memory of previous input & utilize this information when processing new inputs.

What is LSTM?

LSTM is a special type of RNN designed to overcome the vanishing gradient problem and effectively capture long-term dependencies in sequential data.

Memory cell } LSTM contain memory cells that can store information for long period.

Grating Mechanisms } LSTM uses three main gates to control the flow of information into and out of the memory cells.

1. **Forget Gate**:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

This gate decides what fraction of the previous cell state to forget.

2. **Input Gate**:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

This gate determines which new information to add to the cell state.

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

This is the candidate cell state, which contains the new information.

3. **Update Cell State**:

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$$

The cell state is updated by combining the old cell state (partially forgotten) and the new candidate cell state.

4. **Output Gate**:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

This gate determines the output of the LSTM cell.

$$h_t = o_t \cdot \tanh(C_t)$$

The hidden state is updated based on the cell state and the output gate.

What is GRU?

The Gated Recurrent unit, are similar to LSTM Network but a simpler structure. This simplicity can lead to faster training times & fewer computational resources while still maintaining performances comparable to LSTM.

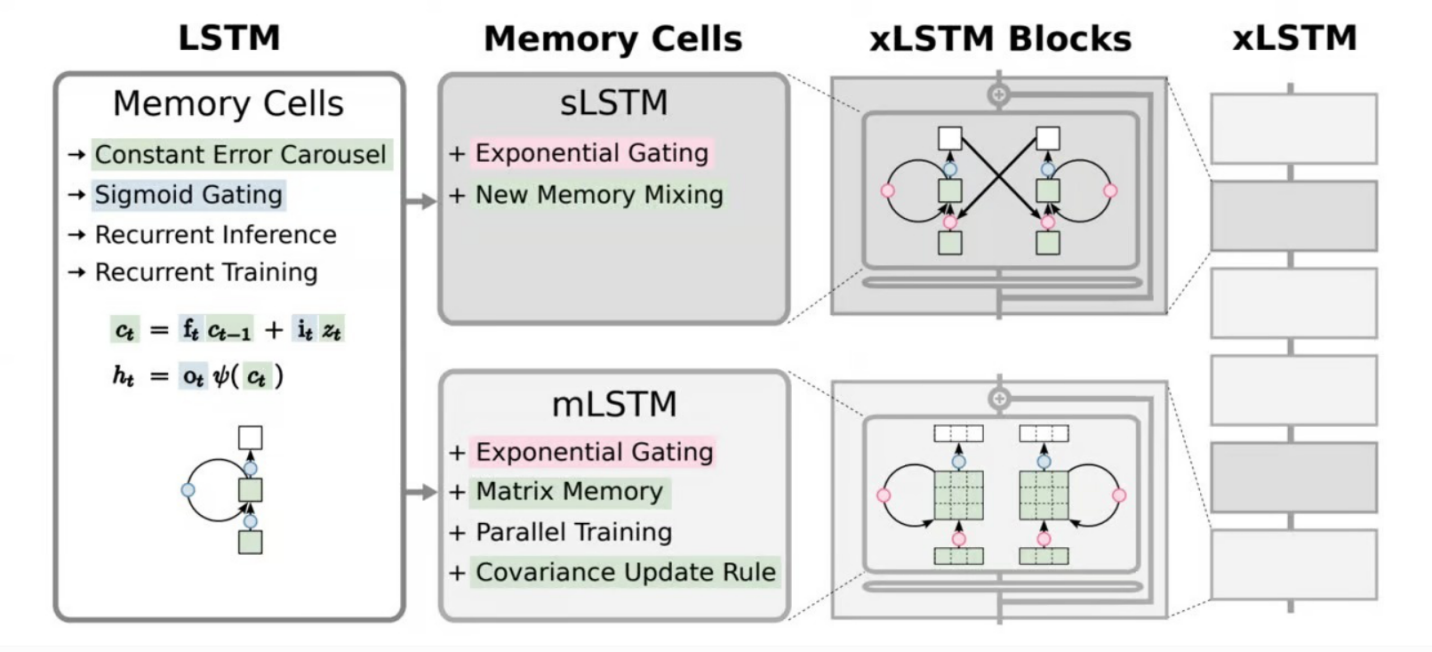
* No separate cell state.

* Simplified Grating Mechanism (Reset & update Gate)

Now, it's back

XLSTM ⟶ sLSTM (Scalar LSTM)
⟶ mLSTM (Matrix LSTM)

SLSTM & mLSTM is optionally joined together to create XLSTM block.



Due to Nature of LSTM, the problem can't be solved using one architecture

In LSTM, each hidden block relies on the output of previous block (sequential manner) it's called memory Mixing which is inefficient.

While, transformer quickly get the entire output with a Matrix Multiplication.

LSTM is not a Scalable to Sequential processing.

One of the key idea of LSTM is Gating, which was rediscovered & reinterpreted in Gated state space model and Mamba.

All Gate activation functions are sigmoid.

$$\sigma(x) = 1 / 1 + e^{-x}$$

earlier, LSTM incorporated constant error carousel along with structured gating Mechanisms for Vanishing gradient problem in RNN.

Limitation of LSTM

* Inability to revise Storage decision, Once information is Stored, traditional LSTMs have limited ability to revise or overwrite it based on new inputs.

* Limited Storage capacity, The scalar nature of LSTM memory cells restricts the volume and complexity of information that can be maintained.

* Lack of parallelizability, The inherent sequential processing between time steps prevents LSTMs from leveraging modern parallel computing architecture effectively.

Working XLSTM

* Exponential Gating, XLSTM introduces exponential gating with appropriate normalization and stabilization techniques which enhances the gating mechanism to address inability of revising storage decision.

The gates (input & forget) may use exponential activations, adjusted by a stabilizing factors to maintain numerical stability.

$$c_t = \mathrm{f}_t \; c_{t-1} + \mathrm{i}_t \; z_t \qquad \text{cell state} \quad (2)$$

$$h_t = \mathrm{o}_t \; \tilde{h}_t \,, \qquad \tilde{h}_t = \psi\left(c_t\right) \qquad \text{hidden state} \quad (3)$$

$$z_t = \varphi\left(\tilde{z}_t\right) \,, \qquad \tilde{z}_t = \boldsymbol{w}_z^\top \boldsymbol{x}_t + r_z h_{t-1} + b_z \qquad \text{cell input} \quad (4)$$

$$\mathrm{i}_t = \sigma\left(\tilde{\mathrm{i}}_t\right) \,, \qquad \tilde{\mathrm{i}}_t = \boldsymbol{w}_\mathrm{i}^\top \boldsymbol{x}_t + r_\mathrm{i} \, h_{t-1} + b_\mathrm{i} \qquad \text{input gate} \quad (5)$$

$$\mathrm{f}_t = \sigma\left(\tilde{\mathrm{f}}_t\right) \,, \qquad \tilde{\mathrm{f}}_t = \boldsymbol{w}_\mathrm{f}^\top \boldsymbol{x}_t + r_\mathrm{f} \, h_{t-1} + b_\mathrm{f} \qquad \text{forget gate} \quad (6)$$

$$\mathrm{o}_t = \sigma\left(\tilde{\mathrm{o}}_t\right) \,, \qquad \tilde{\mathrm{o}}_t = \boldsymbol{w}_\mathrm{o}^\top \boldsymbol{x}_t + r_\mathrm{o} \, h_{t-1} + b_\mathrm{o} \qquad \text{output gate} \quad (7)$$

Sigmoid.

} LSTM Grating System

$$\boldsymbol{C}_t = \mathrm{f}_t \; \boldsymbol{C}_{t-1} + \mathrm{i}_t \; \boldsymbol{v}_t \, \boldsymbol{k}_t^\top \qquad \text{cell state} \quad (19)$$

$$\boldsymbol{n}_t = \mathrm{f}_t \; \boldsymbol{n}_{t-1} + \mathrm{i}_t \; \boldsymbol{k}_t \qquad \text{normalizer state} \quad (20)$$

$$\boldsymbol{h}_t = \mathbf{o}_t \odot \tilde{\boldsymbol{h}}_t \,, \qquad \tilde{\boldsymbol{h}}_t = \boldsymbol{C}_t \; \boldsymbol{q}_t \; / \; \max\left\{ \left| \boldsymbol{n}_t^\top \; \boldsymbol{q}_t \right| , 1 \right\} \qquad \text{hidden state} \quad (21)$$

$$\boldsymbol{q}_t = \boldsymbol{W}_q \, \boldsymbol{x}_t + \boldsymbol{b}_q \qquad \text{query input} \quad (22)$$

$$\boldsymbol{k}_t = \frac{1}{\sqrt{d}} \boldsymbol{W}_k \, \boldsymbol{x}_t + \boldsymbol{b}_k \qquad \text{key input} \quad (23)$$

$$\boldsymbol{v}_t = \boldsymbol{W}_v \, \boldsymbol{x}_t + \boldsymbol{b}_v \qquad \text{value input} \quad (24)$$

$$\mathrm{i}_t = \exp\left(\tilde{\mathrm{i}}_t\right) \,, \qquad \tilde{\mathrm{i}}_t = \boldsymbol{w}_\mathrm{i}^\top \boldsymbol{x}_t + b_\mathrm{i} \qquad \text{input gate} \quad (25)$$

$$\mathrm{f}_t = \sigma\left(\tilde{\mathrm{f}}_t\right) \text{ OR } \exp\left(\tilde{\mathrm{f}}_t\right) , \qquad \tilde{\mathrm{f}}_t = \boldsymbol{w}_\mathrm{f}^\top \boldsymbol{x}_t + b_\mathrm{f} \qquad \text{forget gate} \quad (26)$$

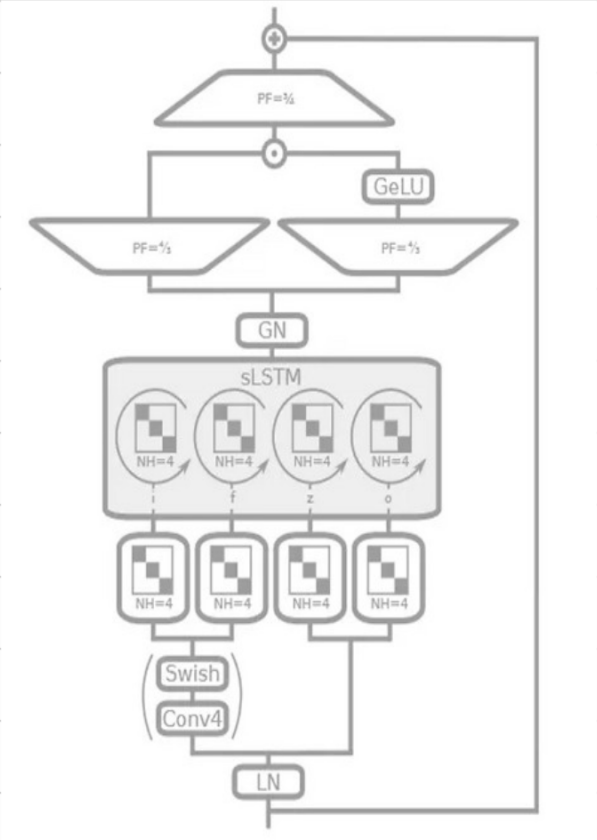
$$\mathbf{o}_t = \sigma\left(\tilde{\mathbf{o}}_t\right) \,, \qquad \tilde{\mathbf{o}}_t = \boldsymbol{W}_\mathbf{o} \, \boldsymbol{x}_t + \boldsymbol{b}_\mathbf{o} \qquad \text{output gate} \quad (27)$$

} XLSTM Grating

Exponential function.

* Modified memory Structure,

(i) SLSTM : Scalar LSTM with Exponential Grating & Memory Mixing



Enhanced version of LSTM with Scalar or sequences level updates.

Includes improvement to the gating Mechanisms (such as exponential gating) and optimization of memory structure.

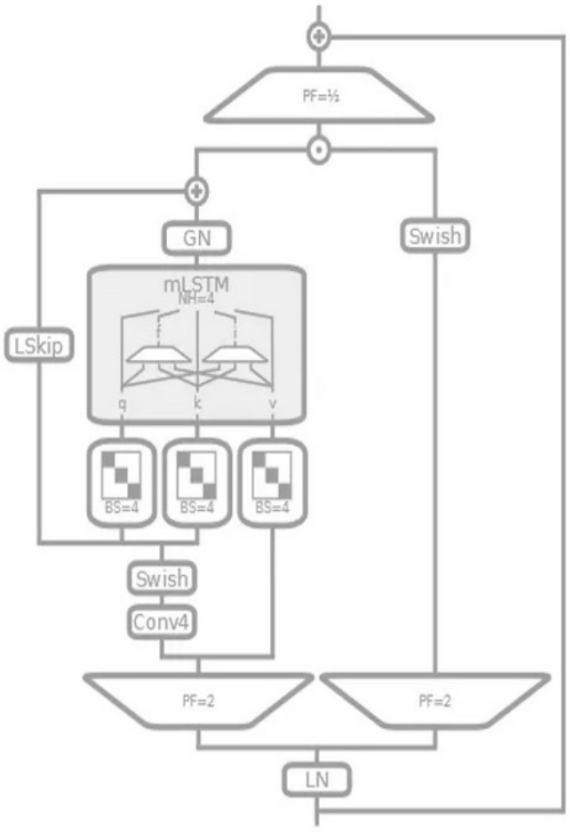
SLSTM, utilizes the post up projections.

* Exponential Grating,
SLSTM incorporates exponential activation functions for the Input & forget gates. Providing more flexible control over information flow.

* Normalization & Stabilization, To prevent numerical instabilities, SLSTM Introduces a normalizer state that tracks the product of input & future forget gates.

* Memory Mixing, SLSTM supports Multiple memory cell and allows for memory Mixing via recurrent connections. Enchances ability to extract complex pattern.

(iii) MLSTM : Matrix LSTM with Enhanced Storage capacities



LSTM variant featuring matrix memory, which allows it to process and store more information in parallel.

It uses matrices instead of scalars to store the LSTM cell states.

MLSTM utilizes prep-up projections.

* Matrix memory, MLSTM utilizes a matrix memory instead of a scalar memory cell, increasing storage capacity and enabling more efficient information retrieval.

* covariance update rule; MLSTM employs a covariance update rule to store and retrieve key-value pairs efficiently.

* parallelizability, By eliminating memory mixing, MLSTM achieves full parallelizability allowing for efficient computation on modern hardware accelerator.

These variants SLSTM & MLSTM can be integrated into residual block architectures forming xLSTM Blocks.

	Context Sentsitive		Deterministic Context Free		Regular					
	Bucket Sort	Missing Duplicate	Mod Arithmetic (w Brackets)	Solve Equation	Cycle Nav	Even Pairs	Mod Arithmetic (w/o Brackets)	Parity	Majority	Majority Count
Llama	0.92 ± 0.02	0.08 ± 0.0	0.02 ± 0.0	0.02 ± 0.0	0.04 ± 0.01	1.0 ± 0.0	0.03 ± 0.0	0.03 ± 0.01	0.37 ± 0.01	0.13 ± 0.0
Mamba	0.69 ± 0.0	0.15 ± 0.0	0.04 ± 0.01	0.05 ± 0.02	0.86 ± 0.04	1.0 ± 0.0	0.05 ± 0.02	0.13 ± 0.02	0.69 ± 0.01	0.45 ± 0.03
Retention	0.13 ± 0.01	0.03 ± 0.0	0.03 ± 0.0	0.03 ± 0.0	0.05 ± 0.01	0.51 ± 0.07	0.04 ± 0.0	0.05 ± 0.01	0.36 ± 0.0	0.12 ± 0.01
Hyena	0.3 ± 0.02	0.06 ± 0.02	0.05 ± 0.0	0.02 ± 0.0	0.06 ± 0.01	0.93 ± 0.07	0.04 ± 0.0	0.04 ± 0.0	0.36 ± 0.01	0.18 ± 0.02
RWKV-4	0.54 ± 0.0	0.21 ± 0.01	0.06 ± 0.0	0.07 ± 0.0	0.13 ± 0.0	1.0 ± 0.0	0.07 ± 0.0	0.06 ± 0.0	0.63 ± 0.0	0.13 ± 0.0
RWKV-5	0.49 ± 0.04	0.15 ± 0.01	0.08 ± 0.0	0.08 ± 0.0	0.26 ± 0.05	1.0 ± 0.0	0.15 ± 0.02	0.06 ± 0.03	0.73 ± 0.01	0.34 ± 0.03
RWKV-6	0.96 ± 0.0	0.23 ± 0.06	0.09 ± 0.01	0.09 ± 0.02	0.31 ± 0.14	1.0 ± 0.0	0.16 ± 0.0	0.22 ± 0.12	0.76 ± 0.01	0.24 ± 0.01
LSTM (Block)	0.99 ± 0.0	0.15 ± 0.0	0.76 ± 0.0	0.5 ± 0.05	0.97 ± 0.03	1.0 ± 0.0	0.91 ± 0.09	1.0 ± 0.0	0.58 ± 0.02	0.27 ± 0.0
LSTM	0.94 ± 0.01	0.2 ± 0.0	0.72 ± 0.04	0.38 ± 0.05	0.93 ± 0.07	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	0.82 ± 0.02	0.33 ± 0.0
xLSTM[0:1]	0.84 ± 0.08	0.23 ± 0.01	0.57 ± 0.09	0.55 ± 0.09	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	0.75 ± 0.02	0.22 ± 0.0
xLSTM[1:0]	0.97 ± 0.0	0.33 ± 0.22	0.03 ± 0.0	0.03 ± 0.01	0.86 ± 0.01	1.0 ± 0.0	0.04 ± 0.0	0.04 ± 0.01	0.74 ± 0.01	0.46 ± 0.0
xLSTM[1:1]	0.7 ± 0.21	0.2 ± 0.01	0.15 ± 0.06	0.24 ± 0.04	0.8 ± 0.03	1.0 ± 0.0	0.6 ± 0.4	1.0 ± 0.0	0.64 ± 0.04	0.5 ± 0.0

Figure 4: Test of xLSTM's exponential gating with memory mixing. Results are given by the scaled accuracy of different models at solving formal language tasks, of which some require state tracking. The different tasks are grouped by the Chomsky hierarchy.

↑ Comparative study

Application of xLSTM

1. Language Modelling.
2. Time Series Analysis
3. Machine Translation.
4. Speech Recognition & Generation.