Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling

Authors: Junyoung Chung, Caglar Gulcehre, KyungHyun Cho and Yoshua Bengio

Presenter: Yu-Wei Lin

Background: Recurrent Neural Network

- Traditional RNNs encounter many difficulties when training long-term dependencies.
    - The vanishing gradient problem/exploding gradient problem.
- There are two approach to solve this problem:
    - Design use new methods to improve or replace stochastic gradient descent (SGD) method
    - Design more sophisticated recurrent unit, such as LSTM, GRU.
- The paper focus on the performance of LSTM and GRU

Research Question

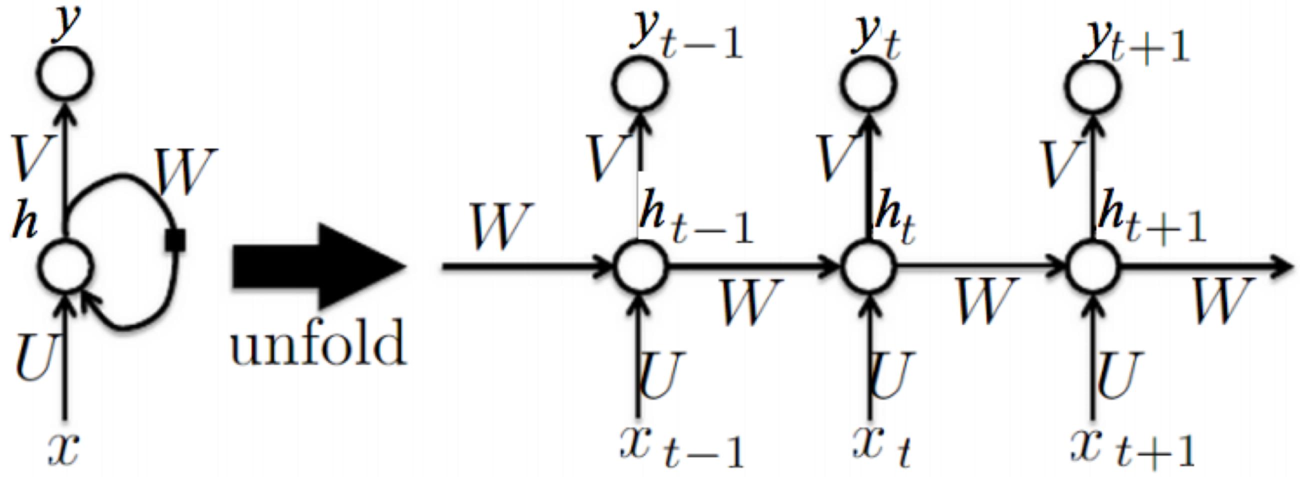
- Do RNNs using recurrent units with gates outperform traditional RNNs?

- Does the LSTM or the GRU perform better as a recurrent unit for tasks such as music and speech prediction?

Approach

- Empirically evaluated recurrent neural networks (RNN) with three widely used recurrent units
  - Traditional tanh unit
  - Long short-term memory (LSTM) unit
  - Gated recurrent unit (GRU)
- The evaluation focused on the task of sequence modeling
  - Dataset: (1) polyphonic music data (2) raw speech signal data.
- Compare their performances using a log-likelihood loss function

Recurrent Neural Networks



- x_t is the input at time step t.
- h_t is the hidden state at time step t.
- h_t is calculated based on the previous hidden state and the input at the current step:
  - $h_t = \int(Ux_t + Wh_{t-1})$
- o_t is the output at step t.
  - E.g., if we wanted to predict the next word in a sentence it would be a vector of probabilities across our vocabulary

Main concept of LSTM

- Closer to how humans process information
  - Control how much of the previous hidden state to forget
  - Control how much of new input to take
- The notion is proposed by Hochreiter and Schmidhuber 1997

Long Short-Term Memory (LSTM)

- Forget Gate (gate 0, forget past)

$$f_t = \sigma\left(W^{(f)} x_t + U^{(f)} h_{t-1}\right)$$

- Input Gate (current cell matters)

$$i_t = \sigma\left(W^{(i)} x_t + U^{(i)} h_{t-1}\right)$$

- New memory cell

$$\tilde{c}_t = \tanh\left(W^{(c)} x_t + U^{(c)} h_{t-1}\right)$$

- Final memory cell

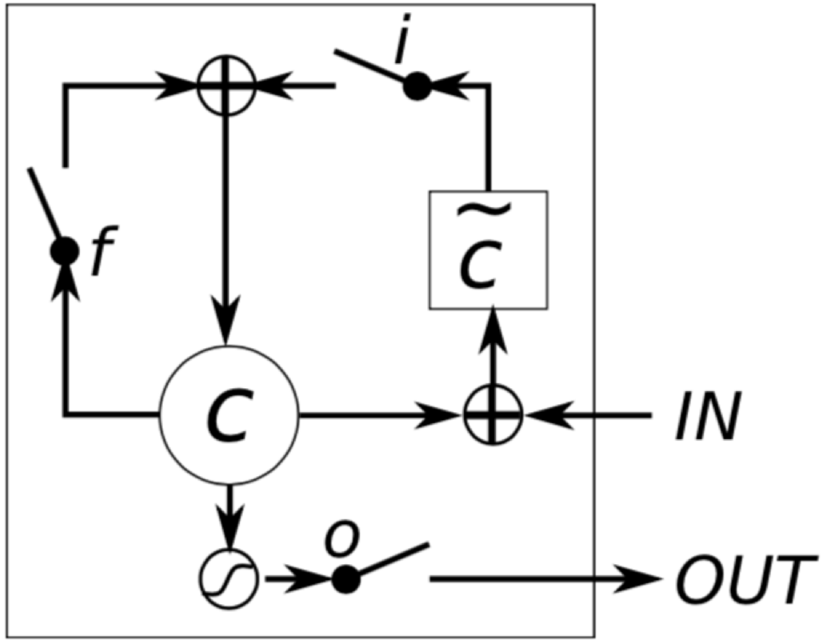
$$c_t = f_t \circ c_{t-1} + i_t \circ \tilde{c}_t$$

- Output Gate (how much cell is exposed)

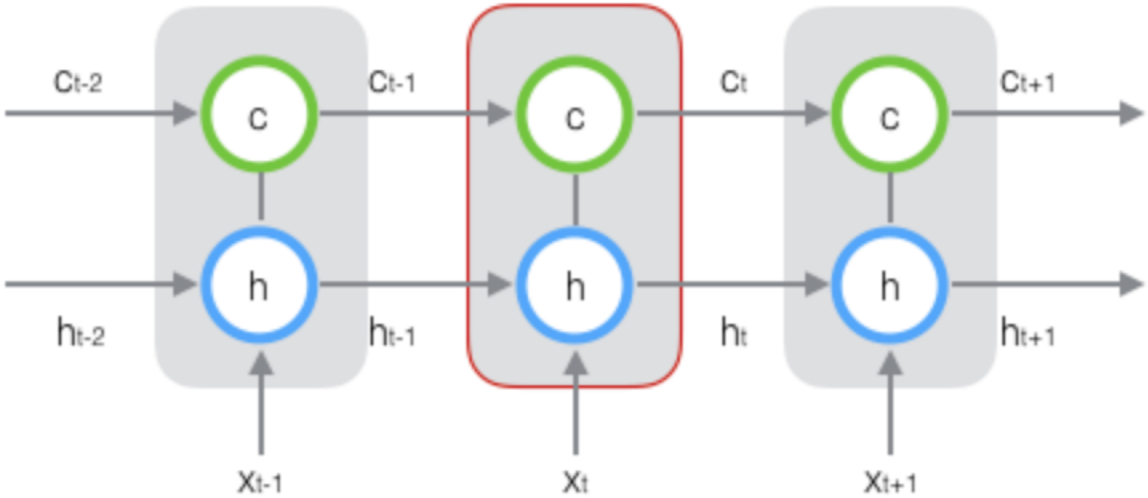
$$o_t = \sigma\left(W^{(o)} x_t + U^{(o)} h_{t-1}\right)$$

- Final hidden state

$$h_t = o_t \circ \tanh(c_t)$$



(a) Long Short-Term Memory



Main concept of Gated Recurrent Unit (GRU)

- LSTMs work well but unnecessarily complicated
- GRU is a variant of LSTM
- Approach:
  - Combine the forgetting gate and input gate in LSTM into a single "Update Gate".
  - Combine the Cell State and Hidden State.
- Computationally less expensive
  - less parameters, less complex structure
- Performance is as good as LSTM

Gated Recurrent Unit (GRU)

- Reset gate: determines how to combine the new input with the previous memory

$$r_t = \sigma\left(W^{(r)}x_t + U^{(r)}h_{t-1}\right)$$

- Update gate: decides how much of the previous memory to keep around

$$z_t = \sigma\left(W^{(z)}x_t + U^{(z)}h_{t-1}\right)$$

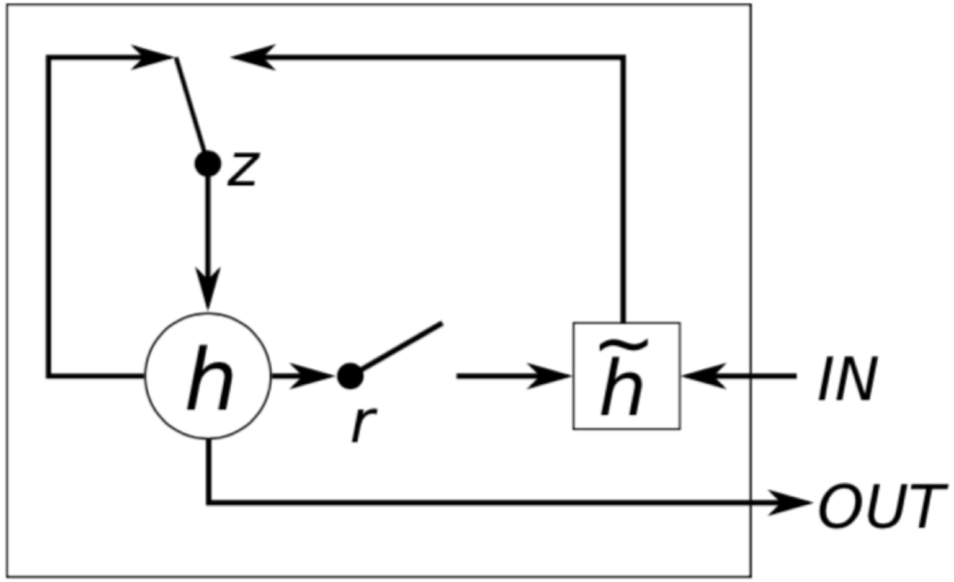
- Candidate hidden layer

$$\tilde{h}_t = \tanh\left(Wx_t + r_t \circ Uh_{t-1}\right)$$

- Final memory at time step combines current and previous time steps:

$$h_t = z_t \circ h_{t-1} + (1 - z_t) \circ \tilde{h}_t$$

- If we set the reset to all 1's and update gate to all 0's, the model is the same as plain RNN model



(b) Gated Recurrent Unit

Advantage of LSTM/GRU

- It is easy for each unit to remember the existence of a specific feature in the input stream for a long series of steps.

- The shortcut paths allow the error to be back-propagated easily without too quickly vanishing
    - Error pass through multiple bounded nonlinearities, which reduces the likelihood of the vanishing gradient.

LSTMs v.s. GRU

LSTM	GRU
Three gates	Two gates
Control the exposure of memory content (cell state)	Expose the entire cell state to other units in the network
Has separate input and forget gates	Performs both of these operations together via update gate
More parameters	Fewer parameters

Model

- The authors built models for each of their three test units (LSTM, GRU, tanh) along the following criteria:
  - Similar numbers of parameters in each network, for fair comparison
  - RMSProp optimization
  - Learning rate chosen to maximize the validation performance from 10 different points from -12 to -6
- The models are tested across four music datasets and two speech datasets.

Unit	# of Units	# of Parameters
Polyphonic music modeling		
LSTM	36	$\approx 19.8 \times 10^3$
GRU	46	$\approx 20.2 \times 10^3$
tanh	100	$\approx 20.1 \times 10^3$
Speech signal modeling		
LSTM	195	$\approx 169.1 \times 10^3$
GRU	227	$\approx 168.9 \times 10^3$
tanh	400	$\approx 168.4 \times 10^3$

Table 1: The sizes of the models tested in the experiments.

Task

- Music dataset
  - Input: the sequence of vectors
  - Output: predict the next time step of the sequence
- Speech signal dataset:
  - Look at 20 consecutive samples to predict the following 10 consecutive samples
  - Input: one-dimensional raw audio signal at each time step
  - Output: the next time 10 consecutive step of the sequence

Result - average negative log-likelihood

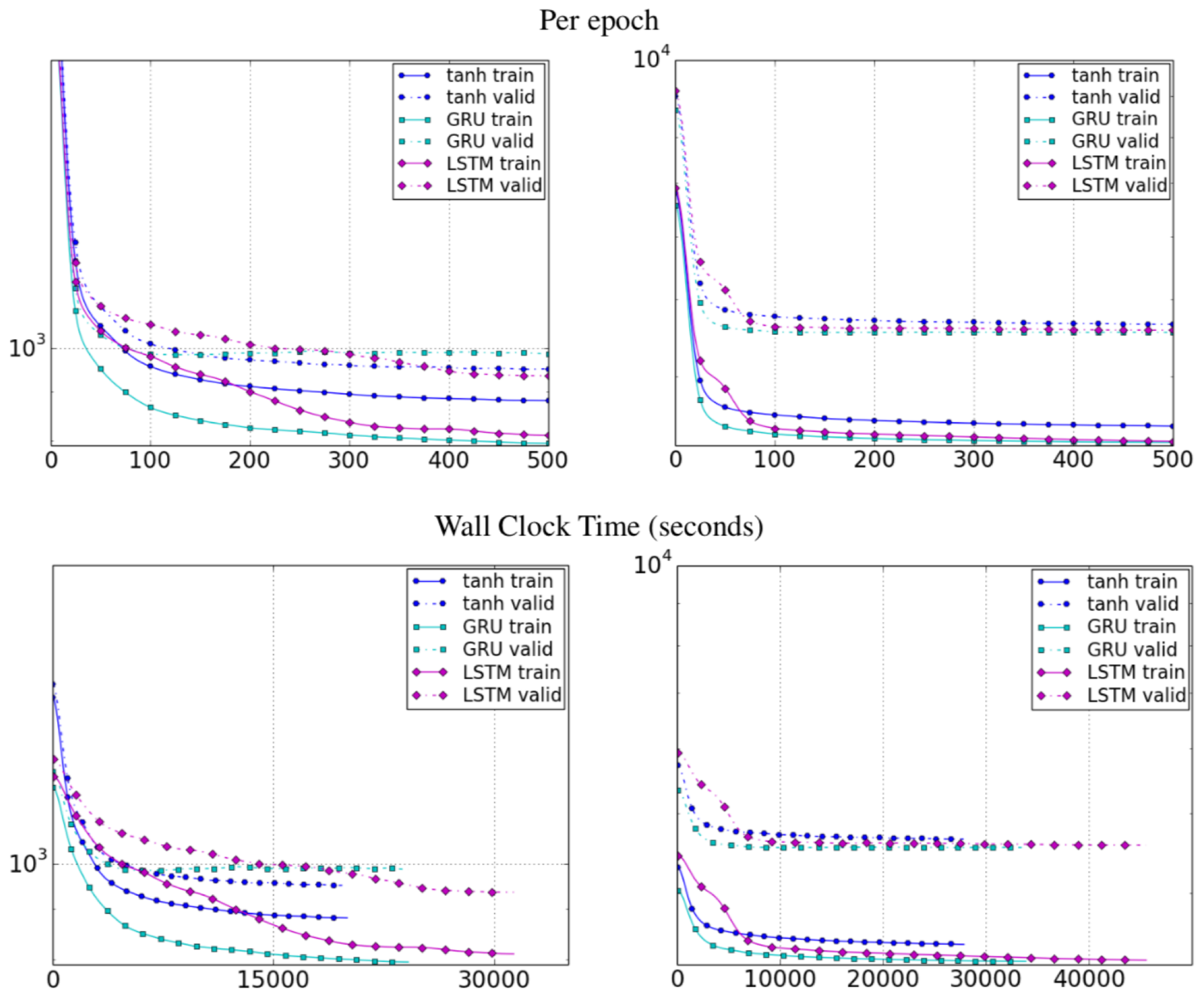
- Music datasets
  - The GRU-RNN outperformed all the others (LSTM-RNN and tanh-RNN)
  - All the three models performed closely to each other
- Ubisoft datasets
  - the RNNs with the gating units clearly outperformed the more traditional tanh-RNN

			tanh	GRU	LSTM
Music Datasets	Nottingham	train	3.22	2.79	3.08
		test	**3.13**	3.23	3.20
	JSB Chorales	train	8.82	6.94	8.15
		test	9.10	**8.54**	8.67
	MuseData	train	5.64	5.06	5.18
		test	6.23	**5.99**	6.23
	Piano-midi	train	5.64	4.93	6.49
		test	9.03	**8.82**	9.03
Ubisoft Datasets	Ubisoft dataset A	train	6.29	2.31	1.44
		test	6.44	3.59	**2.70**
	Ubisoft dataset B	train	7.61	0.38	0.80
		test	7.62	**0.88**	1.26

Table 2: The average negative log-probabilities of the training and test sets.

Result - Learning curves

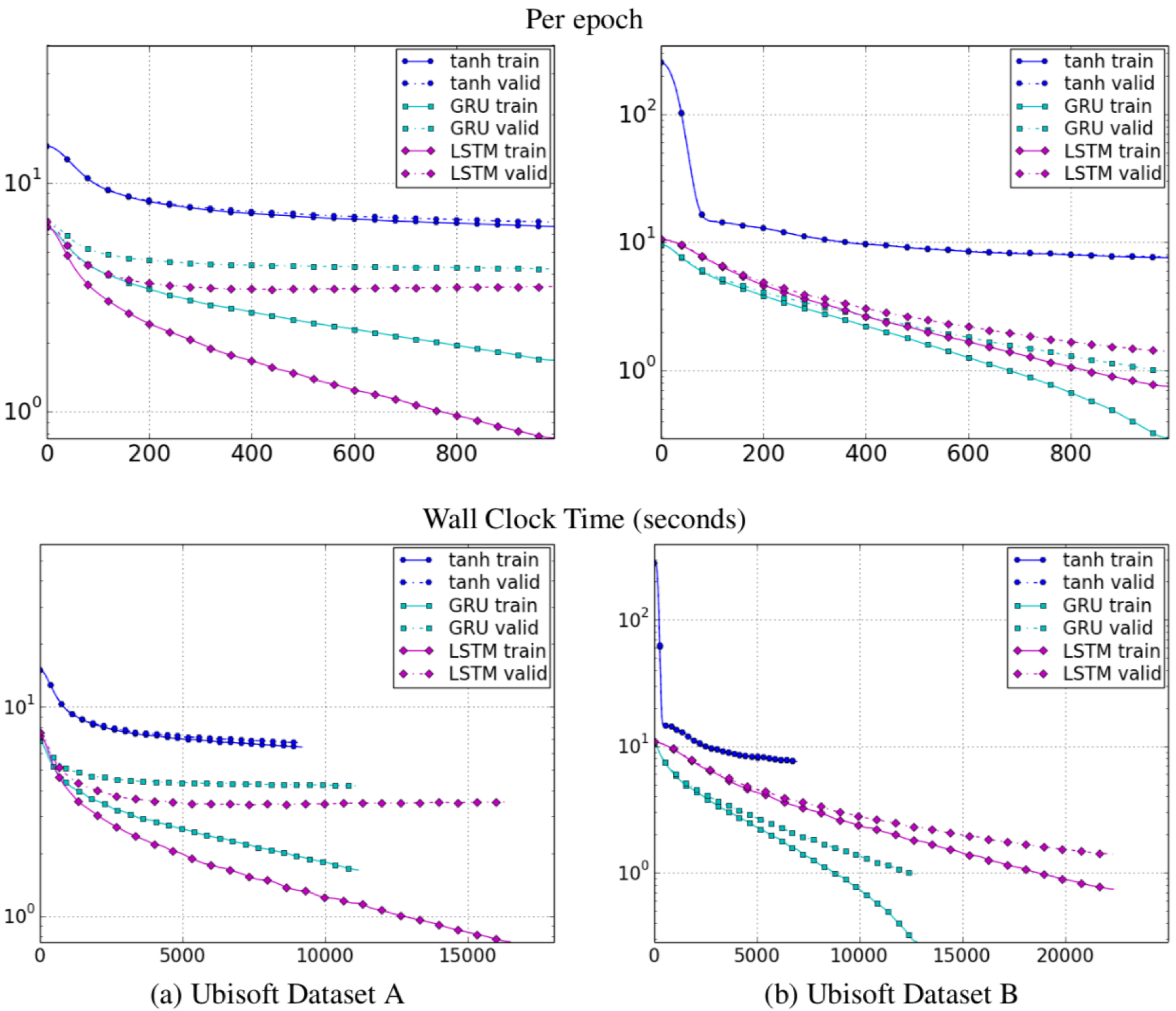
- Learning curves for training and validation sets of different types of units
  - Top: number of iterations
  - Bottom: the wall clock time
- y-axis: the negative-log likelihood of the model shown in log-scale.
- GRU-RNN makes faster progress in terms of both the number of updates and actual CPU time.



(a) Nottingham Dataset (b) MuseData Dataset

Result - Learning curves Cont'd

- The gated units (LSTM and GRU) well outperformed the tanh unit
- The GRU-RNN once again producing the best results



Take ways

- Music datasets
  - The GRU-RNN reached the inching better performance.
  - All of the models performed relatively closely
- Speech datasets
  - The gated units well outperformed the tanh unit
  - The GRU-RNN produce the best results both in terms of accuracy and training time.
- Gated units are superior to recurrent neural networks (RNNs)
- The performance of the two gated units (LTM and RGU) cannot be clearly distinguished.

Thank you !