Towards Binary-Valued Gates for Robust LSTM Training

Zhuohan Li, Di He, Fei Tian, Wei Chen, Tao Qin, Liwei Wang, Tie-Yan Liu

Peking University & Microsoft Research Asia

ICML | 2018

Long Short-Term Memory (LSTM) RNN

- $h_t, c_t = \mathrm{LSTM}(h_{t-1}, c_{t-1}, x_t)$

- $f_t = \sigma\left(W_{xf}x_t + W_{hf}h_{t-1} + b_f\right)$
- $i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i)$
- $g_t = \tanh\left(W_{xg}x_t + W_{hg}h_{t-1} + b_g\right)$
- $o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o)$
- $c_t = f_t \odot c_{t-1} + i_t \odot g_t$
- $h_t = o_t \odot \tanh(c_t)$

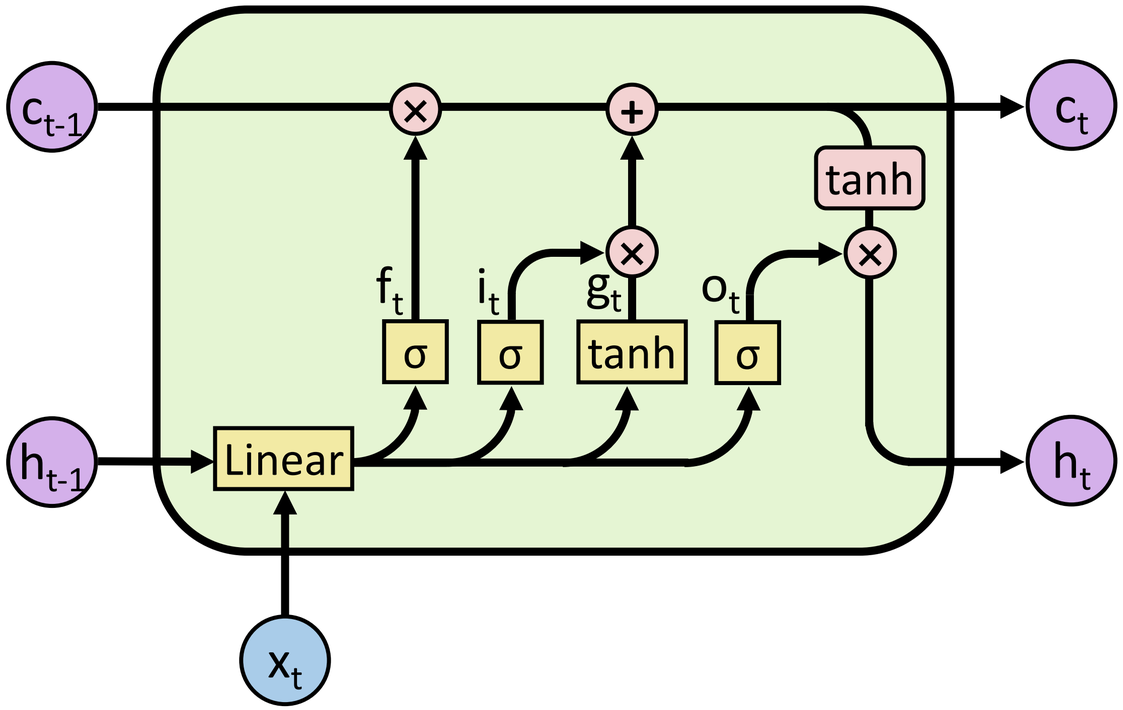


Figure credit to: Christopher Olah, "Understanding LSTM Networks"

Long Short-Term Memory (LSTM) RNN

- $h_t, c_t = \mathrm{LSTM}(h_{t-1}, c_{t-1}, x_t)$

- $f_t = \sigma\left(W_{xf} x_t + W_{hf} h_{t-1} + b_f\right)$
- $i_t = \sigma(W_{xi} x_t + W_{hi} h_{t-1} + b_i)$
- $g_t = \tanh\left(W_{xg} x_t + W_{hg} h_{t-1} + b_g\right)$
- $o_t = \sigma(W_{xo} x_t + W_{ho} h_{t-1} + b_o)$
- $c_t = f_t \odot c_{t-1} + i_t \odot g_t$
- $h_t = o_t \odot \tanh(c_t)$

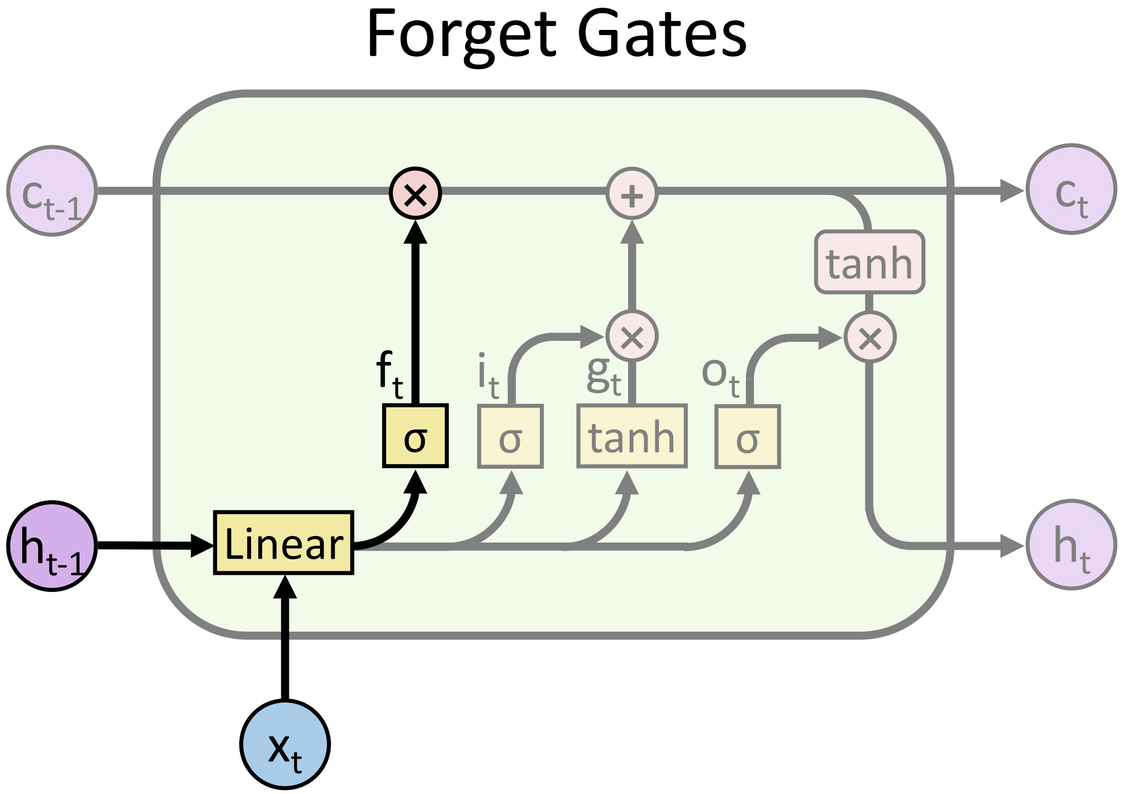


Figure credit to: Christopher Olah, "Understanding LSTM Networks"

Long Short-Term Memory (LSTM) RNN

- $h_t, c_t = \mathrm{LSTM}(h_{t-1}, c_{t-1}, x_t)$

- $f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f)$
- $i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i)$
- $g_t = \tanh(W_{xg}x_t + W_{hg}h_{t-1} + b_g)$
- $o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o)$
- $c_t = f_t \odot c_{t-1} + i_t \odot g_t$
- $h_t = o_t \odot \tanh(c_t)$

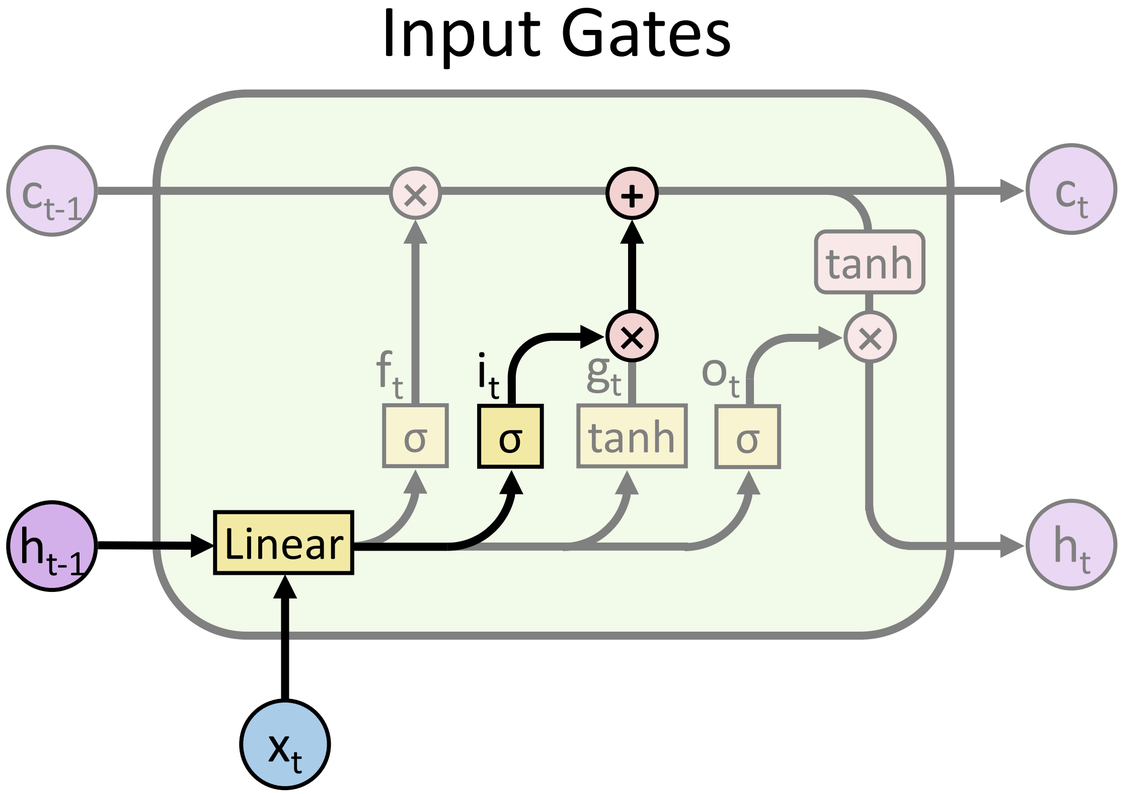


Figure credit to: Christopher Olah, "Understanding LSTM Networks"

Long Short-Term Memory (LSTM) RNN

- $h_t, c_t = \mathrm{LSTM}(h_{t-1}, c_{t-1}, x_t)$

- $f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f)$
- $i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i)$
- $g_t = \tanh(W_{xg}x_t + W_{hg}h_{t-1} + b_g)$
- $o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o)$
- $c_t = f_t \odot c_{t-1} + i_t \odot g_t$
- $h_t = o_t \odot \tanh(c_t)$

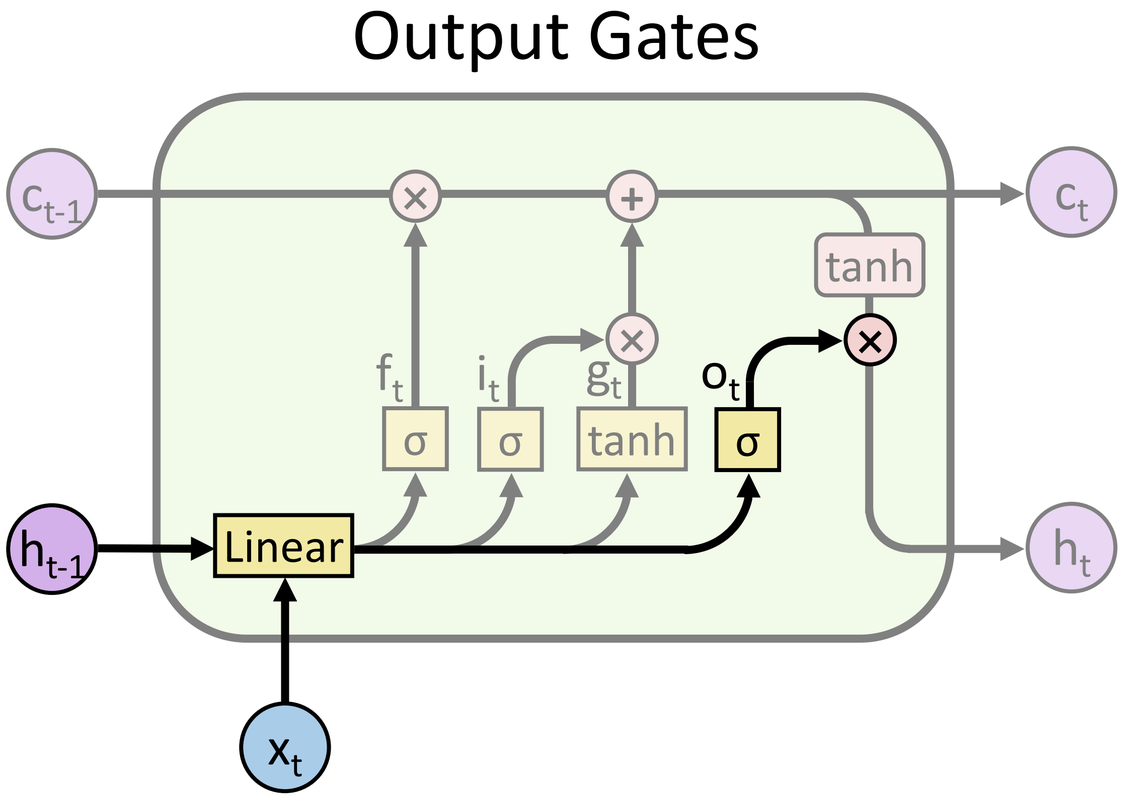
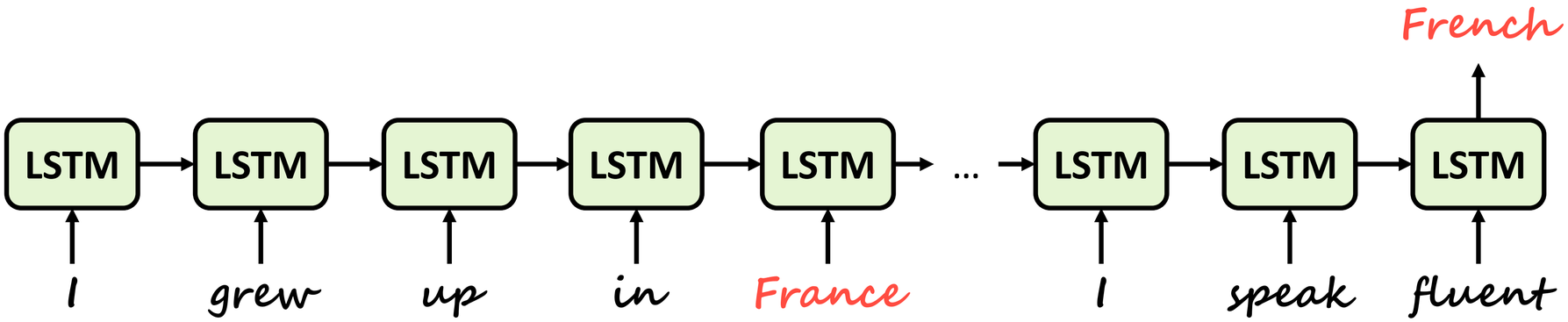


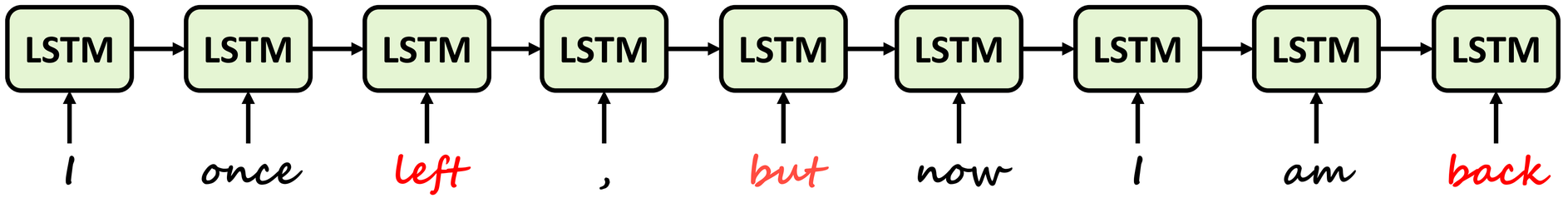
Figure credit to: Christopher Olah, "Understanding LSTM Networks"

Example: Input Gates & Forget Gates



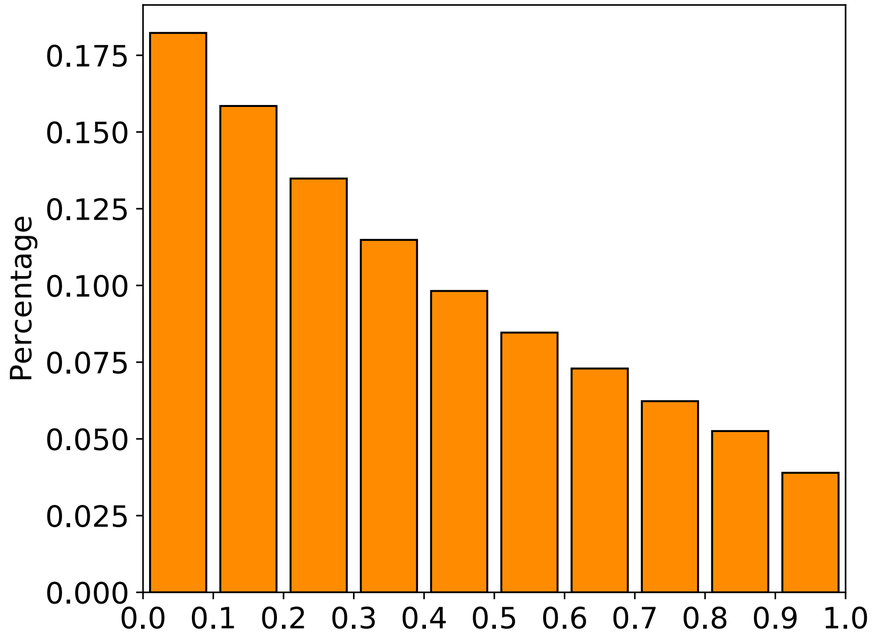
- When the LSTM sees "France", the **input gate** will open and the LSTM will remember the information
- At the subsequent timesteps, the **forget gates** will also be open (take value 1) to keep the information. Finally the LSTM will use this information to predict word "French"

Example: Input Gates & Forget Gates

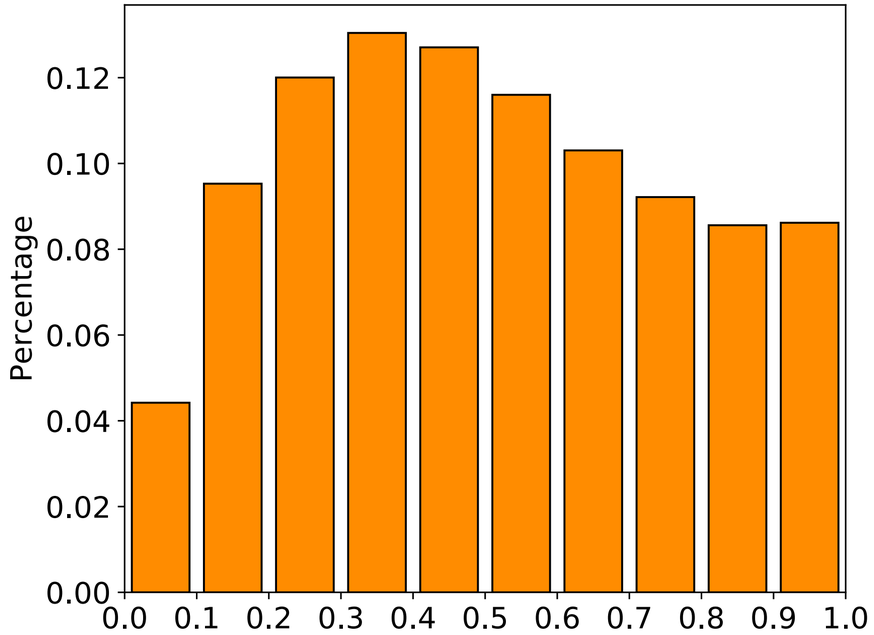


- When the LSTM sees "but" and "back", the **forget gates** should be closed (take value 0) to forget the information of "left"

Histograms of Gate Distributions in LSTM



LSTM Input gates



LSTM Forget gates

Based on the gate outputs of the first-layer LSTM in the decoder from 10000 sentence pairs IWSLT14 German→English training sets

Training LSTM Gates Towards Binary Values

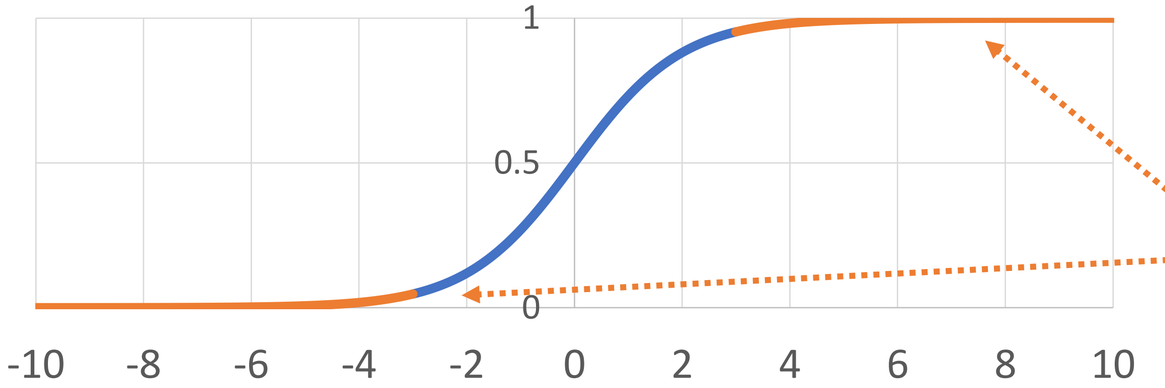
Push the **gate values** to the **boundary** of range (0, 1)

Well aligns with **the original purpose** of gates: to get the information in or skip by "opening" or "closing"

Ready for further compression by pushing the activation function to be binarized

Enables **better generalization**

Ready for Further Compression & Better Generalization



Saturation area of Sigmoid function

The output falls in the saturation area

Parameters in the gates perturb

Change to **the output of the gates** will be small

Change to **the final loss** will also be little

Robust to **model compression**

Better test performance
Flat region generalizes better

Sharpened Sigmoid

- Straight forward idea: **sharpen the Sigmoid function** by using a smaller temperature $\tau < 1$

$$f_{W,b}(x) = \sigma((Wx + b)/\tau) = \sigma\big((W/\tau)x + (b/\tau)\big)$$

- This is equivalent to **rescale** the weight initialization and the gradient

Harm the **optimization process**

Cannot guarantee the outputs to be **close to the boundary**

Gumbel-Softmax Estimator

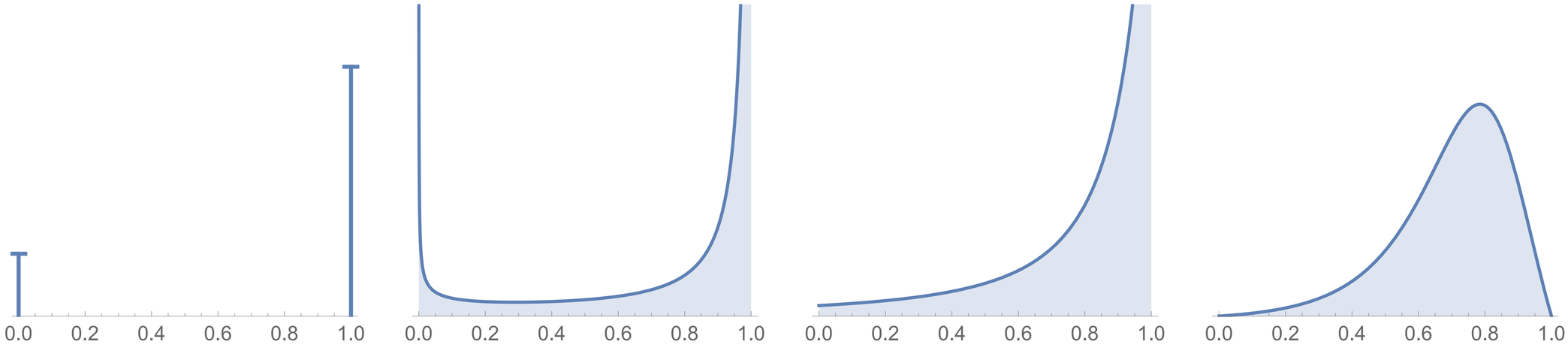
- In our special case, we leverage the **Gumbel-Softmax estimator** to estimate the Bernoulli distribution $D_\alpha \sim B(\sigma(\alpha))$ with prob. $\sigma(\alpha)$
- Define

$$G(\alpha, \tau) = \sigma\left(\frac{\alpha + \log U - \log(1-U)}{\tau}\right),$$

where $U \sim \text{Uniform}(0,1)$, then the following holds for $\epsilon \in (0, 1/2)$:

$$P(D_\alpha = 1) - (\tau/4)\log(1/\epsilon) \leq P(G(\alpha,\tau) \geq 1-\epsilon) \leq P(D_\alpha = 1)$$
$$P(D_\alpha = 0) - (\tau/4)\log(1/\epsilon) \leq P(G(\alpha,\tau) \leq \epsilon) \leq P(D_\alpha = 0)$$

Gumbel-Softmax Estimator



$\tau = 0$ $\tau = 1/2$ $\tau = 1$ $\tau = 2$

Probability density functions of Gumbel-Softmax estimators with different temperature τ

Gumbel-Gate LSTM (G^2-LSTM)

- $h_t, c_t = \mathrm{LSTM}(h_{t-1}, c_{t-1}, x_t)$

- $f_t = G\left(W_{xf}x_t + W_{hf}h_{t-1} + b_f, \tau\right)$
- $i_t = G(W_{xi}x_t + W_{hi}h_{t-1} + b_i, \tau)$
- $g_t = \tanh\left(W_{xg}x_t + W_{hg}h_{t-1} + b_g\right)$
- $o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o)$
- $c_t = f_t \odot c_{t-1} + i_t \odot g_t$
- $h_t = o_t \odot \tanh(c_t)$

In the forward pass during training, we **independently sample** all forget and input gates at each timestep, and update G^2-LSTM

In the backward pass, we use **standard gradient-based method** to update model parameters, since all components are differentiable

Experiments

- Language Modeling
  - Penn Treebank
- Machine Translation
  - IWSLT'14 German→English
  - WMT'14 English→German

Sensitivity Analysis

- Compress the gate-related parameters to show the robustness of our learned models

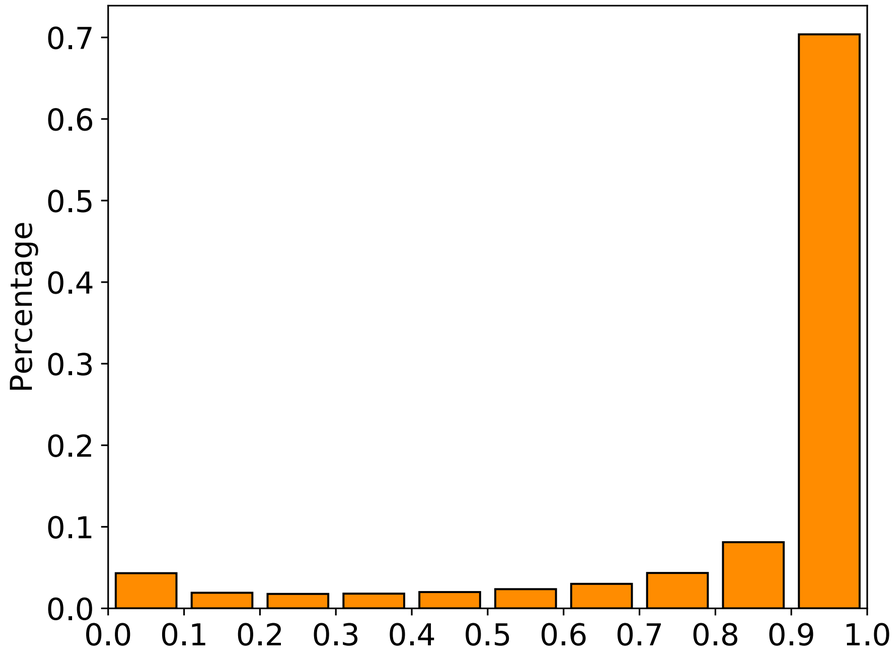
- **Low-precision compression**
  - Reduce the support set of the parameters by $\mathbf{round}_r = \mathrm{round}(x/r) \cdot r$
  - Further clip the rounded value to a fixed range using $\mathbf{clip}_c = \mathrm{clip}(x, -c, c)$

- **Low-rank compression**
  - Compress the parameter matrices by **singular value decomposition** (SVD)
  - Reduce the model size and lead to fast matrix multiplication

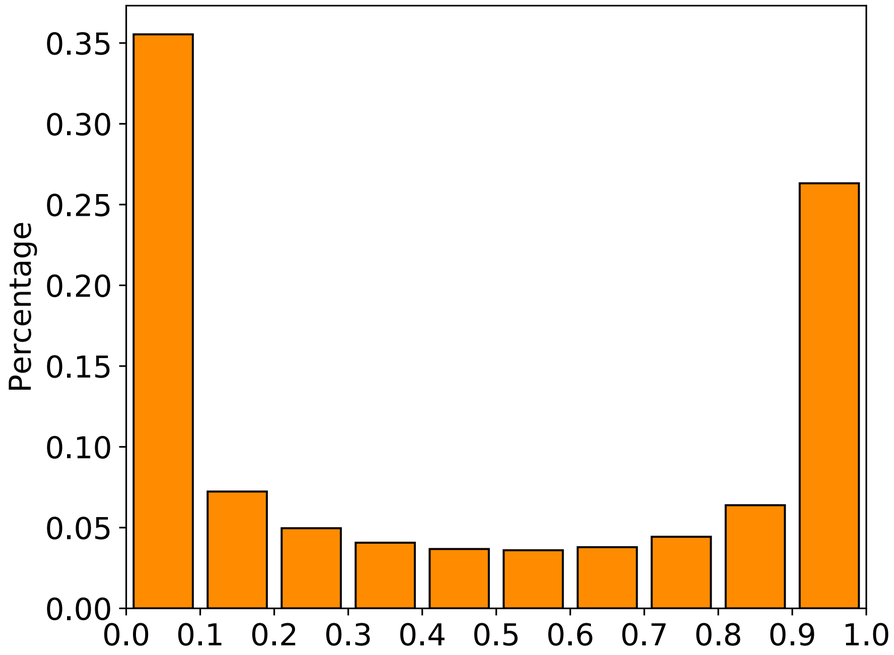
Experimental Results

Model	Original	Round	Round & Clip	SVD	SVD+
Penn Treebank (Perplexity)					
Baseline	52.8	53.2 (+0.4)	53.6 (+0.8)	56.6 (+3.8)	65.5 (+12.7)
Sharpened Sigmoid	53.2	53.5 (+0.3)	53.6 (**+0.4**)	54.6 (+1.4)	60.0 (+6.8)
G^2-LSTM	**52.1**	**52.2** (**+0.1**)	**52.8** (+0.7)	**53.3** (**+1.2**)	**56.0** (**+3.9**)
IWSLT'14 German→English (BLEU)					
Baseline	31.00	28.65 (-2.35)	21.97 (-9.03)	30.52 (-0.48)	29.56 (-1.44)
Sharpened Sigmoid	29.73	27.08 (-2.65)	25.14 (-4.59)	29.17 (-0.53)	28.82 (-0.91)
G^2-LSTM	**31.95**	**31.44** (**-0.51**)	**31.44** (**-0.51**)	**31.62** (**-0.33**)	**31.28** (**-0.67**)
WMT'14 English→German (BLEU)					
Baseline	21.89	16.22 (-5.67)	16.03 (-5.86)	21.15 (-0.74)	19.99 (-1.90)
Sharpened Sigmoid	21.64	16.85 (-4.79)	16.72 (-4.92)	20.98 (-0.66)	19.87 (-1.77)
G^2-LSTM	**22.43**	**20.15** (**-2.28**)	**20.29** (**-2.14**)	**22.16** (**-0.27**)	**21.84** (**-0.51**)

Histograms of Gate Distributions in G^2-LSTM



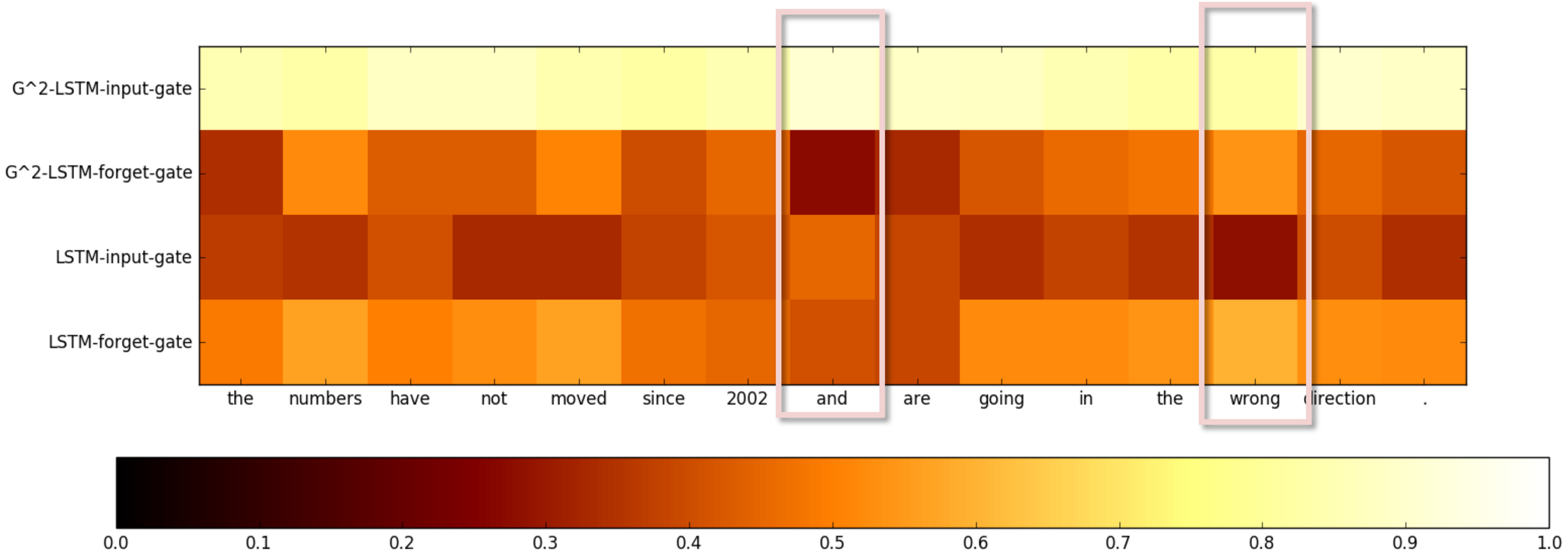
G^2-LSTM Input gates



G^2-LSTM Forget gates

Based on the gate outputs of the first-layer G^2-LSTM in the decoder from the same 10000 sentence pairs IWSLT14 German→English training sets

Visualization of Average Gate Values



Summary

- A new training algorithm for LSTM by leveraging the recently developed Gumbel-Softmax estimator
- Push the values of the input and forget gates to 0 or 1, leading to robust LSTM models
- Experiments on language modeling and machine translation demonstrated the effectiveness of the proposed training algorithm

Thanks!
Poster #63

Zhuohan is applying for a Ph.D. in Fall 2018! Please contact if you are interested!

Contact:

北京大学 PEKING UNIVERSITY Zhuohan Li (lizhuohan@pku.edu.cn)

Microsoft Tao Qin (taoqin@microsoft.com)