Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, Yoshua Bengjo

Ⅰ. Introduction

Recurrent neural networks는 input 길이와 output의 길이가 다양할 때 최근 좋은 결과를 보여주고 있다. 이 논문의 저자들은 최근 성공을 vanilla RNN으로는 이러한 성공을 거의 달성하지 못했다. 성공적인 실험에서는 long short-term memory units과 같은 정교한 recurrent hidden units을 가진 RNN이었다. 이 복잡한 recurrent units 중에, 두 개의 가까이 연관된 변수를 평가하는 LSTM과 GRU로 실험하였다. LSTM은 long-term dependency를 가진 sequence-based task를 잘 처리하는 것으로 알려져 있고 GRU는 최근에 도입되었고(2014년) 기계 번역에서 사용된다. 이 논문에서는 두 개의 유닛과 더 전통적인 tanh 유닛을 sequence modeling task로 평가한다. 세 개의 다성음 음악 datasets과 각각의 sample이 raw speech representation인 Ubisoft로부터 제공된 두 개의 내부 datasets을 사용한다. GRU와 LSTM이 성능은 비슷하지만 GRU과 CPU 연산량과 연산에 필요한 parameter 개수에 관해서 LSTM보다 효율적이다.

Ⅱ. Background : RNN

RNN은 전통적인 feedforward neural network의 확장으로 각각의 시간이 이전의 시간과 의존적인 recurrent hidden state를 가짐으로써 다양한 길이의 sequence input을 다룬다. 주어진 sequence x=$(x_1, x_2, \ldots, x_T)$이라 하면 RNN은 recurrent hidden state h_t를 아래와 같이 갱신한다.

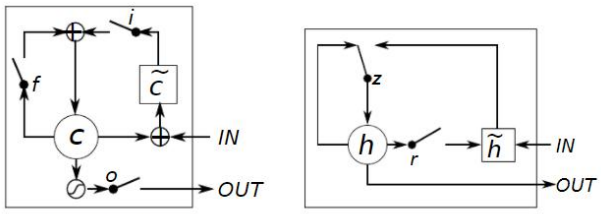
$$\mathbf{h}_t = \begin{cases} 0, & t = 0 \\ \phi\left(\mathbf{h}_{t-1}, \mathbf{x}_t\right), & \text{otherwise} \end{cases} \qquad \mathbf{h}_t = g\left(W\mathbf{x}_t + U\mathbf{h}_{t-1}\right),$$

ϕ는 nonlinear function으로 logistic sigmoid의 구성요소 같은 것이다. 선택적으로 RNN은 다양한 길이가 다시 될 수 있는 output y=$(y_1, y_2, \ldots, y_T)$를 갖는다. G는 smooth하고 bounded function으로 logistic function 또는 hyperbolic tangent function이다. 일반적인 RNN은 현재 상태 h_t가 주어진 sequence의 다음 원소에 대한 확률 분포를 결과로 내고 이 생산적인 모델은 sequence의 끝을 나타내기 위한 특별한 output symbol를 사용함으로써 다양한 길이의 sequence에 대한 분포도 잡을 수 있다. Sequence probability는 $p(x_1, \ldots, x_T) = p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_1, x_2) \cdots p(x_T \mid x_1, \ldots, x_{T-1}).$

로 마지막 원소는 특별한 end-of-sequence-value이다. 모델은 각각 조건 확률 분포를

$p(x_t \mid x_1, \ldots, x_{t-1}) = g(h_t)$, 로 나타낸다.

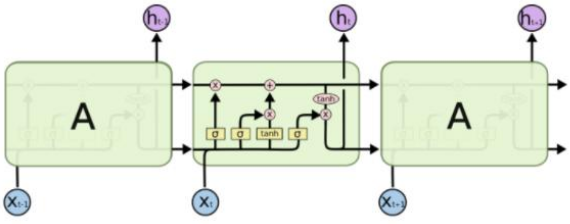
RNN으로는 gradients과 감소되거나 폭등하는 경향이 있어서 long-term dependency를 포착하기 어려웠다. 이로 인해 gradient 크기 변화 뿐만 아니라 장기 종속성의 효과가 단기 종속성의 영향으로 숨겨져 있기 때문에(순서 길이에 비해 기하 급수적으로 작음) gradient 기반 최적화 방법이 어려움을 겪는다. 이 문제의 부정적인 영향을 줄이기 위한 노력이 크게 두가지가 있다. 그 중 하나는 간단한 확률적 경사 하강법(stochastic gradient descent)보다 더 나은 학습 알고리즘을 고안하는 것이다. 가장 단순한 clipped gradient를 사용하거나(gradient vector의 norm으로 clip) 2차 도함수가 1차 도함수만큼 같은 증가 패턴을 따른다면 두 번째 방법을 사용하는 것이 덜 민감하다. 또 다른 접근방법은 더 복잡한 활성화함수를 만드는 것이다. Gating units을 사용하여 간단한 요소별 비선형성에 따른 정밀 변환으로 구성한다. 이것이 LSTM과 GRU이다.



(a) Long Short-Term Memory (b) Gated Recurrent Unit

Ⅲ. GRU

1. LSTM



$$f_t = \sigma(W_{xh\_f} x_t + W_{hh\_f} h_{t-1} + b_{h\_f})$$
$$i_t = \sigma(W_{xh\_i} x_t + W_{hh\_i} h_{t-1} + b_{h\_i})$$
$$o_t = \sigma(W_{xh\_o} x_t + W_{hh\_o} h_{t-1} + b_{h\_o})$$
$$g_t = \tanh(W_{xh\_g} x_t + W_{hh\_g} h_{t-1} + b_{h\_g})$$
$$c_t = f_t \odot c_{t-1} + i_t \odot g_t$$
$$h_t = o_t \odot \tanh(c_t)$$

- f_t (forget gate) : σ를 통하여 previous state c_{t-1}를 얼마나 기억할지 결정
- i_t (input gate) : σ를 통하여 새로운 값 g_t를 얼마나 반영할지 결정
- o_t(output gate) : σ를 통하여 다음 cell에서 current state c_t에 얼마나 영향을 줄지 결정

- g_t(cell input activation function) : 새로운 input x_t과 전달받은 이전의 값 h_{t-1}을 같이 연산하여 새로운 정보를 생성

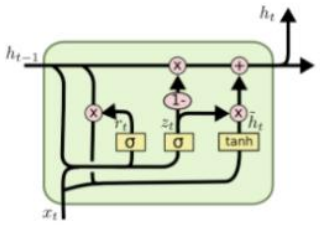
- c_t(cell state) : forget gate f_t와 previous state c_{t-1}를 element-wise product하여 c_{t-1}가 얼마나 남을지를 결정. Input gate i_t 와 g_t를 element-wise product한다음 더하여 current state c_t를 갱신한다.

- h_t(hidden state) : o_t와 곱해져서 다음 cell에 현재의 정보를 넘겨준다.

- σ(sigmoid) : 0~1 범위의 값을 출력하므로 각 gate에서 얼마나 영향을 줄지를 결정한다. 0을 출력할 경우에는 완전히 영향을 주지 않고 1을 출력하면 그대로 정보가 전달된다.

- ⊙(element-wise product)

2. GRU : LSTM의 gate를 재구성하여 간단해진 구조



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

- z_t(update gate) : 새로운 정보를 어떻게 구성할지를 결정. 새로운 input x_t와 previous hidden state를 고려하여 새로운 정보를 얼마나 current hidden state에 반영할지를 결정. z_t는 새로운 정보 $\widetilde{h_t}$에 곱해져 얼마나 반영될지를 결정한다. 1-z_t는 과거의 정보 h_{t-1}에 곱해져 얼마나 반영될지를 결정한다. LSTM의 input gate와 forget gate의 역할을 합친 것이다.

- r_t(reset gate) : 과거의 정보를 t-th time step에서 만들어지는 정보에 얼마나 반영할지만 고려함. 이전의 기억을 지우는 역할을 하지는 않는다. 새로운 input x_t과 과거의 정보 h_{t-1}를 고려했을 때 과거의 정보 h_{t-1}에 곱해져 얼마나 잊을지를 결정한다. LSTM의 forget gate와 비슷해 보이지만 새로운 정보 $\widetilde{h_t}$구성에만 영향을 미친다는 점에서 다르다.

- $\widetilde{h_t}$(temporal hidden state) : t-th time step에 형성된 새로운 정보를 의미한다. 과거의 정보 h_{t-1}를 reset gate r_t에 통과시켜 정보를 지우고 새로운 input x_t와 연산하여 새로운 정보 $\widetilde{h_t}$를 생성한다.

- h_t(current hidden state) : 새로운 정보 $\widetilde{h_t}$와 이전의 정보 h_{t-1}를 update gate z_t를 통해 각각 얼마나 반영할지를 결정. LSTM의 cell state와 hidden state가 합쳐져 있는 형태.

- σ(sigmoid) : 0~1 범위의 값을 출력하므로 각 gate에서 얼마나 영향을 줄지를 결정.

3. Discussion

1) RNN vs LSTM, GRU

- RNN : 현재 가지고 있는 정보를 계속 activation을 통해 계속 변형. $\widetilde{h_t}$가 따로 존재하지 않고 새로운 input x_t를 그대로 받아들인다.

- LSTM, GRU : 현재 가지고 있는 정보는 건들지 않고 새로운 정보를 만들어 더해준다. 과거의 정보가 activation을 통해 바뀌지 않으므로 장기기억을 더 잘하고 이전의 정보에 직접적으로 연산을 해주지 않으므로 shortcut path를 형성하여 gradient를 잘 전달할 수 있다. 새로운 input x_t가 과거의 정보 h_{t-1}을 고려하여 새로운 정보 $\widetilde{h_t}$를 구성.

2) LSTM vs GRU

- LSTM : output gate을 통하여 현재 가지고 있는 정보를 다음 time step에 얼마나 노출시킬지 조절한다. 이전의 정보를 얼마나 잊을지(forget gate)와 새로운 정보를 얼마나 받아들일지(input gate)가 독립적으로 분리되어 있다.

- GRU : 현재 가지고 있는 모든 정보를 time step에 노출시킨다. 기억의 망각과 갱신이 update gate를 통하여 한 번에 처리된다.

IV. Experiements Setting

1. Tasks and Datasets

$$\max_{\boldsymbol{\theta}} \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} \log p\left(x_t^n \mid x_1^n, \ldots, x_{t-1}^n; \boldsymbol{\theta}\right)$$

LSTM unit, GRU, tanh unit을 다성음 음악 모델링과 스피치 신호 모델링으로 비교한다. log likelihood를 최대화하는 방향으로 학습한다. Θ는 model parameter의 set이다. Polyphonic music modeling의 경우 Nottingham, JSB Chorales, MuseData and Piano-midi 데이터셋을 이용하였다. 이 데이터셋들은 각각 symbol이 a93-, 96-, 105-, 108-dimensional binary vector이다. Output unit으로는 sigmoid function을 사용하였다.

두 개의 내부 datasets으로는 Ubisoft에 의해 제공되는 speech signal modeling을 사용하였다. 각각의 sequence는 1차원의 raw audio signal이고 각각의 step에서 다음의 10개의 연속적인 sample들을 예측하기 위한 20개의 연속적인 sample들을 보기 위해 RNN을 이용한다. Dataset의 두 개의 다른 버전을 사용하였는데 하나는 길이가 500인 sequence(Ubisoft A)이고 다른 하나는 길이가 800인 sequence(Ubisoft B)이다. Ubisoft A와

Ubisoft B는 각각 7230개와 800개의 sequence를 갖는다. Output layer로 20개의 구성요소를 가진 Gaussians의 혼합을 사용하였다.

2. Models

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.

			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.

LSTM, GRU, tanh는 각각 대략 같은 개수의 파라미터를 가지고 있다. 의도적으로 모델들을 오버피팅을 피하고 비교를 쉽게 하기 위해 최대한 작게 만들었다. Neural networks에서 hidden units의 다른 유형 비교는 이미 예전에 Table1에서 행해졌다. Table1은 model 크기의 상세 정보이다. 각각의 모델을 RMSProp로 훈련시켰고 고정된 표준편차 0.075로 weight noise를 사용하였다. 매 갱신마다 gradient가 1보다 크면 exploding gradient를 방지하기 위해 norm을 1로 다시 조정한다. U(-12, -6)에서 샘플링 된 10개의 무작위로 선택된 log-uniform 후보 외에 validation performance를 최대화하기 위해 learning rate를 선택한다. Validation set은 early-stop training을 위해 사용되기도 한다.

Ⅴ. Results and Analysis & Conclusion

다성음 음악 데이터셋의 경우, GRU가 Nottingham을 제외하고 다른 모델보다 뛰어났다. LSTM은 Ubisoft A에서 최고였고 GRU는 Ubisoft B에서 최고였다.

GRU가 갱신 횟수와 실제 CPU time에서 더 빠른 진전을 보였다. Tanh-RNN의 각 갱신에 대한 계산 요구 사항은 다른 모델보다 훨씬 적지만 각 갱신마다 많은 진전을 이루지 못했고 훨씬 더 나쁜 수준에서 진행을 중단했다. 이러한 결과는 기존의 recurrent units에 비해 gating 장치의 장점을 명확하게 보여준다. Convergence는 종종 더 빠르고 최종은 더 낫다. 그러나 이 결과는 LSTM과 GRU를 비교하는데 결정적이지는 않고 gated recurrent

unit의 선택은 데이터셋과 해당 작업에 크게 좌우될 수 있다.

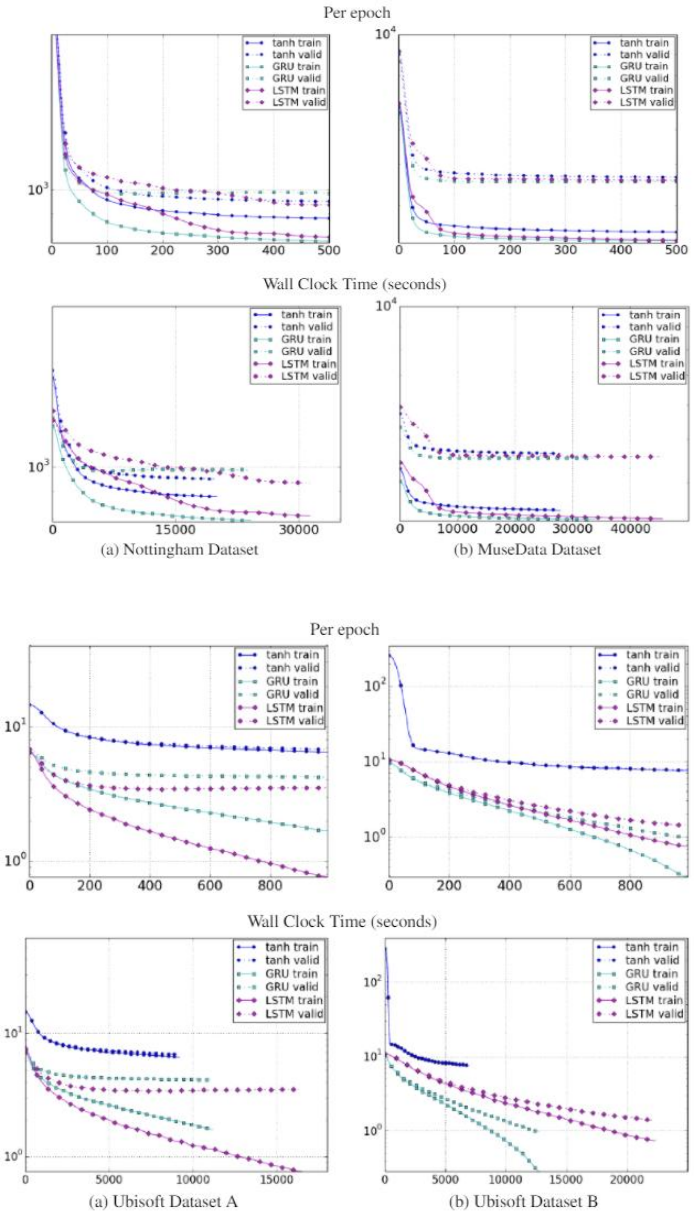


Figure 3: Learning curves for training and validation sets of different types of units with respect to (top) the number of iterations and (bottom) the wall clock time. x-axis is the number of epochs and y-axis corresponds to the negative-log likelihood of the model shown in log-scale.

이 평가는 gated units인 LSTM과 GRU가 tanh unit보다 뛰어나다는 것을 보였고 특히 raw speech signal modeling에서는 더욱 그랬다. 그러나, 이 두개의 gating units이 더 낫다고 확실히 말할 수는 없다. 이 논문은 예비 실험으로 간주된다. Gated unit이 학습을 돕는 방법을 더 잘 이해하고 각 구성 요소의 기여도를 분리하면 향후 더 철저한 실험이 필요하다.

https://www.researchgate.net/publication/269416998_Empirical_Evaluation_of_Gated_Recurrent_Neural_Networks_on_Sequence_Modeling