Boulder

Natural Language Processing; Neural Machine Translation

YouTube Playlist

Maziar Raissi

Assistant Professor

Department of Applied Mathematics

University of Colorado Boulder

maziar.raissi@colorado.edu

Neural Machine Translation by Jointly Learning to Align and Translate

Boulder

YouTube Video

Translation

$$\arg\max_y p(y|x)$$

x: source sentence, y: target sentence

RNN Encoder-Decoder

$x = (x_1, x_2, \ldots, x_{T_x}) \mapsto c$ — sequence of vectors → vector

$h_t = f(x_t, h_{t-1}) \rightarrow$ LSTM

$c = q(\{h_1, \cdots, h_{T_x}\}) = h_{T_x}$

$h_t \in \mathbb{R}^n \rightarrow$ hidden state at time t

encoder

$$p(y) = \prod_{t=1}^{T_y} \underbrace{p(y_t|\{y_1, \ldots, y_{t-1}\}, c)}_{g(y_{t-1}, s_t, c)}$$

$y = (y_1, y_2, \ldots, y_{T_y})$

$s_t \rightarrow$ hidden state of the RNN

decoder

Learning to Align & Translate

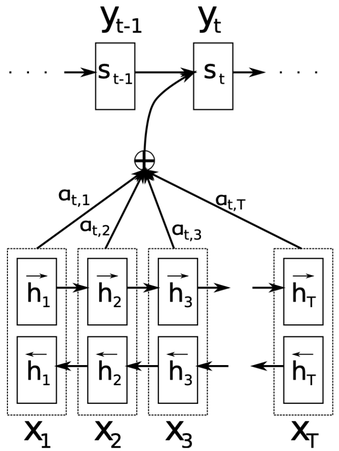
$$p(y_t|\{y_1, \ldots, y_{t-1}\}, x) = g(y_{t-1}, s_t, c_t)$$

$$s_t = f(s_{t-1}, y_{t-1}, c_t)$$

$$c_t = \sum_{t'=1}^{T_x} \alpha_{tt'} h_{t'} \rightarrow \text{context vector}$$

$$\alpha_{tt'} = \frac{\exp(e_{tt'})}{\sum_{\tau=1}^{T_x} \exp(e_{t\tau})}$$

$e_{tt'} = a(s_{t-1}, h_{t'}) \rightarrow$ alignment score (attention mechanism)



$$s_i = f(s_{i-1}, y_{i-1}, c_i) = (1 - z_i) \circ s_{i-1} + z_i \circ \tilde{s}_i,$$

$$\tilde{s}_i = \tanh\left(W \underbrace{e(y_{i-1})}_{\text{embedding}} + U\left[r_i \circ s_{i-1}\right] + C c_i\right)$$

$$z_i = \sigma\left(W_z e(y_{i-1}) + U_z s_{i-1} + C_z c_i\right) \rightarrow \text{update gates}$$

$$r_i = \sigma\left(W_r e(y_{i-1}) + U_r s_{i-1} + C_r c_i\right) \rightarrow \text{reset gates}$$

encoder: ignore c_i, $s_i \leftrightarrow h_i$, $y_{i-1} \leftrightarrow x_i$

$$a(s_{i-1}, h_j) = v_a^\top \tanh\left(W_a s_{i-1} + U_a h_j\right)$$

$$p(y_i|s_i, y_{i-1}, c_i) \propto \exp\left(y_i^\top W_o t_i\right)$$

$$t_i = \left[\max\left\{\tilde{t}_{i,2j-1}, \tilde{t}_{i,2j}\right\}\right]_{j=1,\ldots,l}^\top \rightarrow \text{maxout}$$

$$\tilde{t}_i = U_o s_{i-1} + V_o E y_{i-1} + C_o c_i$$

BLUE Score

Bilingual Evaluation Understudy

https://www.youtube.com/watch?v=DejHQYAGb7Q

$$p_n = \frac{\sum_{C \in \{Candidates\}} \sum_{n\text{-}gram \in C} Count_{clip}(n\text{-}gram)}{\sum_{C' \in \{Candidates\}} \sum_{n\text{-}gram' \in C'} Count(n\text{-}gram')}$$

p_n: modified n-gram precision score

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \qquad w_n = 1/N$$

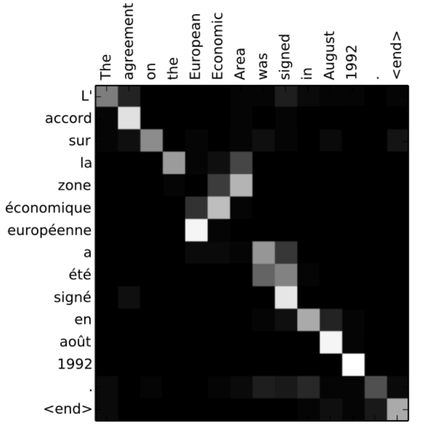
BP: brevity penalty

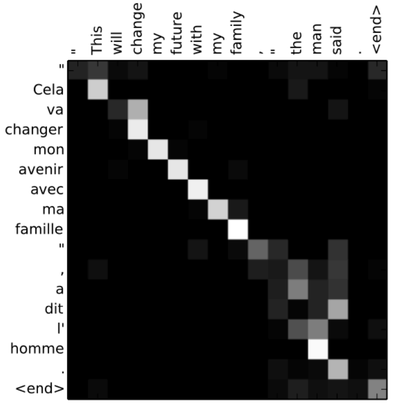
$$\text{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \le r \end{cases}$$

$$\log \text{BLEU} = \min(1 - \frac{r}{c}, 0) + \sum_{n=1}^{N} w_n \log p_n$$

$r \rightarrow$ reference translation length

$c \rightarrow$ candidate translation length

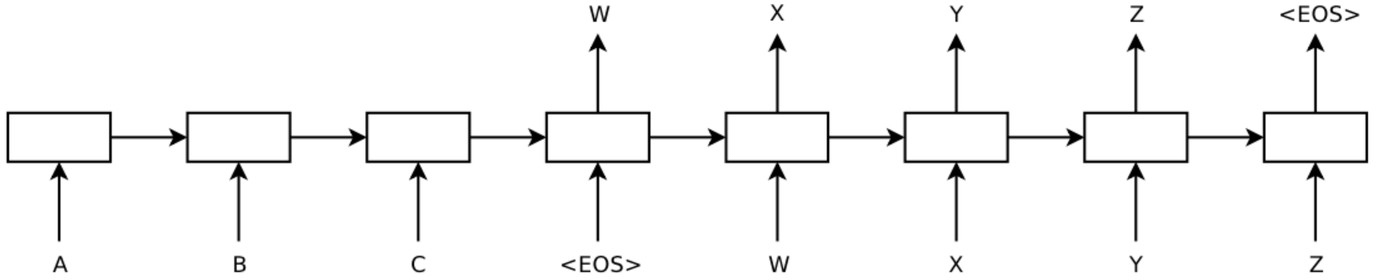




Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." *arXiv preprint arXiv:1409.0473* (2014).

Sequence to Sequence Learning with Neural Networks

YouTube Video



$$(x_1, \ldots, x_T) \xrightarrow{RNN} (y_1, \ldots, y_T)$$

$$h_t = \text{sigm}\left(W^{\text{hx}} x_t + W^{\text{hh}} h_{t-1}\right)$$

$$y_t = W^{\text{yh}} h_t$$

$$T \neq T'$$

$$p(y_1, \ldots, y_{T'} | x_1, \ldots, x_T) = \prod_{t=1}^{T'} \underbrace{p(y_t | v, y_1, \ldots, y_{t-1})}_{\text{softmax}}$$

1) Two different LSTMs: input & output sequences
2) Deep LSTMs
3) Reverse the order of the words of the input sentence $c, b, a \mapsto \alpha, \beta, \gamma$ rather than $a, b, c \mapsto \alpha, \beta, \gamma$

$$1/|\mathcal{S}| \sum_{(T,S)\in\mathcal{S}} \log p(T|S) \rightarrow \text{training objective}$$

https://www.youtube.com/watch?v=RLWuzLLSIgw

$$\hat{T} = \arg\max_T p(T|S) \rightarrow \text{beam search}$$

Method	test BLEU score (ntst14)
Bahdanau et al. [2]	28.45
Baseline System [29]	33.30
Single forward LSTM, beam size 12	26.17
Single reversed LSTM, beam size 12	30.59
Ensemble of 5 reversed LSTMs, beam size 1	33.00
Ensemble of 2 reversed LSTMs, beam size 12	33.27
Ensemble of 5 reversed LSTMs, beam size 2	34.50
Ensemble of 5 reversed LSTMs, beam size 12	**34.81**

Method	test BLEU score (ntst14)
Baseline System [29]	33.30
Cho et al. [5]	34.54
State of the art [9]	**37.0**
Rescoring the baseline 1000-best with a single forward LSTM	35.61
Rescoring the baseline 1000-best with a single reversed LSTM	35.85
Rescoring the baseline 1000-best with an ensemble of 5 reversed LSTMs	**36.5**
Oracle Rescoring of the Baseline 1000-best lists	∼45

Type	Sentence
model	Ulrich UNK , membre du conseil d' administration du constructeur automobile Audi , affirme qu' il s' agit d' une pratique courante depuis des années pour que les téléphones portables puissent être collectés avant les réunions du conseil d' administration afin qu' ils ne soient pas utilisés comme appareils d' écoute à distance .
Truth	Ulrich Hackenberg , membre du conseil d' administration du constructeur automobile Audi , déclare que la collecte des téléphones portables avant les réunions du conseil , afin qu' ils ne puissent pas être utilisés comme appareils d' écoute à distance , est une pratique courante depuis des années .

Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks." *Advances in neural information processing systems*. 2014.

Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation

Boulder

YouTube Video

Encoder $X = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N)$

$$h_j^{\langle t \rangle} = z_j h_j^{\langle t-1 \rangle} + (1 - z_j)\tilde{h}_j^{\langle t \rangle}$$

$$\tilde{h}_j^{\langle t \rangle} = \tanh\left([\mathbf{W}e(\mathbf{x}_t)]_j + \left[\mathbf{U}\left(\mathbf{r} \odot \mathbf{h}_{\langle t-1 \rangle}\right)\right]_j\right)$$

$$z_j = \sigma\left([\mathbf{W}_z e(\mathbf{x}_t)]_j + \left[\mathbf{U}_z \mathbf{h}_{\langle t-1 \rangle}\right]_j\right)$$

$$r_j = \sigma\left([\mathbf{W}_r e(\mathbf{x}_t)]_j + \left[\mathbf{U}_r \mathbf{h}_{\langle t-1 \rangle}\right]_j\right)$$

$$\mathbf{c} = \tanh\left(\mathbf{V}\mathbf{h}^{\langle N \rangle}\right)$$

Decoder $Y = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_M)$

$$\mathbf{h}'^{\langle 0 \rangle} = \tanh\left(\mathbf{V}'\mathbf{c}\right)$$

$$h'^{\langle t \rangle}_j = z'_j h'^{\langle t-1 \rangle}_j + (1 - z'_j)\tilde{h}'^{\langle t \rangle}_j$$

$$\tilde{h}'^{\langle t \rangle}_j = \tanh\left(\left[\mathbf{W}'e(\mathbf{y}_{t-1})\right]_j + r'_j\left[\mathbf{U}'\mathbf{h}'_{\langle t-1 \rangle} + \mathbf{C}\mathbf{c}\right]\right)$$

$$z'_j = \sigma\left(\left[\mathbf{W}'_z e(\mathbf{y}_{t-1})\right]_j + \left[\mathbf{U}'_z \mathbf{h}'_{\langle t-1 \rangle}\right]_j + \left[\mathbf{C}_z \mathbf{c}\right]_j\right)$$

$$r'_j = \sigma\left(\left[\mathbf{W}'_r e(\mathbf{y}_{t-1})\right]_j + \left[\mathbf{U}'_r \mathbf{h}'_{\langle t-1 \rangle}\right]_j + \left[\mathbf{C}_r \mathbf{c}\right]_j\right)$$

$$p(y_{t,j} = 1 \mid \mathbf{y}_{t-1}, \ldots, \mathbf{y}_1, X) = \frac{\exp\left(\mathbf{g}_j \mathbf{s}_{\langle t \rangle}\right)}{\sum_{j'=1}^{K} \exp\left(\mathbf{g}_{j'} \mathbf{s}_{\langle t \rangle}\right)}$$

$$s_i^{\langle t \rangle} = \max\left\{s'^{\langle t \rangle}_{2i-1}, s'^{\langle t \rangle}_{2i}\right\}$$

$$\mathbf{s}'^{\langle t \rangle} = \mathbf{O}_h \mathbf{h}'^{\langle t \rangle} + \mathbf{O}_y \mathbf{y}_{t-1} + \mathbf{O}_c \mathbf{c}$$

$$\mathbf{G} = \mathbf{G}_l \mathbf{G}_r$$

$$\mathbf{G}_r \in \mathbb{R}^{500 \times 1000}$$

$$\mathbf{G}_l \in \mathbb{R}^{K \times 500}$$

Statistical Machine Translation

$$p(\mathbf{f} \mid \mathbf{e}) \propto p(\mathbf{e} \mid \mathbf{f})p(\mathbf{f})$$

$p(\mathbf{f})$: language model

$p(\mathbf{e} \mid \mathbf{f})$: translation model

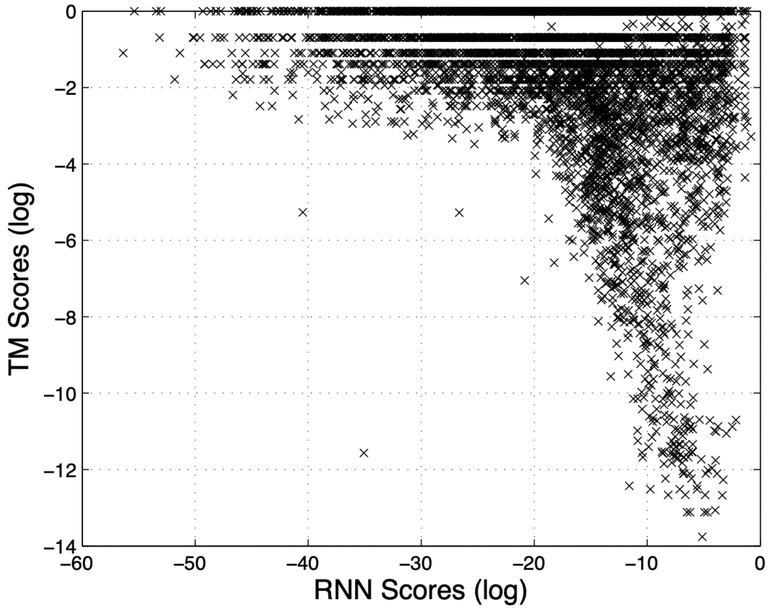
$$\log p(\mathbf{f} \mid \mathbf{e}) = \sum_{n=1}^{N} w_n f_n(\mathbf{f}, \mathbf{e}) + \log Z(\mathbf{e})$$

f_n: n-th feature

Use RNN Encoder–Decoder Phrase Pairs' scores as additional features in the log-linear model

$p_{\boldsymbol{\theta}}(\mathbf{y} \mid \mathbf{x}) \rightarrow$ score

phrase table



Models	BLEU dev	BLEU test
Baseline Moses	30.64	33.30
RNN	31.20	33.87

https://github.com/moses-smt/mosesdecoder

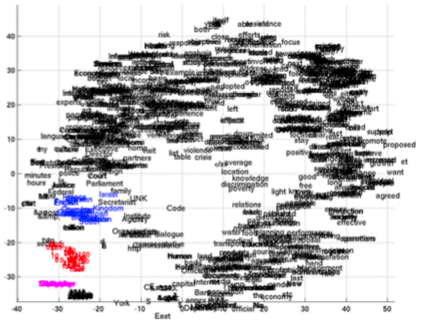
top scoring target phrases

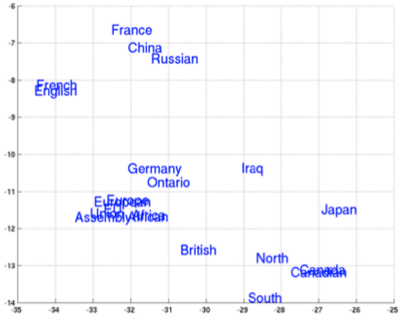
Source	Translation Model	RNN Encoder–Decoder
at the end of the	[a la fin de la] [f la fin des années] [être suprimés à la fin de la]	[à la fin du] [à la fin des] [à la fin de la]
for the first time	[r © pour la premirëre fois] [été donnés pour la première fois] [été commémorée pour la première fois]	[pour la première fois] [pour la première fois ,] [pour la première fois que]
in the United States and	[? aux ?tats-Unis et] [été ouvertes aux États-Unis et] [été constatées aux États-Unis et]	[aux Etats-Unis et] [des Etats-Unis et] [des États-Unis et]
, as well as	[?s , qu'] [?s , ainsi que] [?re aussi bien que]	[, ainsi qu'] [, ainsi que] [, ainsi que les]
one of the most	[?t ?l' un des plus] [?l' un des plus] [être retenue comme un de ses plus]	[l' un des] [le] [un des]

(a) Long, frequent source phrases

Source	Translation Model	RNN Encoder–Decoder
, Minister of Communications and Transport	[Secrétaire aux communications et aux transports :] [Secrétaire aux communications et aux transports]	[Secrétaire aux communications et aux transports] [Secrétaire aux communications et aux transports :]
did not comply with the	[vestimentaire , ne correspondaient pas à des] [susmentionnée n' était pas conforme aux] [présentées n' étaient pas conformes à la]	[n' ont pas respecté les] [n' était pas conforme aux] [n' ont pas respecté la]
parts of the world .	[© gions du monde .] [régions du monde considérées .] [région du monde considérée .]	[parties du monde .] [les parties du monde .] [des parties du monde .]
the past few days .	[le petit texte .] [cours des tout derniers jours .] [les tout derniers jours .]	[ces derniers jours .] [les derniers jours .] [cours des derniers jours .]
on Friday and Saturday	[vendredi et samedi à la] [vendredi et samedi à] [se déroulera vendredi et samedi ,]	[le vendredi et le samedi] [le vendredi et samedi] [vendredi et samedi]

(b) Long, rare source phrases

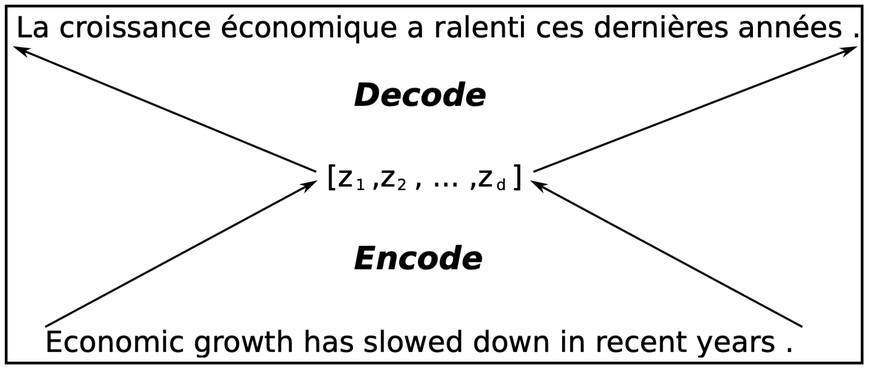




Cho, Kyunghyun, et al. "Learning phrase representations using RNN encoder-decoder for statistical machine translation." *arXiv preprint arXiv:1406.1078* (2014).

On the Properties of Neural Machine Translation: Encoder–Decoder Approaches

YouTube Video



– sentence length – vocabulary size

RNN with Gated Hidden Neurons

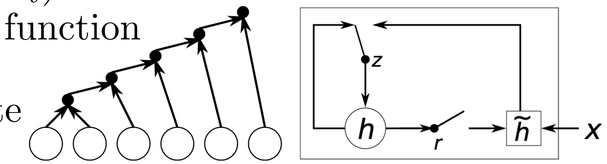
$x = (x_1, x_2, \ldots, x_T) \rightarrow$ variable length sequence

$h^{(t)} = f(h^{(t-1)}, x_t) \rightarrow$ hidden state

$f \rightarrow$ activation function

$r \rightarrow$ reset gate

$z \rightarrow$ update gate



$$p(x) = \prod_{t=1}^{T} p(x_t | x_{t-1}, \ldots, x_1) \rightarrow \text{language modeling}$$

$$p(x_{t,j} = 1 | x_{t-1}, \ldots, x_1) = \frac{\exp(w_j h_{<t>})}{\sum_{j'=1}^{K} \exp(w_{j'} h_{<t>})}$$

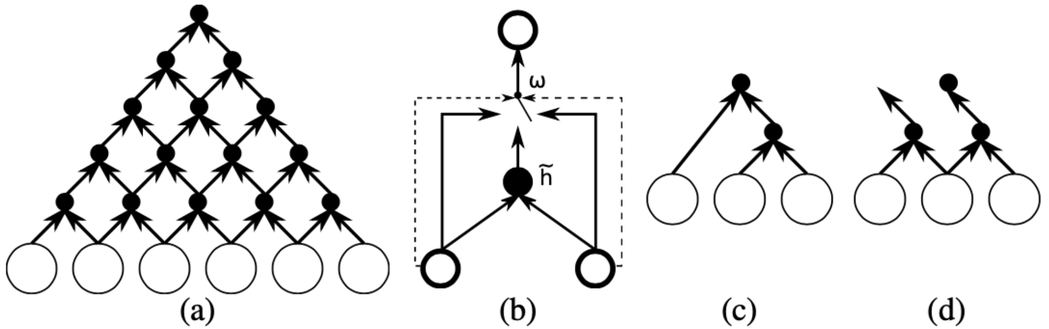
1-of-K encoding

Gated Recursive CNN

Parameters at each level are shared throughout the network

$x = (x_1, x_2, \ldots, x_T) \rightarrow$ input sequence $\quad x_j \in \mathbb{R}^d$

$h_j^{(t)} \rightarrow$ activation of the j-th hidden unit at recursion level $t \in [1, T-1]$



$$h_j^{(t)} = w_c \tilde{h}_j^{(t)} + w_\ell h_{j-1}^{(t-1)} + w_r h_j^{(t-1)}$$

$$h_j^{(0)} = U x_j \qquad [w_c, w_\ell, w_r] = \text{softmax}(G^\ell h_{j-1}^{(t-1)} + G^r h_j^{(t-1)})$$

$$\tilde{h}_j^{(t)} = \phi(W^\ell h_{j-1}^{(t-1)} + W^r h_j^{(t-1)})$$

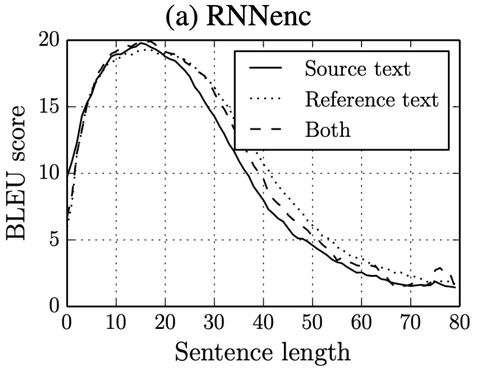
$\phi \rightarrow$ element-wise nonlinearity

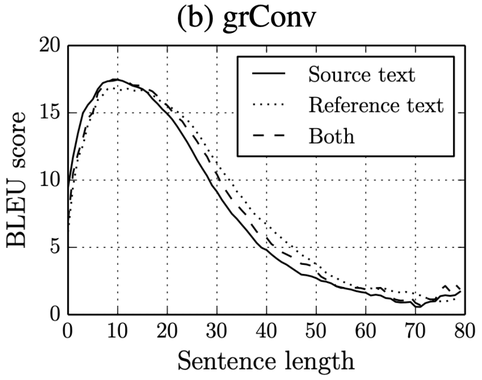
Source	And there are thorny mechanical questions that must be resolved during that time, like how to balance the state's mandate of "adequate access" to licensed marijuana with its prohibitions on cannabis businesses within 1,000 feet of a school, park, playground or child care center.
Reference	Pendant ce temps, des questions pratiques restent en suspens: comment équilibrer le mandat de l'état qui garantit un accès approprié à la marijuana agréée et interdit l'installation de commerces de vente de cannabis dans un rayon de 30 km autour d'une école, d'un parc, d'un terrain de jeu ou d'une crèche.
RNNEnc	Il y a des questions préventives qui se posent quant à l'équilibre des droits de l'enfant dans les limites d'une entreprise de collecte de sang.
grConv	De façon générale, il y a des raisons de sécurité pour que les entreprises aient accès à des milliers de centres de pêche, d'eau ou de recherche.
Moses	Et il y a des problèmes mécaniques complexes qui doivent être résolues au cours de cette période, comme la manière d'équilibrer le mandat de "l'accès adéquat" permis de marijuana avec l'interdiction du cannabis aux entreprises de 1000 pieds d'une école de jeu ou de parc, le service de garde.

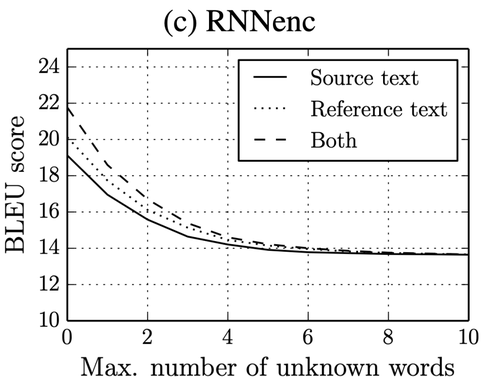
(a) Long Sentences

Source	According to them, one can find any weapon at a low price right now.
Reference	Selon eux, on peut trouver aujourd'hui à Moscou n'importe quelle arme pour un prix raisonnable.
RNNEnc	Selon eux, on peut se trouver de l'arme à un prix trop bas.
grConv	En tout cas, ils peuvent trouver une arme à un prix très bas à la fois.
Moses	Selon eux, on trouve une arme à bas prix pour l'instant.

(b) Short Sentences





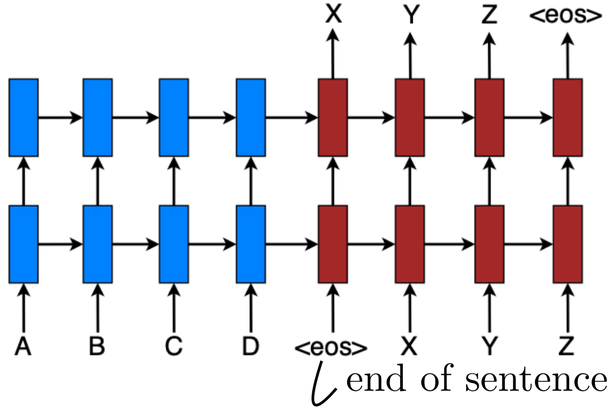


Cho, Kyunghyun, et al. "On the properties of neural machine translation: Encoder-decoder approaches." *arXiv preprint arXiv:1409.1259* (2014).

Effective Approaches to Attention-based Neural Machine Translation

Boulder

YouTube Playlist



$p(y|x)$

$x = (x_1, \ldots, x_n) \rightarrow$ source sentence

$y = (y_1, \ldots, y_m) \rightarrow$ target sentence

$$\log p(y|x) = \sum_{j=1}^{m} \log p\left(y_j | y_{<j}, \boldsymbol{s}\right)$$

$\mathbf{s} \rightarrow$ source sentence representation

$$p\left(y_j | y_{<j}, \boldsymbol{s}\right) = \text{softmax}\left(\underbrace{g\left(\boldsymbol{h}_j\right)}_{\text{vocabulary sized vector}}\right)$$

$\boldsymbol{h}_j = f(\boldsymbol{h}_{j-1}, \boldsymbol{s}) \rightarrow$ RNN hidden unit

RNN, GRU, or LSTM

$$J_t = \sum_{(x,y)\in\mathbb{D}} \underbrace{-\log p(y|x)}_{\text{training corpus}}$$

Attention-based Models

$p(y_t|y_{<t}, x) = \text{softmax}(\boldsymbol{W}_s \tilde{\boldsymbol{h}}_t)$

$\tilde{\boldsymbol{h}}_t = \tanh(\boldsymbol{W}_c[\boldsymbol{c}_t; \boldsymbol{h}_t])$

$\mathbf{h}_t \rightarrow$ target hidden state

$\mathbf{c}_t \rightarrow$ source-side context vector

Global Attention

$\boldsymbol{a}_t(s) = \text{align}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s)$

$$= \frac{\exp\left(\text{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s)\right)}{\sum_{s'} \exp\left(\text{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_{s'})\right)}$$

$\bar{\mathbf{h}}_s \rightarrow$ source hidden state

$$\text{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s) = \begin{cases} \boldsymbol{h}_t^\top \bar{\boldsymbol{h}}_s & \textit{dot} \\ \boldsymbol{h}_t^\top \boldsymbol{W}_a \bar{\boldsymbol{h}}_s & \textit{general} \\ \boldsymbol{v}_a^\top \tanh\left(\boldsymbol{W}_a[\boldsymbol{h}_t; \bar{\boldsymbol{h}}_s]\right) & \textit{concat} \end{cases}$$

content-based function

$\boldsymbol{a}_t = \text{softmax}(\boldsymbol{W}_a \boldsymbol{h}_t) \rightarrow$ location-based function

Local Attention

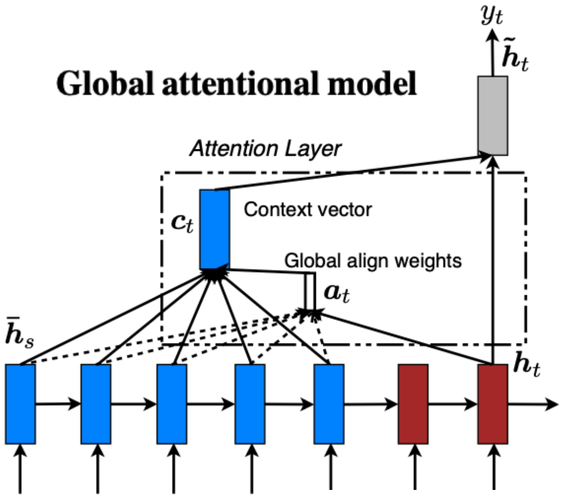
$p_t \rightarrow$ aligned position $[p_t - D, p_t + D]$ $\mathbf{a}_t \in \mathbb{R}^{2D+1}$

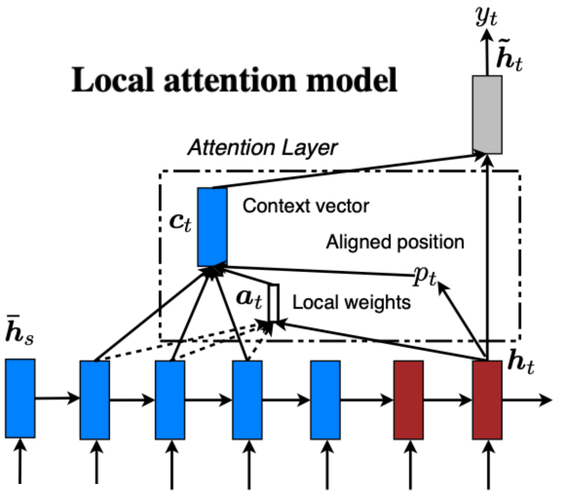
$p_t = t \rightarrow$ Monotonic alignment (local-m)

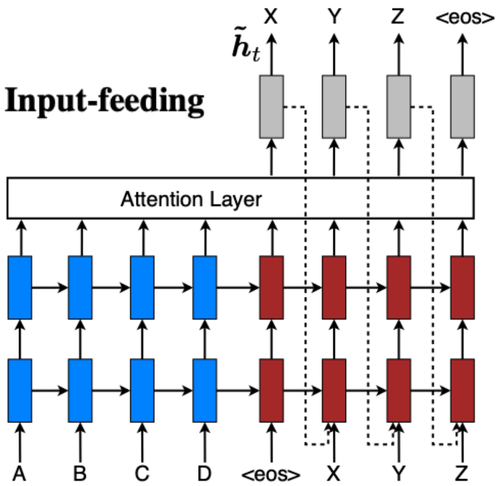
Predictive alignment (local-p)

$p_t = S \cdot \text{sigmoid}(\boldsymbol{v}_p^\top \tanh(\boldsymbol{W}_p \boldsymbol{h}_t))$

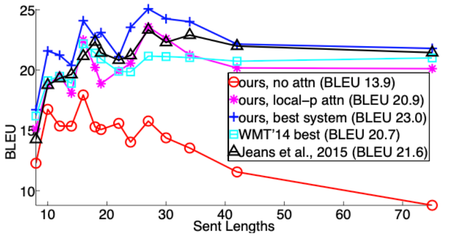
$$\boldsymbol{a}_t(s) = \text{align}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s) \exp\left(-\frac{(s - p_t)^2}{2\sigma^2}\right) \quad \sigma = \frac{D}{2}$$







System	Ppl	BLEU	
		Before	After unk
global (location)	6.4	18.1	19.3 (+1.2)
global (dot)	6.1	18.6	20.5 (+1.9)
global (general)	6.1	17.3	19.1 (+1.8)
local-m (dot)	>7.0	x	x
local-m (general)	6.2	18.6	20.4 (+1.8)
local-p (dot)	6.6	18.0	19.6 (+1.9)
local-p (general)	**5.9**	**19**	**20.9 (+1.9)**



Luong, Minh-Thang, Hieu Pham, and Christopher D. Manning. "Effective approaches to attention-based neural machine translation." *arXiv preprint arXiv:1508.04025* (2015).

Boulder

Neural Machine Translation Of Rare Words With Subword Units

YouTube Video

Byte Pair Encoding (BPE)

A simple data compression technique that iteratively replaces the most frequent pair of bytes in a sequence with a single, unused byte.

Byte: Character or character sequences

Byte Pair Encoding (BPE) merge operations learned from dictionary {'low', 'lower', 'newest', 'widest'}

Algorithm 1 Learn BPE operations

```
import re, collections

def get_stats(vocab):
  pairs = collections.defaultdict(int)
  for word, freq in vocab.items():
    symbols = word.split()
    for i in range(len(symbols)-1):
      pairs[symbols[i],symbols[i+1]] += freq
  return pairs

def merge_vocab(pair, v_in):
  v_out = {}
  bigram = re.escape(' '.join(pair))
  p = re.compile(r'(?<!\S)' + bigram + r'(?!\S)')
  for word in v_in:
    w_out = p.sub(''.join(pair), word)
    v_out[w_out] = v_in[word]
  return v_out

vocab = {'l o w </w>' : 5, 'l o w e r </w>' : 2,
         'n e w e s t </w>':6, 'w i d e s t </w>':3}
num_merges = 10
for i in range(num_merges):
  pairs = get_stats(vocab)
  best = max(pairs, key=pairs.get)
  vocab = merge_vocab(best, vocab)
  print(best)
```

```
{'l o w </w>': 5, 'l o w e r </w>': 2, 'n e w e s t </w>': 6, 'w i d e s t </w>': 3}

('e', 's')
{'l o w </w>': 5, 'l o w e r </w>': 2, 'n e w es t </w>': 6, 'w i d es t </w>': 3}

('es', 't')
{'l o w </w>': 5, 'l o w e r </w>': 2, 'n e w est </w>': 6, 'w i d est </w>': 3}

('est', '</w>')
{'l o w </w>': 5, 'l o w e r </w>': 2, 'n e w est</w>': 6, 'w i d est</w>': 3}

('l', 'o')
{'lo w </w>': 5, 'lo w e r </w>': 2, 'n e w est</w>': 6, 'w i d est</w>': 3}

('lo', 'w')
{'low </w>': 5, 'low e r </w>': 2, 'n e w est</w>': 6, 'w i d est</w>': 3}

('n', 'e')
{'low </w>': 5, 'low e r </w>': 2, 'ne w est</w>': 6, 'w i d est</w>': 3}

('ne', 'w')
{'low </w>': 5, 'low e r </w>': 2, 'new est</w>': 6, 'w i d est</w>': 3}

('new', 'est</w>')
{'low </w>': 5, 'low e r </w>': 2, 'newest</w>': 6, 'w i d est</w>': 3}

('low', '</w>')
{'low</w>': 5, 'low e r </w>': 2, 'newest</w>': 6, 'w i d est</w>': 3}

('w', 'i')
{'low</w>': 5, 'low e r </w>': 2, 'newest</w>': 6, 'wi d est</w>': 3}
```

At test time, the out-of-vocabulary (OOV) word 'lowest' would be segmented into 'low est.'.

Corpus statistics for German

segmentation	# tokens	# types	# UNK
none	100 m	1 750 000	1079
characters	550 m	3000	0
character bigrams	306 m	20 000	34
character trigrams	214 m	120 000	59
compound splitting△	102 m	1 100 000	643
morfessor*	109 m	544 000	237
hyphenation◇	186 m	404 000	230
BPE	112 m	63 000	0
BPE (joint)	111 m	82 000	32
character bigrams (shortlist: 50 000)	129 m	69 000	34

sequence length (# tokens) & vocabulary size (# types)

system	sentence
source	health research institutes
reference	Gesundheitsforschungsinstitute
WDict	Forschungsinstitute
C2-50k	Fo|rs|ch|un|gs|in|st|it|ut|io|ne|n
BPE-60k	Gesundheits|forsch|ungsinstitu|ten
BPE-J90k	Gesundheits|forsch|ungsin|stitute
source	asinine situation
reference	dumme Situation
WDict	asinine situation → UNK → asinine
C2-50k	as|in|in|e situation → As|in|en|si|tu|at|io|n
BPE-60k	as|in|ine situation → A|in|line-|Situation
BPE-J90K	as|in|ine situation → As|in|in-|Situation

system	sentence
source	Mirzayeva
reference	Мирзаева (Mirzaeva)
WDict	Mirzayeva → UNK → Mirzayeva
C2-50k	Mi|rz|ay|ev|a → Ми|рз|ае|ва (Mi|rz|ae|va)
BPE-60k	Mirz|ayeva → Мир|за|ева (Mir|za|eva)
BPE-J90k	Mir|za|yeva → Мир|за|ева (Mir|za|eva)
source	rakfisk
reference	ракфиска (rakfiska)
WDict	rakfisk → UNK → rakfisk
C2-50k	ra|kf|is|k → ра|кф|ис|к (ra|kf|is|k)
BPE-60k	rak|fisk → пра|ф|иск (pra|fisk)
BPE-J90k	rak|fisk → рак|ф|иска (rak|fiska)

Sennrich, Rico, Barry Haddow, and Alexandra Birch. "Neural machine translation of rare words with subword units." *arXiv preprint arXiv:1508.07909* (2015).

Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation

YouTube Video

$(X, Y) \rightarrow$ source and target sentence pair

$X = x_1, x_2, x_3, ..., x_M$

$Y = y_1, y_2, y_3, ..., y_N$

$\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_M} = EncoderRNN(x_1, x_2, x_3, ..., x_M)$

$$P(Y|X) = P(Y|\mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3}, ..., \mathbf{x_M}) = \prod_{i=1}^{N} P(y_i|y_0, y_1, y_2, ..., y_{i-1}; \mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3}, ..., \mathbf{x_M})$$

$P(y_i|y_0, y_1, y_2, y_3, ..., y_{i-1}; \mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3}, ..., \mathbf{x_M})$

$s_t = AttentionFunction(\mathbf{y}_{i-1}, \mathbf{x}_t) \quad \forall t, \quad 1 \leq t \leq M$

$\mathbf{y}_{i-1}$: decoder-RNN output

$$p_t = \exp(s_t) / \sum_{t=1}^{M} \exp(s_t) \qquad \forall t, \quad 1 \leq t \leq M$$

$$\mathbf{a}_i = \sum_{t=1} p_t . \mathbf{x}_t \rightarrow \text{attention context}$$

Residual Connections

$$\mathbf{c}_t^i, \mathbf{m}_t^i = \text{LSTM}_i(\mathbf{c}_{t-1}^i, \mathbf{m}_{t-1}^i, \mathbf{x}_t^{i-1}; \mathbf{W}^i)$$

$$\mathbf{x}_t^i = \mathbf{m}_t^i + \mathbf{x}_t^{i-1}$$

$$\mathbf{c}_t^{i+1}, \mathbf{m}_t^{i+1} = \text{LSTM}_{i+1}(\mathbf{c}_{t-1}^{i+1}, \mathbf{m}_{t-1}^{i+1}, \mathbf{x}_t^i; \mathbf{W}^{i+1})$$

$\mathbf{m}_t^i$ and $\mathbf{c}_t^i$ are the hidden states and memory states

8 LSTM layers Bi-directional Encoder for First Layer

Wordpiece Model

Word: Jet makers feud over seat width with big orders at stake

wordpieces: _J et _makers _fe ud _over _seat _width _with _big _orders _at _stake

$$\mathcal{D} \equiv \{(X^{(i)}, Y^{*(i)})\}_{i=1}^{N}$$

$$\mathcal{O}_{\text{ML}}(\boldsymbol{\theta}) = \sum_{i=1}^{N} \log P_\theta(Y^{*(i)} \mid X^{(i)})$$

$$\mathcal{O}_{\text{RL}}(\boldsymbol{\theta}) = \sum_{i=1}^{N} \sum_{Y \in \mathcal{Y}} P_\theta(Y \mid X^{(i)}) \, r(Y, Y^{*(i)})$$

$r(Y, Y^{*(i)})$: GLEU score (see paper)

$$\mathcal{O}_{\text{Mixed}}(\boldsymbol{\theta}) = \alpha * \mathcal{O}_{\text{ML}}(\boldsymbol{\theta}) + \mathcal{O}_{\text{RL}}(\boldsymbol{\theta})$$

Quantizable Model and Quantized Inference

$$\mathbf{c'}_t^i, \mathbf{m}_t^i = \text{LSTM}_i(\mathbf{c}_{t-1}^i, \mathbf{m}_{t-1}^i, \mathbf{x}_t^{i-1}; \mathbf{W}^i)$$

$$\mathbf{c}_t^i = \max(-\delta, \min(\delta, \mathbf{c'}_t^i)) \in [-\delta, \delta]$$

$$\mathbf{x'}_t^i = \mathbf{m}_t^i + \mathbf{x}_t^{i-1}$$

$$\mathbf{x}_t^i = \max(-\delta, \min(\delta, \mathbf{x'}_t^i)) \in [-\delta, \delta]$$

$$\mathbf{c'}_t^{i+1}, \mathbf{m}_t^{i+1} = \text{LSTM}_{i+1}(\mathbf{c}_{t-1}^{i+1}, \mathbf{m}_{t-1}^{i+1}, \mathbf{x}_t^i; \mathbf{W}^{i+1})$$

$$\mathbf{c}_t^{i+1} = \max(-\delta, \min(\delta, \mathbf{c'}_t^{i+1})) \in [-\delta, \delta]$$

$$\mathbf{W} = [\mathbf{W}_1, \mathbf{W}_2, \mathbf{W}_3, \mathbf{W}_4, \mathbf{W}_5, \mathbf{W}_6, \mathbf{W}_7, \mathbf{W}_8]$$

$$\mathbf{i}_t = \text{sigmoid}(\mathbf{W}_1 \mathbf{x}_t + \mathbf{W}_2 \mathbf{m}_t)$$

$$\mathbf{i'}_t = \tanh(\mathbf{W}_3 \mathbf{x}_t + \mathbf{W}_4 \mathbf{m}_t)$$

$$\mathbf{f}_t = \text{sigmoid}(\mathbf{W}_5 \mathbf{x}_t + \mathbf{W}_6 \mathbf{m}_t)$$

$$\mathbf{o}_t = \text{sigmoid}(\mathbf{W}_7 \mathbf{x}_t + \mathbf{W}_8 \mathbf{m}_t)$$

$$\mathbf{c}_t = \mathbf{c}_{t-1} \odot \mathbf{f}_t + \mathbf{i'}_t \odot \mathbf{i}_t$$

$$\mathbf{m}_t = \mathbf{c}_t \odot \mathbf{o}_t$$

$$\mathbf{v_t} = \mathbf{W_s} * \mathbf{y_t}$$

$$\mathbf{v'_t} = \max(-\gamma, \min(\gamma, \mathbf{v_t}))$$

$$\mathbf{p_t} = softmax(\mathbf{v'_t})$$

replace all the floating point operations with fixed-point integer operations with either 8-bit or 16-bit resolution

weight matrix $\mathbf{W}$ is represented using an 8-bit integer matrix $\mathbf{WQ}$ and a float vector $\mathbf{s}$

$$\mathbf{s}_i = \max(\text{abs}(\mathbf{W}[i, :]))$$

$$\mathbf{WQ}[i, j] = \text{round}(\mathbf{W}[i, j] / \mathbf{s}_i \times 127.0)$$

$\mathbf{c}_t^i$ and $\mathbf{x}_t^i$ are represented using 16-bit integers

matrix multiplications are done using 8-bit integer multiplication

All other operations (sigmoid, tanh) and $(\odot, +)$ are done using 16-bit integer operations

beam search:

$$s(Y, X) = \log(P(Y|X)) / lp(Y) + cp(X; Y)$$

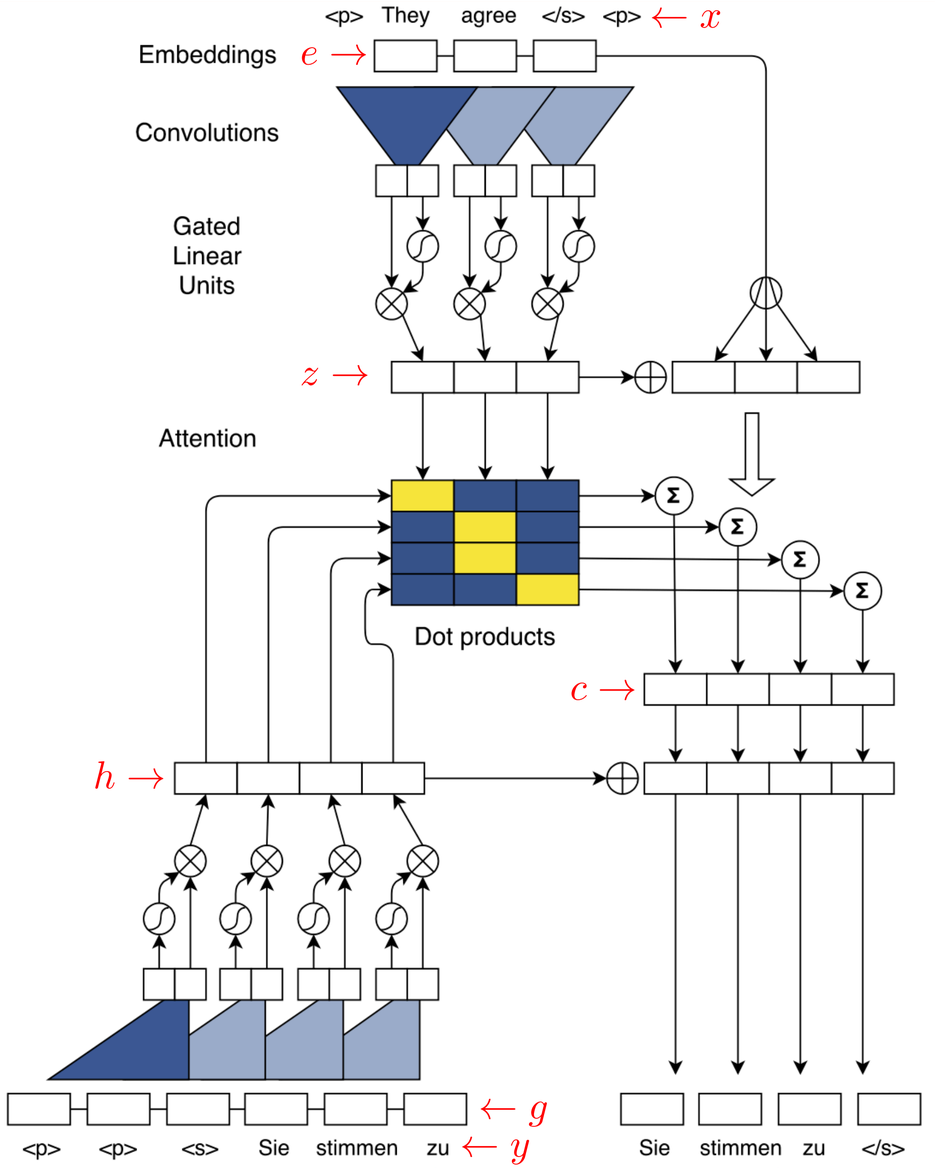
$$lp(Y) = \frac{(5 + |Y|)^\alpha}{(5 + 1)^\alpha}$$

$$cp(X; Y) = \beta * \sum_{i=1}^{|X|} \log(\min(\sum_{j=1}^{|Y|} p_{i,j}, 1.0)),$$

Wu, Yonghui, et al. "Google's neural machine translation system: Bridging the gap between human and machine translation." *arXiv preprint arXiv:1609.08144* (2016).

Convolutional Sequence to Sequence Learning

YouTube Video



Position Embeddings

$x = (x_1, \ldots, x_m) \rightarrow$ input elements

$w = (w_1, \ldots, w_m) \rightarrow$ embedding of x in distributional space

$w_j \in \mathbb{R}^f \rightarrow$ a column in an embedding matrix $\mathcal{D} \in \mathbb{R}^{f \times V}$

$p = (p_1, \ldots, p_m) \rightarrow$ position embedding (sense of order)

$p_j \in \mathbb{R}^f$

$e = (e_1, \ldots, e_m) \rightarrow$ input element representation

$e_j = w_j + p_j$

$g = (g_1, \ldots, g_n) \rightarrow$ output element representation (similarly) (fed back into the decoder network)

$h^l = (h^l_1, \ldots, h^l_n) \rightarrow$ output of the l-th block for the decoder network

$z^l = (z^l_1, \ldots, z^l_m) \rightarrow$ output of the l-th block for the encoder network

Block: One dimensional convolution followed by a non-linearity

$X \in \mathbb{R}^{k \times d} \rightarrow$ input to the convolution

↳ concatenation of k input elements embedded in d dimensions

$X \in \mathbb{R}^{kd} \rightarrow$ flatten

$W \in \mathbb{R}^{2d \times kd}, b_w \in \mathbb{R}^{2d} \rightarrow$ parameters of the convolutional kernel

$Y = WX + b_w \in \mathbb{R}^{2d} \rightarrow$ convolution

$Y = [A, B] \in \mathbb{R}^{2d}, A, B \in \mathbb{R}^d$

$v([A, B]) = A \odot \sigma(B) \in \mathbb{R}^d \rightarrow$ GLU (Gated Linear Units)

$h^l_i = v(W^l[h^{l-1}_{i-k/2}, \ldots, h^{l-1}_{i+k/2}] + b^l_w) + h^{l-1}_i \rightarrow$ residual connections

$p(y_{i+1}|y_1, \ldots, y_i, x) = \text{softmax}(W_o h^L_i + b_o) \in \mathbb{R}^T$

Multi-step Attention

$h^l_i \rightarrow$ current decoder state

$g_i \rightarrow$ embedding of the previous target element

$$d^l_i = W^l_d h^l_i + b^l_d + g_i$$

↳ decoder state summary

$a^l_{ij} \rightarrow$ attention of state i and source element j for decoder layer l

$z^u_j \rightarrow$ output of the the last encoder block u

$$a^l_{ij} = \frac{\exp\left(d^l_i \cdot z^u_j\right)}{\sum_{t=1}^m \exp\left(d^l_i \cdot z^u_t\right)}$$

$$c^l_i = \sum_{j=1}^m a^l_{ij}(z^u_j + e_j)$$

↳ conditional input to the current decoder layer

WMT'16 English-Romanian	BLEU
Sennrich et al. (2016b) GRU (BPE 90K)	28.1
ConvS2S (Word 80K)	29.45
ConvS2S (BPE 40K)	30.02

WMT'14 English-German	BLEU
Luong et al. (2015) LSTM (Word 50K)	20.9
Kalchbrenner et al. (2016) ByteNet (Char)	23.75
Wu et al. (2016) GNMT (Word 80K)	23.12
Wu et al. (2016) GNMT (Word pieces)	24.61
ConvS2S (BPE 40K)	25.16

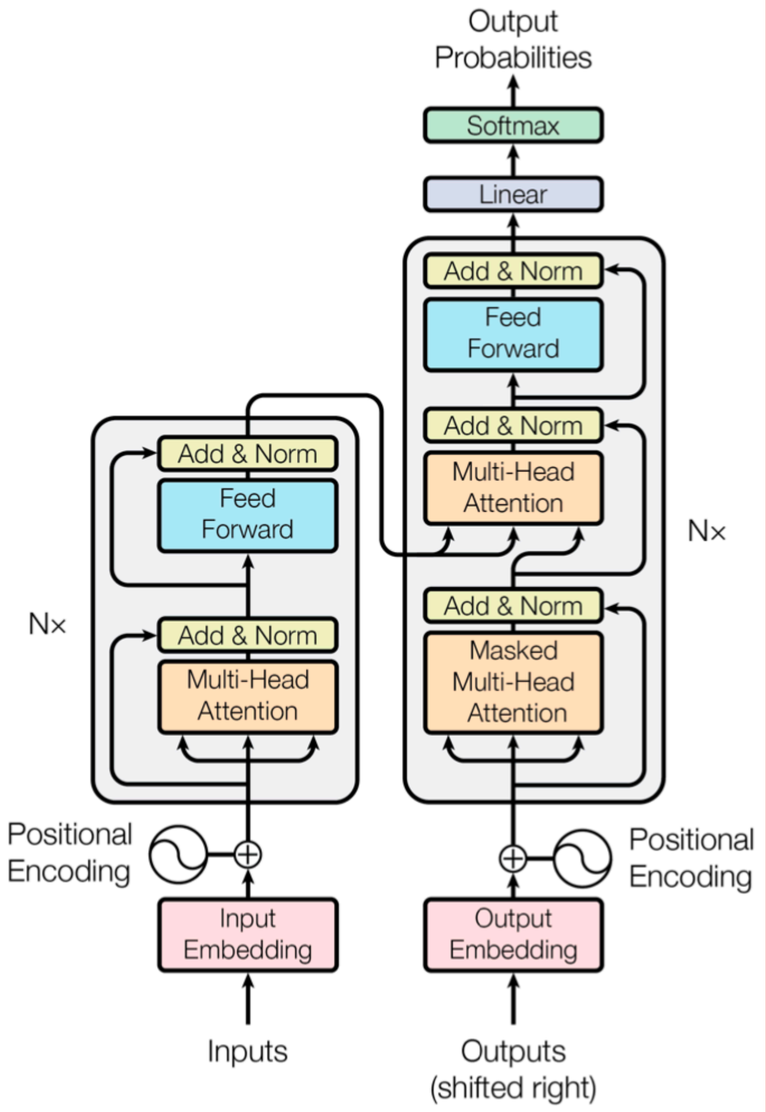
WMT'14 English-French	BLEU
Wu et al. (2016) GNMT (Word 80K)	37.90
Wu et al. (2016) GNMT (Word pieces)	38.95
Wu et al. (2016) GNMT (Word pieces) + RL	39.92
ConvS2S (BPE 40K)	40.51

Gehring, Jonas, et al. "Convolutional sequence to sequence learning." *arXiv preprint arXiv:1705.03122* (2017).

Attention Is All You Need

Boulder

YouTube Video



$(x_1, x_2, \ldots, x_n) \rightarrow$ input sequence of symbol representations

$(z_1, z_2, \ldots, z_n) \rightarrow$ sequence of continuous representations

$(y_1, y_2, \ldots, y_m) \rightarrow$ output sequence of symbols

$\text{encoder} : (x_1, \ldots, x_n) \mapsto (z_1, \ldots, z_n)$

$\text{decoder} : (y_0, \ldots, y_{t-1}; z_1, \ldots, z_n) \mapsto p(y_t | y_0, \ldots, y_{t-1}; z_1, \ldots, z_n)$

$x_t \in \{1, \ldots, |V|\} \qquad V \rightarrow$ input vocabulary

$E \in \mathbb{R}^{|V| \times d_{\text{model}}} \rightarrow$ input embedding matrix

$e_t = E(x_t) \in \mathbb{R}^{d_{\text{model}}}$

$p_t \in \mathbb{R}^{d_{\text{model}}} \rightarrow$ positional encoding

$$p_t^{2i-1} = \sin\left(\frac{t}{10000^{\frac{2i-1}{d_{\text{model}}}}}\right)$$

$$p_t^{2i} = \cos\left(\frac{t}{10000^{\frac{2i}{d_{\text{model}}}}}\right)$$

$\epsilon_t = e_t + p_t \in \mathbb{R}^{d_{\text{model}}}$

$q_t^\ell = W_\ell^q \epsilon_t \in \mathbb{R}^d \rightarrow$ query $(W_\ell^q \in \mathbb{R}^{d \times d_{\text{model}}})$

$k_t^\ell = W_\ell^q \epsilon_t \in \mathbb{R}^d \rightarrow$ key $(W_\ell^k \in \mathbb{R}^{d \times d_{\text{model}}})$

$v_t^\ell = W_\ell^v \epsilon_t \in \mathbb{R}^d \rightarrow$ value $(W_\ell^v \in \mathbb{R}^{d \times d_{\text{model}}})$

$$a_{tt'}^\ell = \frac{q_t^{\ell T} k_{t'}^\ell}{\sqrt{d}} \qquad d = d_{\text{model}}/m$$

$$\alpha_{tt'}^\ell = \frac{\exp(a_{tt'}^\ell)}{\sum_{\tau=1}^n \exp(a_{t\tau}^\ell)}$$

$$h_t^\ell = \sum_{t'=1}^n \alpha_{tt'}^\ell v_{t'}^\ell \in \mathbb{R}^d$$

$$h_t = [h_t^1; \ldots; h_t^m] \in \mathbb{R}^{d_{\text{model}}}$$

$$o_t = W^o h_t \in \mathbb{R}^{d_{\text{model}}}$$

$$W^o \in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}}$$

$$f_t = \epsilon_t + o_t \in \mathbb{R}^{d_{\text{model}}}$$

$$\mu_t = \text{mean}_{i=1,\ldots,d_{\text{model}}} f_t^i \in \mathbb{R}$$

$$\sigma_t = \text{std}_{i=1,\ldots,d_{\text{model}}} f_t^i \in \mathbb{R} \qquad g_t = \frac{f_t - \mu_t}{\sigma_t}$$

$$\text{FFN}(x) = \underbrace{\max(0, xW_1 + b_1)}_{\in \mathbb{R}^{d_{\text{ff}}}} W_2 + b_2 \in \mathbb{R}^{d_{\text{model}}}$$

$$h_t = \sum_{t'=1}^{t-1} \alpha_{tt'} v_{t'} \rightarrow \text{masking \& shift right}$$

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [15]	23.75			
Deep-Att + PosUnk [32]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [31]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [8]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [26]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [32]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [31]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [8]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	**$3.3 \cdot 10^{18}$**	
Transformer (big)	**28.4**	**41.0**	$2.3 \cdot 10^{19}$	

Vaswani, Ashish, et al. "Attention is all you need." *Advances in neural information processing systems*. 2017.

Reformer: The Efficient Transformer

YouTube Video

$\text{Attention}(Q, K, V) = \text{softmax}(QK^T)V$

$Q, K, V \in \mathbb{R}^{b \times l \times d} \rightarrow$ query, key, value

$b \rightarrow$ batch size

$l \rightarrow$ sequence length

$d \rightarrow$ dimension of queries, keys and values

$QK^T \in \mathbb{R}^{b \times l \times l}$

– memory footprint

– computational cost

Memory-efficient attention

If $l = 64K$ and $b = 1$, the memory footprint of QK^T is $64K \times 64K \times \text{Float32} \approx 16\text{GB}$.

The attention can be computed for each query q_t separately.

$$o_t = \sum_{s \in \mathcal{P}_t} \exp(q_t \cdot k_s - z(t, \mathcal{P}_t))v_s$$

$z(t, \mathcal{P}_t) \rightarrow$ normalization term in softmax

$\mathcal{P}_t = \{s : s \leq t\} \rightarrow$ set that t attends to

Locality Sensitive Hashing (LHS)

Computational complexity is quadratic in l.

Softmax is dominated by the largest elements!

Only need to focus on keys closest to q_t.

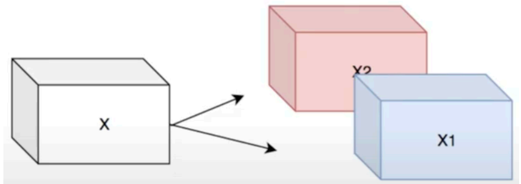
Nearby vectors get the same hash with high probability and distant ones do not.

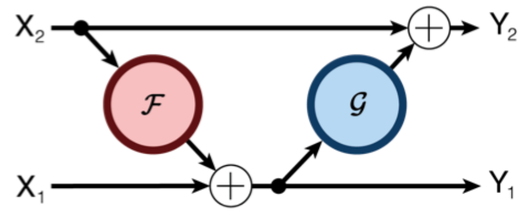
$h \rightarrow$ hash function

$\mathcal{P}_t = \{s : s \leq t, h(k_s) = h(q_t)\}$

Computational complexity can be reduced to $O(l \log l)$ in l.

Reversible Transformer



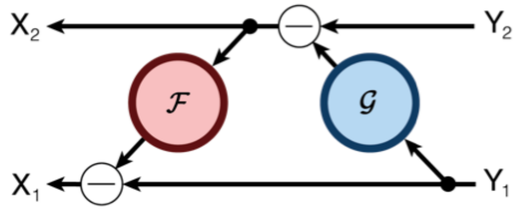


$\mathcal{F} \rightarrow$ Attention

$\mathcal{G} \rightarrow$ FeedForward

$$y_1 = x_1 + \mathcal{F}(x_2)$$

$$y_2 = x_2 + \mathcal{G}(y_1)$$



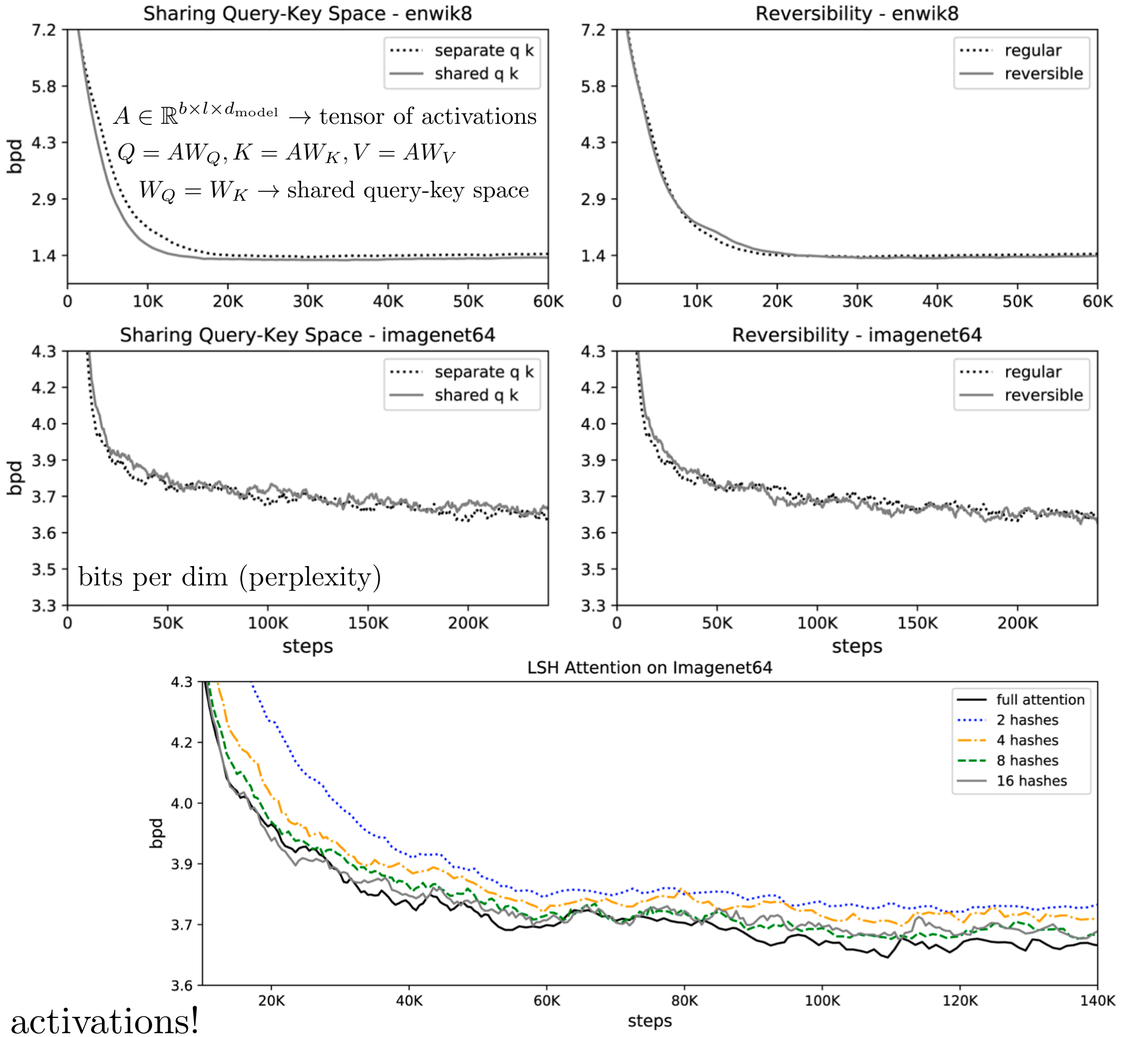
$$x_2 = y_2 - \mathcal{G}(y_1)$$

$$x_1 = y_1 - \mathcal{F}(x_2)$$

Backpropagation without storing activations!

Chunking across positions in a sequance

$Y_2 = [Y_2^{(1)}; \ldots; Y_2^{(c)}] \quad Y_2^{(i)} = X_2^{(i)} + \mathcal{G}(Y_1^{(i)})$

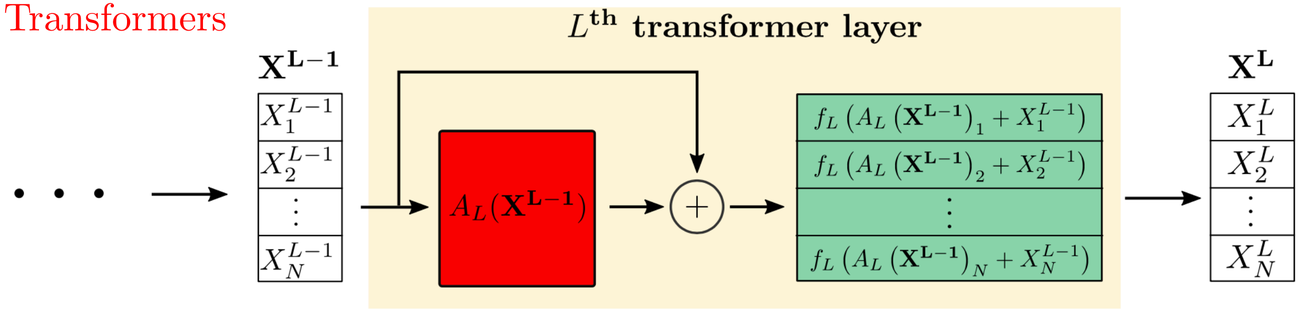


Kitaev, Nikita, Łukasz Kaiser, and Anselm Levskaya. "Reformer: The efficient transformer." *arXiv preprint arXiv:2001.04451* (2020).

Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention

Boulder

Transformers



$x \in \mathbb{R}^{N \times F} \rightarrow$ a sequence of N feature vectors of dimension F

$T : \mathbb{R}^{N \times F} \rightarrow \mathbb{R}^{N \times F}$

$\hookrightarrow$ transformer (composition of L transformer layers)

$T = T_L \circ T_{L-1} \circ \ldots \circ T_1$

$T_l(x) = f_l(A_l(x) + x)$

$\hookrightarrow$ small two layer feedforward network transforming each feature vector independently

$A_l \rightarrow$ self-attention function

$Q = xW_Q, K = xW_K, V = xW_V$

$W_Q, W_K \in \mathbb{R}^{F \times D}, W_V \in \mathbb{R}^{F \times M}$

$$A_l(x) = V' = \underbrace{\text{softmax}}_{\text{row-wise}}\left(\frac{QK^T}{\sqrt{D}}\right)V$$

$$V'_i = \frac{\sum_{j=1}^{N} \text{sim}(Q_i, K_j)V_j}{\sum_{j=1}^{N} \text{sim}(Q_i, K_j)}, \text{sim}(q,k) := \exp\left(\frac{q^T k}{\sqrt{D}}\right)$$

Linearized Attention

$\text{sim}(q,k) = \phi(q)^T\phi(k), \phi \rightarrow$ feature representation of the kernel

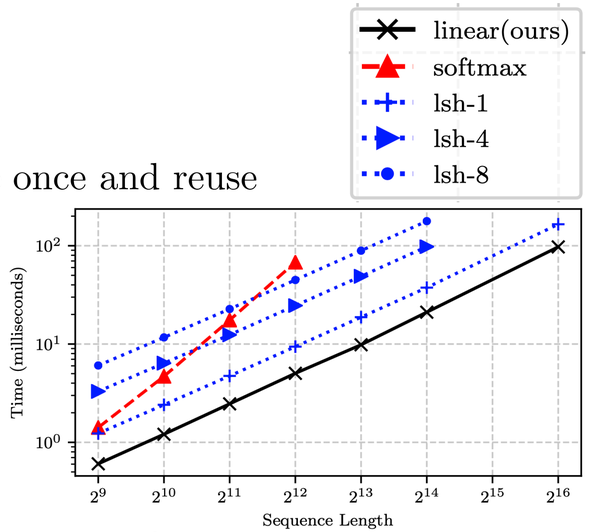
$$\phi(x) = \text{elu}(x) + 1$$

$$V'_i = \frac{\sum_{j=1}^{N} \phi(Q_i)^T\phi(K_j)V_j}{\sum_{j=1}^{N} \phi(Q_i)^T\phi(K_j)}$$

$=:S \rightarrow$ compute once and reuse

$$= \frac{\phi(Q_i)^T \overbrace{\sum_{j=1}^{N} \phi(K_j)V_j^T}^{=:S}}{\phi(Q_i)^T \underbrace{\sum_{j=1}^{N} \phi(K_j)}_{=:Z}}$$

$=:Z \rightarrow$ compute once and reuse



Causal Masking

$=:S_i \rightarrow$ can be computed from S_{i-1}

$$V'_i = \frac{\phi(Q_i)^T \overbrace{\sum_{j=1}^{i} \phi(K_j)V_j^T}^{=:S_i}}{\phi(Q_i)^T \underbrace{\sum_{j=1}^{i} \phi(K_j)}_{=:Z_i}}$$

$=:Z_i \rightarrow$ can be computed from Z_{i-1}

Transformers are RNNs

$s, z \rightarrow$ attention, normalizer memory

$s_0 = 0, z_0 = 0$

$s_i = s_{i-1} + \phi(x_i W_K)(x_i W_V)^T$

$z_i = z_{i-1} + \phi(x_i W_K)$

$$y_i = f_l\left(\frac{\phi(x_i W_Q)^T s_i}{\phi(x_i W_Q)^T z_i} + x_i\right)$$

Gradient Computation

A naive implementation would require storing all intermediate values S_i in order to compute the gradients! See the paper for a derivation of the gradients of the numerator as cumulative sums!

Katharopoulos, Angelos, et al. "Transformers are rnns: Fast autoregressive transformers with linear attention." *International Conference on Machine Learning*. PMLR, 2020.

Boulder

Questions?

YouTube Playlist