

# Effective Approaches to Attention-based Neural Machine Translation

Minh-Thang Luong , Hieu Pham, Christopher D. Manning

Lan Li (present)

# Outline

Abstract

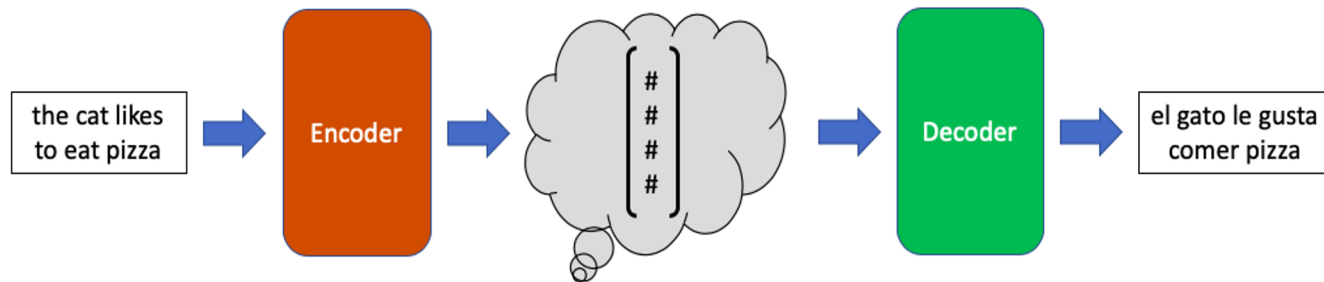
Introduction

Related Work

Models & Comparison

Experiment

Takeaways



# Abstract

Claims: “This paper examines two simple and effective classes of attentional mechanism: a *global* approach which always attends to *all* source words and a *local* one that only looks at a *subset* of source words at a time.”

Key-result: “Our ensemble model using different attention architectures yields a new state-of-the-art result in the WMT’15 English to German translation task with **25.9** BLEU points, an improvement of **1.0** BLEU points over the existing best system backed by NMT and an n-gram reranker.”

# Introduction

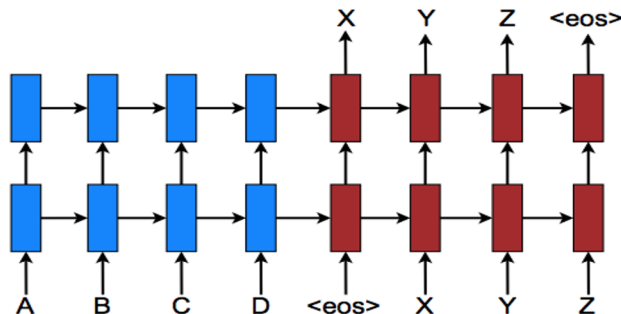
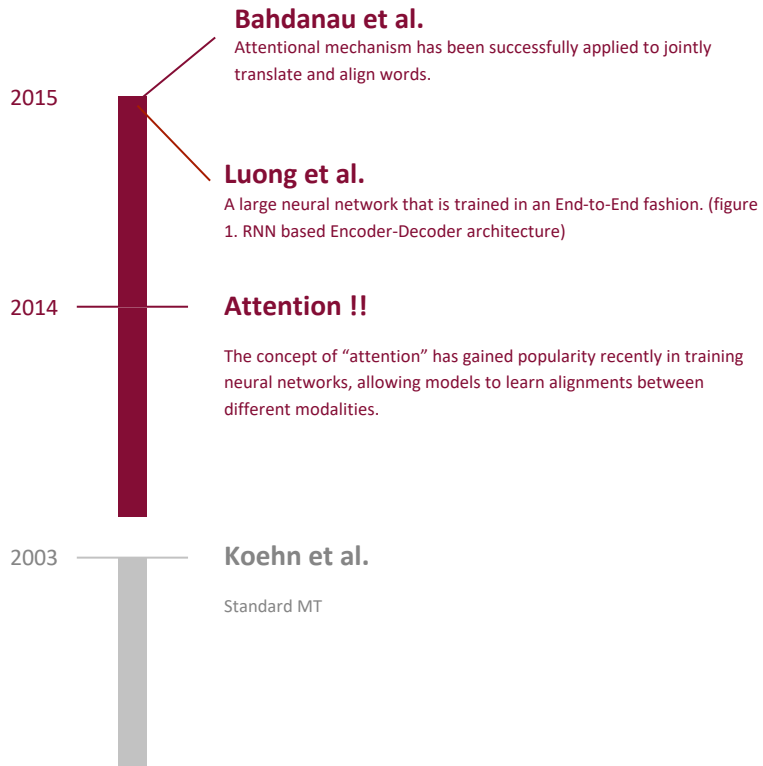


Figure 1: Neural machine translation as a stacking recurrent architecture for translating a source sequence A, B, C, D into a target sequence X, Y, Z . Here <eos> marks the end of a sentence





## Related Work → NMT

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

$$p(y_j | y_{<j}, \mathbf{s}) = \text{softmax}(g(\mathbf{h}_j))$$

$$\mathbf{h}_j = f(\mathbf{h}_{j-1}, \mathbf{s}),$$

where

$g$ : transformation function that outputs a vocabulary size vector

$h$ : RNN hidden unit

$f$ : computes the current hidden state given the previously hidden state.

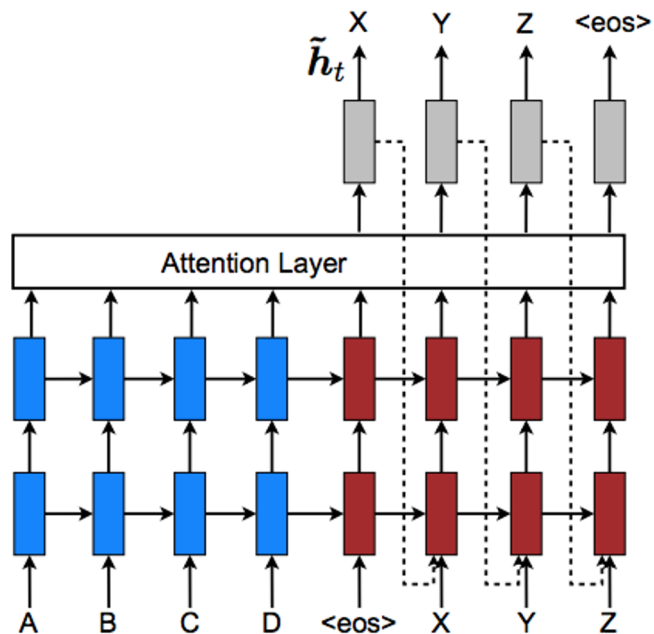
NMT two components:

1. An encoder which computes a representation  $\mathbf{S}$  for each source sentence.
2. A decoder which generates translation one word at a time and hence decomposes the conditional probability. (RNN architecture)

Training objective:

$$J_t = \sum_{(x,y) \in \mathbb{D}} -\log p(y|x)$$

# Related Work



Encoder

# Attention-based Models: Global

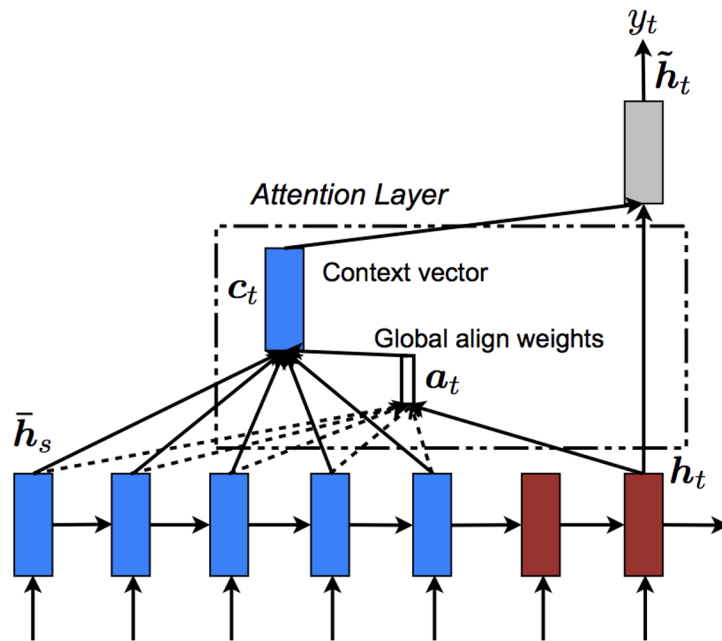
$$\tilde{h}_t = \tanh(\mathbf{W}_c[\mathbf{c}_t; \mathbf{h}_t])$$

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

- Global attentional model

$$\begin{aligned} \mathbf{a}_t(s) &= \text{align}(\mathbf{h}_t, \bar{\mathbf{h}}_s) \\ &= \frac{\exp(\text{score}(\mathbf{h}_t, \bar{\mathbf{h}}_s))}{\sum_{s'} \exp(\text{score}(\mathbf{h}_t, \bar{\mathbf{h}}_{s'}))} \end{aligned}$$

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



$\mathbf{h}(t)$ : Hidden target state

$\mathbf{c}(t)$ : Source side context vector

$\mathbf{y}(t)$ : Current target word

$\mathbf{h\_bar}(t)$ : Attentional hidden state

$\mathbf{a}(t)$ : Alignment vector

# Comparison to (Bahdanau et al., 2015)

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)	<b>5.9</b>	<b>19</b>	<b>20.9 (+1.9)</b>

For content-based functions, our implementation of **concat** does not yield good performances and more analysis should be done to understand the reason...

1. Global : “we simply use hidden states at the top LSTM layers in both the encoder and decoder”;

Previous: use the concatenation of the forward and backward source hidden states in the bi-directional encoder and target hidden states in their non-stacking uni-directional decoder

1. Global: computation path is simpler

Previous: build from the previous hidden state

1. Previous: only experimented with one alignment function: the concat product.

# Attention-based Models: Local

Local Attentional Model

- Small window of context and is differentiable.
- The local alignment vector  $a(t)$  is now fixed-dimensional

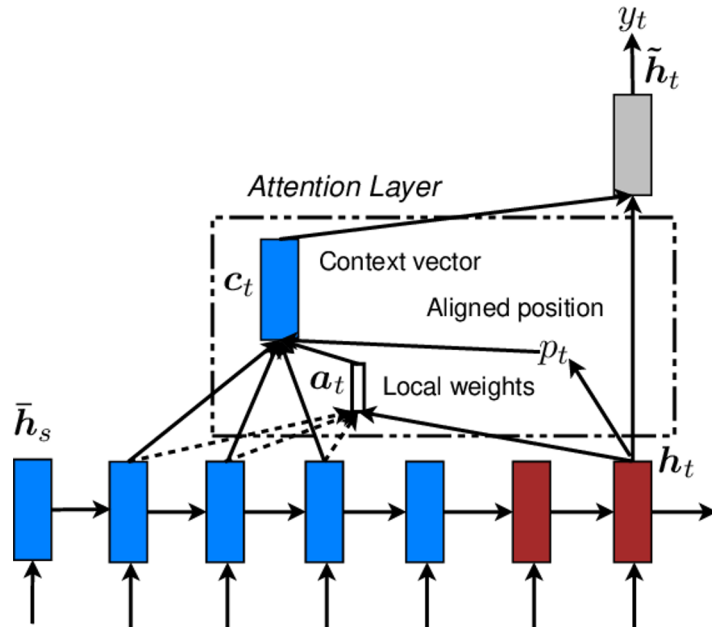
Monotonic Alignment (local-m) :- Global Attention

$$\begin{aligned} a_t(s) &= \text{align}(\mathbf{h}_t, \bar{\mathbf{h}}_s) \\ &= \frac{\exp(\text{score}(\mathbf{h}_t, \bar{\mathbf{h}}_s))}{\sum_{s'} \exp(\text{score}(\mathbf{h}_t, \bar{\mathbf{h}}_{s'}))} \end{aligned}$$

Predictive align

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

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



$\mathbf{W}(\mathbf{p})$  and  $\mathbf{v}(\mathbf{p})$  are models parameters which will be learned to predict positions.  
 $\mathbf{S}$  is the source sentence length  
 $\mathbf{p}(\mathbf{t}): [0, S]$

# Input-feeding Approach

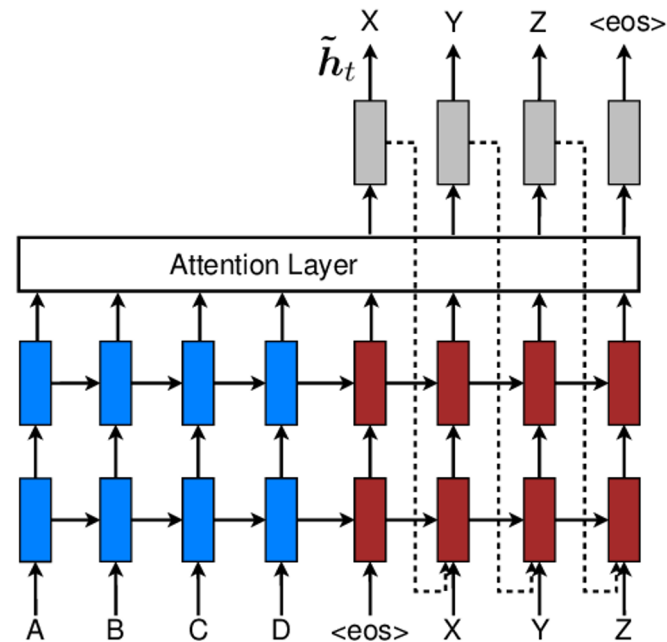
- In the proposed attention mechanisms the attention decisions are made **independently**.

How?

-  $\tilde{h}_t$  is concatenated with inputs at the next time steps as illustrated.

Advantages:

1. Make the model fully aware of the previous alignment choices.
2. Create a very deep network spanning both horizontally and vertically



Input-feeding approach - Attention vectors  $\tilde{h}_t$  are fed as inputs to the next time steps to inform the model about past alignment decisions

# Experiment (WMT' 14 & 15 English-German)

System	Ppl	BLEU
Winning WMT' 14 system – <i>phrase-based + large LM</i> (Buck et al., 2014)		20.7
<i>Existing NMT systems</i>		
RNNsearch (Jean et al., 2015)		16.5
RNNsearch + unk replace (Jean et al., 2015)		19.0
RNNsearch + unk replace + large vocab + <i>ensemble</i> 8 models (Jean et al., 2015)		<b>21.6</b>
<i>Our NMT systems</i>		
Base	10.6	11.3
Base + reverse	9.9	12.6 (+1.3)
Base + reverse + dropout	8.1	14.0 (+1.4)
Base + reverse + dropout + global attention ( <i>location</i> )	7.3	16.8 (+2.8)
Base + reverse + dropout + global attention ( <i>location</i> ) + feed input	6.4	18.1 (+1.3)
Base + reverse + dropout + local-p attention ( <i>general</i> ) + feed input	5.9	19.0 (+0.9)
Base + reverse + dropout + local-p attention ( <i>general</i> ) + feed input + unk replace		20.9 (+1.9)
<i>Ensemble</i> 8 models + unk replace		<b>23.0</b> (+2.1)

System	BLEU
SOTA – <i>NMT + 5-gram rerank</i> (MILA)	24.9
Our ensemble 8 models + unk replace	<b>25.9</b>

**WMT'15 English-German results** -NIST  
BLEU scores of the existing WMT'15 SOTA  
system and our best one on newstest2015.

**WMT'14 English-German results** - shown are the perplexities (ppl) and the tokenized BLEU scores of various systems on newstest 2014. We highlight the **best** system in bold and give progressive improvements in *italic* between consecutive systems. *Local-p* refers to the local attention with predictive alignments. We indicate for each attention model the alignment score function used in parentheses.

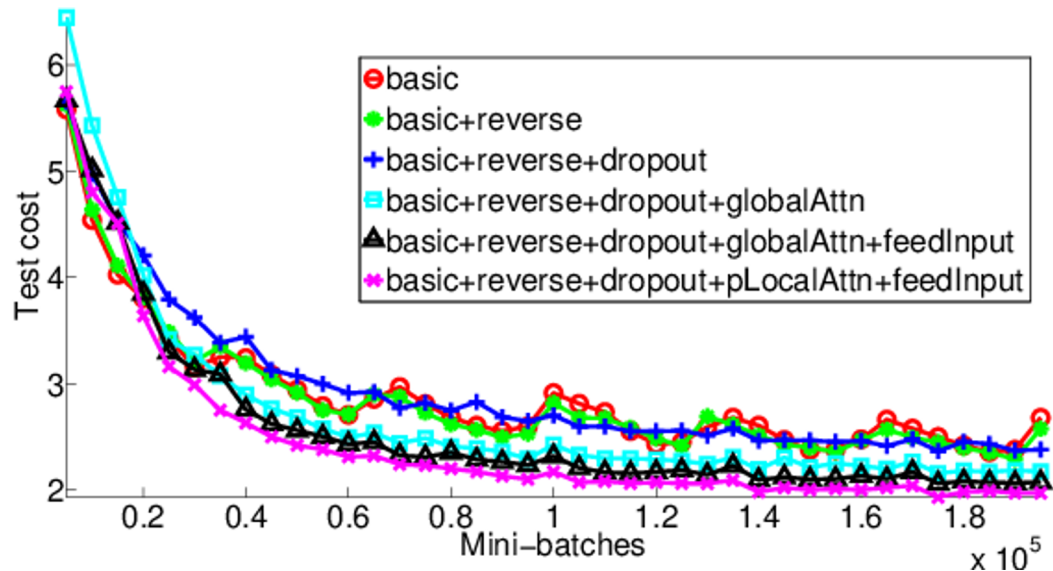
# Experiment (WMT'15 German-English)

System	Ppl.	BLEU
<i>WMT'15 systems</i>		
SOTA – <i>phrase-based</i> (Edinburgh)		<b>29.2</b>
NMT + 5-gram rerank (MILA)		27.6
<i>Our NMT systems</i>		
Base (reverse)	14.3	16.9
+ global ( <i>location</i> )	12.7	19.1 (+2.2)
+ global ( <i>location</i> ) + feed	10.9	20.1 (+1.0)
+ global ( <i>dot</i> ) + drop + feed		22.8 (+2.7)
+ global ( <i>dot</i> ) + drop + feed + unk	9.7	24.9 (+2.1)

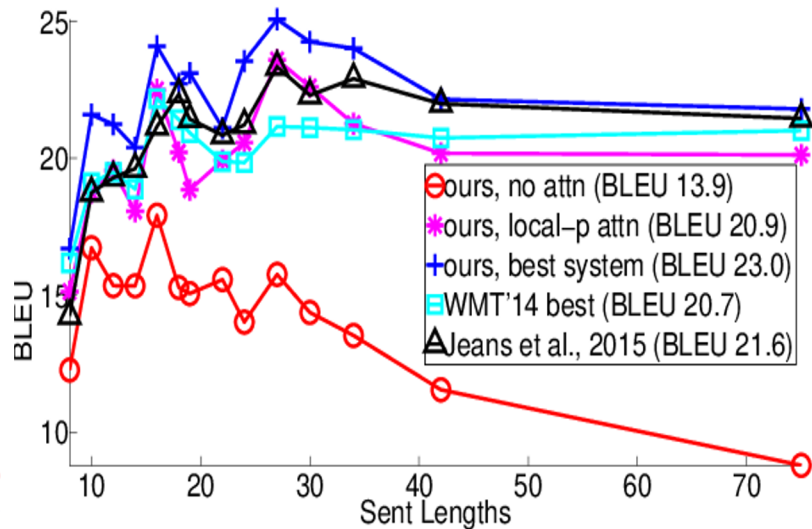
**WMT' 15 German-English results** - performance of various systems. The *base* system already includes source reversing on which we add *global* attention, *dropout*, input *feeding*, and *unk* (universal token) replacement.



# Experiment analysis



**Learning curves** – test cost (ln perplexity) on newstest2014 for English-German NMTs as training progresses



**Length Analysis** - the translation quality does not degrade as sentences become longer. Our best model (blue + curve) outperforms all other systems in all length buckets.

# Takeaways


1. This work proposes two simple and effective attentional mechanisms for NMT: global which always looks at **all source** positions and local one which only attends to a **subset of source** positions at a time.
2. This work compared **various alignment functions** and shed light on which functions are the best for which attentional models.
3. The dependencies between previous alignment information and current alignment decisions take into consideration.
4. Attentional **beats** non-attentional

# Neural Machine Translation of **Rare Words** with **Subword Units**

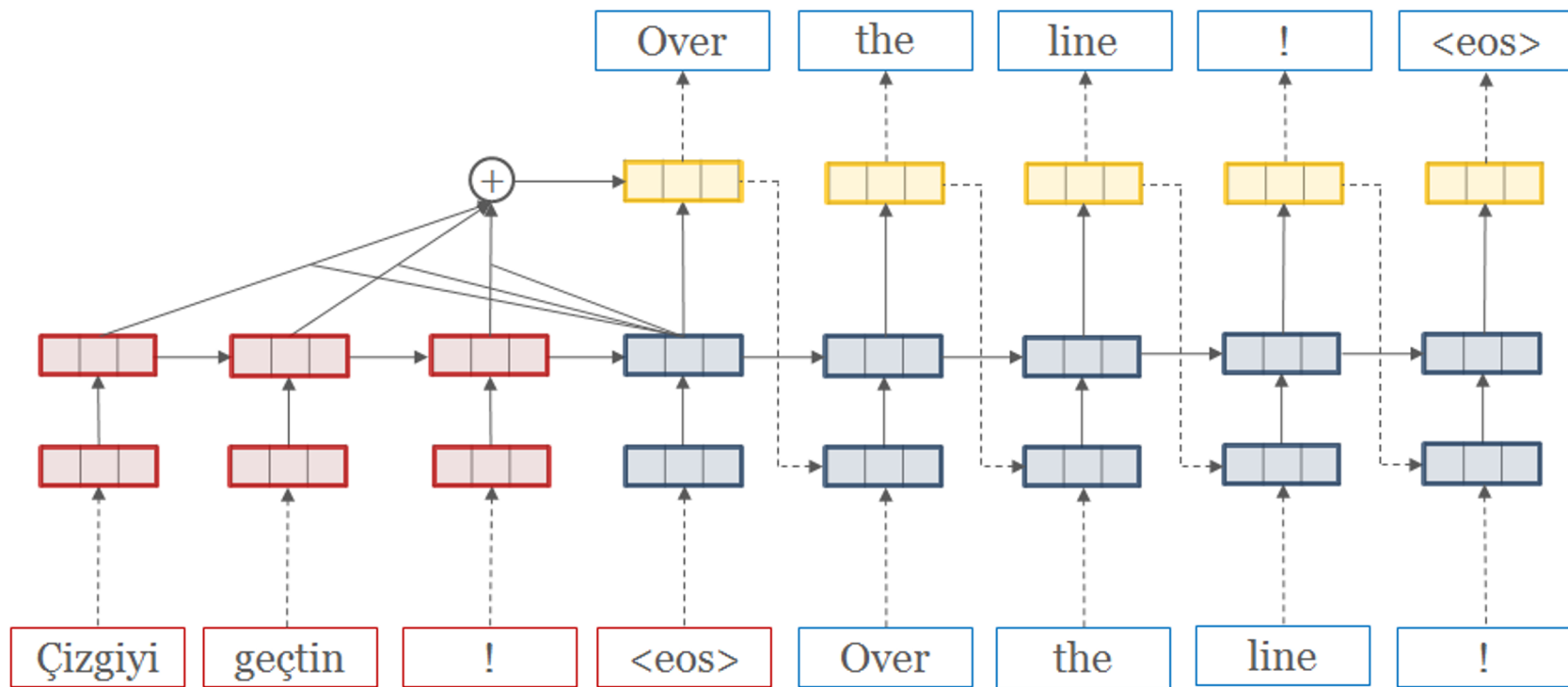
Rico Sennrich, Barry Haddow, Alexandra Birch

Presented by: Wei Liu

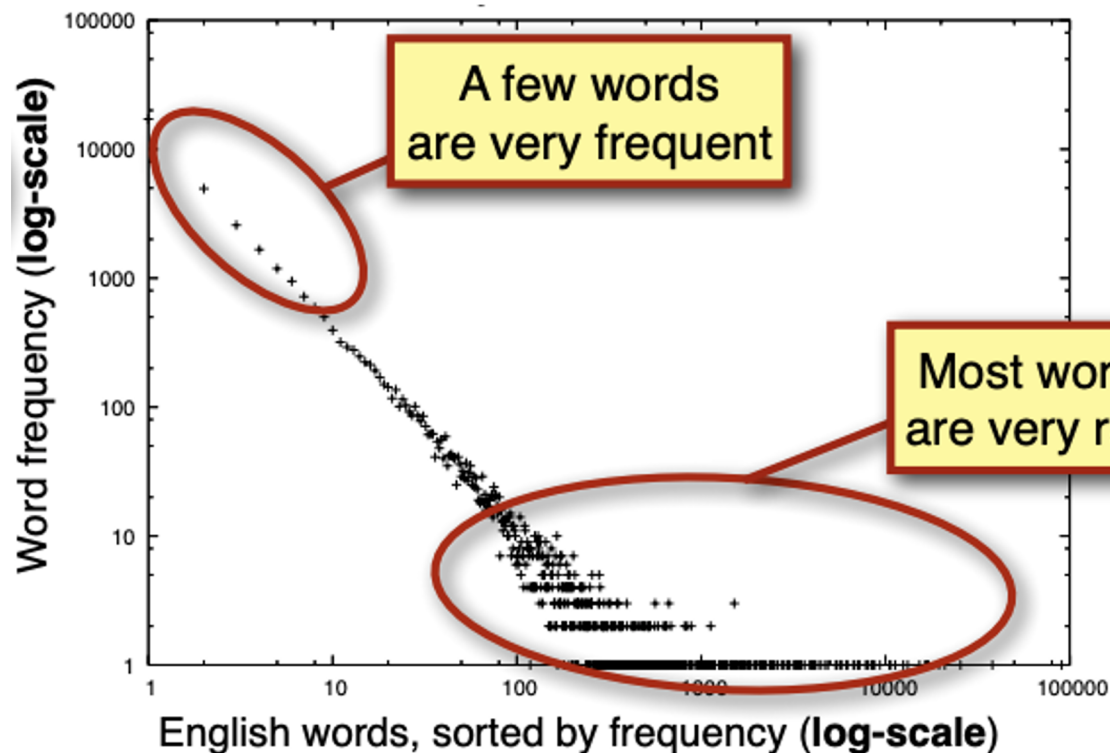
# Outline

- 
- Motivation
  - Contribution
    - Byte Pair Encoding for word segmentation
    - Variants of Byte Pair Encoding
  - Model
  - Evaluation
  - Conclusion

# Recap: NMT



# Motivation



$w_1 = the, w_2 = to, \dots, w_{5346} = computer, \dots$

# Motivation



***German: Donaudampfschiffahrtselektrizitätenhaupt-  
betriebswer-kbauunterbeamten-gesellschaft***

***English:***

***Association for Subordinate Officials of the Main  
Maintenance Building of the Danube Steam Shipping  
Electrical Services***

# Motivation



## Transparent Word:

Words that are translatable by a competent translator even if they are novel to him/her.

- Named Entities
  - Barack Obama (English)
  - バラクオバマ (Japanese)
- Cognates and Loanwords
  - Claustrophobia (English)
  - Klaustrophobie (German)
- Morphologically complex words
  - Solar System(English)
  - Sonnensystem(German)





**Solution?**  
**Goto subword**  
**level!**

# Contribution



---

## 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)
```

---

# Contribution



What is Byte Pair Encoding?

→ aa**ab**daa**ba**c

→ Z**ab**dZ**ab**c

Z=aa

→ Z**Y**dZ**Y**ac

Y=ab

Z=aa

→ XdXac

X=ZY

Y=ab

Z=aa

### Dictionary

5 low  
2 lower  
6 new **es** t  
3 wide **es** t

### Vocabulary

l, o, w, e, r, n, w, s, t, i, d

Start with all characters  
in vocab

### Dictionary

5 low  
2 lower  
6 new **es** t  
3 wide **es** t

### Vocabulary

l, o, w, e, r, n, w, s, t, i, d, es

Add a pair (e, s) with freq 9

### Dictionary

5 low  
2 lower  
6 newest  
3 widest

### Vocabulary

l, o, w, e, r, n, w, s, t, i, d, es, **est**

Add a pair (es, t) with freq 9

### Dictionary

5 low  
2 lower  
6 newest  
3 widest

### Vocabulary

l, o, w, e, r, n, w, s, t, i, d, es, est, **lo**

Add a pair (l, o) with freq 7

# Variants

1. Learn **two independent** encodings.  
One for the source vocabulary,  
one for the target vocabulary.
1. Learn **one** encoding on  
the **union** of the two vocabularies.

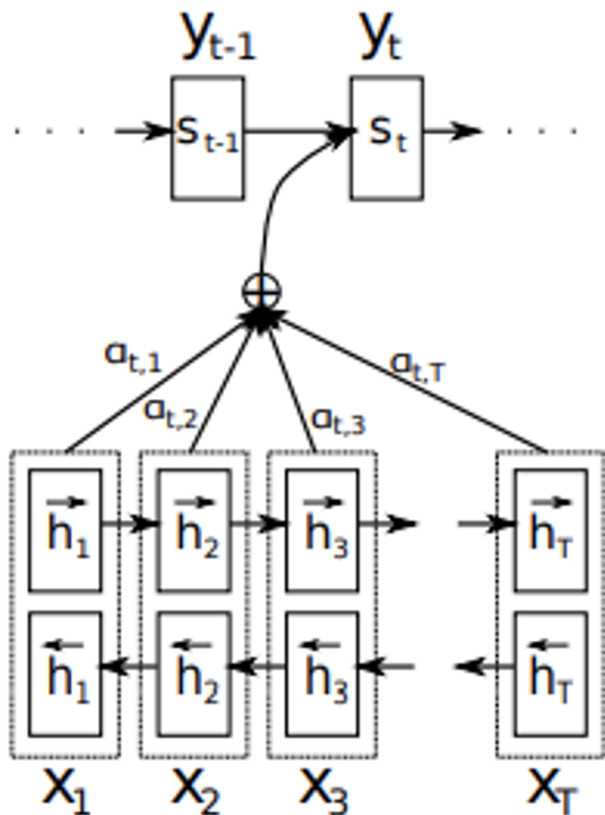
Note: For languages use different alphabet,  
like Russian and English, first transliterate  
Russian vocabulary into Latin characters.

# Transliteration

## RUSSIAN VIRTUAL KEYBOARD

а	а	и	i	с	s	ъ	"
б	b	й	j	т	t	ы	y
в	v,w	к	k	у	u	ь	'
г	g	л	l	ф	f	э	je,ä
д	d	м	m	х	h,x	ю	ju, yu,ü
е	e	н	n	ц	c	я	ja, ya,q
ё	jo, yo,ö	о	o	ч	ch	№	#
ж	zh	п	p	ш	sh	«	
з	z	р	r	щ	shh	»	

# Model: Neural Machine Translation by Jointly Learning to Align and Translate (Bahdanau et al. 2015)



*Encoder:*  
Bidirectional  
Gated Recurrent Unit

*Decoder:*  
Recurrent Neural  
Network



# Evaluation



English → German

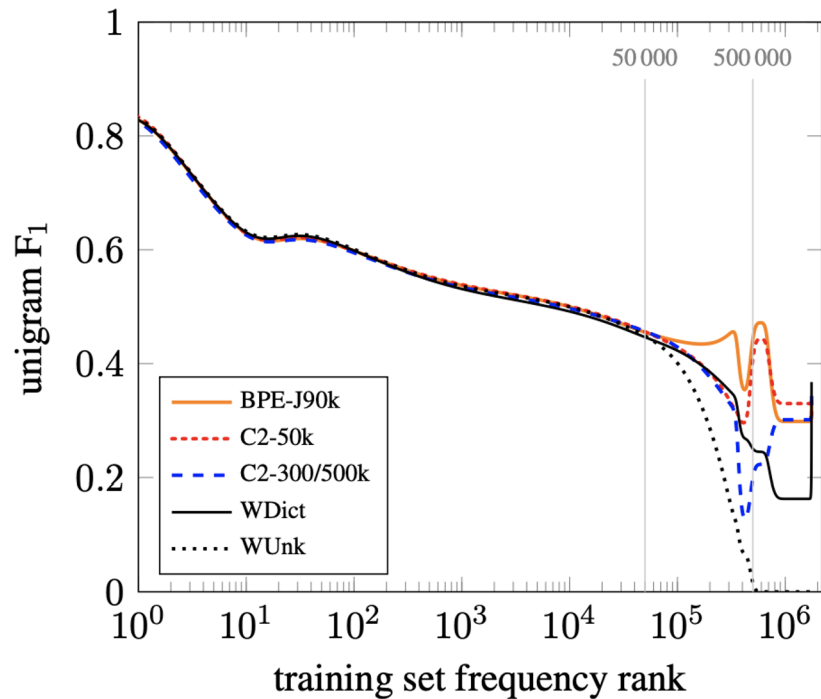
Basic BPE →  
Joint BPE →

name	segmentation	shortlist	vocabulary		BLEU		CHRF3		unigram F <sub>1</sub> (%)		
			source	target	single	ens-8	single	ens-8	all	rare	OOV
syntax-based (Sennrich and Haddow, 2015)					24.4	-	55.3	-	59.1	46.0	37.7
WUnk	-	-	300 000	500 000	20.6	22.8	47.2	48.9	56.7	20.4	0.0
WDict	-	-	300 000	500 000	22.0	24.2	50.5	52.4	58.1	36.8	<b>36.8</b>
C2-50k	char-bigram	50 000	60 000	60 000	<b>22.8</b>	<b>25.3</b>	51.9	53.5	58.4	40.5	30.9
BPE-60k	BPE	-	60 000	60 000	<b>21.5</b>	<b>24.5</b>	<b>52.0</b>	53.9	<b>58.4</b>	<b>40.9</b>	<b>29.3</b>
BPE-J90k	BPE (joint)	-	90 000	90 000	<b>22.8</b>	<b>24.7</b>	51.7	<b>54.1</b>	<b>58.5</b>	<b>41.8</b>	<b>33.6</b>

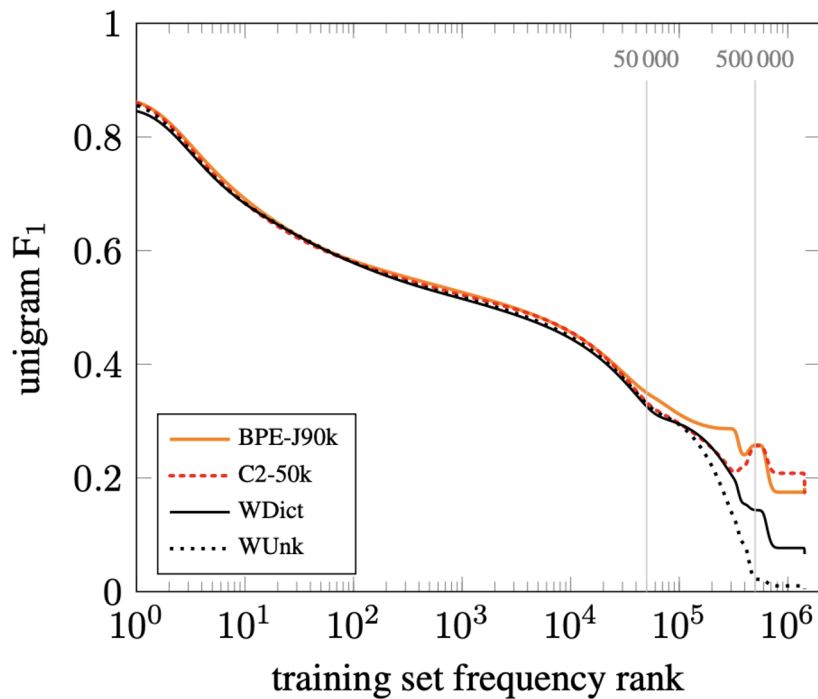
English → Russian

name	segmentation	shortlist	vocabulary		BLEU		CHRF3		unigram F <sub>1</sub> (%)		
			source	target	single	ens-8	single	ens-8	all	rare	OOV
phrase-based (Haddow et al., 2015)					24.3	-	53.8	-	56.0	31.3	16.5
WUnk	-	-	300 000	500 000	18.8	22.4	46.5	49.9	54.2	25.2	0.0
WDict	-	-	300 000	500 000	19.1	22.8	47.5	51.0	54.8	26.5	6.6
C2-50k	char-bigram	50 000	60 000	60 000	<b>20.9</b>	<b>24.1</b>	49.0	51.6	55.2	27.8	17.4
BPE-60k	BPE	-	60 000	60 000	<b>20.5</b>	<b>23.6</b>	<b>49.8</b>	52.7	<b>55.3</b>	<b>29.7</b>	<b>15.6</b>
BPE-J90k	BPE (joint)	-	90 000	100 000	<b>20.4</b>	<b>24.1</b>	49.7	<b>53.0</b>	<b>55.8</b>	<b>29.7</b>	<b>18.3</b>

# Evaluation

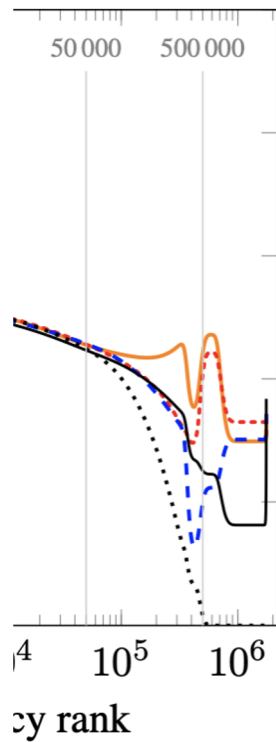


English → German

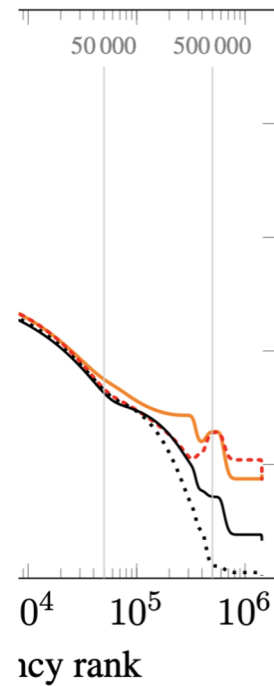


English → Russian

# Evaluation



English → German



English → Russian

# Conclusion

What is Byte Pair Encoding?

- It is just a subword-level encoding technique.

What's the advantage of using it?

- Better accuracy for the translation of rare words.
- Relative lower vocabulary size compared to character n-grams.

What's the drawback?

- Longer training time. Backprop through time over a much longer sequence.
- Longer runtime.

Is it still being used now?

- Yes, very often. For example, RoBERTa, Google NMT.

# Convolutional Sequence to Sequence Learning

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, Yann N. Dauphin (Facebook 2017)

Presenter: Yujia Qiu

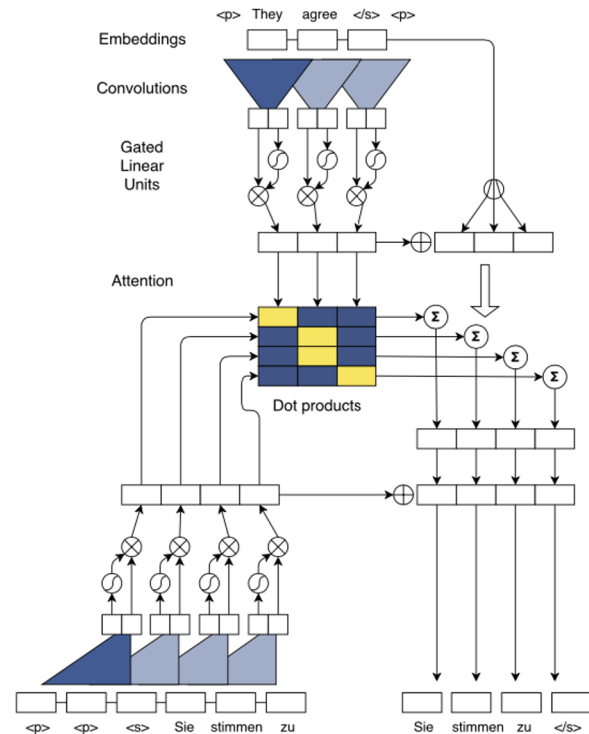


# Motivation

- RNNs maintain a hidden state of the entire past that prevents parallel computation within a sequence. CNN does not depend on previous time step -> Parallelization.
- CNN creates a hierarchical structure provides a shorter path to capture long-range dependencies compared to RNN
  - RNN  $O(n)$  -> CNN  $O(n/k)$

# Model Architecture

- Embedding
  - Embed  $x = (x_1, \dots, x_m)$  to  $w = (w_1, \dots, w_m)$
  - Position embeddings  $p = (p_1, \dots, p_m)$
  - $e = (w_1 + p_1, \dots, w_m + p_m)$
- Output of decoder states  $h$
- Output of encoder states  $z$



# Convolutional Block Architecture

- 1-D Convolution (kernel width  $k$ )

$$\begin{matrix} W \in \mathbb{R}^{2d \times kd} & b_w \in \mathbb{R}^{2d} \\ \hat{X} \in \mathbb{R}^{k \times d} & \longrightarrow & Y \in \mathbb{R}^{2d} \end{matrix}$$

- Non-linearity (GLU)  $Y = [A \ B] \in \mathbb{R}^{2d}$ :

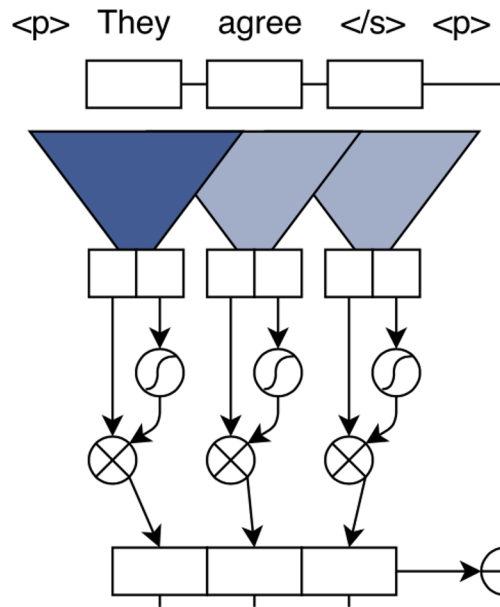
- Gated linear units

$$v([A \ B]) = A \otimes \sigma(B)$$

Embeddings

Convolutions

Gated  
Linear  
Units





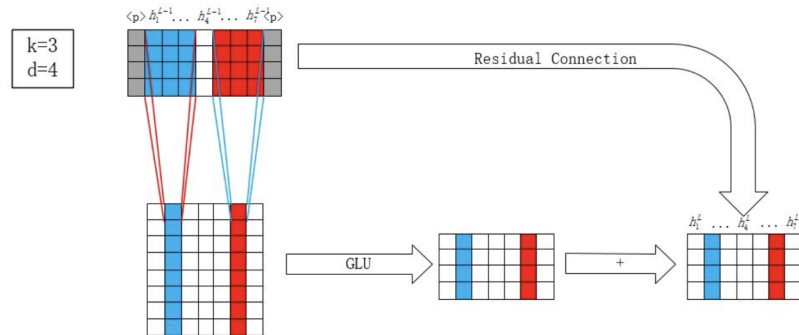
# Convolutional Block Architecture

To enable deep convolutional networks, residual connections are added from the input of each convolution to the output of the block

$$h_i^l = v(W^l[h_{i-k/2}^{l-1}, \dots, h_{i+k/2}^{l-1}] + b_w^l) + h_i^{l-1}$$

After the last decoder, compute distribution over the  $T$  possible next target elements  $y_{i+1}$ ,

$$p(y_{i+1} | y_1, \dots, y_i, \mathbf{x}) = \text{softmax}(W_o h_i^L + b_o) \in \mathbb{R}^T$$



# Multi-step Attention

- Combine current decoder  $h_i$  with an embedding of previous target element  $g_i$

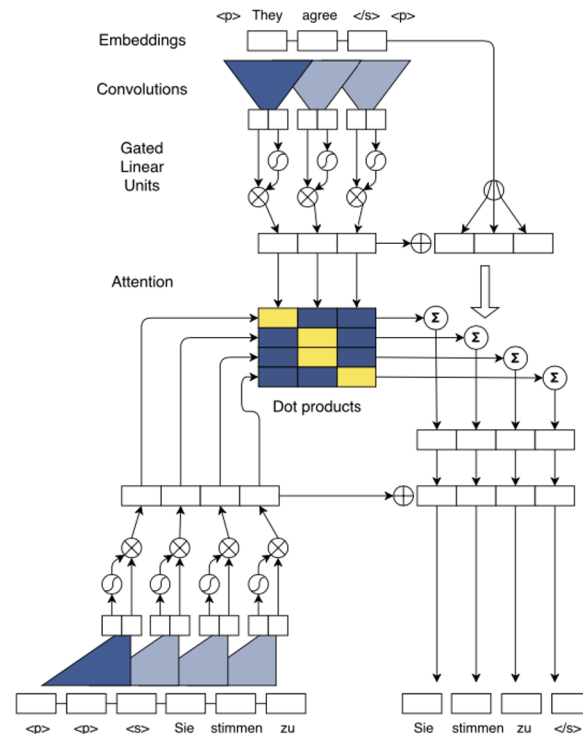
$$d_i^l = W_d^l h_i^l + b_d^l + g_i$$

- Attention (Decoder  $d_i$  and  $z_j$  of last encoder block  $u$ )

$$a_{ij}^l = \frac{\exp(d_i^l \cdot z_j^u)}{\sum_{t=1}^m \exp(d_i^l \cdot z_t^u)}$$

- Conditional input  $c_i$ , weighted sum over  $z_j$ 
  - $e_j$  provides point information, which is beneficial

$$c_i^l = \sum_{j=1}^m a_{ij}^l (z_j^u + e_j)$$





# Normalization & Initialization

- Normalization

- Multiply the sum of input and output of a residual block by  $\sqrt{0.5}$  to halve the variance of the sum
- Conditional input  $c_i$  is a weighted sum of  $m$  vectors, then the variance is scaling by  $m\sqrt{1/m}$ . Multiply by  $m$  to scale up the inputs to their original size.
- Convolutional decoder with multiple attentions, scale the gradients for the encoder layers by the number of attention mechanisms used.

- Initialization

- All embeddings are initialized from a normal distribution with mean 0 and std 1
- For layers whose output is not directly fed to a gated linear unit, initialize weights from  $\mathcal{N}(0, \sqrt{1/n_l})$   
 $n_l$  is the number of input connections to each neuron  $\rightarrow$  make the variance retained.
- For layers followed by GLU activation, weights are  $\mathcal{N}(0, \sqrt{4/n_l})$  if variance are small
- Apply dropouts to restore the variance.



# Datasets

- WMT'16 English-Romanian (2.8M sentences pairs)
- WMT'14 English-German (4.5M sentences pairs)
- WMT'14 English-French (35.5M sentences pairs)



# Results

WMT'16 English-Romanian	BLEU
<a href="#">Sennrich et al. (2016b)</a> GRU (BPE 90K)	28.1
ConvS2S (Word 80K)	29.45
ConvS2S (BPE 40K)	30.02

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

WMT'14 English-French	BLEU
<a href="#">Wu et al. (2016)</a> GNMT (Word 80K)	37.90
<a href="#">Wu et al. (2016)</a> GNMT (Word pieces)	38.95
<a href="#">Wu et al. (2016)</a> GNMT (Word pieces) + RL	39.92
ConvS2S (BPE 40K)	40.51

Table 1. Accuracy on WMT tasks compared to previous work. ConvS2S and GNMT results are averaged over several runs.



# Results

WMT'14 English-German	BLEU
Wu et al. (2016) GNMT	26.20
Wu et al. (2016) GNMT + RL	26.30
ConvS2S	26.43
WMT'14 English-French	BLEU
Zhou et al. (2016)	40.4
Wu et al. (2016) GNMT	40.35
Wu et al. (2016) GNMT + RL	41.16
ConvS2S	41.44
ConvS2S (10 models)	41.62

Table 2. Accuracy of ensembles with eight models. We show both likelihood and Reinforce (RL) results for GNMT; Zhou et al. (2016) and ConvS2S use simple likelihood training.



# Generation Speed

	BLEU	Time (s)
GNMT GPU (K80)	31.20	3,028
GNMT CPU 88 cores	31.20	1,322
GNMT TPU	31.21	384
ConvS2S GPU (K40) $b=1$	33.45	327
ConvS2S GPU (M40) $b=1$	33.45	221
ConvS2S GPU (GTX-1080ti) $b=1$	33.45	142
ConvS2S CPU 48 cores $b=1$	33.45	142
ConvS2S GPU (K40) $b=5$	34.10	587
ConvS2S CPU 48 cores $b=5$	34.10	482
ConvS2S GPU (M40) $b=5$	34.10	406
ConvS2S GPU (GTX-1080ti) $b=5$	34.10	256

Table 3. CPU and GPU generation speed in seconds on the development set of WMT’14 English-French. We show results for different beam sizes  $b$ . GNMT figures are taken from [Wu et al. \(2016\)](#). CPU speeds are not directly comparable because [Wu et al. \(2016\)](#) use a 88 core machine versus our 48 core setup.



# Results

	PPL	BLEU
ConvS2S	6.64	21.7
-source position	6.69	21.3
-target position	6.63	21.5
-source & target position	6.68	21.2

*Table 4.* Effect of removing position embeddings from our model in terms of validation perplexity (valid PPL) and BLEU.

Position embeddings allow the model to identify the source and target sequence. Removing source position embedding results in a larger accuracy decrease than target position embeddings.

Model can learn relative position information within the contexts visible to encoder & decoder





# My thoughts

- Advantages:
  - Accuracy improvement
  - Fast speed
- Disadvantages:
  - It needs more parameters tuning when doing normalization & initialization
  - Limited range of dependency
    - kernel width  $k$ , the dependency will only be  $\alpha(k-1)+1$  inputs

# Phrase-Based & Neural Unsupervised Machine Translation

G. Lample et al. (2018)

Presenter: Ashwin Ramesh



# Outline

Machine Translation (MT) Background

Principles of Unsupervised MT

Unsupervised NMT and PBSMT

Experiments

Results

Conclusion





# Outline

## Machine Translation (MT) Background

Principles of Unsupervised MT

Unsupervised NMT and PBSMT

Experiments

Results

Conclusion





# **Background : Supervised Machine Translation**



## Background : Supervised Machine Translation

- Using large bilingual text corpus, you train an encoder-decoder pair to translate from source sentences to target sentences.

# Background : Supervised Machine Translation

- Using large bilingual text corpus, you train an encoder-decoder pair to translate from source sentences to target sentences.
- Problem:

# Background : Supervised Machine Translation

- Using large bilingual text corpus, you train an encoder-decoder pair to translate from source sentences to target sentences.
- **Problem:** Many language pairs do not have large parallel text corpora, these are referred to as *low-resource* languages.



# Background : Supervised Machine Translation

- Using large bilingual text corpus, you train an encoder-decoder pair to translate from source sentences to target sentences.
- **Problem:** Many language pairs do not have large parallel text corpora, these are referred to as *low-resource* languages.
- **Solution:**

# Background : Supervised Machine Translation

- Using large bilingual text corpus, you train an encoder-decoder pair to translate from source sentences to target sentences.
- **Problem:** Many language pairs do not have large parallel text corpora, these are referred to as *low-resource* languages.
- **Solution:** Automatically generate source and target sentence pairs to turn unsupervised into supervised!

# Background : Unsupervised Machine Translation

- Builds on two previous works

# Background : Unsupervised Machine Translation

- Builds on two previous works
  - G. Lample, A. Conneau, L. Denoyer, and M. Ranzato. 2018. Unsupervised machine translation using monolingual corpora only. In International Conference on Learning Representations (ICLR).
  - Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2018. Unsupervised neural machine translation. In International Conference on Learning Representations (ICLR)

# Background : Unsupervised Machine Translation

- Builds on two previous works
  - G. Lample, A. Conneau, L. Denoyer, and M. Ranzato. 2018. Unsupervised machine translation using monolingual corpora only. In International Conference on Learning Representations (ICLR).
  - Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2018. Unsupervised neural machine translation. In International Conference on Learning Representations (ICLR)
- Distills and improves on the 3 common principles underlying the success of the above works.



# Outline

## Machine Translation (MT) Background

Principles of Unsupervised MT

Unsupervised NMT and PBSMT

Experiments

Results

Conclusion





# Outline

Machine Translation (MT) Background

**Principles of Unsupervised MT**

Unsupervised NMT and PBSMT

Experiments

Results

Conclusion





# Principles of Unsupervised MT : Algorithm

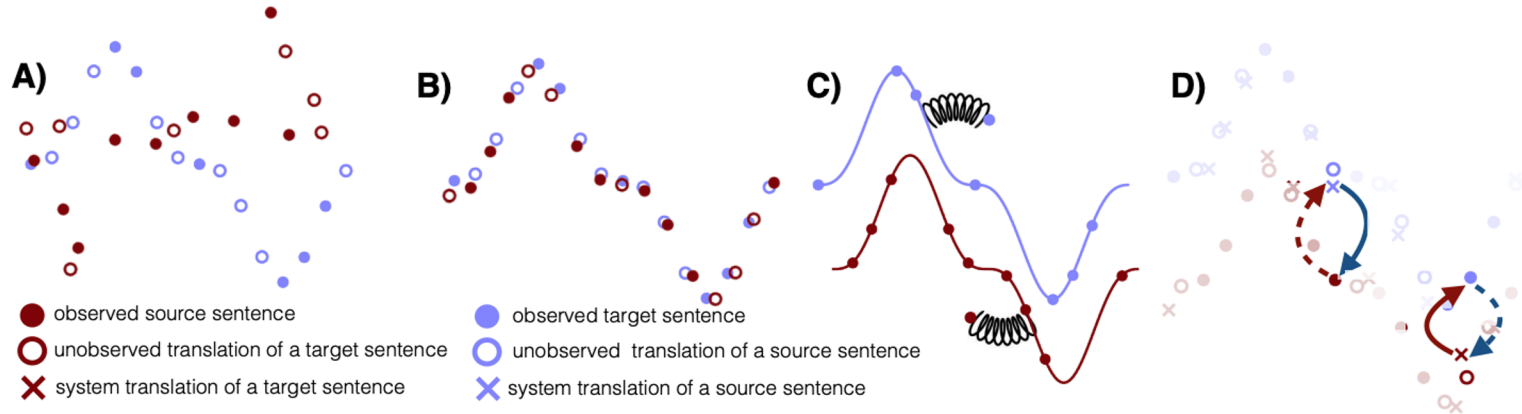




# Principles of Unsupervised MT : Algorithm

1. Initialize Translation Models  $p^{(0)}_{s \rightarrow t}$  and  $p^{(0)}_{t \rightarrow s}$  .

# Principles of Unsupervised MT : Language Models



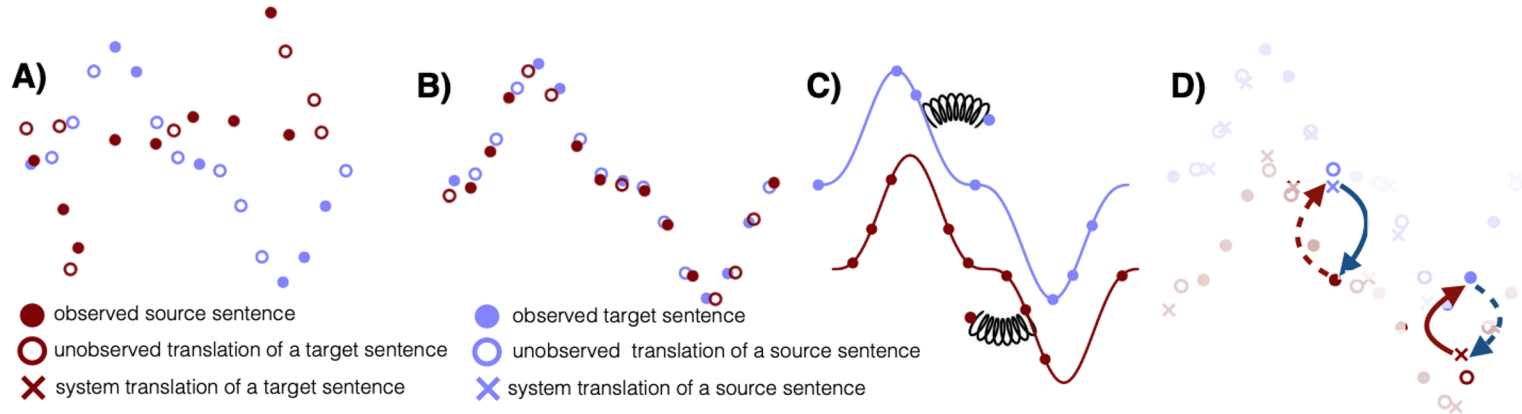
# Principles of Unsupervised MT : Algorithm

1. Initialize Translation Models  $p^{(0)}_{s \rightarrow t}$  and  $p^{(0)}_{t \rightarrow s}$  .

# Principles of Unsupervised MT : Algorithm

1. **Initialize Translation Models**  $P^{(0)}_{s \rightarrow t}$  and  $P^{(0)}_{t \rightarrow s}$  .
2. **Language models** : Learn two language models,  $P_s$  and  $P_t$  , over source and target languages.

# Principles of Unsupervised MT : Initialization



# Principles of Unsupervised MT : Algorithm

1. **Initialize Translation Models**  $P^{(0)}_{s \rightarrow t}$  and  $P^{(0)}_{t \rightarrow s}$  .
2. **Language models** : Learn two language models,  $P_s$  and  $P_t$  , over source and target languages.

# Principles of Unsupervised MT : Algorithm

1. **Initialize Translation Models**  $P^{(0)}_{s \rightarrow t}$  and  $P^{(0)}_{t \rightarrow s}$  .
2. **Language models** : Learn two language models,  $P_s$  and  $P_t$  , over source and target languages.
3. **for**  $k = 1$  **to**  $N$  **do**

end

# Principles of Unsupervised MT : Algorithm

1. **Initialize Translation Models**  $P^{(0)}_{s \rightarrow t}$  and  $P^{(0)}_{t \rightarrow s}$  .
2. **Language models** : Learn two language models,  $P_s$  and  $P_t$  , over source and target languages.
3. **for**  $k = 1$  **to**  $N$  **do**
  - i. **Back Translation** : Use  $P^{(k-1)}_{s \rightarrow t}$  ,  $P^{(k-1)}_{t \rightarrow s}$  ,  $P_s$  and  $P_t$  to generate source and target sentences

end

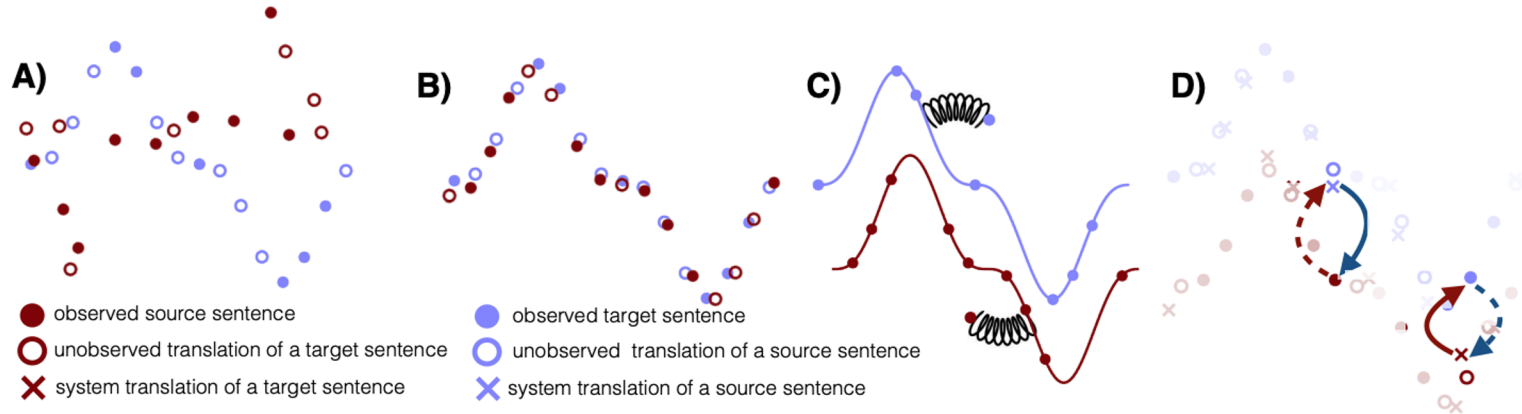


# Principles of Unsupervised MT : Algorithm

1. **Initialize Translation Models**  $P^{(0)}_{s \rightarrow t}$  and  $P^{(0)}_{t \rightarrow s}$ .
2. **Language models** : Learn two language models,  $P_s$  and  $P_t$ , over source and target languages.
3. **for**  $k = 1$  **to**  $N$  **do**
  - i. **Back Translation** : Use  $P^{(k-1)}_{s \rightarrow t}$ ,  $P^{(k-1)}_{t \rightarrow s}$ ,  $P_s$  and  $P_t$  to generate source and target sentences
  - i. Train new translation models  $P^{(k)}_{s \rightarrow t}$  and  $P^{(k)}_{t \rightarrow s}$ , using the generated sentences and  $P_s$  and  $P_t$ .

**end**

# Principles of Unsupervised MT : Back Translation





# Outline

Machine Translation (MT) Background

**Principles of Unsupervised MT**

Unsupervised NMT and PBSMT

Experiments

Results

Conclusion





# Outline

Machine Translation (MT) Background

Principles of Unsupervised MT

**Unsupervised NMT and PBSMT**

Experiments

Results

Conclusion





# Unsupervised NMT : Models



# Unsupervised NMT : Models

2 types of models

# Unsupervised NMT : Models

2 types of models

- LSTM-based
  - Encoder, decoder : 3-layer bidirectional LSTM.
  - Encoders and decoders share LSTM weights across source and target

# Unsupervised NMT : Models

2 types of models

- LSTM-based
  - Encoder, decoder : 3-layer bidirectional LSTM.
  - Encoders and decoders share LSTM weights across source and target
- Transformer-based
  - 4 -layer encoder and decoder



# Unsupervised NMT : Initialization

2 main contributions :

# Unsupervised NMT : Initialization

2 main contributions :

- *Byte-Pair Encodings* (BPEs) were used.
  - Reduce vocabulary size
  - Eliminate the presence of unknown words in the output translation

# Unsupervised NMT : Initialization

2 main contributions :

- *Byte-Pair Encodings* (BPEs) were used.
  - Reduce vocabulary size
  - Eliminate the presence of unknown words in the output translation
- Learn token embeddings from the byte pair tokenization of joint corpora and use these to initialize the lookup tables in the encoder and decoder.

# Unsupervised NMT : Language Modelling

- Language modelling is accomplished via denoising auto-encoding.

# Unsupervised NMT : Language Modelling

- Language modelling is accomplished via denoising auto-encoding.
- The language model aims to minimize :

$$\mathcal{L}^{lm} = \mathbb{E}_{x \sim \mathcal{S}}[-\log P_{s \rightarrow s}(x|C(x))] + \mathbb{E}_{y \sim \mathcal{T}}[-\log P_{t \rightarrow t}(y|C(y))]$$

$C$  is a noise model and  $P_{s \rightarrow s}$  and  $P_{t \rightarrow t}$  are the composite encoder-decoder pairs for the source and target languages respectively.



# Unsupervised NMT : Back-Translation

# Unsupervised NMT : Back-Translation

- Let  $x \in S$  and  $y \in T$ 
  - $u^*(y) = \operatorname{argmax}_u P^{(k-1)}_{t \rightarrow s}(u | y).$
  - $v^*(x) = \operatorname{argmax}_v P^{(k-1)}_{s \rightarrow t}(v | x).$

# Unsupervised NMT : Back-Translation

- Let  $x \in S$  and  $y \in T$ 
  - $u^*(y) = \operatorname{argmax}_u P^{(k-1)}_{t \rightarrow s}(u | y).$
  - $v^*(x) = \operatorname{argmax}_v P^{(k-1)}_{s \rightarrow t}(v | x).$
- The pairs  $(u^*(y), y)$  and  $(x, v^*(x))$  are automatically generated parallel sentences that can be used to train  $P^{(k)}_{s \rightarrow t}$  and  $P^{(k)}_{t \rightarrow s}$  using the back-translation principle.



# Unsupervised NMT : Back-Translation

- The models are trained by minimizing:

$$\mathcal{L}^{back} = \mathbb{E}_{y \sim \mathcal{T}}[-\log P_{s \rightarrow t}(y|u^*(y))] + \mathbb{E}_{x \sim \mathcal{S}}[-\log P_{t \rightarrow s}(x|v^*(x))].$$

# Unsupervised NMT : Back-Translation

- The models are trained by minimizing:

$$\mathcal{L}^{back} = \mathbb{E}_{y \sim \mathcal{T}}[-\log P_{s \rightarrow t}(y|u^*(y))] + \mathbb{E}_{x \sim \mathcal{S}}[-\log P_{t \rightarrow s}(x|v^*(x))].$$

- The models are not trained via back-propagation through the reverse model but rather just by minimizing  $L_{back} + L_{lm}$  at every iteration of stochastic gradient descent.



# Unsupervised PBSMT : Models



# Unsupervised PBSMT : Models

- PBSMT :
  - $\operatorname{argmax}_y P(y|x) = \operatorname{argmax}_y P(x|y) P(y).$
  - $P(x|y)$  : phrase tables
  - $P(y)$  : language model

# Unsupervised PBSMT : Models

- PBSMT :
  - $\operatorname{argmax}_y P(y|x) = \operatorname{argmax}_y P(x|y) P(y).$
  - $P(x|y)$  : phrase tables
  - $P(y)$  : language model
- PBSMT uses a smoothed  $n$ -gram language model.

# Unsupervised PBSMT : Initialization

# Unsupervised PBSMT : Initialization

- Need to populate source-target and target-source phrase tables!

# Unsupervised PBSMT : Initialization

- Need to populate source-target and target-source phrase tables!
  - Conneau et al. (2018) : Infer bilingual dictionary from 2 monolingual corpora.



# Unsupervised PBSMT : Initialization

- Need to populate source-target and target-source phrase tables!
  - Conneau et al. (2018) : Infer bilingual dictionary from 2 monolingual corpora.
  - Phrase tables are populated with scores using :

$$p(t_j|s_i) = \frac{e^{\frac{1}{T} \cos(e(t_j), We(s_i))}}{\sum_k e^{\frac{1}{T} \cos(e(t_k), We(s_i))}},$$



# Unsupervised PBSMT : Language Modelling



# Unsupervised PBSMT : Language Modelling

- Smoothed n-gram language models are learned using KenLM (Heafield, 2011).

# Unsupervised PBSMT : Language Modelling

- Smoothed n-gram language models are learned using KenLM (Heafield, 2011).
- These remain fixed throughout back-translation iterations.



# Unsupervised PBSMT : Back-Translation Algorithm



# Unsupervised PBSMT : Back-Translation Algorithm

- Learn  $P^{(0)}_{s \rightarrow t}$  from phrase tables and language model, and get  $D^{(0)}_t$  using  $P^{(0)}_{s \rightarrow t}$  on source corpus.

# Unsupervised PBSMT : Back-Translation Algorithm

- Learn  $P^{(0)}_{s \rightarrow t}$  from phrase tables and language model, and get  $D^{(0)}_t$  using  $P^{(0)}_{s \rightarrow t}$  on source corpus.
- **for**  $k = 1$  **to**  $N$  **do**
  - Train  $P^{(k)}_{t \rightarrow s}$  using  $D^{(k-1)}_t$ .
  - **Back Translation** :  $P^{(k)}_{t \rightarrow s}$  on target corpus gives  $D^{(k)}_s$
  - Train  $P^{(k)}_{s \rightarrow t}$  using  $D^{(k)}_s$ .
  - **Back Translation** :  $P^{(k)}_{s \rightarrow t}$  on source corpus gives  $D^{(k)}_t$

**end**



# Outline

Machine Translation (MT) Background

Principles of Unsupervised MT

**Unsupervised NMT and PBSMT**

Experiments

Results

Conclusion







# Outline

Machine Translation (MT) Background

Principles of Unsupervised MT

Unsupervised Phrase-Based Statistical MT

## Experiments

Results

Conclusion





# Experiments : Datasets



## Experiments : Datasets

- 5 language pairs : English-French, English-German, English-Romanian, English-Russian, and English-Urdu
- WMT monolingual News Crawl datasets from 2007-2017 for training
- *newstest 2014* for *en-fr*, *newstest 2016* for *en-de*, *en-ro* and *en-ru* for evaluation
- For Urdu, LDC2010T21 and LDC2010T23 corpora with 1800 sentences for validation and test, respectively.



# Experiments : Initialization

## Experiments : Initialization

- For NMT, the two monolingual corpora were concatenated and fastText (Bojanowski et al., 2017) was used to generate a cross-lingual BPE embedding with embedding dimension of 512.

## Experiments : Initialization

- For NMT, the two monolingual corpora were concatenated and fastText (Bojanowski et al., 2017) was used to generate a cross-lingual BPE embedding with embedding dimension of 512.
- For PBSMT, n-gram embeddings are created for the source and target corpora independently, then aligned using the MUSE library.

## Experiments : Initialization

- For NMT, the two monolingual corpora were concatenated and fastText (Bojanowski et al., 2017) was used to generate a cross-lingual BPE embedding with embedding dimension of 512.
- For PBSMT, n-gram embeddings are created for the source and target corpora independently, then aligned using the MUSE library.
  - Only the 300k most frequent phrases are considered and aligned to their 200 nearest neighbors in the target space.
  - This creates 60 million phrase pairs which are scored using

$$p(t_j|s_i) = \frac{e^{\frac{1}{T} \cos(e(t_j), We(s_i))}}{\sum_k e^{\frac{1}{T} \cos(e(t_k), We(s_i))}}$$



# Experiments : Training

For NMT





# Experiments : Training

For NMT

- Dimensionality of hidden layers and embeddings is set to 512

# Experiments : Training

For NMT

- Dimensionality of hidden layers and embeddings is set to 512
- The adam optimizer is used with learning rate  $10^{-4}$ .

# Experiments : Training

For NMT

- Dimensionality of hidden layers and embeddings is set to 512
- The adam optimizer is used with learning rate  $10^{-4}$ .
- Batch\_size = 32



# Experiments : Training

For PBSMT



# Experiments : Training

For PBSMT

- Translate 5 million randomly sampled sentences per iteration



# Outline

Machine Translation (MT) Background

Principles of Unsupervised MT

Unsupervised Phrase-Based Statistical MT

## Experiments

Results

Conclusion





# Outline

Machine Translation (MT) Background

Principles of Unsupervised MT

Unsupervised Phrase-Based Statistical MT

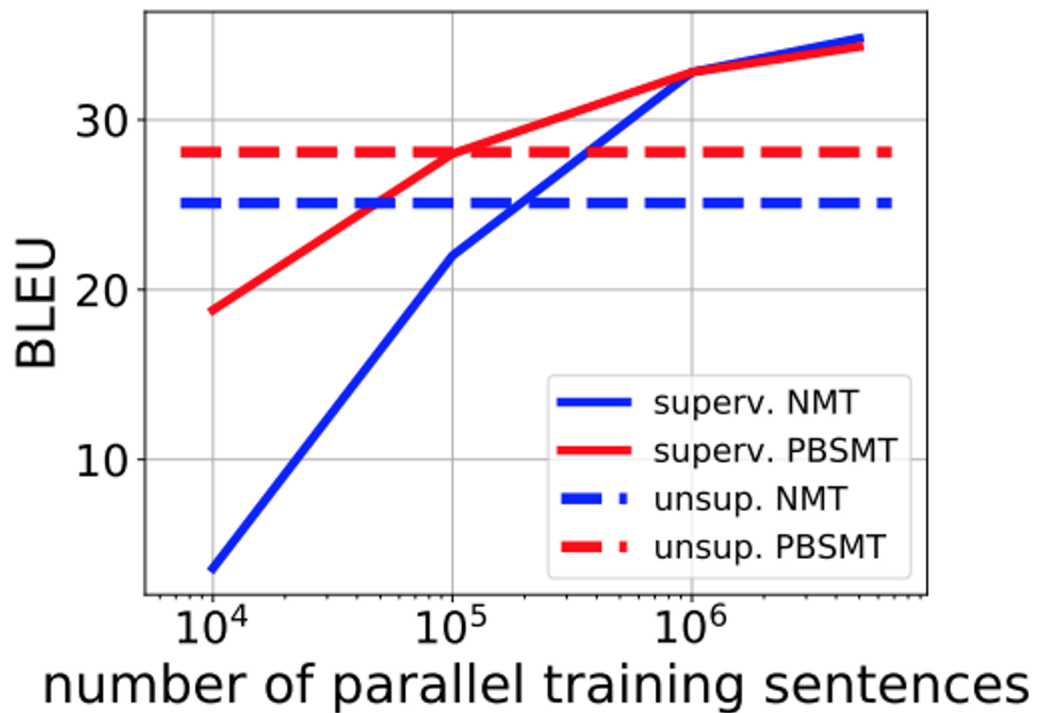
Experiments

**Results**

Conclusion



## Results : NMT





## Results : NMT

Model	en-fr	fr-en	en-de	de-en
(Artetxe et al., 2018)	15.1	15.6	-	-
(Lample et al., 2018)	15.0	14.3	9.6	13.3
(Yang et al., 2018)	17.0	15.6	10.9	14.6
NMT (LSTM)	24.5	23.7	14.7	19.6
NMT (Transformer)	25.1	24.2	17.2	21.0
PBSMT (Iter. 0)	16.2	17.5	11.0	15.6
PBSMT (Iter. n)	<b>28.1</b>	27.2	17.9	22.9
NMT + PBSMT	27.1	26.3	17.5	22.1
PBSMT + NMT	27.6	<b>27.7</b>	<b>20.2</b>	<b>25.2</b>

# Results

	en → fr	fr → en	en → de	de → en	en → ro	ro → en	en → ru	ru → en
<i>Unsupervised PBSMT</i>								
Unsupervised phrase table	-	17.50	-	15.63	-	14.10	-	8.08
Back-translation - Iter. 1	24.79	26.16	15.92	22.43	18.21	21.49	11.04	15.16
Back-translation - Iter. 2	27.32	26.80	17.65	22.85	20.61	22.52	12.87	16.42
Back-translation - Iter. 3	27.77	26.93	17.94	22.87	21.18	22.99	13.13	16.52
Back-translation - Iter. 4	27.84	27.20	17.77	22.68	21.33	23.01	13.37	16.62
Back-translation - Iter. 5	28.11	27.16	-	-	-	-	-	-
<i>Unsupervised NMT</i>								
LSTM	24.48	23.74	14.71	19.60	-	-	-	-
Transformer	25.14	24.18	17.16	21.00	21.18	19.44	7.98	9.09
<i>Phrase-based + Neural network</i>								
NMT + PBSMT	27.12	26.29	17.52	22.06	21.95	23.73	10.14	12.62
PBSMT + NMT	27.60	27.68	20.23	25.19	25.13	23.90	13.76	16.62



# Outline

Machine Translation (MT) Background

Principles of Unsupervised MT

Unsupervised Phrase-Based Statistical MT

Experiments

**Results**

Conclusion





# Outline

Machine Translation (MT) Background

Principles of Unsupervised MT

Unsupervised Phrase-Based Statistical MT

Experiments

Results

**Conclusion**





## **Conclusion : Summary**



## Conclusion : Summary

- Unsupervised machine translation performed with back-translation of large monolingual corpora can perform as well as supervised MT which has parallel data requirements.

## Conclusion : Summary

- Unsupervised machine translation performed with back-translation of large monolingual corpora can perform as well as supervised MT which has parallel data requirements.
- Tuning the NMT model with the data generated from PBSMT performs at the current state of the art for unsupervised machine translation methods

# Synchronous Bidirectional Neural Machine Translation

Long Zhou, Jiajun Zhang, and Chengqing Zong. TACL, vol 7, 2019.

Presented by Yang Yu





# Unidirectional encoder-decoder model

- Generates target translation in one direction (left to right)
- Suffers from unbalanced outputs
- Decoding relies on history information but pays no attention to future information

Model	The first 4 tokens	The last 4 tokens
L2R	<b>40.21%</b>	35.10%
R2L	35.67%	<b>39.47%</b>

Table 1: Translation accuracy of the first 4 tokens and last 4 tokens in NIST Chinese-English translation tasks. L2R denotes left-to-right decoding and R2L means right-to-left decoding for conventional NMT.

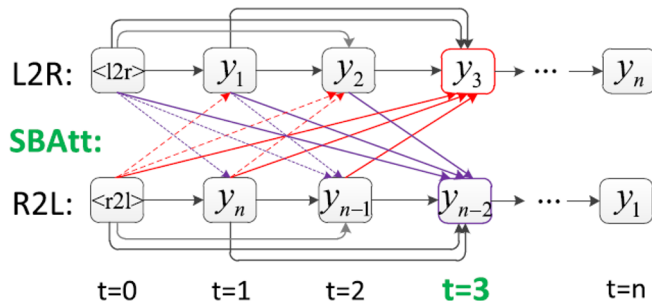


# Attempts to solve this problem

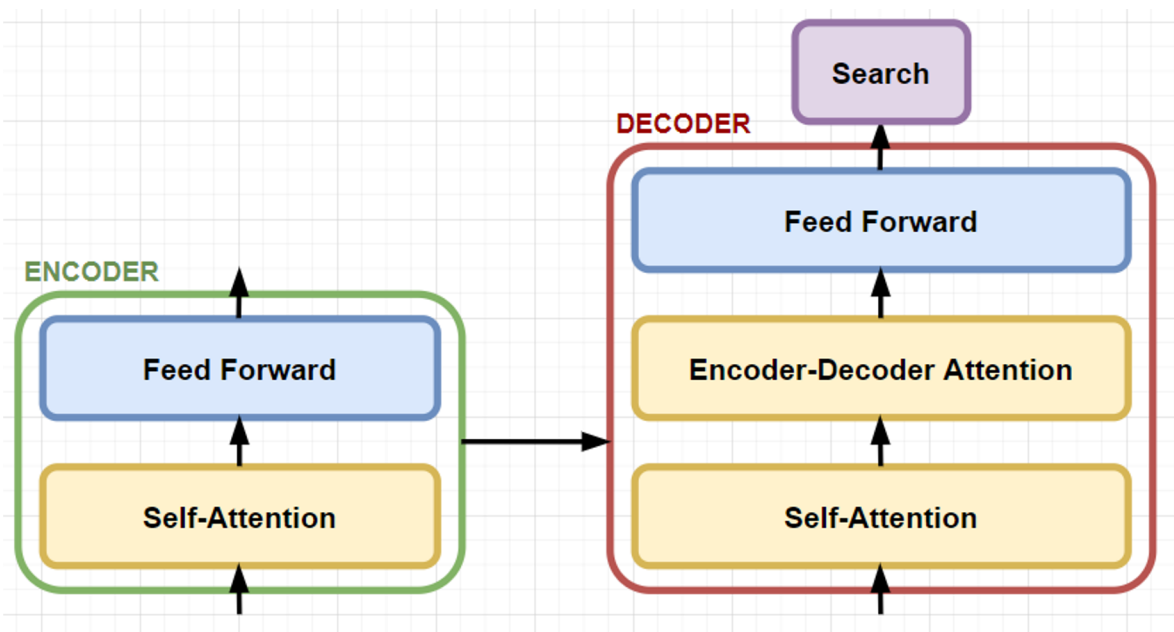
- Independent bidirectional decoder
  - Train two NMT models, one L2R and one R2L
  - Evaluate the translation candidates together
- Asynchronous bidirectional decoding
  - Adding a backward decoder
  - Only the forward decoder can use information from the backward decoder

# Synchronous Bidirectional NMT (SB-NMT) Model

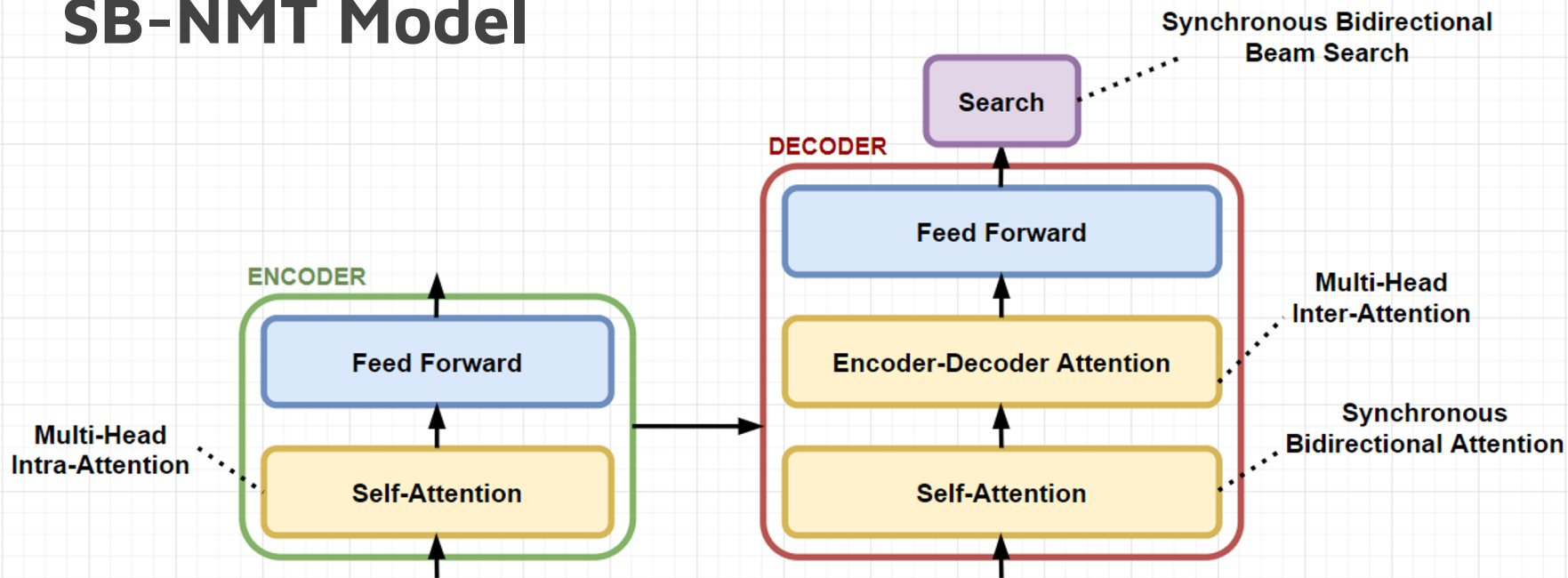
- Single decoder to bidirectionally generate target sentences
- Capable of optimizing bidirectional decoding simultaneously
- Uses a beam search algorithm, the single decoder model is faster and more compact



# SB-NMT Model

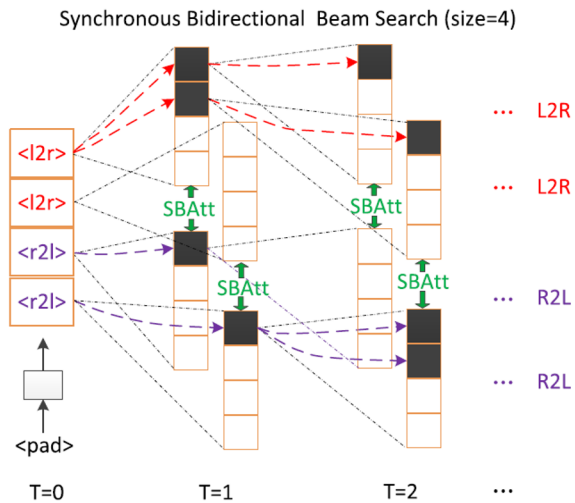


# SB-NMT Model



# Synchronous Bidirectional Beam Search

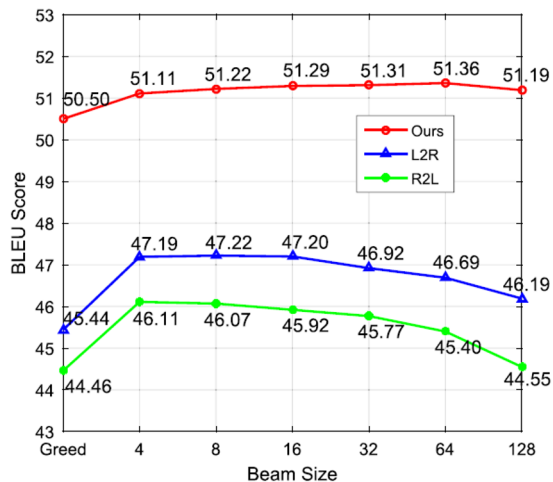
1. For each time step, choose half of the beam for L2R, half for R2L
2. After the final time step, translation result with highest probability will be the final result.





# Synchronous Bidirectional Beam Search

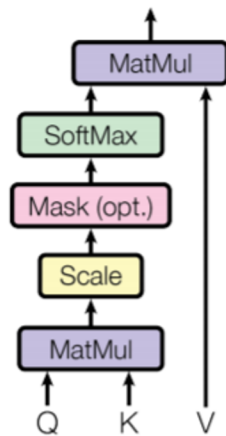
- Effect of different beam sizes was investigated



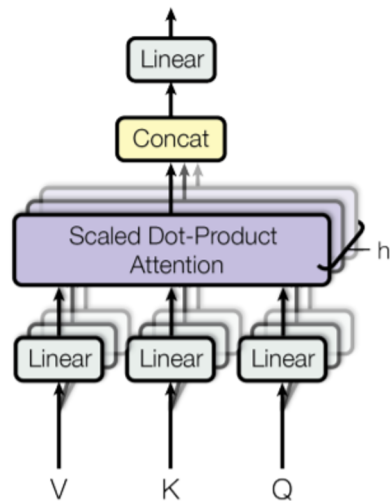
# Synchronous Bidirectional Attention

- Based on the Transformer model with Scaled Dot-Product Attention and Multi-Head Attention proposed by Vaswani et. al. (NIPS 2017)

Scaled Dot-Product Attention



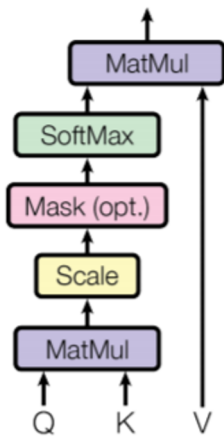
Multi-Head Attention





# Synchronous Bidirectional Attention

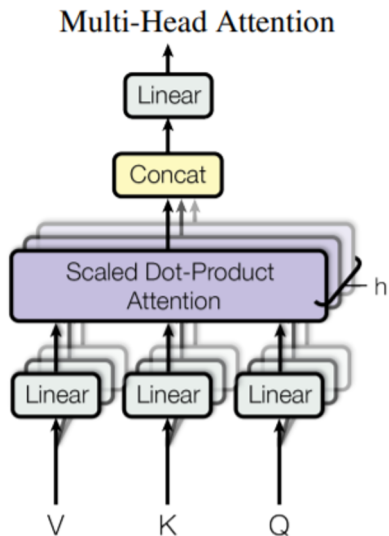
## Scaled Dot-Product Attention



- Similar to a retrieval process: maps a query and a set of key-value pairs to output

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

# Synchronous Bidirectional Attention



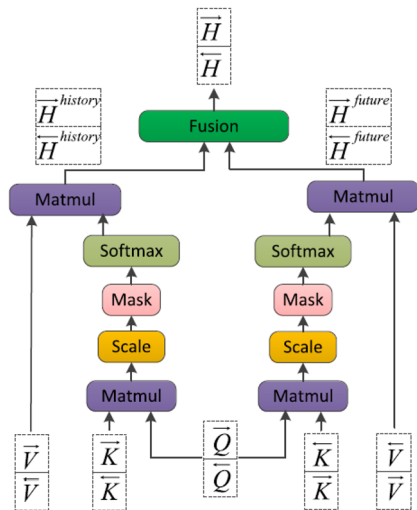
- Allows the model to attend to information from different representation subspaces at different positions

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)W^O$$

where  $\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$

# Synchronous Bidirectional Attention

Synchronous Bidirectional Dot-Product Attention



- Used for decoder self-attention
- Allows future information to combine with history information

$$\vec{H}^{history} = \text{Attention}(\vec{Q}, \vec{K}, \vec{V})$$

$$\vec{H}^{future} = \text{Attention}(\vec{Q}, \vec{K}, \vec{V})$$

$$\vec{H} = \text{Fusion}(\vec{H}^{history}, \vec{H}^{future})$$



# Choices for Fusion Function

- Linear Interpolation

$$\vec{H} = \vec{H}^{history} + \lambda * \vec{H}^{future}$$

- Nonlinear Interpolation
  - $\tanh$  or  $\text{relu}$  as activation function

$$\vec{H} = \vec{H}^{history} + \lambda * AF(\vec{H}^{future})$$

- Gated Mechanism

$$r_t, z_t = \sigma(W^g[\vec{H}^{history}, \vec{H}^{future}])$$

$$\vec{H} = r_t \odot \vec{H}^{history} + z_t \odot \vec{H}^{future}$$



## Choices for Fusion Function

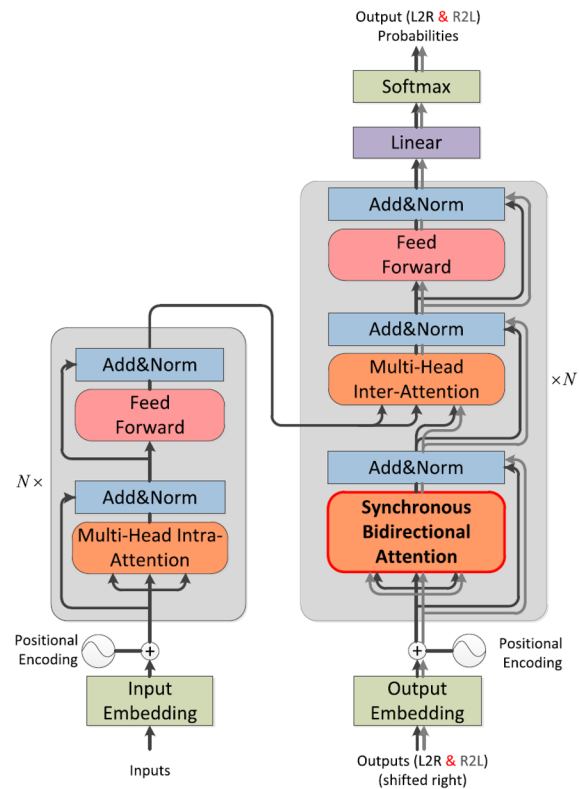
Fusion		$\lambda = 0.1$	$\lambda = 0.5$	$\lambda = 1.0$
Linear		51.05	50.71	46.98
Nonlinear	<i>tanh</i>	50.99	50.72	50.96
	<i>relu</i>	50.79	50.57	50.71
Gate		50.51		

Sensitive to  $\lambda$

Robust

More parameters

# SB-NMT Model





## Experiments - translation quality

Model	TEST
GNMT $\ddagger$ (Wu et al., 2016)	24.61
Conv $\ddagger$ (Gehring et al., 2017)	25.16
AttIsAll $\ddagger$ (Vaswani et al., 2017)	28.40
Transformer <sup>11</sup>	27.72
Transformer (R2L)	27.13
Rerank-NMT	27.81
ABD-NMT	28.22
Our Model	<b>29.21</b>

Table 4: Results of WMT14 English-German translation using case-sensitive BLEU. Results with  $\ddagger$  mark are taken from the corresponding papers.

Model	DEV	TEST
Transformer	35.28	31.02
Transformer (R2L)	35.22	30.57
Our Model	<b>36.38</b>	<b>32.06</b>

Table 5: Results of WMT18 Russian-English translation using case-insensitive tokenized BLEU.



## Experiments - translation quality

Model	DEV	MT03	MT04	M05	MT06	AVE	$\Delta$
Moses	37.85	37.47	41.20	36.41	36.03	37.78	-9.41
RNMT	42.43	42.43	44.56	41.94	40.95	42.47	-4.72
Transformer	48.12	47.63	48.32	47.51	45.31	47.19	-
Transformer (R2L)	47.81	46.79	47.01	46.50	44.13	46.11	-1.08
Rerank-NMT	49.18	48.23	48.91	48.73	46.51	48.10	+0.91
ABD-NMT	48.28	49.47	48.01	48.19	47.09	48.19	+1.00
Our Model	<b>50.99</b>	<b>51.87</b>	<b>51.50</b>	<b>51.23</b>	<b>49.83</b>	<b>51.11</b>	<b>+3.92</b>

Table 3: Evaluation of translation quality for Chinese-English translation tasks using case-insensitive BLEU scores. All results of our model are significantly better than Transformer and Transformer (R2L) ( $p < 0.01$ ).





## Experiments - translation speed

Model	Param	Speed	
		<i>Train</i>	<i>Test</i>
Transformer	207.8M	2.07	19.97
Transformer (R2L)	207.8M	2.07	19.81
Rerank-NMT	415.6M	1.03	6.51
ABD-NMT	333.8M	1.18	7.20
Our Model	207.8M	1.26	17.87

Table 6: Statistics of parameters, training, and testing speeds. *Train* denotes the number of global training steps processed per second at the same batch-size sentences; *Test* indicates the amount of translated sentences in 1 second.



# Experiments - unbalanced outputs

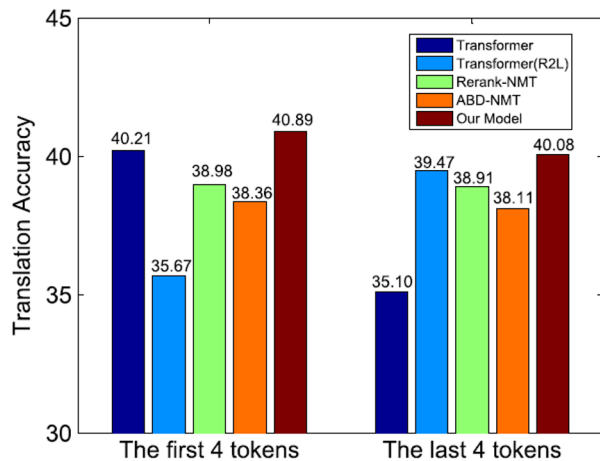


Figure 7: Translation accuracy of the first and last 4 tokens for Transformer, Transformer (R2L), Rerank-NMT, ABD-NMT, and our proposed model.

# Experiments - long sentences

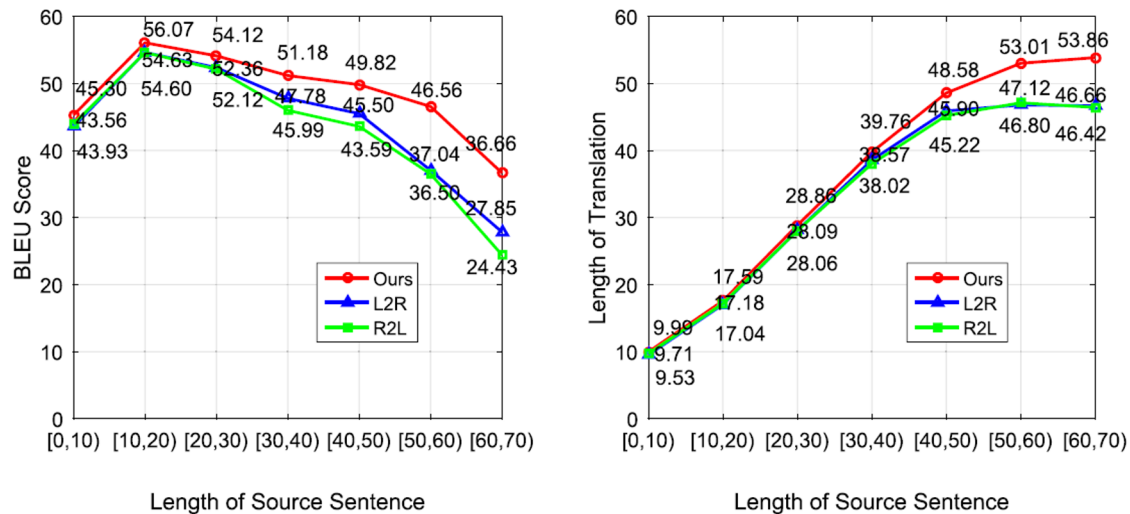


Figure 9: Performance of translations on the test set with respect to the lengths of the source sentences.



## Experiments - subject evaluation

Model	Over-Trans		Under-Trans	
	Ratio	$\Delta$	Ratio	$\Delta$
L2R	0.07%	-	7.85%	-
R2L	0.14%	-	7.81%	-
Ours	0.07%	-0.00%	5.42%	-30.6%

Table 7: Subjective evaluation on over-translation and under-translation for Chinese-English. Ratio denotes the percentage of source words which are over- or under-translated;  $\Delta$  indicates relative improvement.



## Future work

- Fine tuning of parameters, e.g.  $\lambda$ , choice of fusion function
- Application to other tasks, e.g. sequence labeling, abstractive summarization, and image captioning

**Thank you!**

**Questions?**

