Neural Machine Translation Basics II

Rico Sennrich

University of Edinburgh

Modelling Translation

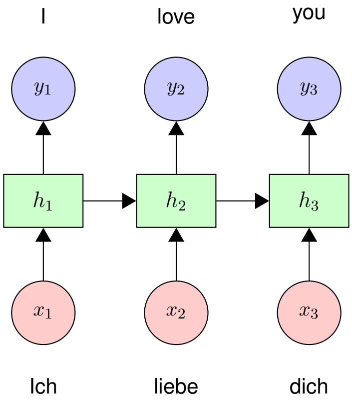
- Suppose that we have:
  - a source sentence S of length m $(x_1, \ldots, x_m)$
  - a target sentence T of length n $(y_1, \ldots, y_n)$
- We can express translation as a probabilistic model

$$T^* = \arg\max_T p(T|S)$$

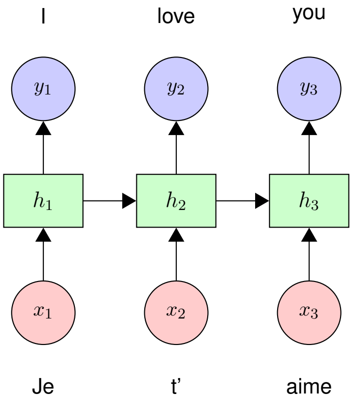
- Expanding using the chain rule gives

$$\begin{aligned} p(T|S) &= p(y_1, \ldots, y_n | x_1, \ldots, x_m) \\ &= \prod_{i=1}^{n} p(y_i | y_1, \ldots, y_{i-1}, x_1, \ldots, x_m) \end{aligned}$$

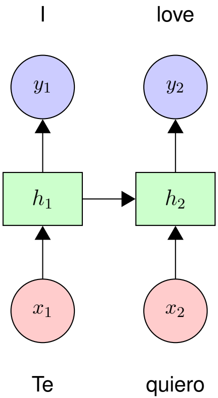
Language Model for Translation?



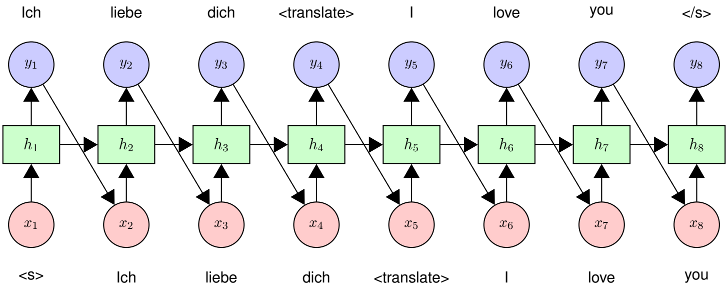
Language Model for Translation?



Language Model for Translation?



Language Model for Translation?



Differences Between Translation and Language Model

- Target-side language model:

$$p(T) = \prod_{i=1}^{n} p(y_i|y_1, \ldots, y_{i-1})$$

- Translation model:

$$p(T|S) = \prod_{i=1}^{n} p(y_i|y_1, \ldots, y_{i-1}, x_1, \ldots, x_m)$$

- We could just treat sentence pair as one long sequence, but:
  - We do not care about $p(S)$
  - We may want different vocabulary, network architecture for source text

Differences Between Translation and Language Model

- Target-side language model:

$$p(T) = \prod_{i=1}^{n} p(y_i|y_1, \dots, y_{i-1})$$

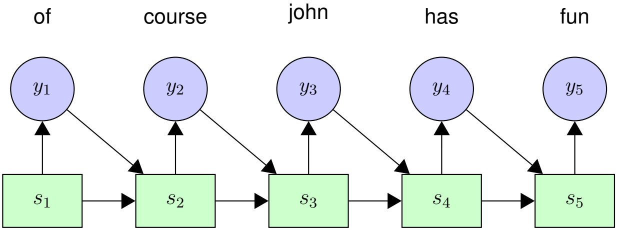
- Translation model:

$$p(T|S) = \prod_{i=1}^{n} p(y_i|y_1, \dots, y_{i-1}, x_1, \dots, x_m)$$

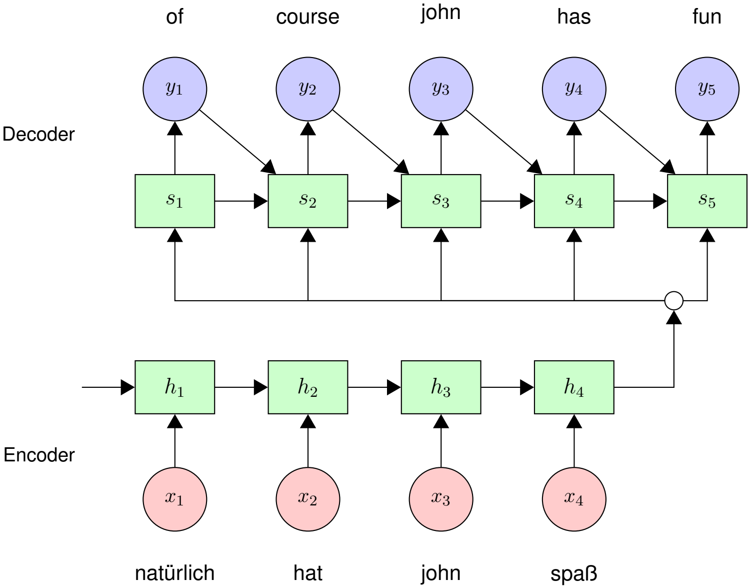
- We could just treat sentence pair as one long sequence, but:
  - We do not care about $p(S)$
  - We may want different vocabulary, network architecture for source text

$\rightarrow$ Use separate RNNs for source and target.

Encoder-Decoder for Translation

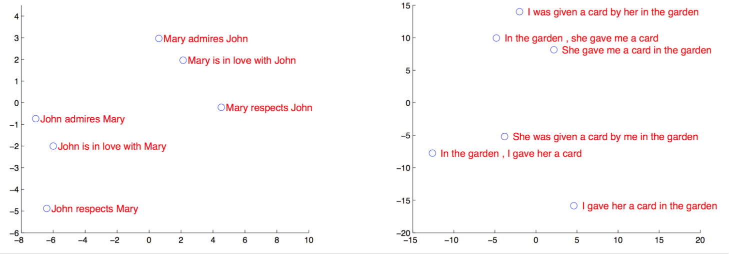


Encoder-Decoder for Translation



Summary vector

- Last encoder hidden-state “summarises” source sentence
- With multilingual training, we can potentially learn language-independent meaning representation

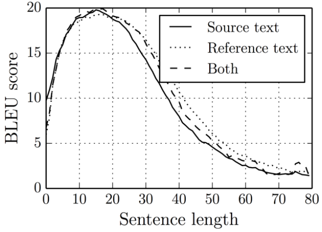


[Sutskever et al., 2014]

Summary vector as information bottleneck

Problem: Sentence Length

- Fixed sized representation degrades as sentence length increases
- Reversing source brings some improvement [Sutskever et al., 2014]

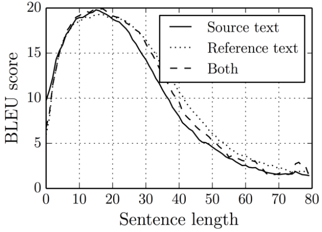


[Cho et al., 2014]

Summary vector as information bottleneck

Problem: Sentence Length

- Fixed sized representation degrades as sentence length increases
- Reversing source brings some improvement [Sutskever et al., 2014]

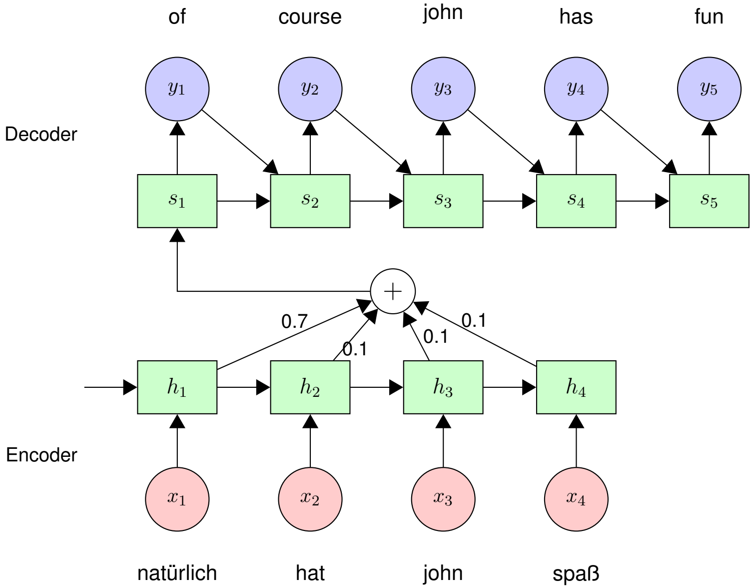


[Cho et al., 2014]

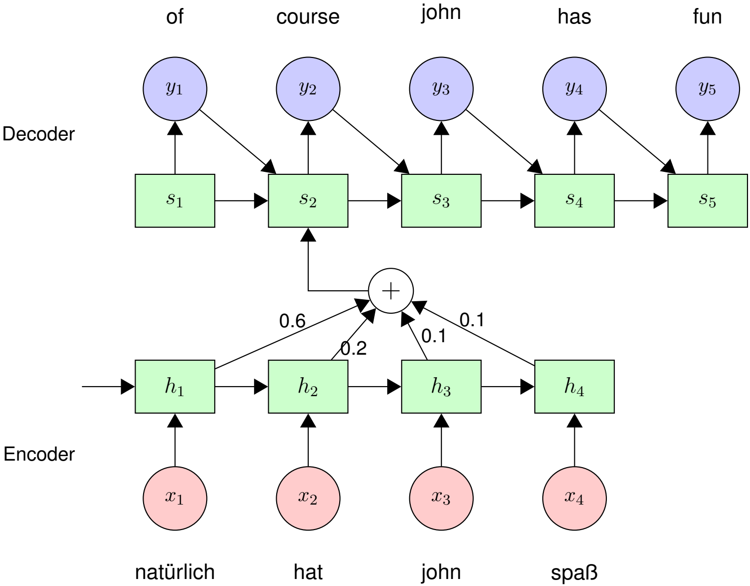
Solution: Attention

- Compute *context vector* as weighted average of source hidden states
- Weights computed by feed-forward network with softmax activation

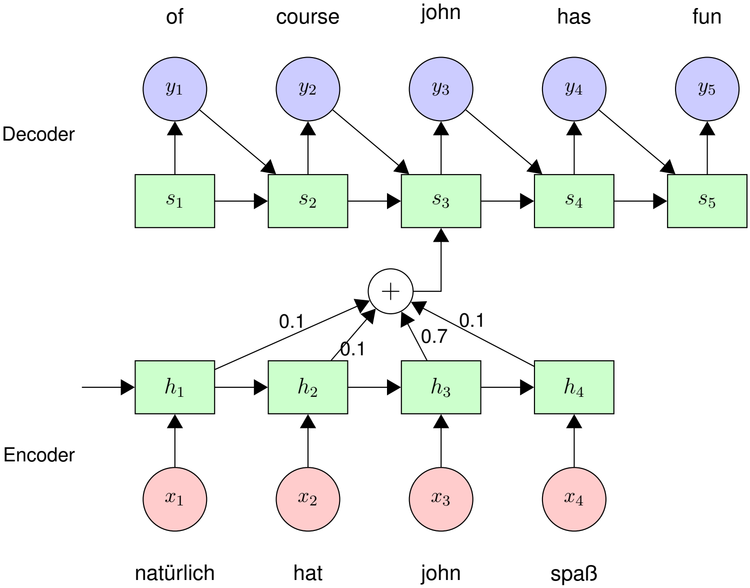
Encoder-Decoder with Attention



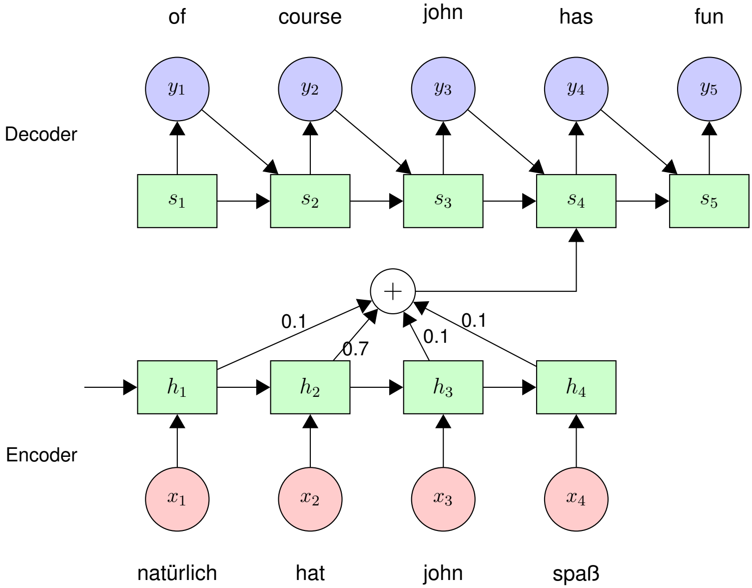
Encoder-Decoder with Attention



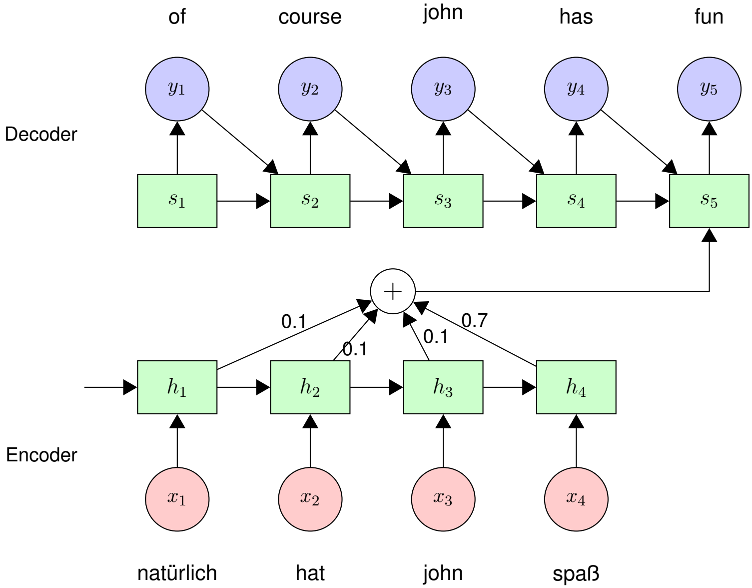
Encoder-Decoder with Attention



Encoder-Decoder with Attention



Encoder-Decoder with Attention



Attentional encoder-decoder: Maths

simplifications of model by [Bahdanau et al., 2015] (for illustration)

- plain RNN instead of GRU
- simpler output layer
- we do not show bias terms
- decoder follows *Look, Update, Generate* strategy [Sennrich et al., 2017]
- Details in `https://github.com/amunmt/amunmt/blob/master/contrib/notebooks/dl4mt.ipynb`

notation

- W, U, E, C, V are weight matrices (of different dimensionality)
  - E one-hot to embedding (e.g. $50000 \cdot 512$)
  - W embedding to hidden (e.g. $512 \cdot 1024$)
  - U hidden to hidden (e.g. $1024 \cdot 1024$)
  - C context (2x hidden) to hidden (e.g. $2048 \cdot 1024$)
  - V_o hidden to one-hot (e.g. $1024 \cdot 50000$)
- separate weight matrices for encoder and decoder (e.g. E_x and E_y)
- input X of length T_x; output Y of length T_y

Attentional encoder-decoder: Maths

encoder

$$\overrightarrow{h}_j = \begin{cases} 0, & \text{, if } j = 0 \\ \tanh(\overrightarrow{W}_x E_x x_j + \overrightarrow{U}_x h_{j-1}) & \text{, if } j > 0 \end{cases}$$

$$\overleftarrow{h}_j = \begin{cases} 0, & \text{, if } j = T_x + 1 \\ \tanh(\overleftarrow{W}_x E_x x_j + \overleftarrow{U}_x h_{j+1}) & \text{, if } j \le T_x \end{cases}$$

$$h_j = (\overrightarrow{h}_j, \overleftarrow{h}_j)$$

Attentional encoder-decoder: Maths

decoder

$$
s_i = \begin{cases} \tanh(W_s \overleftarrow{h}_i), & \text{, if } i = 0 \\ \tanh(W_y E_y y_{i-1} + U_y s_{i-1} + C c_i) & \text{, if } i > 0 \end{cases}
$$

$$
t_i = \tanh(U_o s_i + W_o E_y y_{i-1} + C_o c_i)
$$

$$
y_i = \text{softmax}(V_o t_i)
$$

attention model

$$
e_{ij} = v_a^\top \tanh(W_a s_{i-1} + U_a h_j)
$$

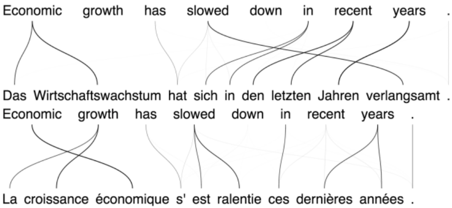
$$
\alpha_{ij} = \text{softmax}(e_{ij})
$$

$$
c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j
$$

Attention model

attention model

- side effect: we obtain “alignment” between source and target sentence
- applications:
  - visualisation
  - replace unknown words with back-off dictionary [Jean et al., 2015]
  - ...



Kyunghyun Cho
http://devblogs.nvidia.com/parallelforall/introduction-neural-machine-translation-gpus-part-3/

Attention is not alignment

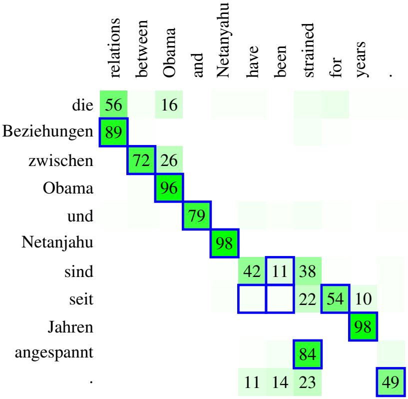


Figure 8: Word alignment for English–German: comparing the attention model states (green boxes with probability in percent if over 10) with alignments obtained from fast-align (blue outlines).

[Koehn and Knowles, 2017]

Attention is not alignment

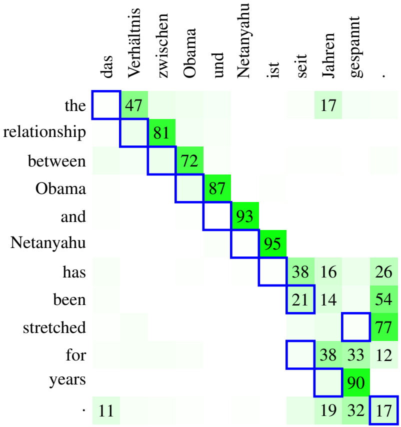


Figure 9: Mismatch between attention states and desired word alignments (German–English).

[Koehn and Knowles, 2017]

Attention model

question: how can model translate correctly, if attention is one-off?

Attention model

question: how can model translate correctly, if attention is one-off?

answer: information can also flow along recurrent connections, so there is no guarantee that attention corresponds to alignment

What If You Want Attention to be Alignment-like?

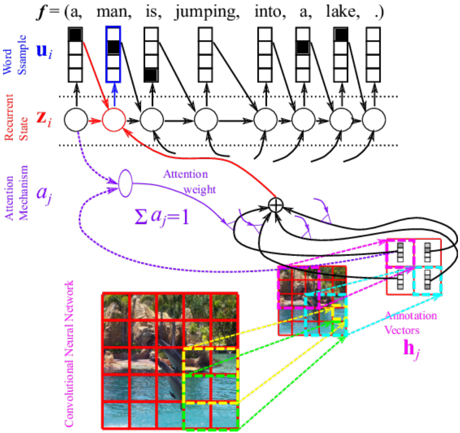
One Solution: Guided Alignment Training [Chen et al., 2016]

1. compute alignment with external tool (IBM models)
2. if multiple source words align to same target words, normalize so that $\sum_j A_{ij} = 1$
3. modify objective function of NMT training:
   - minimize target sentence cross-entropy (as before)
   - minimize divergence between model attention α and external alignment A:

$$H(A, \alpha) = -\frac{1}{T_y} \sum_{i=1}^{T_y} \sum_{j=1}^{T_x} A_{ij} \log \alpha_{ij}$$

Attention model

attention model also works with images:



[Cho et al., 2015]

Attention model

Fig. 5. Examples of the attention-based model attending to the correct object (*white* indicates the attended regions, *underlines* indicated the corresponding word) [22]

[Cho et al., 2015]

Application of Encoder-Decoder Model

Scoring (a translation)

p(La, croissance, économique, s'est, ralentie, ces, dernières, années, . |
Economic, growth, has, slowed, down, in, recent, year, .) = ?

Decoding (a source sentence)

Generate the most probable translation of a source sentence

$$y^* = \mathrm{argmax}_y\, p(y|\text{Economic, growth, has, slowed, down, in, recent, year, .})$$

Decoding

exact search

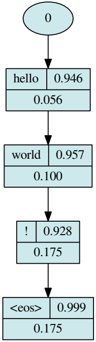
- generate every possible sentence T in target language
- compute score $p(T|S)$ for each
- pick best one

- intractable: $|\text{vocab}|^N$ translations for output length N
  $\rightarrow$ we need approximative search strategy

Decoding

approximative search/1: **greedy search**

- at each time step, compute probability distribution $P(y_i|S, y_{<i})$
- select y_i according to some heuristic:
  - sampling: sample from $P(y_i|S, y_{<i})$
  - greedy search: pick $\text{argmax}_y \, p(y_i|S, y_{<i})$
- continue until we generate *<eos>*



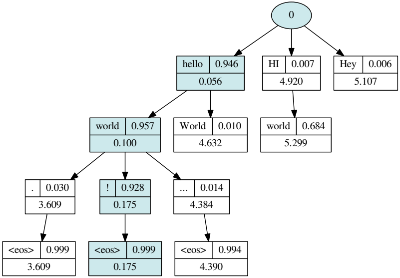
- efficient, but suboptimal

Decoding

approximative search/2: **beam search**

- maintain list of K hypotheses (beam)
- at each time step, expand each hypothesis k: $p(y_i^k|S, y_{<i}^k)$
- select K hypotheses with highest total probability:

$$\prod_i p(y_i^k|S, y_{<i}^k)$$

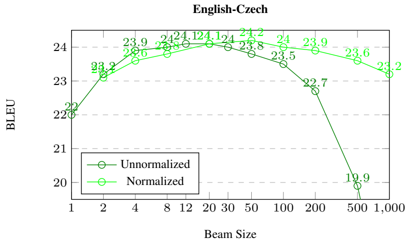


$K = 3$

- relatively efficient . . . beam expansion parallelisable
- currently default search strategy in neural machine translation
- small beam ($K \approx 10$) offers good speed-quality trade-off

Beam Search in Practice

- translation quality goes down with large beam size (!)
- pruning hypotheses to small beam removes states that are improbable, but have low entropy (label bias)
- models have a bias towards short translation
  $\rightarrow$ cost is often normalized by length
- if model starts copying input, future copying has high probability



[Koehn and Knowles, 2017]

Ensembles

- combine decision of multiple classifiers by voting
- ensemble will reduce error if these conditions are met:
  - base classifiers are accurate
  - base classifiers are diverse (make different errors)

Ensembles in NMT

- vote at each time step to explore same search space
  (better than decoding with one, reranking n-best list with others)
- voting mechanism: typically average (log-)probability

$$\log P(y_i|S, y_{<i}) = \frac{\sum_{m=1}^{M} \log P_m(y_i|S, y_{<i})}{M}$$

- requirements for voting at each time step:
  - same output vocabulary
  - same factorization of Y
  - but: internal network architecture may be different
- we still use reranking in some situations
  example: combine left-to-right decoding and right-to-left decoding

Text Representation

how do we represent text in NMT?

- 1-hot encoding
  - lookup of word embedding for input
  - probability distribution over vocabulary for output
- large vocabularies
  - increase network size
  - decrease training and decoding speed
- typical network vocabulary size: 10 000–100 000 symbols

vocabulary		representation of "cat" 1-hot vector	representation of "cat" embedding
0 1 2 . 1024	the cat is . mat	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ . \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0.1 \\ 0.3 \\ 0.7 \\ 0.5 \end{bmatrix}$

Problem

translation is open-vocabulary problem

- many training corpora contain millions of word types
- productive word formation processes (compounding; derivation) allow formation and understanding of unseen words
- names, numbers are morphologically simple, but open word classes

Non-Solution: Ignore Rare Words

- replace out-of-vocabulary words with UNK
- a vocabulary of 50 000 words covers 95% of text

this gets you 95% of the way...
... if you only care about automatic metrics

Non-Solution: Ignore Rare Words

- replace out-of-vocabulary words with UNK
- a vocabulary of 50 000 words covers 95% of text

this gets you 95% of the way...
... if you only care about automatic metrics

why 95% is not enough

rare outcomes have high self-information

source	The **indoor temperature** is very pleasant.	
reference	Das **Raumklima** ist sehr angenehm.	
[Bahdanau et al., 2015]	Die **UNK** ist sehr angenehm.	✗
[Jean et al., 2015]	Die **Innenpool** ist sehr angenehm.	✗
[**Sennrich**, Haddow, Birch, ACL 2016]	Die **Innen+ temperatur** ist sehr angenehm.	✓

Solution 1: Back-off Models

back-off models [Jean et al., 2015, Luong et al., 2015]

- replace rare words with UNK at training time
- when system produces UNK, align UNK to source word, and translate this with back-off method

source	The **indoor temperature** is very pleasant.	
reference	Das **Raumklima** ist sehr angenehm.	
[Bahdanau et al., 2015]	Die **UNK** ist sehr angenehm.	✗
[Jean et al., 2015]	Die **Innenpool** ist sehr angenehm.	✗

limitations

- compounds: hard to model 1-to-many relationships
- morphology: hard to predict inflection with back-off dictionary
- names: if alphabets differ, we need transliteration
- alignment: attention model unreliable

Solution 2: Subword Models

MT is an open-vocabulary problem

- compounding and other productive morphological processes
  - they charge a carry-on bag fee.
  - sie erheben eine Hand|gepäck|gebühr.
- names
  - Obama(English; German)
  - Обама (Russian)
  - オバマ (o-ba-ma) (Japanese)
- technical terms, numbers, etc.

Subword units

segmentation algorithms: wishlist

- **open-vocabulary NMT**: encode *all* words through small vocabulary
- encoding generalizes to unseen words
- small text size
- good translation quality

our experiments [Sennrich et al., 2016]

- after preliminary experiments, we propose:
  - character n-grams (with shortlist of unsegmented words)
  - segmentation via *byte pair encoding* (BPE)

Byte pair encoding for word segmentation

bottom-up character merging

- starting point: character-level representation
  → computationally expensive
- compress representation based on information theory
  → byte pair encoding [Gage, 1994]
- repeatedly replace most frequent symbol pair ('A','B') with 'AB'
- hyperparameter: when to stop
  → controls vocabulary size

word	freq
'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

vocabulary:
l o w</w> w e r</w> n s t</w> i d

Byte pair encoding for word segmentation

bottom-up character merging

- starting point: character-level representation
  $\rightarrow$ computationally expensive
- compress representation based on information theory
  $\rightarrow$ byte pair encoding [Gage, 1994]
- repeatedly replace most frequent symbol pair ('A','B') with 'AB'
- hyperparameter: when to stop
  $\rightarrow$ controls vocabulary size

word	freq
'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

vocabulary:
l o w</w> w e r</w> n s t</w> i d
es

Byte pair encoding for word segmentation

bottom-up character merging

- starting point: character-level representation
  $\rightarrow$ computationally expensive
- compress representation based on information theory
  $\rightarrow$ byte pair encoding [Gage, 1994]
- repeatedly replace most frequent symbol pair ('A','B') with 'AB'
- hyperparameter: when to stop
  $\rightarrow$ controls vocabulary size

word	freq
'l o w</w>'	5
'l o w e r</w>'	2
'n e w **est</w>**'	6
'w i d **est</w>**'	3

vocabulary:
l o w</w> w e r</w> n s t</w> i d
es **est</w>**

Byte pair encoding for word segmentation

bottom-up character merging

- starting point: character-level representation
  $\rightarrow$ computationally expensive
- compress representation based on information theory
  $\rightarrow$ byte pair encoding [Gage, 1994]
- repeatedly replace most frequent symbol pair ('A','B') with 'AB'
- hyperparameter: when to stop
  $\rightarrow$ controls vocabulary size

word	freq
'**lo** w</w>'	5
'**lo** w e r</w>'	2
'n e w **est</w>**'	6
'w i d **est</w>**'	3

vocabulary:
l o w</w> w e r</w> n s t</w> i d
es est</w> **lo**

Byte pair encoding for word segmentation

why BPE?

- open-vocabulary:
  operations learned on training set can be applied to unknown words
- compression of frequent character sequences improves efficiency
  $\rightarrow$ trade-off between text length and vocabulary size

'l o w e s t</w>'

e s	$\rightarrow$	es
es t</w>	$\rightarrow$	est</w>
l o	$\rightarrow$	lo

Byte pair encoding for word segmentation

why BPE?

- open-vocabulary:
  operations learned on training set can be applied to unknown words
- compression of frequent character sequences improves efficiency
  $\rightarrow$ trade-off between text length and vocabulary size

'l o w **es** t</w>'

e s	$\rightarrow$	**es**
es t</w>	$\rightarrow$	est</w>
l o	$\rightarrow$	lo

Byte pair encoding for word segmentation

why BPE?

- open-vocabulary:
  operations learned on training set can be applied to unknown words
- compression of frequent character sequences improves efficiency
  → trade-off between text length and vocabulary size

’l o w **est</w>**’

e s	→	es
es t</w>	→	**est</w>**
l o	→	lo

Byte pair encoding for word segmentation

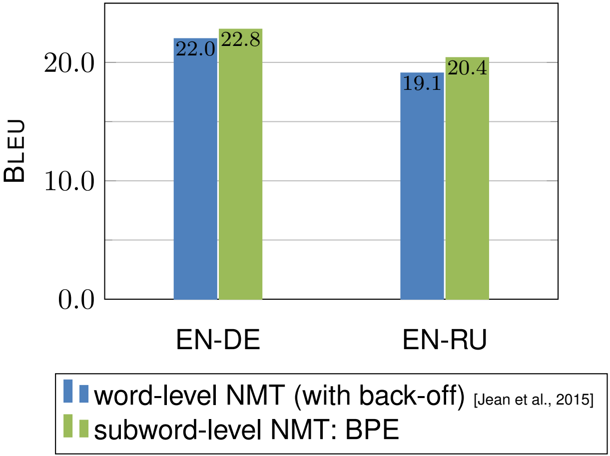
why BPE?

- open-vocabulary:
  operations learned on training set can be applied to unknown words
- compression of frequent character sequences improves efficiency
  → trade-off between text length and vocabulary size

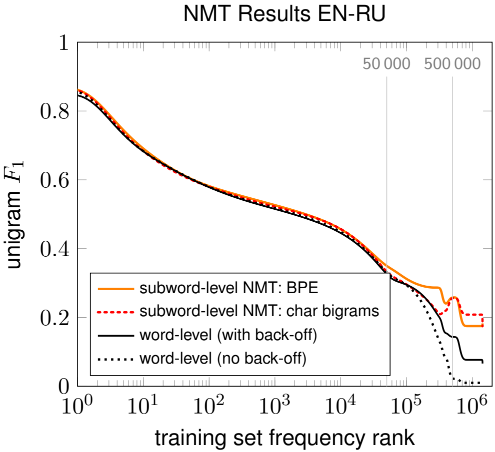
'**lo** w est</w>'

e s	→	es
es t</w>	→	est</w>
l o	→	**lo**

Subword NMT: Translation Quality



Subword NMT: Translation Quality



Examples

system	sentence
source	health research institutes
reference	Gesundheitsforschungsinstitute
word-level (with back-off)	Forschungsinstitute
character bigrams	Fo|rs|ch|un|gs|in|st|it|ut|io|ne|n
BPE	Gesundheits|forsch|ungsin|stitute
source	rakfisk
reference	ракфиска (rakfiska)
word-level (with back-off)	rakfisk → UNK → rakfisk
character bigrams	ra|kf|is|k → ра|кф|ис|к (ra|kf|is|k)
BPE	rak|f|isk → рак|ф|иска (rak|f|iska)

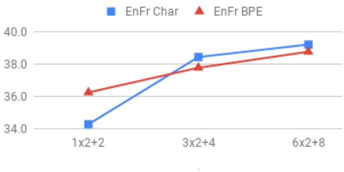
Subword Models: Variants

- morphologically motivated subword units [Sánchez-Cartagena and Toral, 2016, Tamchyna et al., 2017, Huck et al., 2017, Pinnis et al., 2017]
- probabilistic segmentation and sampling [Kudo, 2018]

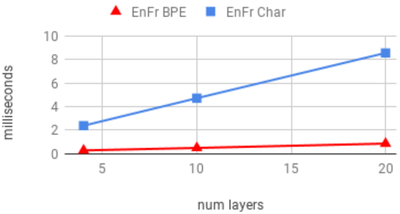
Solution 3: Character-level Models

- advantages:
  - (mostly) open-vocabulary
  - no heuristic or language-specific segmentation
  - neural network can learn from raw character sequences
- drawbacks:
  - increasing sequence length slows training/decoding
    (reported x2–x8 increase in training time)
  - memory requirement: trade-off between embedding matrix size and sequence length
- active research on specialized architectures [Ling et al., 2015, Luong and Manning, 2016, Chung et al., 2016, Lee et al., 2016]

Character-level NMT: Current Results (as of 5 days ago: August 29 2018)



For deep LSTMs, char-level systems can achieve higher BLEU than BPE



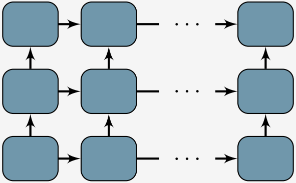
...but training time (per sentence) is 8x higher

[Cherry et al., 2018]

Deep Networks

- increasing model depth often increases model performance
- example: stack RNN:

$$
\begin{aligned}
h_{i,1} &= g(U_1 h_{i-1,1} + W_1 E_x x_i) \\
h_{i,2} &= g(U_2 h_{i-1,2} + W_2 h_{i,1}) \\
h_{i,3} &= g(U_3 h_{i-1,3} + W_3 h_{i,2})
\end{aligned}
$$



Deep Networks

- often necessary to combat vanishing gradient:
  residual connections between layers:

$$
\begin{aligned}
h_{i,1} &= g(U_1 h_{i-1,1} + W_1 E_x x_i) \\
h_{i,2} &= g(U_2 h_{i-1,2} + W_2 h_{i,1}) + \mathbf{h_{i,1}} \\
h_{i,3} &= g(U_3 h_{i-1,3} + W_3 h_{i,2}) + \mathbf{h_{i,2}}
\end{aligned}
$$

Layer Normalization

- if input distribution to NN layer changes, parameters need to adapt to this **covariate shift**
- especially bad: RNN state grows/shrinks as we go through sequence
- normalization of layers reduces shift, and improves training stability
- re-center and re-scale each layer $\mathbf{a}$ (with H units)
- two bias parameters, $\mathbf{g}$ and $\mathbf{b}$, restore original representation power

$$\mu = \frac{1}{H}\sum_{i=1}^{H} a_i$$

$$\sigma = \sqrt{\frac{1}{H}\sum_{i=1}^{H}(a_i - \mu)^2}$$

$$\mathbf{h} = \left[\frac{\mathbf{g}}{\sigma} \odot (\mathbf{a} - \mu) + \mathbf{b}\right]$$

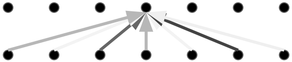
Layer Normalization and Deep Models: Results from UEDIN@WMT17

system	CS→EN 2017	DE→EN 2017	LV→EN 2017	RU→EN 2017	TR→EN 2017	ZH→EN 2017
baseline	27.5	32.0	16.4	31.3	19.7	21.7
+layer normalization	28.2	32.1	17.0	32.3	18.8	22.5
+deep model	28.9	33.5	16.6	32.7	20.6	22.9

- layer normalization and deep models generally improve quality
- layer normalization also speeds up convergence when training (fewer updates needed)

Attention Is All You Need [Vaswani et al., 2017]

- main criticism of recurrent architecture:
  recurrent computations cannot be parallelized
- core idea: instead of recurrence, use attention mechanism to condition hidden states on context
  $\rightarrow$ self-attention
- specifically, attend over previous layer of deep network



https://nlp.stanford.edu/seminar/details/lkaiser.pdf

Attention Is All You Need [Vaswani et al., 2017]

Transformer architecture

- stack of N self-attention layers
- self-attention in decoder is *masked*
- decoder also attends to encoder states
- *Add & Norm*: residual connection and layer normalization
- RNN can learn to count raw text
  Transformer needs *positional encoding*

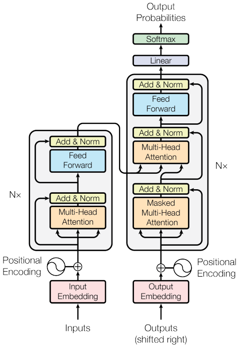


Figure 1: The Transformer - model architecture.

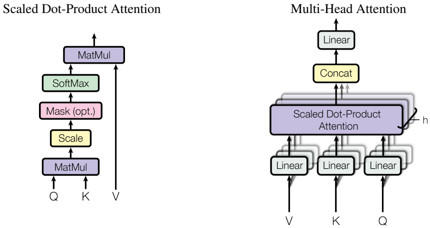
[Vaswani et al., 2017]

Multi-Head Attention

- basic attention mechanism in AIAYN: Scaled Dot-Product Attention

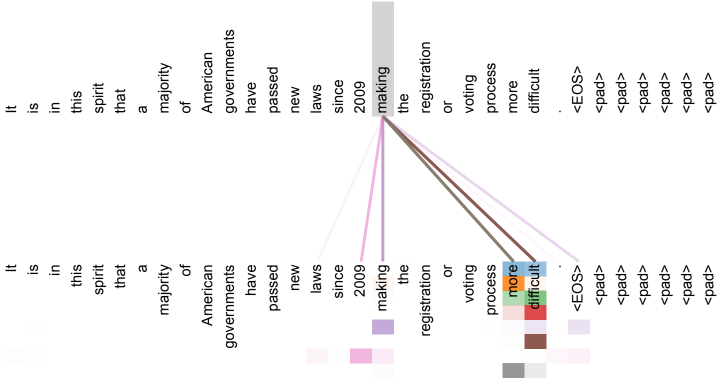
$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

- query Q is decoder/encoder state (for attention/self-attention)
- key K and value V are encoder hidden states
- multi-head attention: use h parallel attention mechanisms with low-dimensional, learned projections of Q, K, and V



Multi-Head Attention

motivation for multi-head attention:
different heads can attend to different states



Comparison

empirical comparison difficult

- some components could be mix-and-matched
  - choice of attention mechanism
  - choice of positional encoding
  - hyperparameters and training tricks
- different test sets and/or evaluation scripts

Comparison

SOCKEYE [Hieber et al., 2017] (EN-DE; newstest2017)

system	BLEU
deep LSTM	25.6
Convolutional	24.6
Transformer	**27.5**

[Chen et al., 2018] (EN-DE; newstest2014)

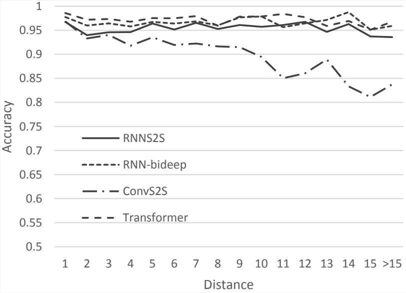
system	BLEU
GNMT (RNN)	24.7
Transformer	27.9
RNMT+ (RNN)	**28.5**

Why Self-Attention?

- our understanding of neural networks lags behind empirical progress
- there are some theoretical arguments why architectures work well... (e.g. self-attention reduces distance in network between words)
- ...but these are very speculative

Targeted Evaluation [Tang et al., 2018]

- quality of long-distance agreement is similar between GRU and Transformer
- strength of Transformer: better word sense disambiguation



Further Reading

- primary literature:
  - Sutskever, Vinyals, Le (2014): Sequence to Sequence Learning with Neural Networks.
  - Bahdanau, Cho, Bengio (2014): Neural Machine Translation by Jointly Learning to Align and Translate.
  - Vaswani et al. (2017): Attention Is All You Need
- secondary literature:
  - Koehn (2017): Neural Machine Translation.
  - Neubig (2017): Neural Machine Translation and Sequence-to-sequence Models: A Tutorial
  - Cho (2015): Natural Language Understanding with Distributed Representation

Bibliography I

Bahdanau, D., Cho, K., and Bengio, Y. (2015).
Neural Machine Translation by Jointly Learning to Align and Translate.
In Proceedings of the International Conference on Learning Representations (ICLR).

Chen, M. X., Firat, O., Bapna, A., Johnson, M., Macherey, W., Foster, G., Jones, L., Schuster, M., Shazeer, N., Parmar, N., Vaswani, A., Uszkoreit, J., Kaiser, L., Chen, Z., Wu, Y., and Hughes, M. (2018).
The Best of Both Worlds: Combining Recent Advances in Neural Machine Translation.
In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 76–86, Melbourne, Australia. Association for Computational Linguistics.

Chen, W., Matusov, E., Khadivi, S., and Peter, J. (2016).
Guided Alignment Training for Topic-Aware Neural Machine Translation.
CoRR, abs/1607.01628.

Cherry, C., Foster, G., Bapna, A., Firat, O., and Macherey, W. (2018).
Revisiting Character-Based Neural Machine Translation with Capacity and Compression.
ArXiv e-prints.

Cho, K., Courville, A., and Bengio, Y. (2015).
Describing Multimedia Content using Attention-based Encoder-Decoder Networks.

Cho, K., van Merrienboer, B., Bahdanau, D., and Bengio, Y. (2014).
On the Properties of Neural Machine Translation: Encoder–Decoder Approaches.
In Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

Chung, J., Cho, K., and Bengio, Y. (2016).
A Character-level Decoder without Explicit Segmentation for Neural Machine Translation.
CoRR, abs/1603.06147.

Bibliography II

Gage, P. (1994).
A New Algorithm for Data Compression.
C Users J., 12(2):23–38.

Hieber, F., Domhan, T., Denkowski, M., Vilar, D., Sokolov, A., Clifton, A., and Post, M. (2017).
Sockeye: A Toolkit for Neural Machine Translation.
CoRR.

Huck, M., Riess, S., and Fraser, A. (2017).
Target-side Word Segmentation Strategies for Neural Machine Translation.
In Proceedings of the Second Conference on Machine Translation, Volume 1: Research Papers, Copenhagen, Denmark. Association for Computational Linguistics.

Jean, S., Cho, K., Memisevic, R., and Bengio, Y. (2015).
On Using Very Large Target Vocabulary for Neural Machine Translation.
In
Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference o
pages 1–10, Beijing, China. Association for Computational Linguistics.

Koehn, P. and Knowles, R. (2017).
Six Challenges for Neural Machine Translation.
In Proceedings of the First Workshop on Neural Machine Translation, pages 28–39, Vancouver. Association for Computational Linguistics.

Kudo, T. (2018).
Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates.
In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

Bibliography III

Lee, J., Cho, K., and Hofmann, T. (2016).
Fully Character-Level Neural Machine Translation without Explicit Segmentation.
ArXiv e-prints.

Ling, W., Trancoso, I., Dyer, C., and Black, A. W. (2015).
Character-based Neural Machine Translation.
ArXiv e-prints.

Luong, M.-T. and Manning, D. C. (2016).
Achieving Open Vocabulary Neural Machine Translation with Hybrid Word-Character Models.
In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1054–1063. Association for Computational Linguistics.

Luong, T., Sutskever, I., Le, Q., Vinyals, O., and Zaremba, W. (2015).
Addressing the Rare Word Problem in Neural Machine Translation.
In
Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference c
pages 11–19, Beijing, China. Association for Computational Linguistics.

Pinnis, M., Krislauks, R., Deksne, D., and Miks, T. (2017).
Neural Machine Translation for Morphologically Rich Languages with Improved Sub-word Units and Synthetic Data.
In Text, Speech, and Dialogue - 20th International Conference, TSD 2017, pages 237–245, Prague, Czech Republic.

Sánchez-Cartagena, V. M. and Toral, A. (2016).
Abu-MaTran at WMT 2016 Translation Task: Deep Learning, Morphological Segmentation and Tuning on Character Sequences.
In Proceedings of the First Conference on Machine Translation, pages 362–370, Berlin, Germany. Association for Computational Linguistics.

Bibliography IV

Sennrich, R., Firat, O., Cho, K., Birch, A., Haddow, B., Hitschler, J., Junczys-Dowmunt, M., Läubli, S., Miceli Barone, A. V., Mokry, J., and Nadejde, M. (2017).
Nematus: a Toolkit for Neural Machine Translation.
In
Proceedings of the Software Demonstrations of the 15th Conference of the European Chapter of the Association for Computational L
pages 65–68, Valencia, Spain.

Sennrich, R., Haddow, B., and Birch, A. (2016).
Neural Machine Translation of Rare Words with Subword Units.
In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1715–1725, Berlin, Germany.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014).
Sequence to Sequence Learning with Neural Networks.
In
Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, pages 3104–3112, Montreal, Quebec, Canada.

Tamchyna, A., Weller-Di Marco, M., and Fraser, A. (2017).
Modeling Target-Side Inflection in Neural Machine Translation.
In Second Conference on Machine Translation (WMT17).

Tang, G., Müller, M., Rios, A., and Sennrich, R. (2018).
Why Self-Attention? A Targeted Evaluation of Neural Machine Translation Architectures.
In EMNLP 2018, Brussels, Belgium. Association for Computational Linguistics.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017).
Attention Is All You Need.
CoRR, abs/1706.03762.