Machine Translation

Kevin Duh

Johns Hopkins University

Number of Languages in the World

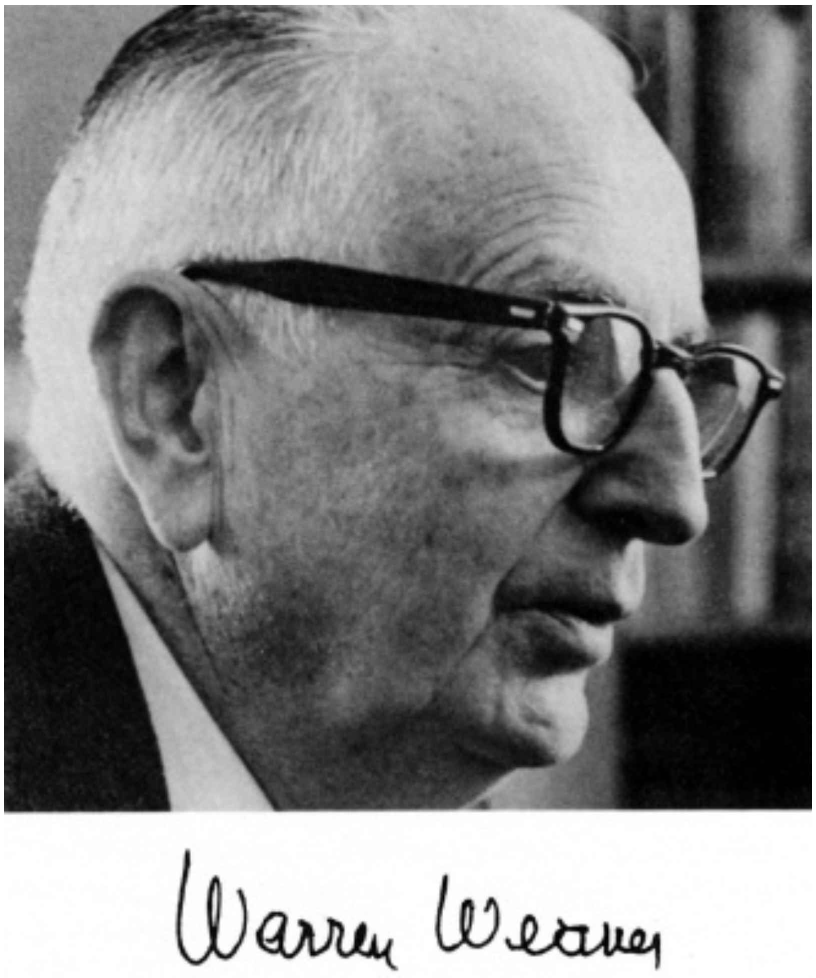
6000

Image courtesy of nasa.gov

MT Applications

- Dissemination:
  - Translate out to many languages, e.g. localization
- Assimilation:
  - Translate into your own language, e.g. cross-lingual search
- Communication
  - Real-time two-way conversation, e.g. the Babelfish!

When I look at an article in Russian, I say:
”This is really written in English, but it has been coded in some strange symbols.
I will now proceed to decode”.



Warren Weaver,
American scientist (1894-1978)

Image courtesy of: Biographical Memoirs of the National Academy of Science, Vol. 57

Progress in MT

Warren Weaver's memo

1947

Founding of SYSTRAN. Development of Rule-based MT (RBMT)

1968

Seminal SMT paper from IBM

1993

DARPA TIDES, GALE, BOLT programs
Open-source of Moses toolkit
Development of Statistical MT (SMT)

Early 2000s

2011-2012: Early deep learning success in speech/vision
2015: Seminal NMT paper (RNN+attention)
2016: Google announces NMT in production
2017: New NMT architecture: Transformer

2010s-Present

Outline

1. Background: Intuitions, SMT
2. NMT: Recurrent Model with Attention
3. NMT: Transformer Model
4. NMT with Large Language Models

Rule-Based Machine Translation (RBMT)

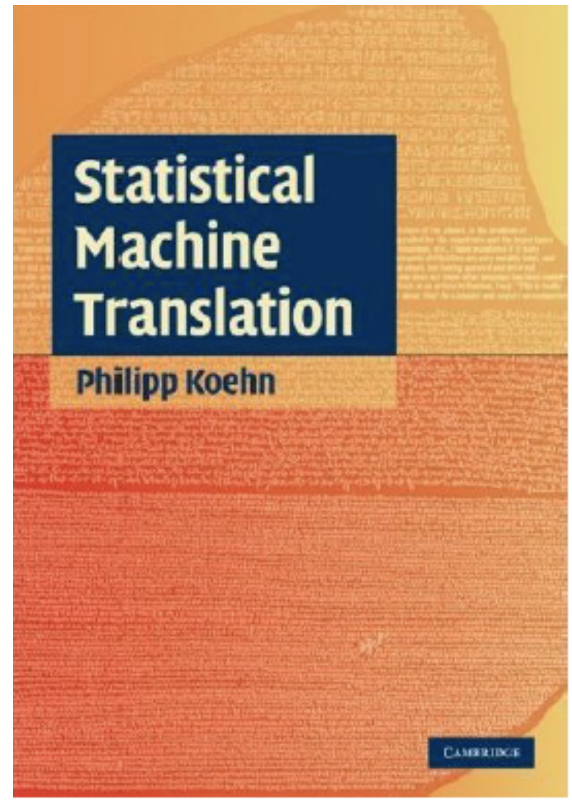
- Rule-based systems:
    - build dictionaries
    - write transformation rules

```
"have" :=

if
   subject(animate)
   and object(owned-by-subject)
then
   translate to "kade...  aahe"
if
   subject(animate)
   and object(kinship-with-subject)
then
   translate to "laa...  aahe"
if
   subject(inanimate)
then
   translate to "madhye...  aahe"
```

Statistical Machine Translation (SMT)

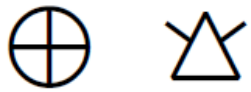
- Data-driven:
  - Learn dictionaries from data
  - Learn transformation "rules" from data
- SMT usually refers to a set of data-driven techniques around 1980-2015. It's often distinguished from neural network models (NMT), but note that NMT also uses statistics!



How to learn from data?

- Assume bilingual text (bitext), a.k.a. parallel text
    - Each sentence in Language A is aligned to its translation in Language B
- Assume we have lots of this. Now, we can proceed to “decode”

1a) evas dlrow-eht

1b) 

2a) dlrow-eht si detcennoc

2b) 

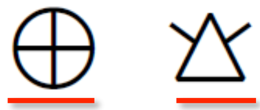
3a) hcraeser si tnatropmi

3b) 

4a) ew eb-ot-mia tseb ni dlrow-eht

4b) 

1a) evas dlrow-eht

1b) 

2a) dlrow-eht si detcennoc

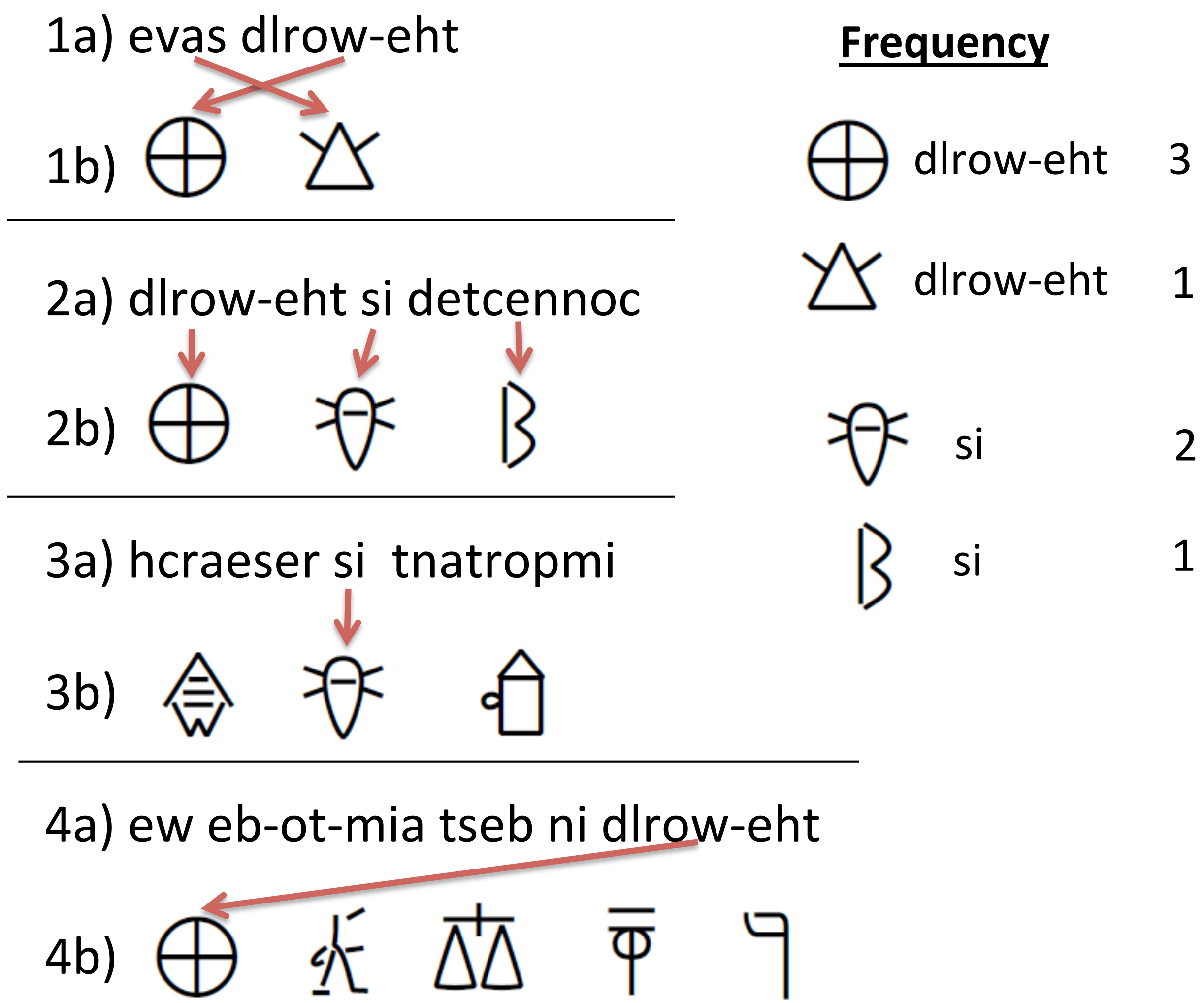
2b) 

3a) hcraeser si tnatropmi

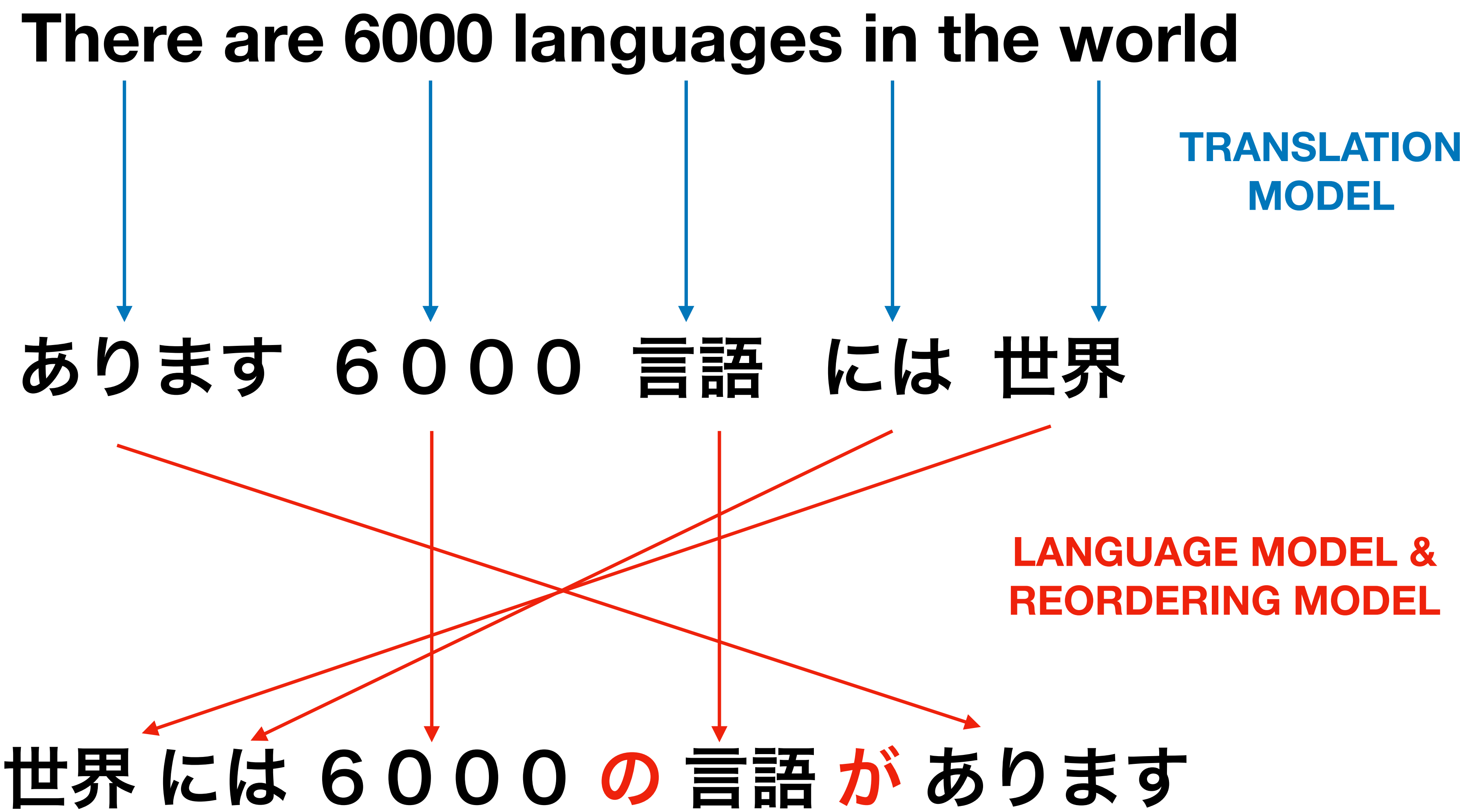
3b) 

4a) ew eb-ot-mia tseb ni dlrow-eht

4b) 


1a) evas dlrow-eht
1b)
2a) dlrow-eht si detcennoc
2b)
3a) hcraeser si tnatropmi
3b)
4a) ew eb-ot-mia tseb ni dlrow-eht
4b)
Frequency
dlrow-eht 3
dlrow-eht 1
si 2
si 1

Inside a SMT system (simplified view)



SMT vs NMT

- Problem Setup:
  - Input: source sentence
  - Output: target sentence
  - Given bitext, learn a model that maps source to target
- SMT models the mapping with several probabilistic models (e.g. translation model, language model)
- NMT models the mapping with a single neural network

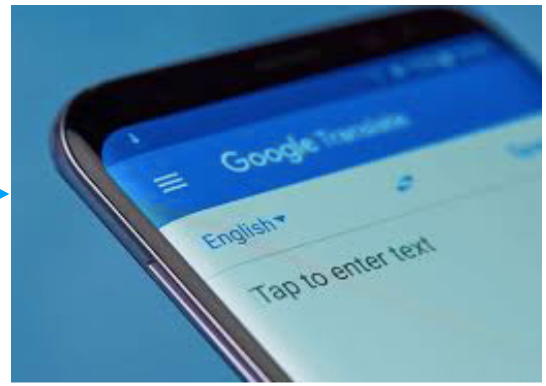
Outline

1. Background: Intuitions, SMT
2. NMT: Recurrent Model with Attention
3. NMT: Transformer Model
4. NMT with Large Language Models

Neural sequence-to-sequence models

- For sequence input:
  - We need an “encoder” to convert arbitrary length input to some fixed-length hidden representation
  - Without this, may be hard to apply matrix operations
- For sequence output:
  - We need a “decoder” to generate arbitrary length output
  - One method: generate one word at a time, until special <stop> token

das Haus ist gross



the house is big

Encoder

"Sentence Vector"

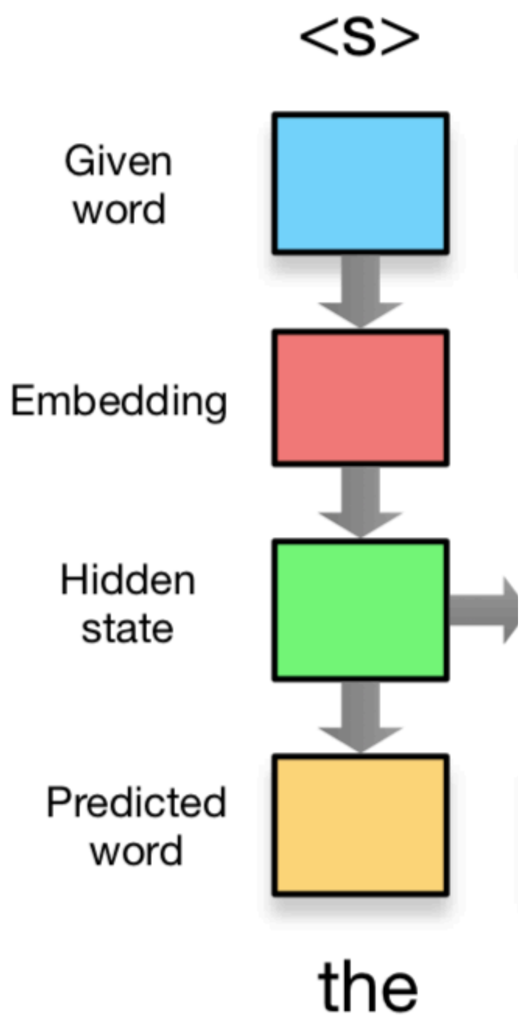
Decoder

das Haus ist gross

step 1: the
step 2: house
step 3: is
step 4: big
step 5: <stop>

Each step applies a softmax over all vocab

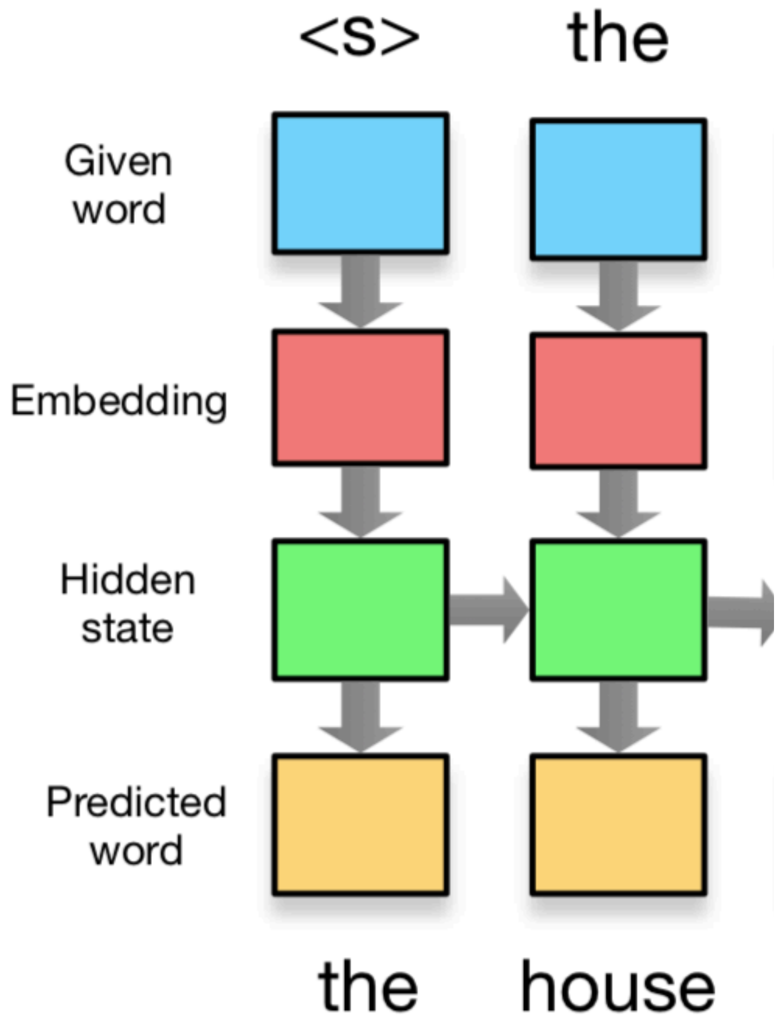
Sequence modeling with a recurrent network



the house is big .

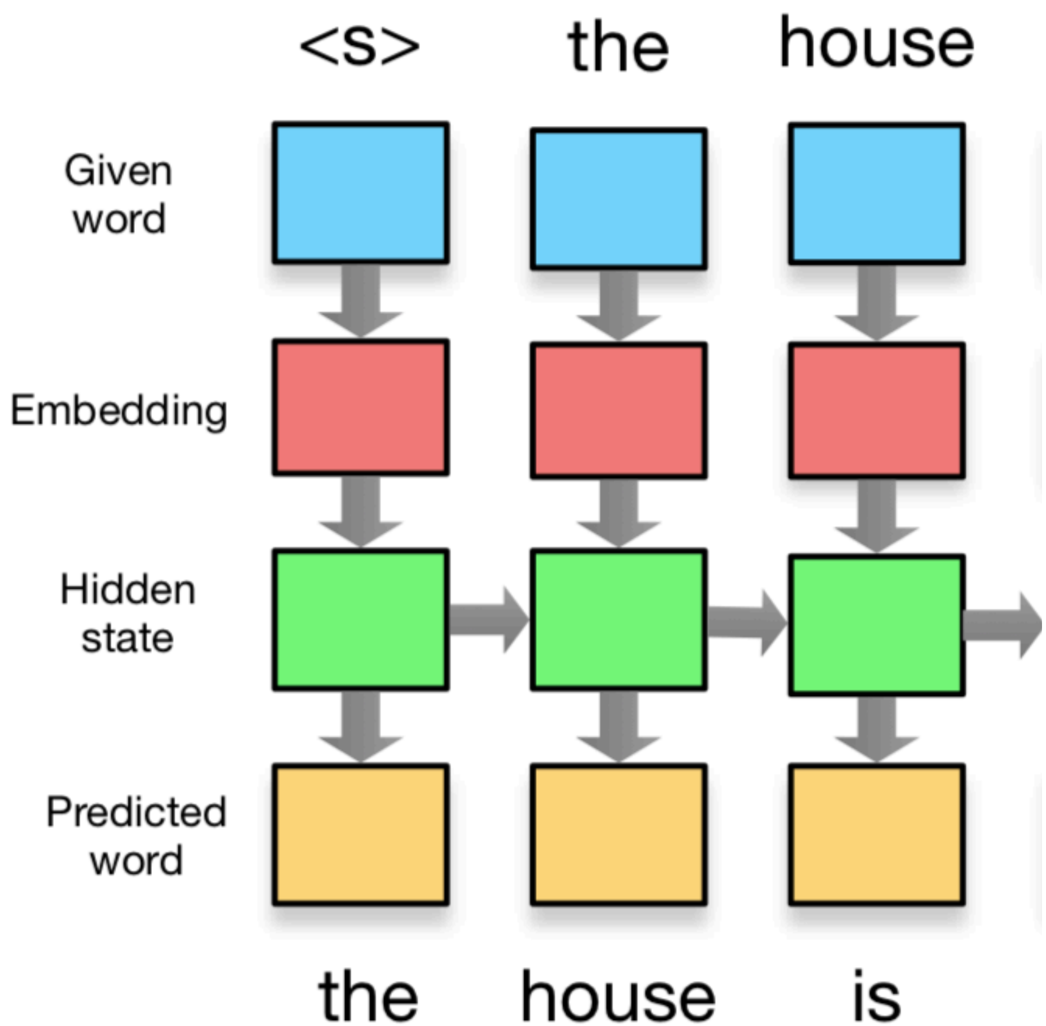
The following animations courtesy of Philipp Koehn: http://mt-class.org/jhu

Sequence modeling with a recurrent network



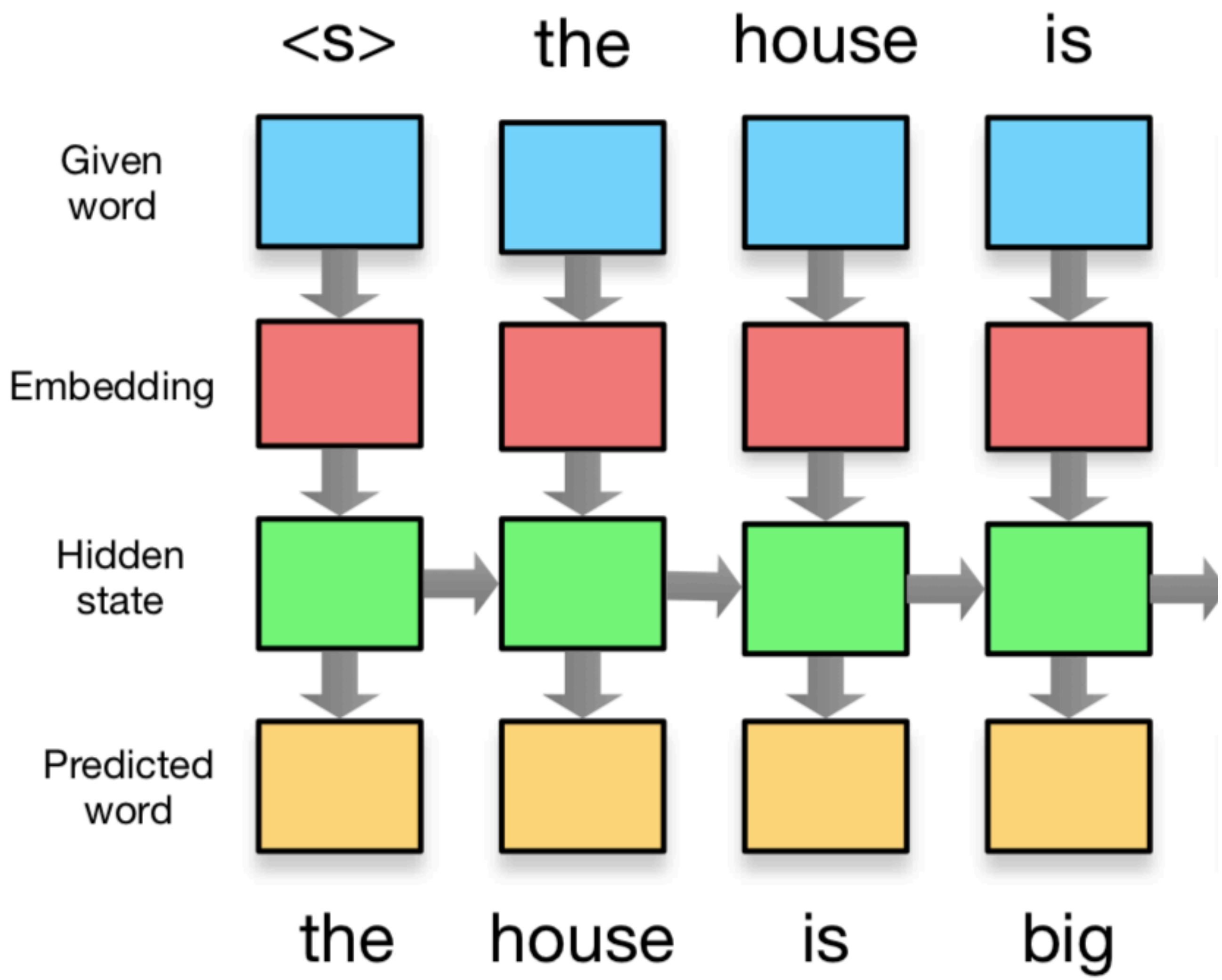
the house is big .

Sequence modeling with a recurrent network



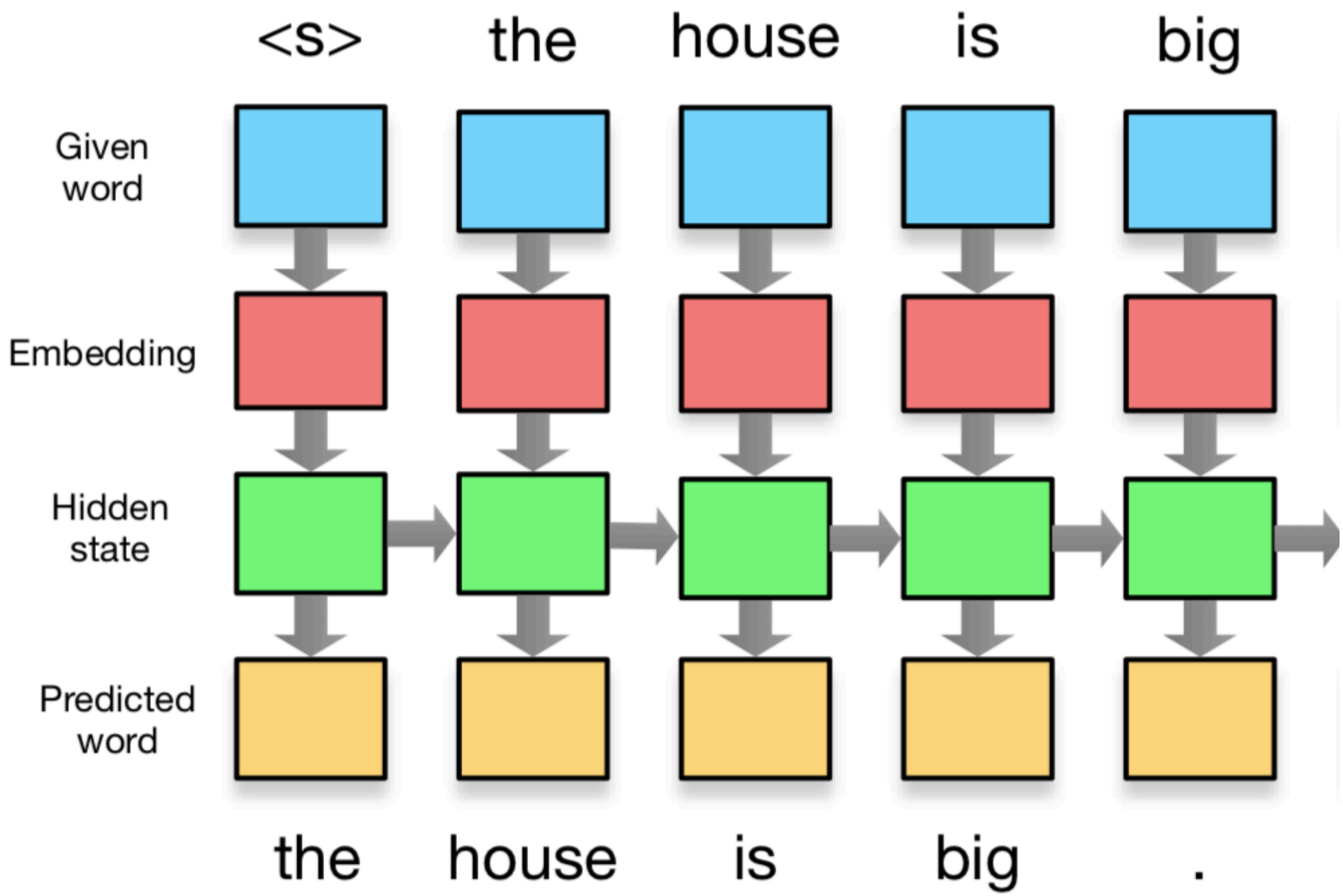
the house is big .

Sequence modeling with a recurrent network



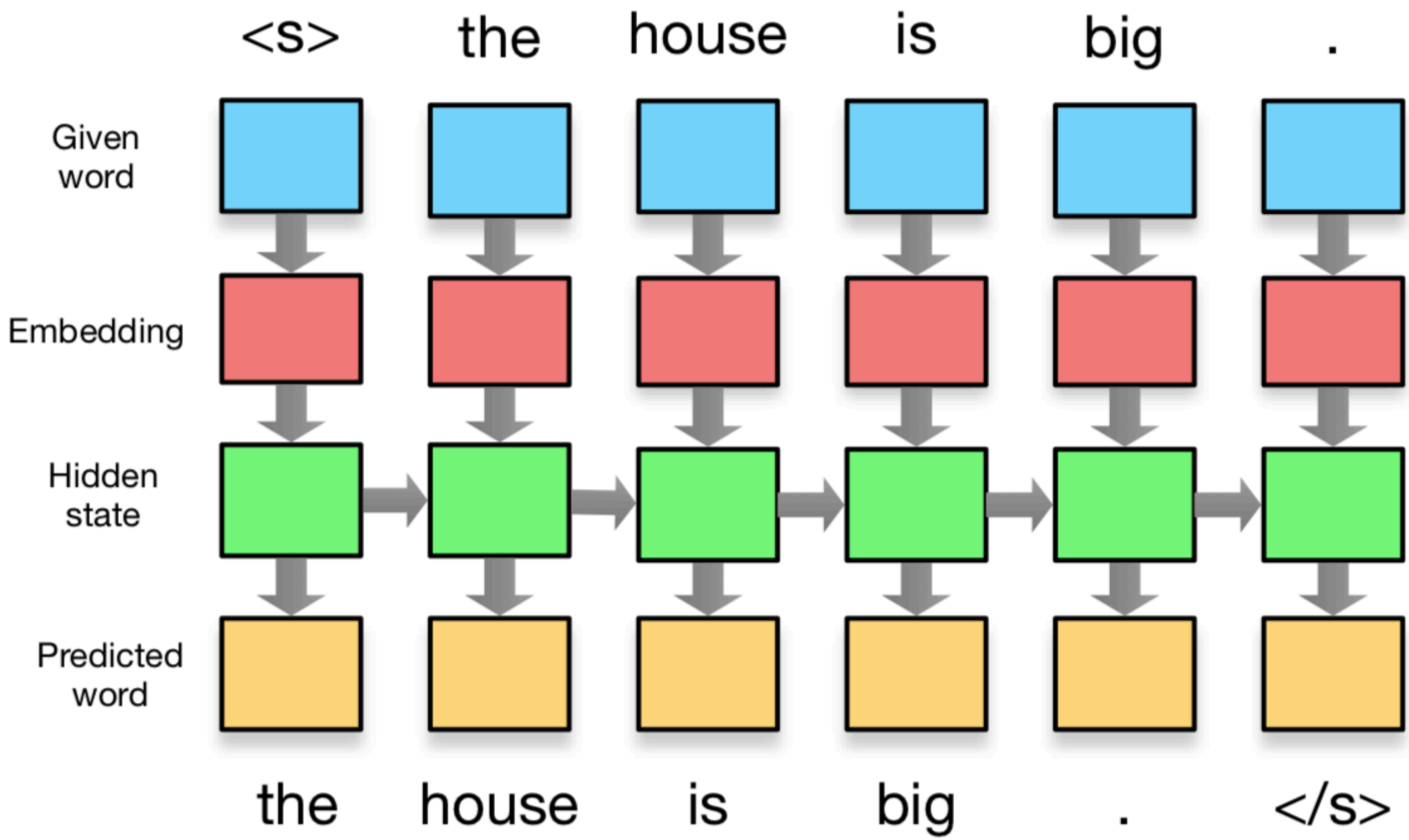
the house is big .

Sequence modeling with a recurrent network



the house is big .

Sequence modeling with a recurrent network

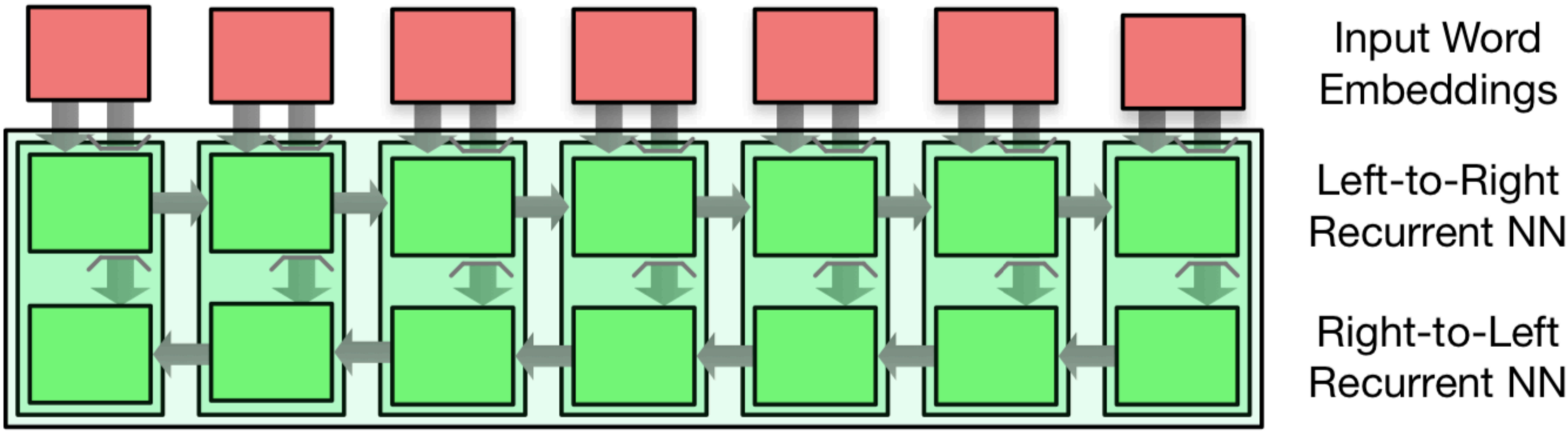


the house is big .

Recurrent models for sequence-to-sequence problems

- We can use these models for both input and output
- For output, there is the constraint of left-to-right generation
- For input, we are provided the whole sentence at once, we can do both left-to-right and right-to-left modeling
- The recurrent units may be based on LSTM, GRU, etc.

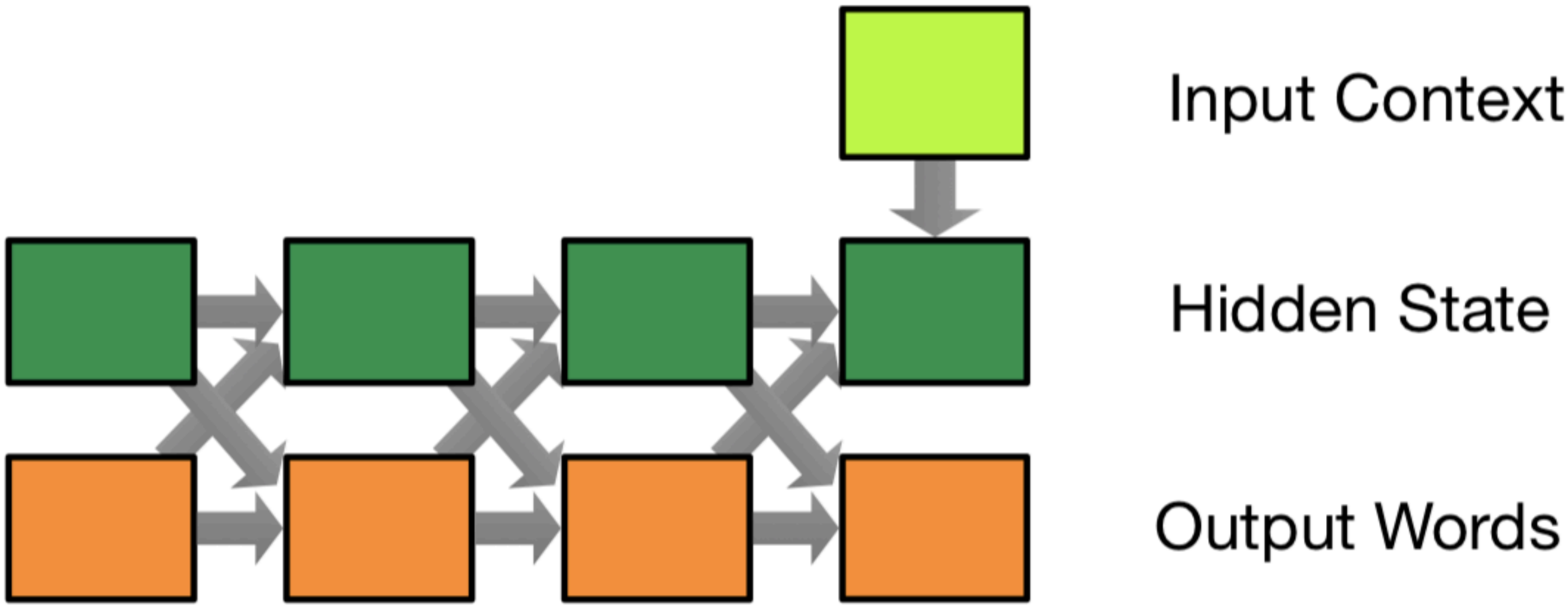
Bidirectional Encoder for Input Sequence



Word embedding: word meaning in isolation
Hidden state of each Recurrent Neural Net (RNN): word meaning in this sentence

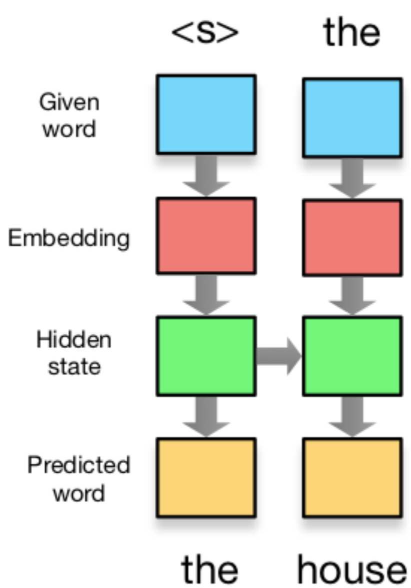
$$\overleftarrow{h_j} = f(\overleftarrow{h_{j+1}}, \bar{E}\, x_j)$$
$$\overrightarrow{h_j} = f(\overrightarrow{h_{j-1}}, \bar{E}\, x_j)$$

Left-to-Right Decoder

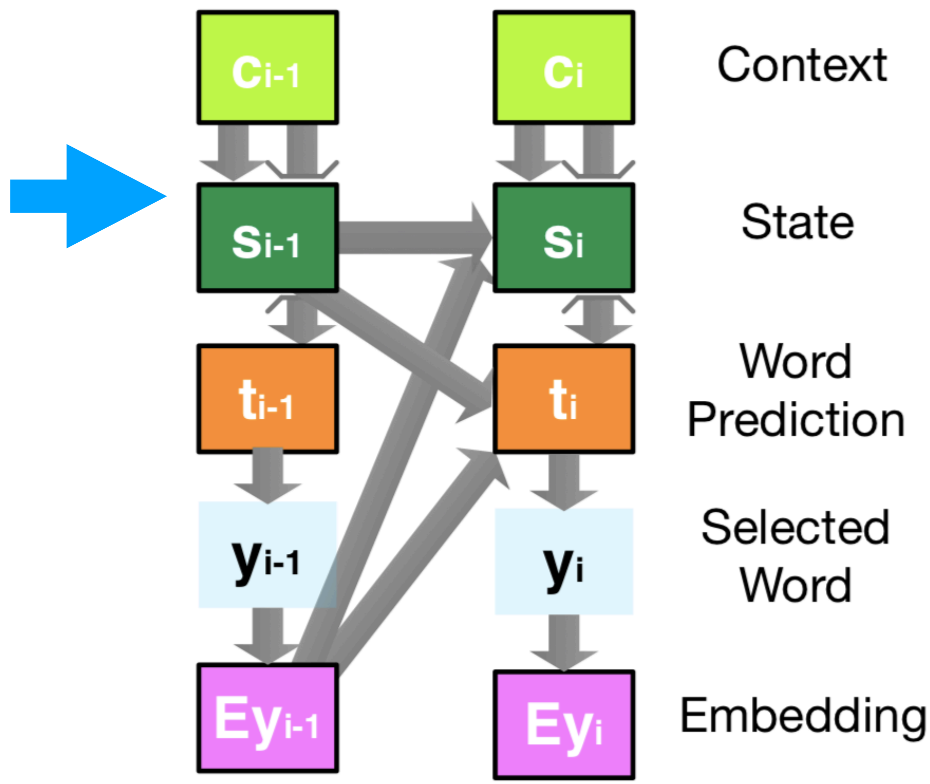


- Input context comes from encoder
- Each output is informed by current hidden state and previous output word
- Hidden state is updated at every step

In detail: each step



(simplified view)



Context contains information from encoder/input

$$s_i = f(s_{i-1},\ Ey_{-1}, c_i)$$

$$t_i = W(Us_{i-1} + VEy_{i-1} + Cc_i)$$

What connects the encoder and decoder

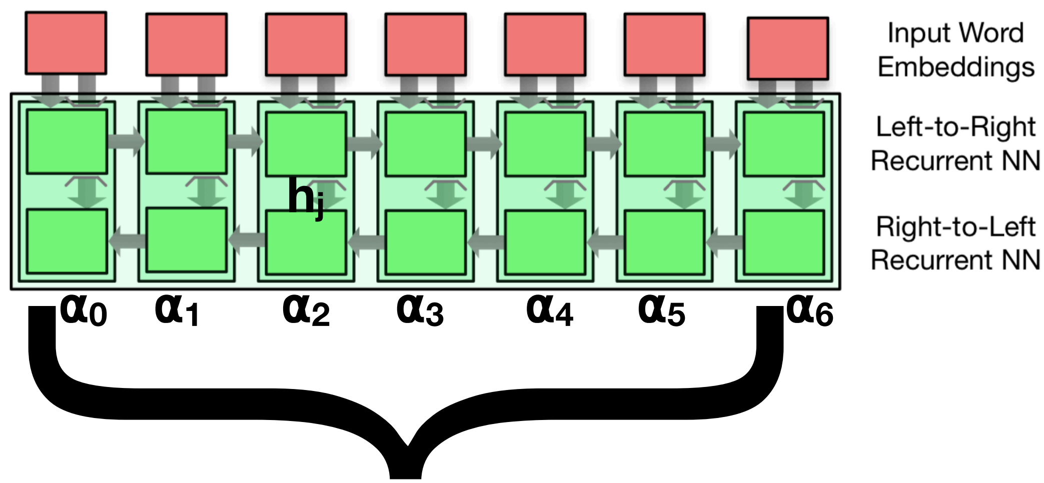
Input context is a fixed-dim vector: weighted average of all L vectors in RNN

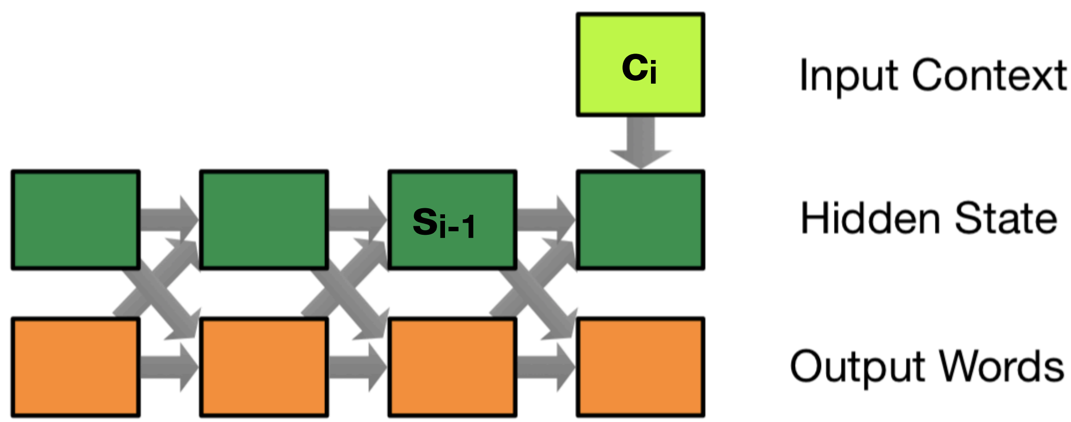
How to compute weighting? Attention mechanism:

$$\alpha_{ij} = \frac{\exp(a(s_{i-1}, h_j))}{\sum_k \exp(a(s_{i-1}, h_k))}$$

$$c_i = \sum_j \alpha_{ij} h_j$$

Note this changes at each step i
What's paid attention has more influence on next prediction

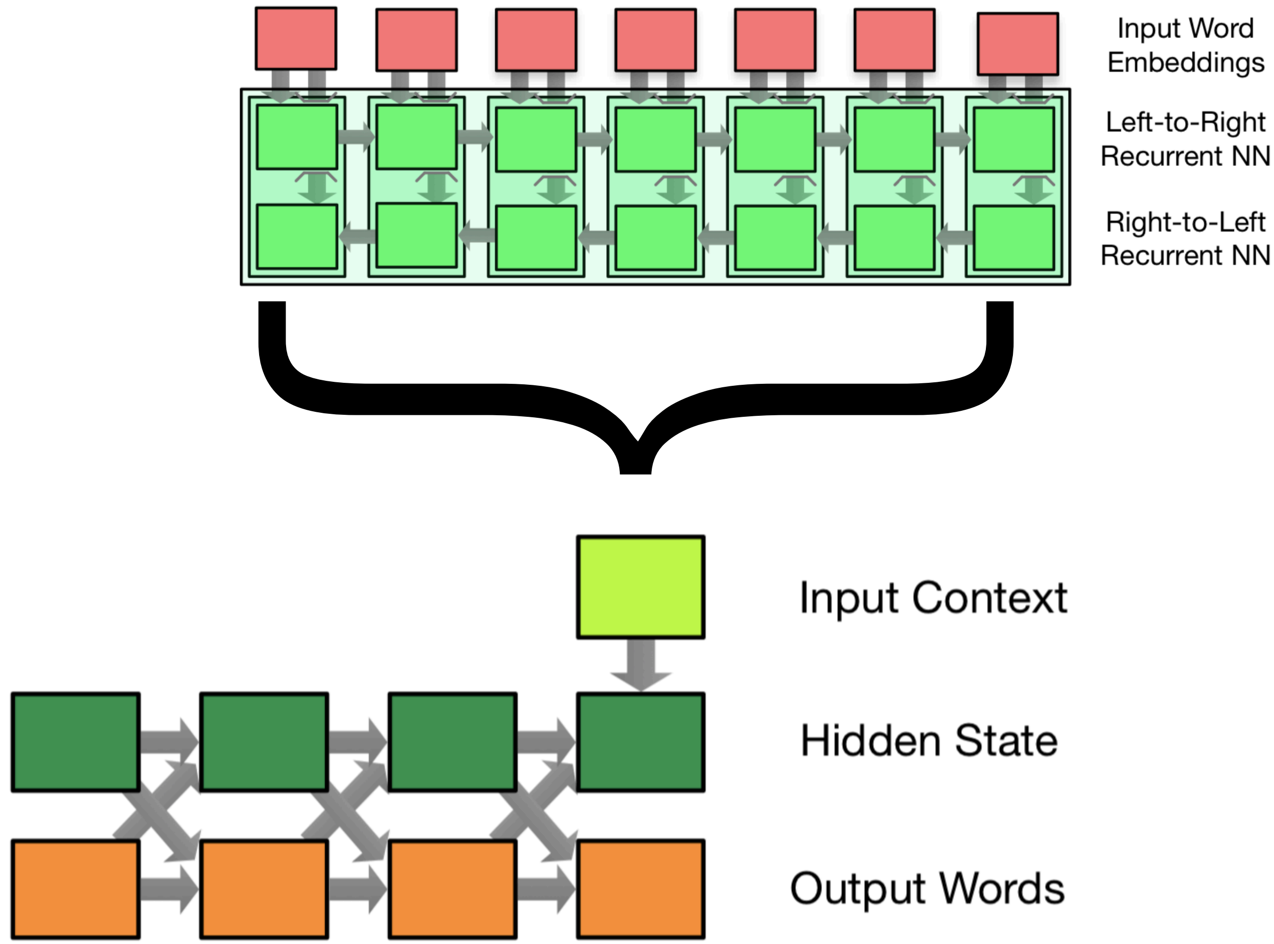




To wrap up: Recurrent models with attention

1. Encoder takes in arbitrary length input

2. Decoder generates output one word at a time, using current hidden state, input context (from attention), and previous output



Note: we can add layers to make this model "deeper"

Outline

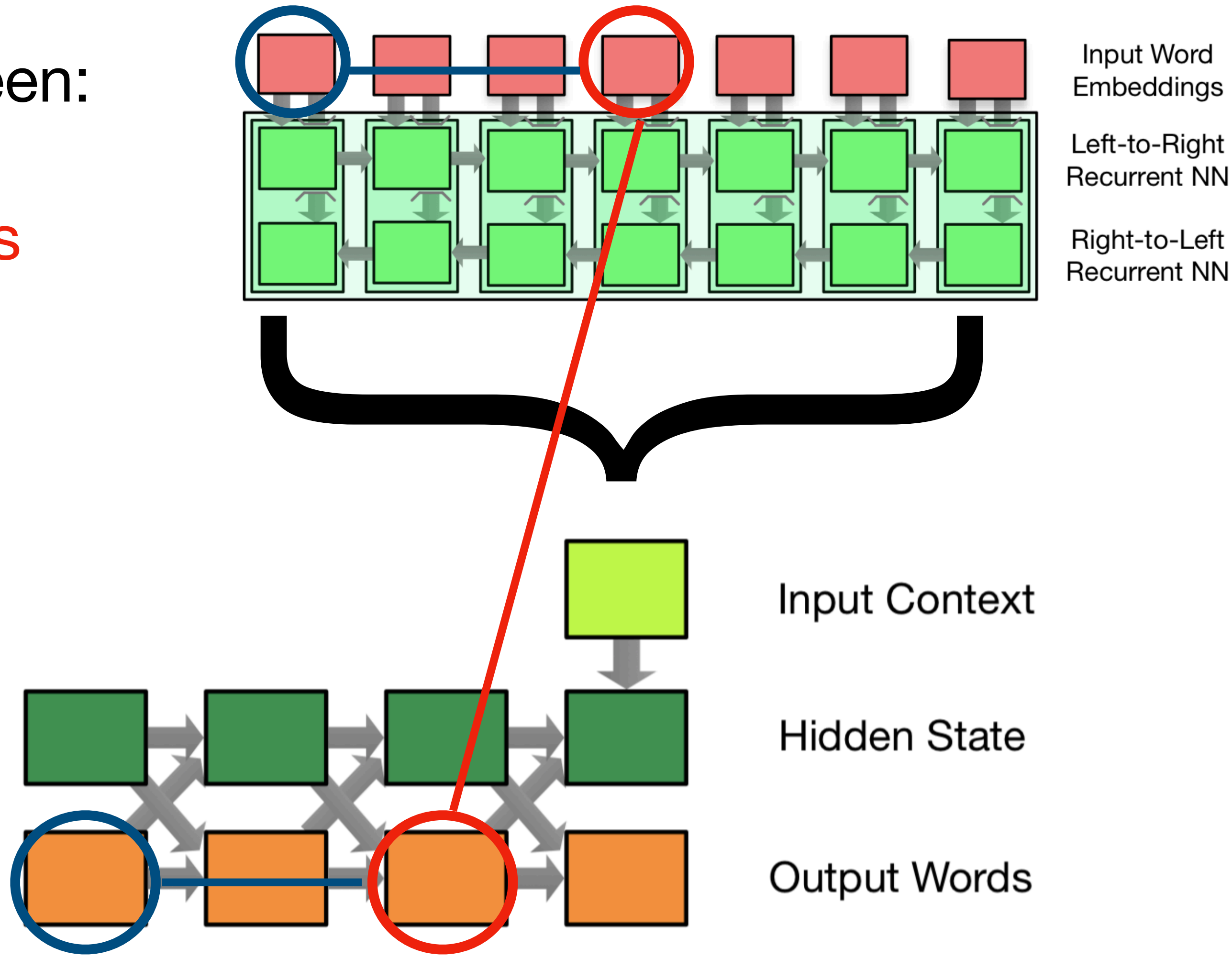
1. Background: Intuitions, SMT
2. NMT: Recurrent Model with Attention
3. NMT: Transformer Model
4. NMT with Large Language Models

Motivation of Transformer Model

- RNNs are great, but have two demerits:
  - Sequential structure is hard to parallelize, may slow down GPU computation
  - Still has to model some kinds of long-term dependency (though addressed by LSTM/GRU)
- Transformers solve the sequence-to-sequence problem using only attention mechanisms, no RNN

Long-term dependency

- Dependencies between:
  - Input-output words
  - Two input words
  - Two output words



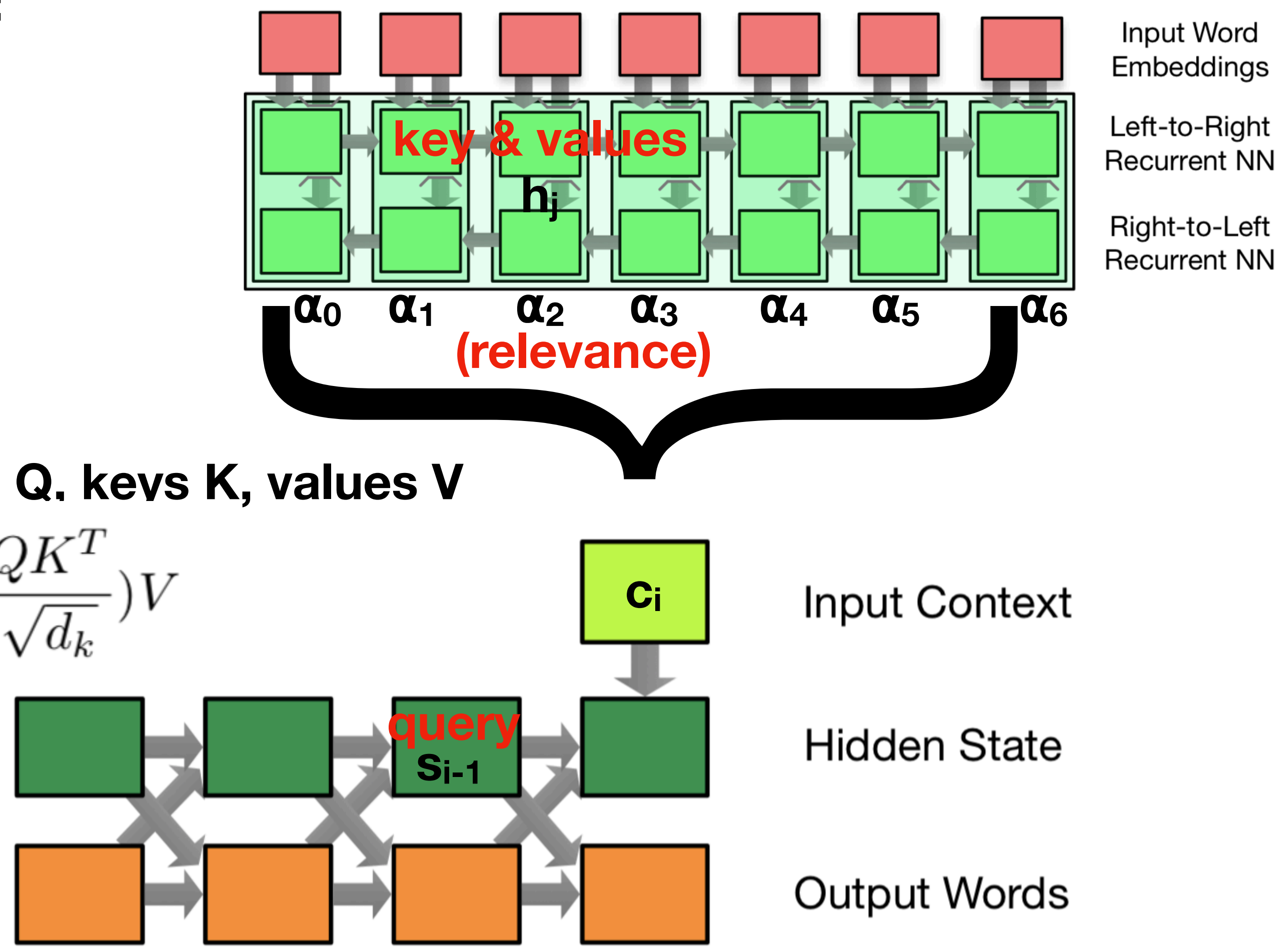
Attention mechanism "shortens" path between input and output words.
What about others?

Attention, more abstractly

Previous attention formulation:

$$\alpha_{ij} = \frac{\exp(a(s_{i-1}, h_j))}{\sum_k \exp(a(s_{i-1}, h_k))}$$

$$c_i = \sum_j \alpha_{ij} h_j$$

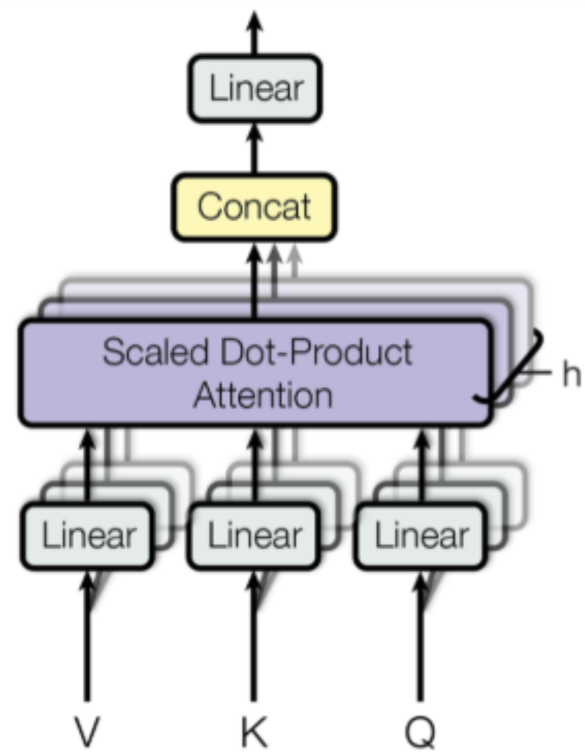


Abstract formulation:
Scaled dot-product for queries Q, keys K, values V

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

Multi-head Attention

- For expressiveness, do at scaled dot-product attention multiple times
- Add different linear transform for each key, query, value

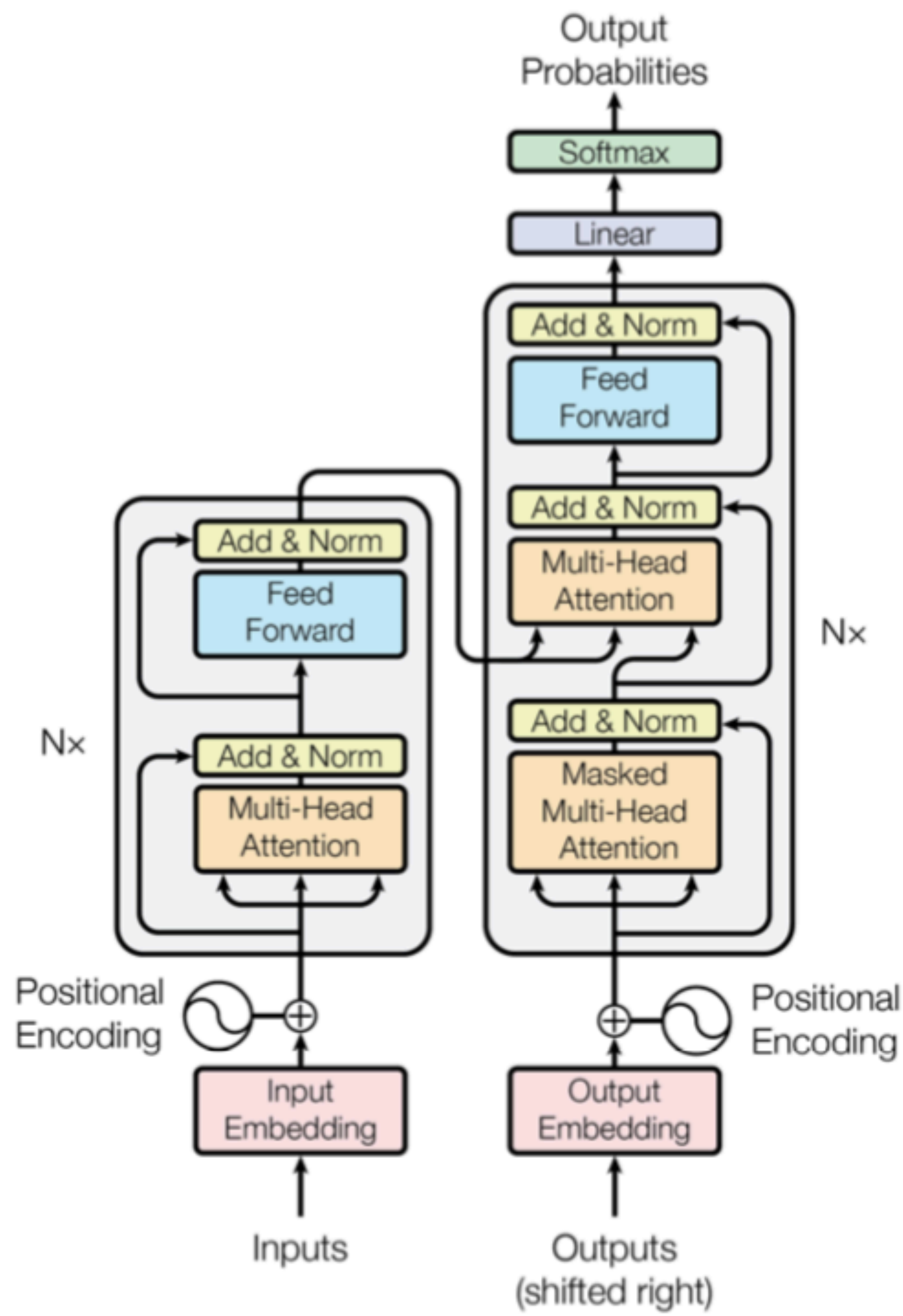


$$\mathrm{MultiHead}(Q,K,V) = \mathrm{Concat}(\mathrm{head_1}, ..., \mathrm{head_h})W^O$$
$$\text{where } \mathrm{head_i} = \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

$$W_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}, W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}, W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v} \quad W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$$

Putting it together

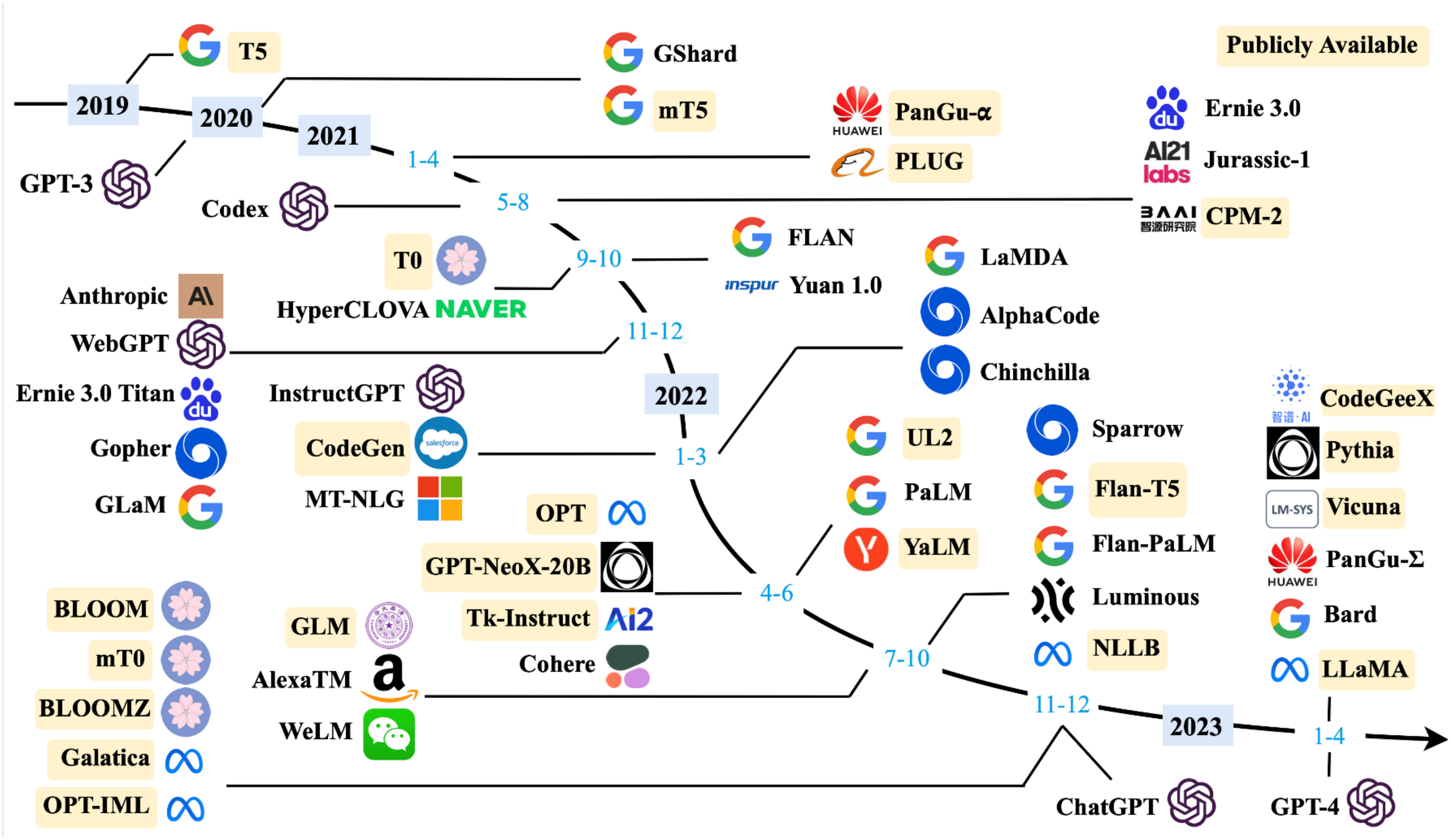
- Multiple (N) layers
- For encoder-decoder attention, Q: previous decoder layer, K and V: output of encoder
- For encoder self-attention, Q/K/V all come from previous encoder layer
- For decoder self-attention, allow each position to attend to all positions up to that position
- Positional encoding for word order



From: https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html

Outline

1. Background: Intuitions, SMT
2. NMT: Recurrent Model with Attention
3. NMT: Transformer Model
4. NMT with Large Language Models



Zhao, et. al. A Survey of Large Language Models. May 2023.
https://arxiv.org/pdf/2303.18223.pdf

Large Language Models (LLMs) for translation: why & why not

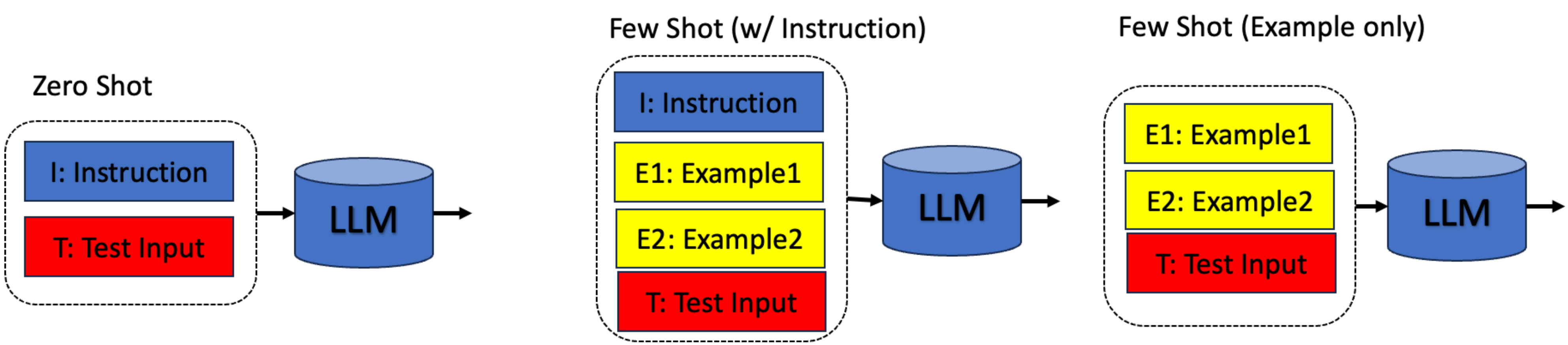
- Pros:
  - One LLM can do many tasks, e.g. translate+summarize
  - Exploits vastly large amounts of data
- Cons
  - Inference cost may be higher than smaller dedicated NMT models
  - Dependence on specific LLMs

Large Language Models (LLMs) for translation: potential?

- In-context learning may allow quicker adaptation of translation to new domains and styles?
- Document-level adaptation is possible with LLM's long input context?

In-Context Learning

I: Instruction	Translate English to French	
E1: Example1	[en]: A discomfort which lasts.	[fr]: Un malaise qui dure
E2: Example2	[en]: HTML is a language for formatting	[fr]: HTML est un langage de formatage
T: Test Input	[en]: After you become comfortable with formatting	[fr]:



Document-Level Translation

- Info may be underspecified at the sentence level
- Document is a clear-cut unit
  - Internally coherent, e.g. one sense per discourse
  - Many practical translation tasks are document-based

Human Expert Translator

Each TED talk is translated as a whole

source: www.ted.com

Summary

1. Background
    - Learning translation knowledge from data
2. Recurrent Model with Attention
    - Bidirectional RNN encoder, RNN decoder, attention-based context vector tying it together
3. Transformer Model
    - Another way to solve sequence problems, without using sequential models
4. NMT with Large Language Models
    - Fine-tuning & In-Context Learning are new paradigms

Questions? Comments?