An Introduction to Neural Machine Translation

Kevin Duh
slides courtesy of Philipp Koehn

August 2018

Outline

- Machine Translation: History & Problem Formulation
- Language Model
- Encoder-Decoder NMT Model
- Training & Inference
- Alternative NMT Models

some history

An Old Idea

Warren Weaver on translation
as code breaking (1947):

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".

Early Efforts and Disappointment

- Excited research in 1950s and 1960s

1954
Georgetown experiment
Machine could translate
250 words and
6 grammar rules

- 1966 ALPAC report:
  - only $20 million spent on translation in the US per year
  - no point in machine translation

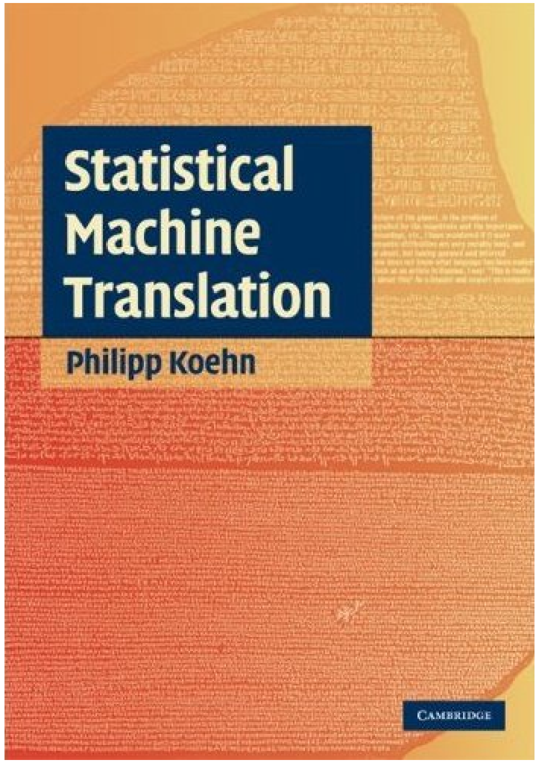
Rule-Based Systems

- Rule-based systems
  - build dictionaries
  - write transformation rules
  - refine, refine, refine
- Météo system for weather forecasts (1976)
- Systran (1968), Logos and Metal (1980s)

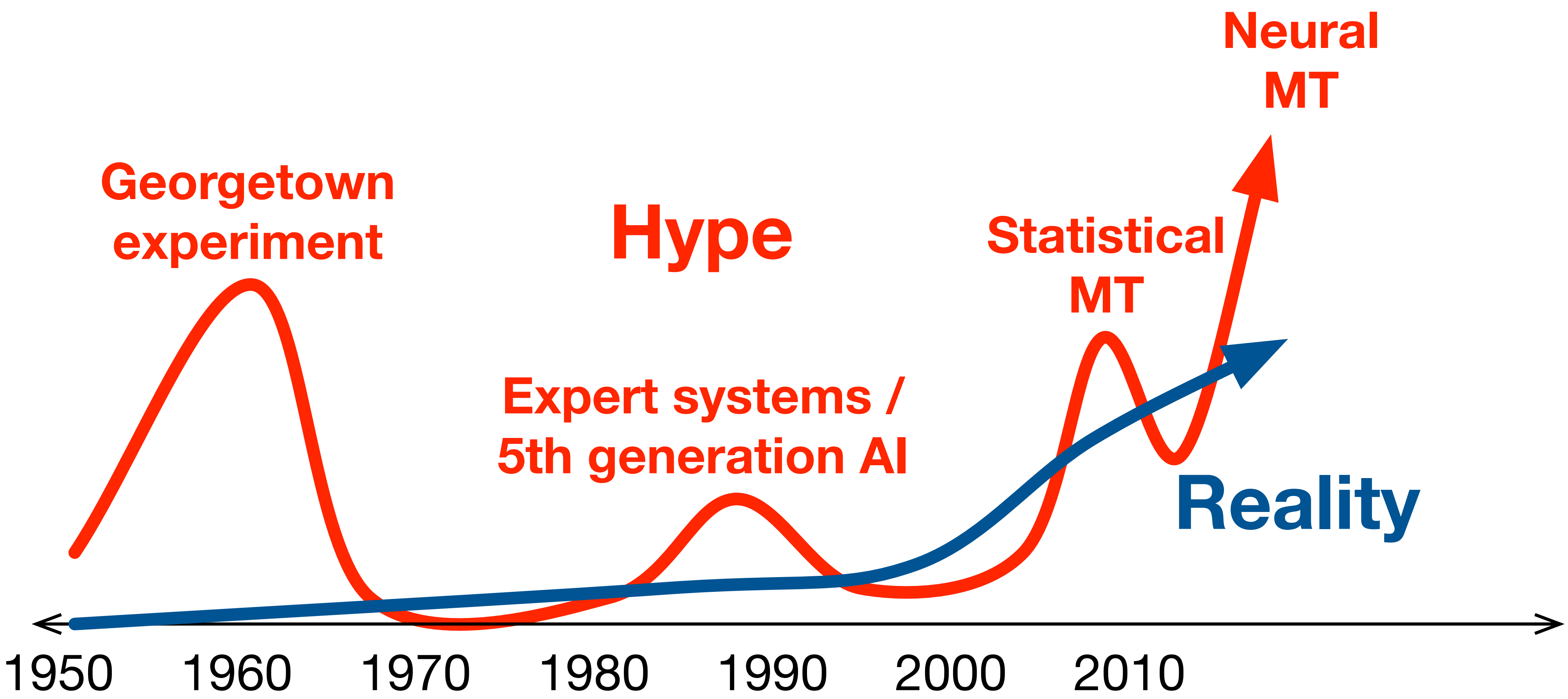
```
"have" :=

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

- 1980s: IBM
- 1990s: increased research
- Mid 2000s: Phrase-Based MT (Moses, Google)
- Around 2010: commercial viability
- Since mid 2010s: neural network models



Hype


Neural
MT
Georgetown
experiment
Hype
Statistical
MT
Expert systems /
5th generation AI
Reality
1950
1960
1970
1980
1990
2000
2010

how good is machine translation?

Machine Translation: Chinese

记者从环保部了解到，《水十条》要求今年年底前直辖市、省会城市、计划单列市建成区基本解决黑臭水体。截至目前，全国224个地级及以上城市共排查确认黑臭水体2082个，其中34.9%完成整治，28.4%正在整治，22.8%正在开展项目前期。

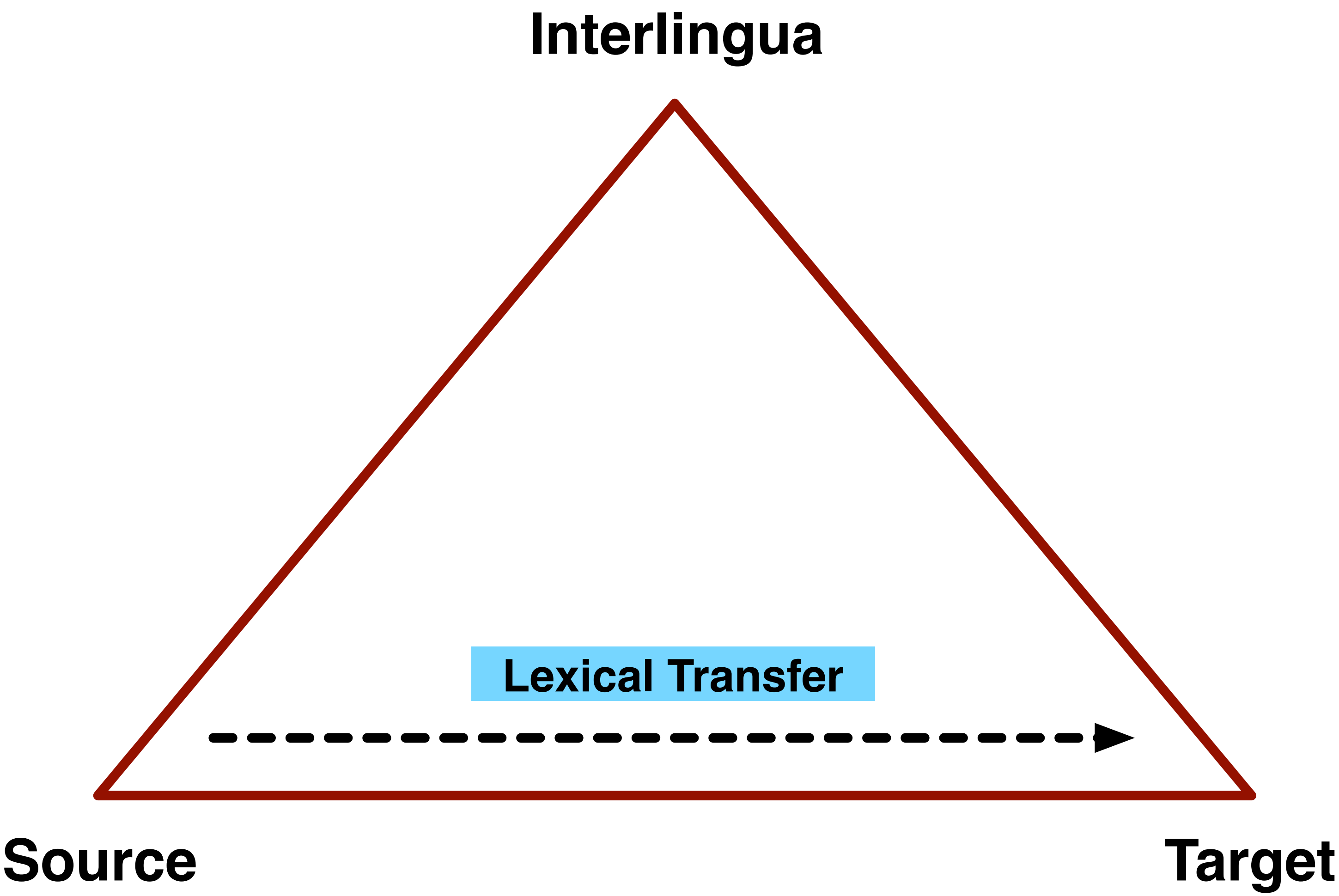
Reporters learned from the Ministry of Environmental Protection, "Water 10" requirements before the end of this year before the municipality, the provincial capital city, plans to build a separate city to solve the basic black and black water. Up to now, the country's 224 prefecture-level and above cities were identified to confirm the black and white water 2082, of which 34.9% to complete the renovation, 28.4% is remediation, 22.8% is carrying out the project early.

Machine Translation: French

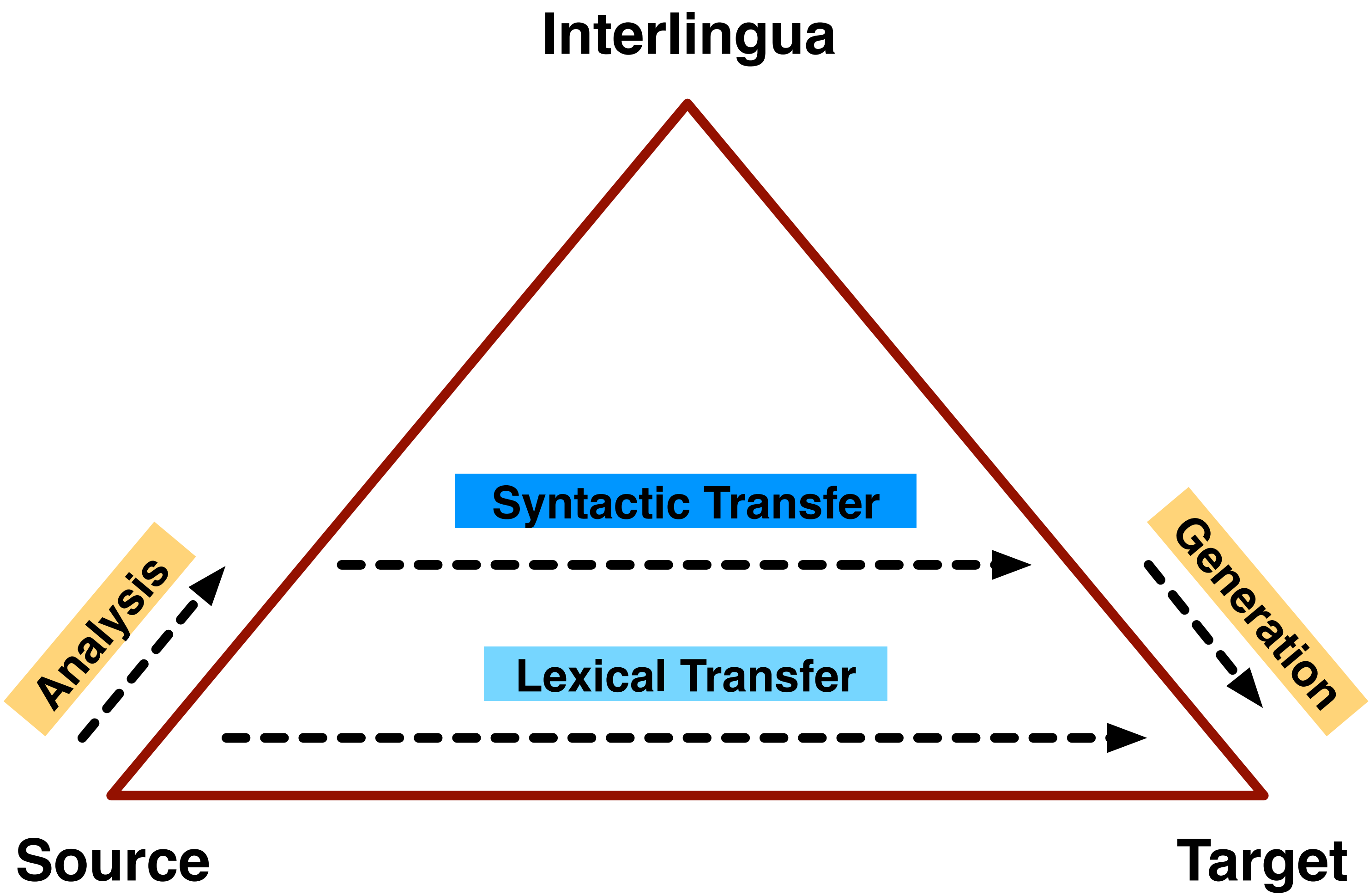
A l'orée de ce débat télévisé inédit dans l'histoire de la Ve République, on attendait une forme de «Tous sur Macron» mais c'est la candidate du Front national qui s'est retrouvée au cœur des premières attaques de ses quatre adversaires d'un soir, favorisées par le premier thème abordé, les questions de société et donc de sécurité, d'immigration et de laïcité.

At the beginning of this televised debate, which was unheard of in the history of the Fifth Republic, a "Tous sur Macron" was expected, but it was the candidate of the National Front who found itself at the heart of the first attacks of its four Opponents of one evening, favored by the first theme tackled, the issues of society and thus security, immigration and secularism.

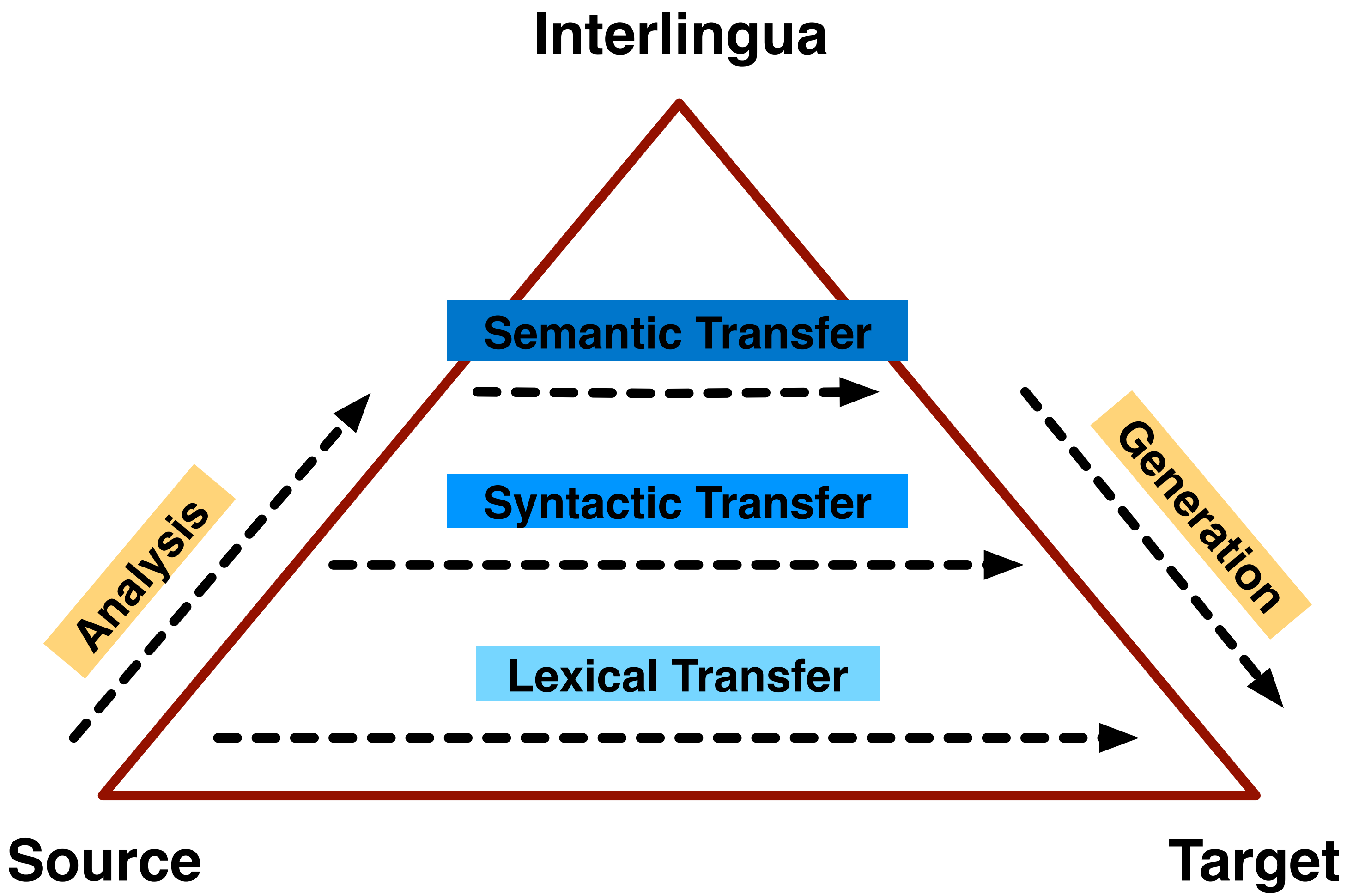
A Clear Plan



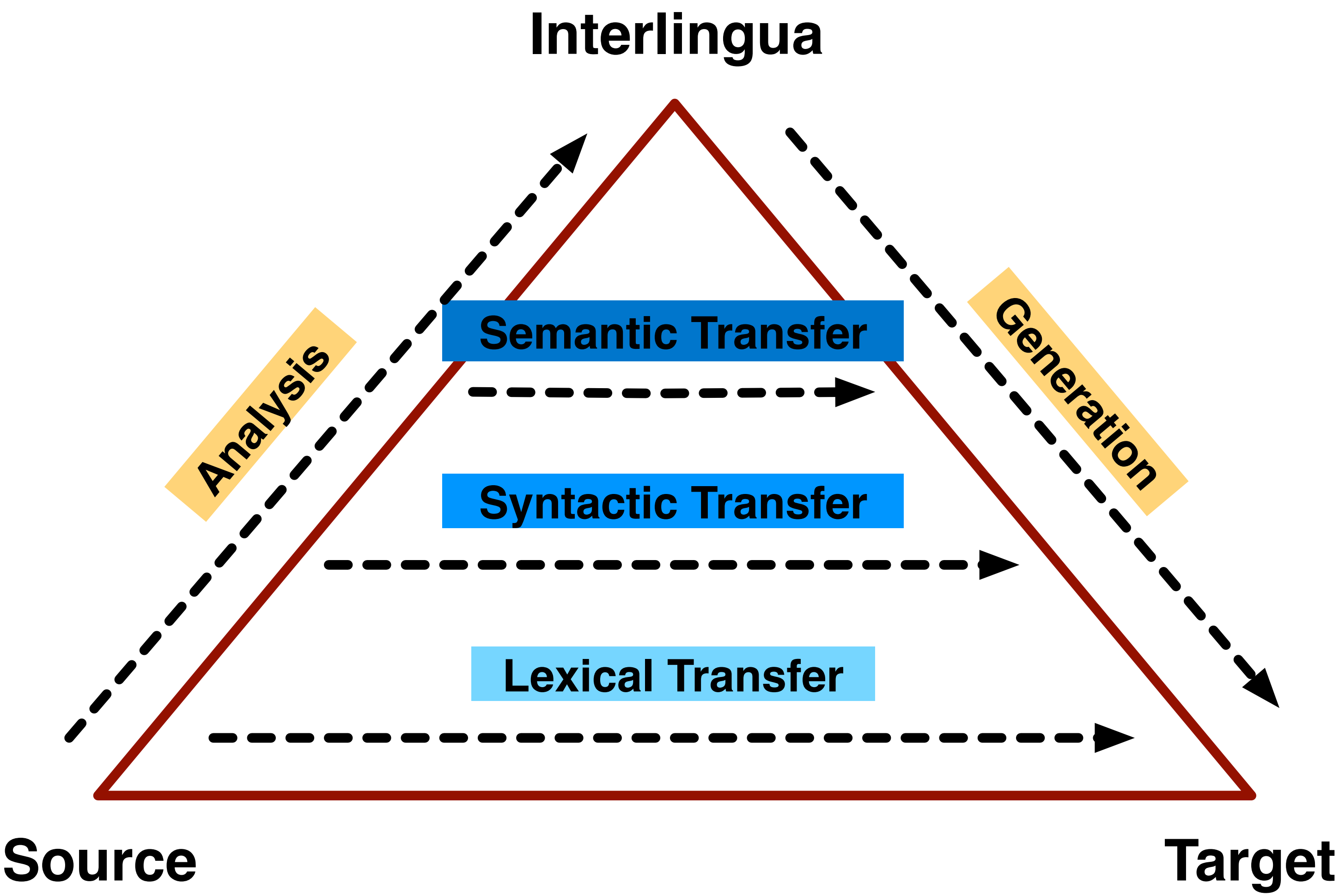
A Clear Plan


Interlingua
Syntactic Transfer
Lexical Transfer
Analysis
Generation
Source
Target

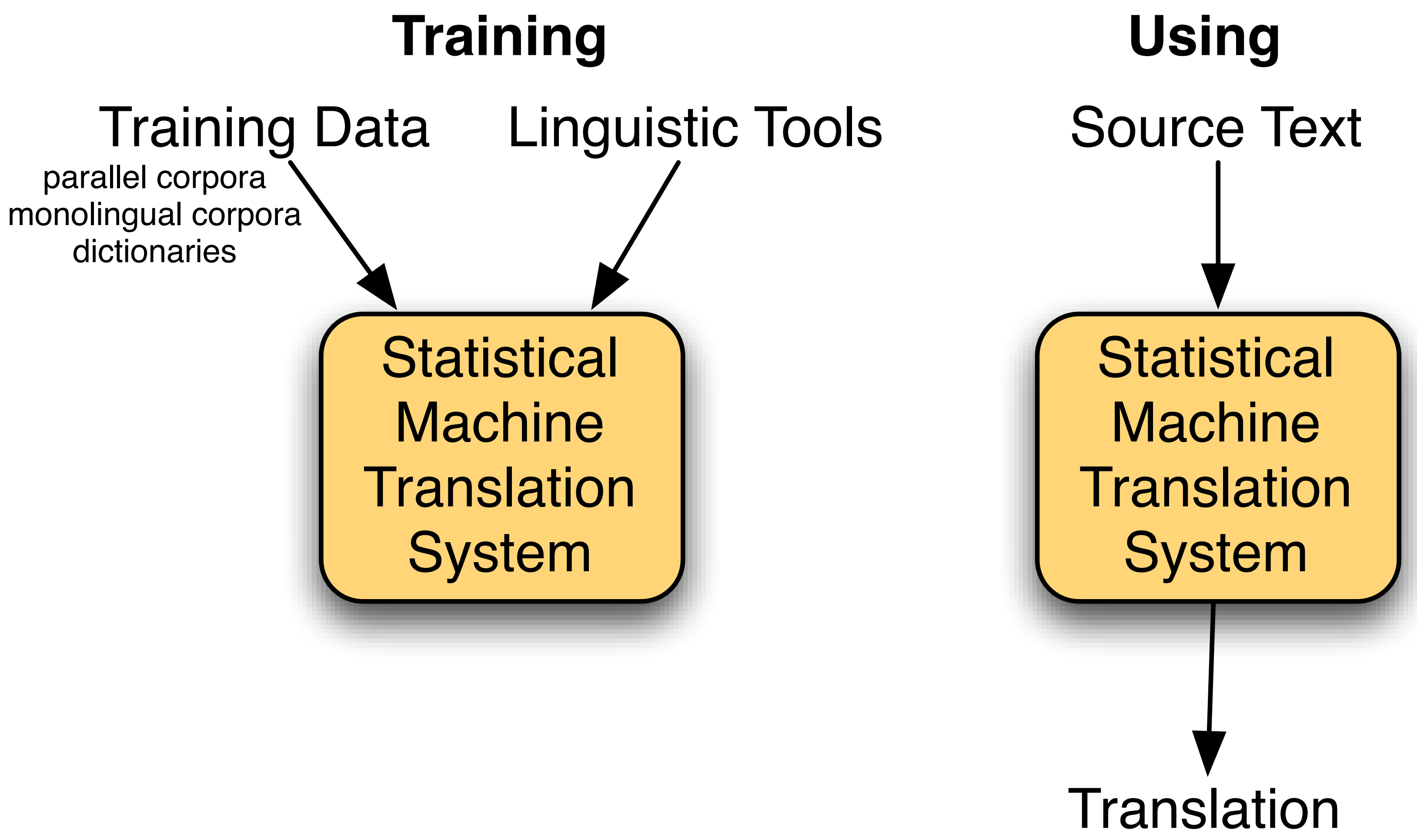
A Clear Plan


Interlingua
Semantic Transfer
Syntactic Transfer
Lexical Transfer
Analysis
Generation
Source
Target

A Clear Plan


Interlingua
Analysis
Generation
Semantic Transfer
Syntactic Transfer
Lexical Transfer
Source
Target

Learning from Data



why is that a good plan?

Word Translation Problems

- Words are ambiguous

He deposited money in a bank account
with a high interest rate.

Sitting on the bank of the Mississippi,
a passing ship piqued his interest.

- How do we find the right meaning, and thus translation?

- Context should be helpful

Learning from Data

- What is the best translation?

$$\text{Sicherheit} \rightarrow \text{security}$$
$$\text{Sicherheit} \rightarrow \text{safety}$$
$$\text{Sicherheit} \rightarrow \text{certainty}$$

Learning from Data

- What is the best translation?

$$\text{Sicherheit} \rightarrow \text{security } 14{,}516$$
$$\text{Sicherheit} \rightarrow \text{safety } 10{,}015$$
$$\text{Sicherheit} \rightarrow \text{certainty } 334$$

- Counts in European Parliament corpus

Learning from Data

- What is the best translation?

Sicherheit → security 14,516
Sicherheit → safety 10,015
Sicherheit → certainty 334

- Phrasal rules

Sicherheitspolitik → security policy 1580
Sicherheitspolitik → safety policy 13
Sicherheitspolitik → certainty policy 0

Lebensmittelsicherheit → food security 51
Lebensmittelsicherheit → food safety 1084
Lebensmittelsicherheit → food certainty 0

Rechtssicherheit → legal security 156
Rechtssicherheit → legal safety 5
Rechtssicherheit → legal certainty 723

Learning from Data

- What is most fluent?

a problem for translation
a problem of translation
a problem in translation

Learning from Data

- What is most fluent?

a problem for translation 13,000

a problem of translation 61,600

a problem in translation 81,700

- Hits on Google

Learning from Data

- What is most fluent?

a problem for translation 13,000

a problem of translation 61,600

a problem in translation 81,700

a translation problem 235,000

Learning from Data

- What is most fluent?

police disrupted the demonstration
police broke up the demonstration
police dispersed the demonstration
police ended the demonstration
police dissolved the demonstration
police stopped the demonstration
police suppressed the demonstration
police shut down the demonstration

Learning from Data

- What is most fluent?

police disrupted the demonstration 2,140
police broke up the demonstration 66,600
police dispersed the demonstration 25,800
police ended the demonstration 762
police dissolved the demonstration 2,030
police stopped the demonstration 722,000
police suppressed the demonstration 1,400
police shut down the demonstration 2,040

Terminology

- Text, Corpus, Data is often used interchangeably.
- Parallel Text (Bitext): sentence-aligned text
  $\{(s1, t1), (s2, t2), \ldots, (s99999, t999999)\}$
- Source text $\rightarrow$ Target text
- We use the training data (bitext) to estimate parameters of our model
- We use the Development/Validation data to select models or tune a few hyperparameters.
- Finally, report results on Test data, i.e. translate source text of test and compare with target text of test.

Questions?

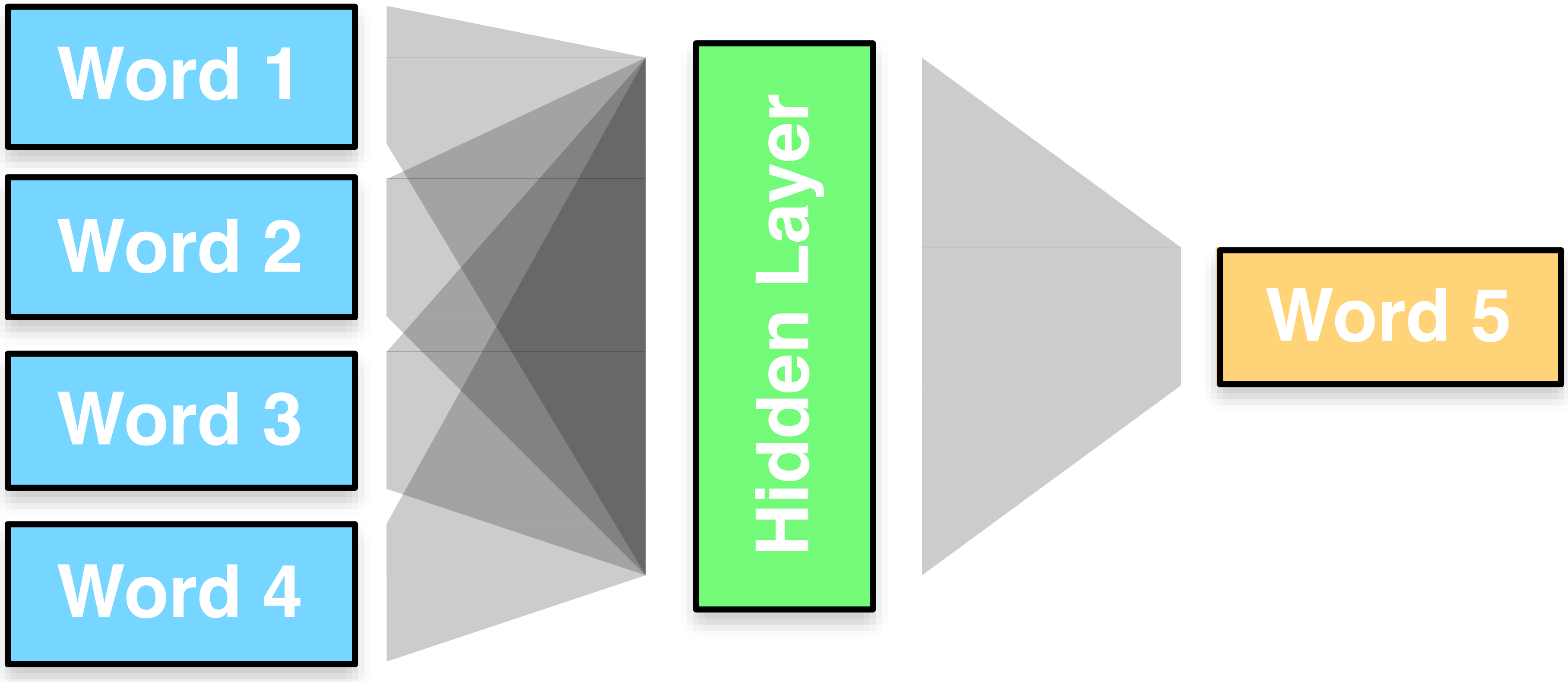
Outline

- Machine Translation: History & Problem Formulation
- Language Model
- Encoder-Decoder NMT Model
- Training & Inference
- Alternative NMT Models

Language Model (a very important task!)

- Goal: Given the past words, predict the next word

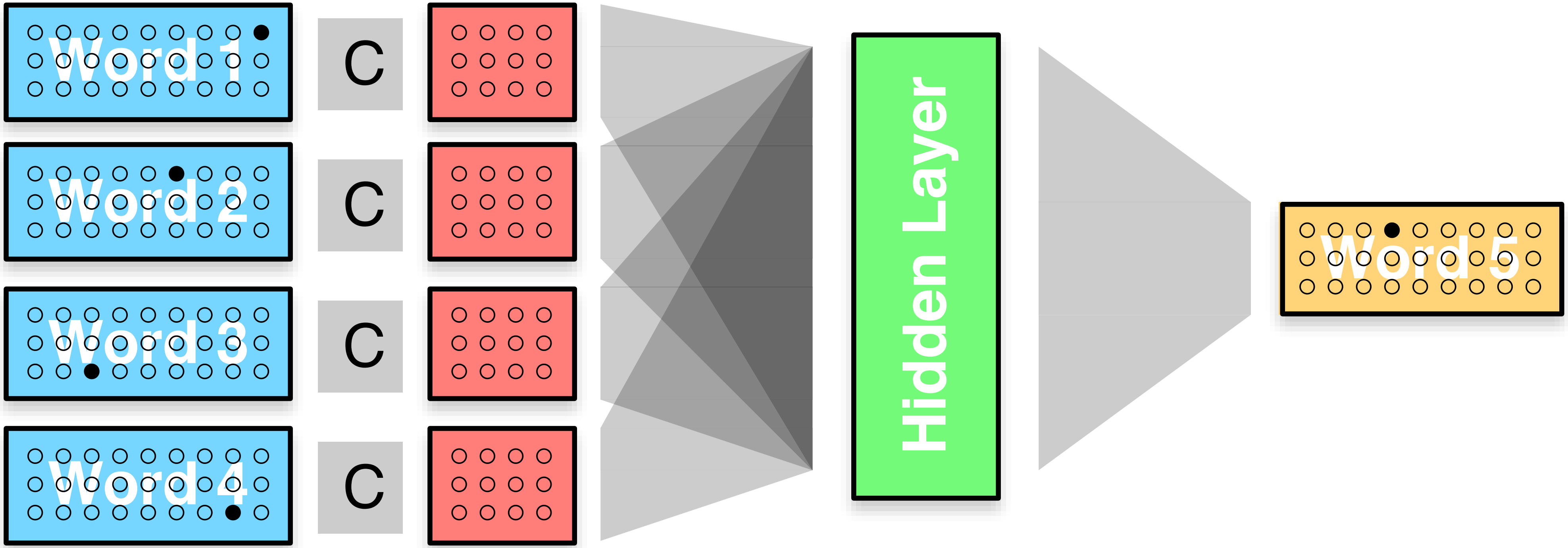
- $p(w_i|w_1, ..., w_{i-1}) \simeq p(w_i|w_{i-4}, w_{i-3}, w_{i-2}, w_{i-1})$



Representing Words

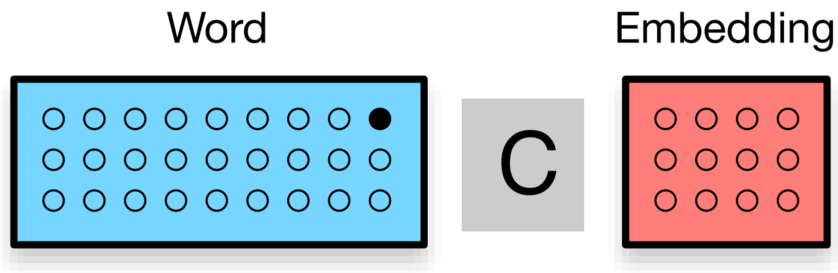
- Words are represented with a one-hot vector, e.g.,
  - dog = (0,0,0,0,1,0,0,0,0,....)
  - cat = (0,0,0,0,0,0,0,1,0,....)
  - eat = (0,1,0,0,0,0,0,0,0,....)
- To model $|V|$ words, one-hot vector is length $|V|$
- We would like to map words to smaller-dimension word embeddings first

Feedforward Neural Language Model



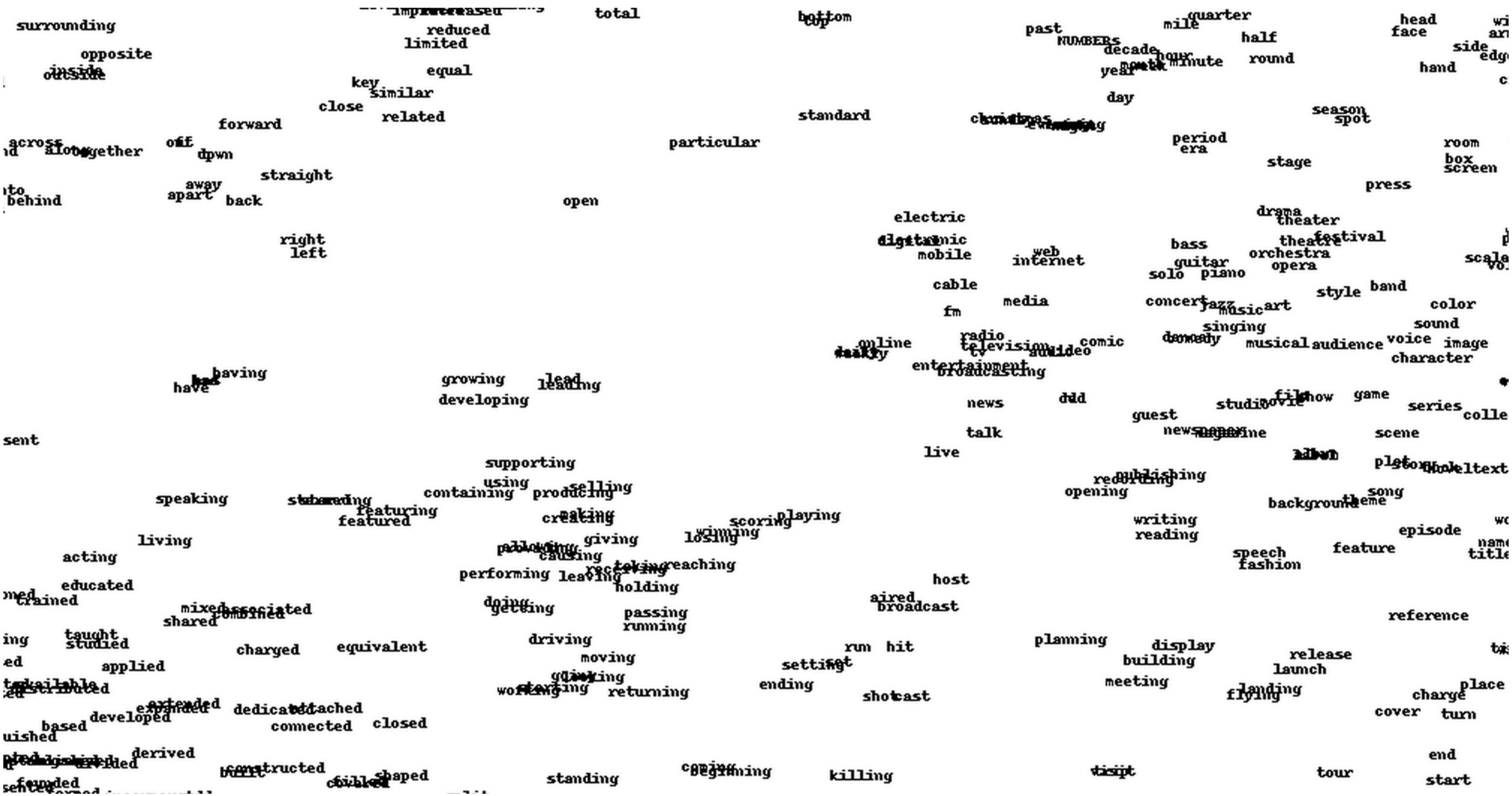
- Map each word first into a lower-dimensional real-valued space
- Shared weight matrix C

Word Embeddings

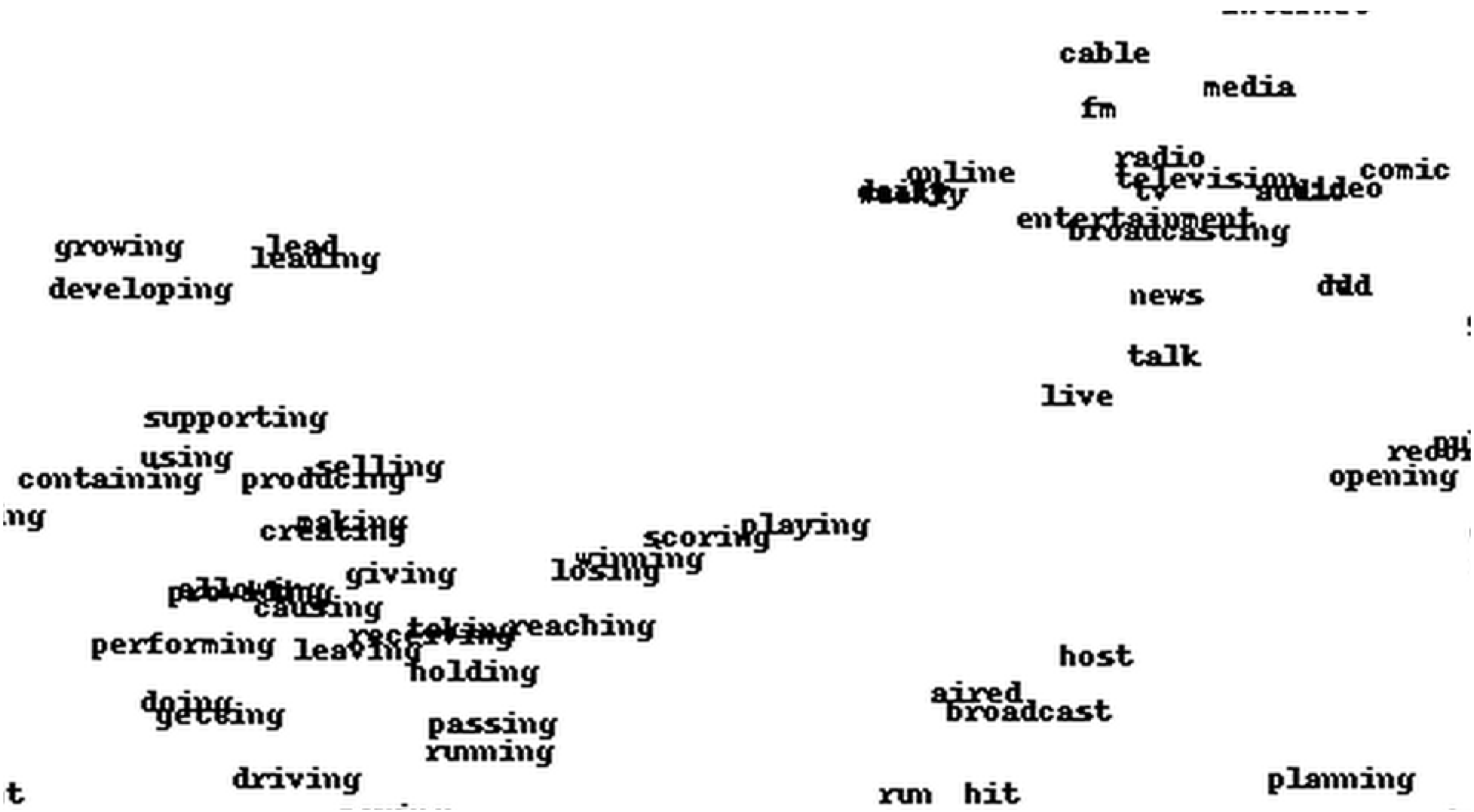


- By-product: embedding of word into continuous space
- Similar contexts $\rightarrow$ similar embedding

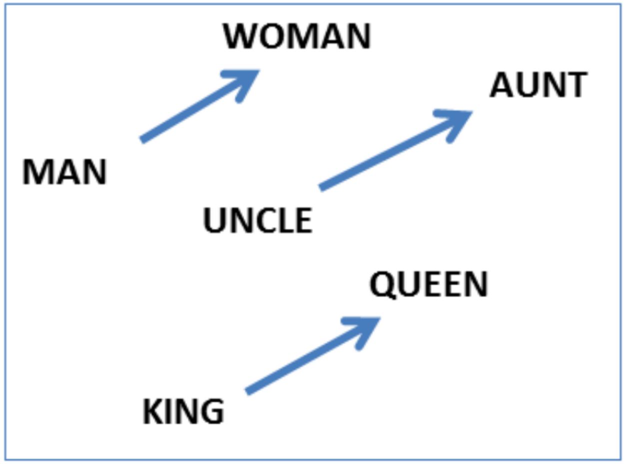
Word Embeddings

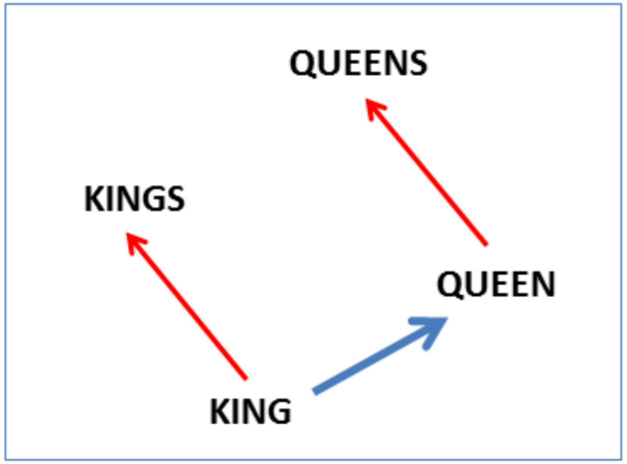


Word Embeddings



Are Word Embeddings Magic?

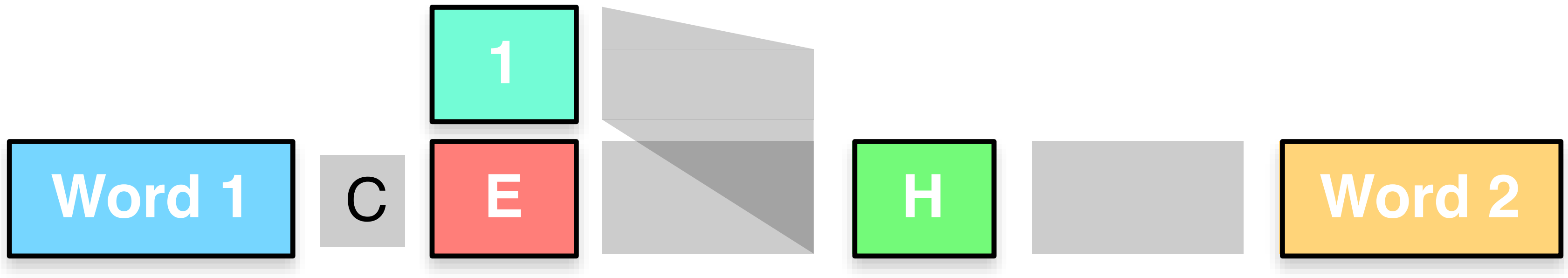




- Morphosyntactic regularities (Mikolov et al., 2013)
  - adjectives base form vs. comparative, e.g., good, better
  - nouns singular vs. plural, e.g., year, years
  - verbs present tense vs. past tense, e.g., see, saw
- Semantic regularities
  - clothing is to shirt as dish is to bowl
  - evaluated on human judgment data of semantic similarities

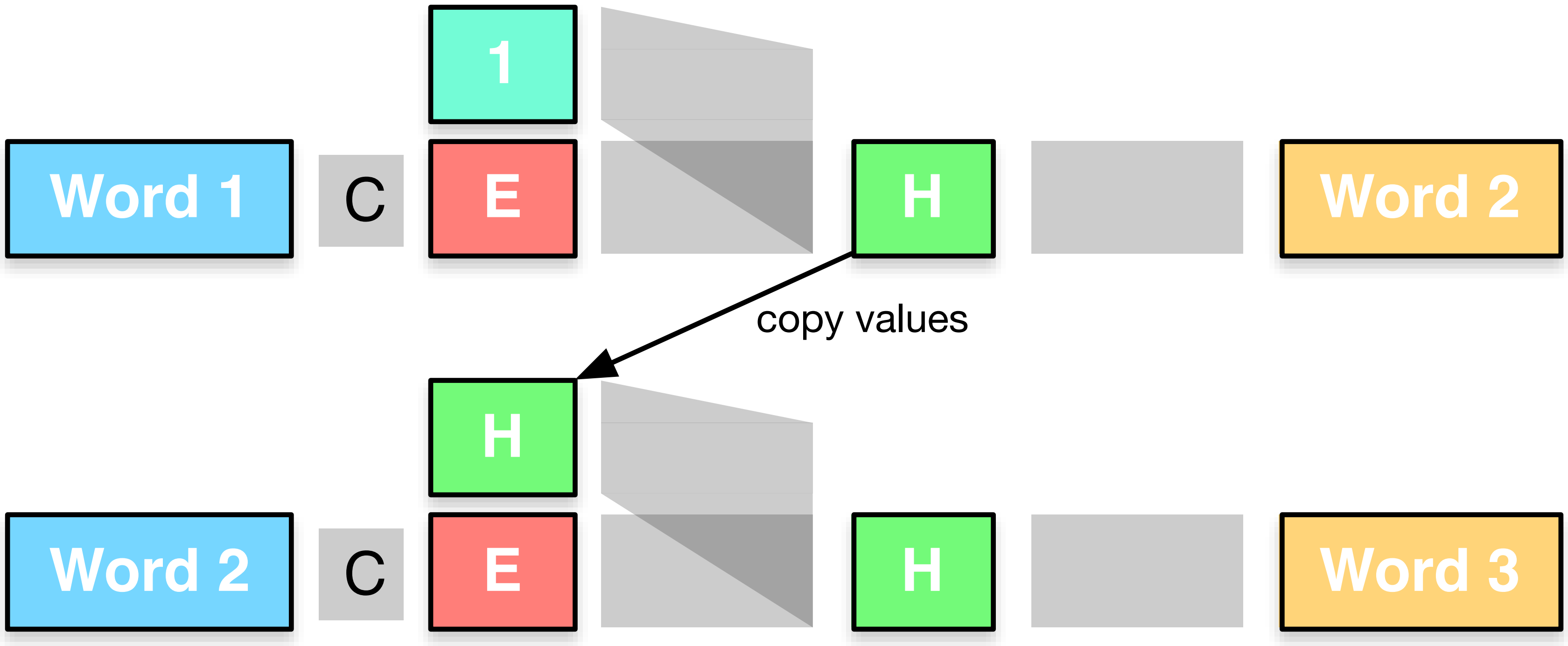
recurrent neural networks

Recurrent Neural Networks

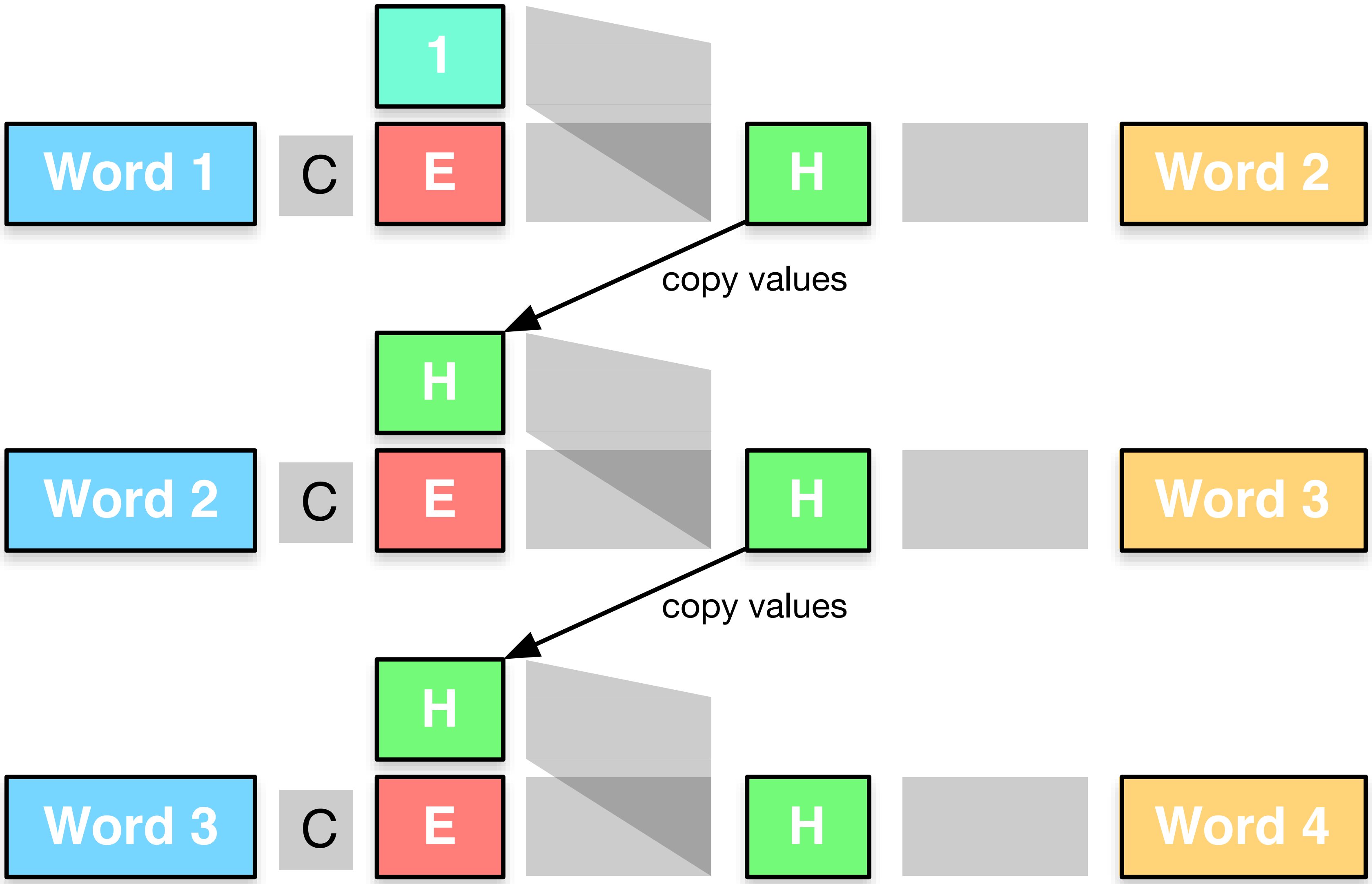


- Start: predict second word from first
- Mystery layer with nodes all with value 1

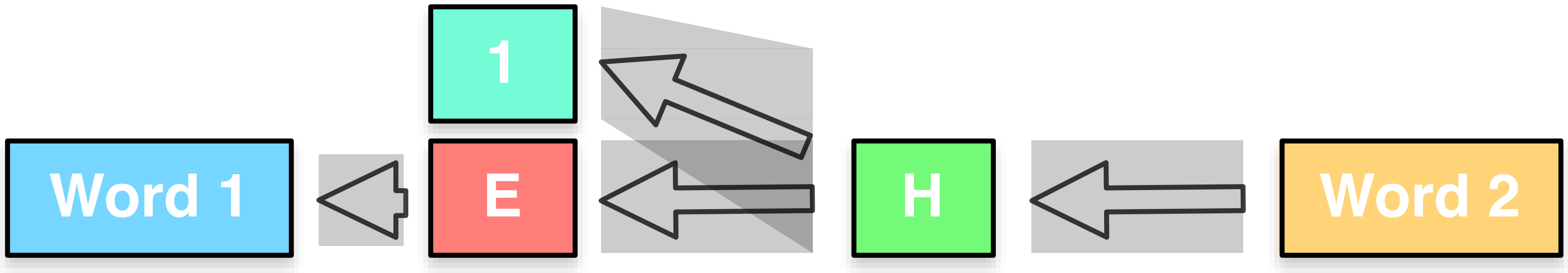
Recurrent Neural Networks



Recurrent Neural Networks

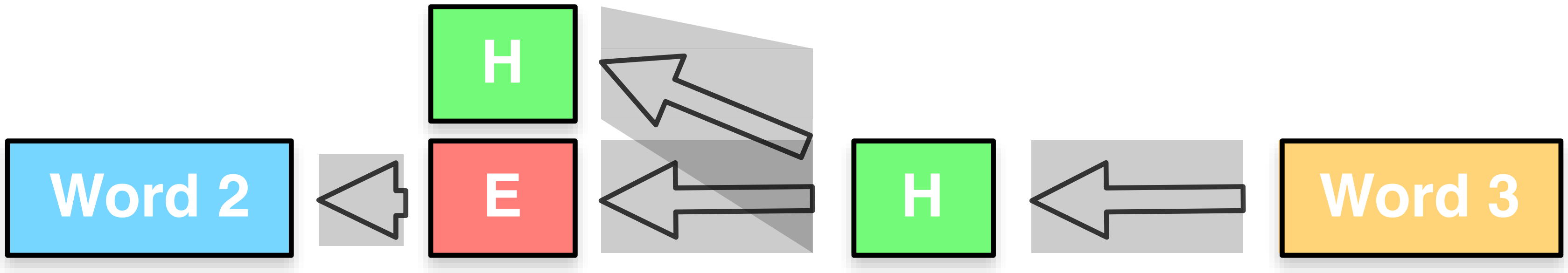


Training



- Process first training example
- Update weights with back-propagation

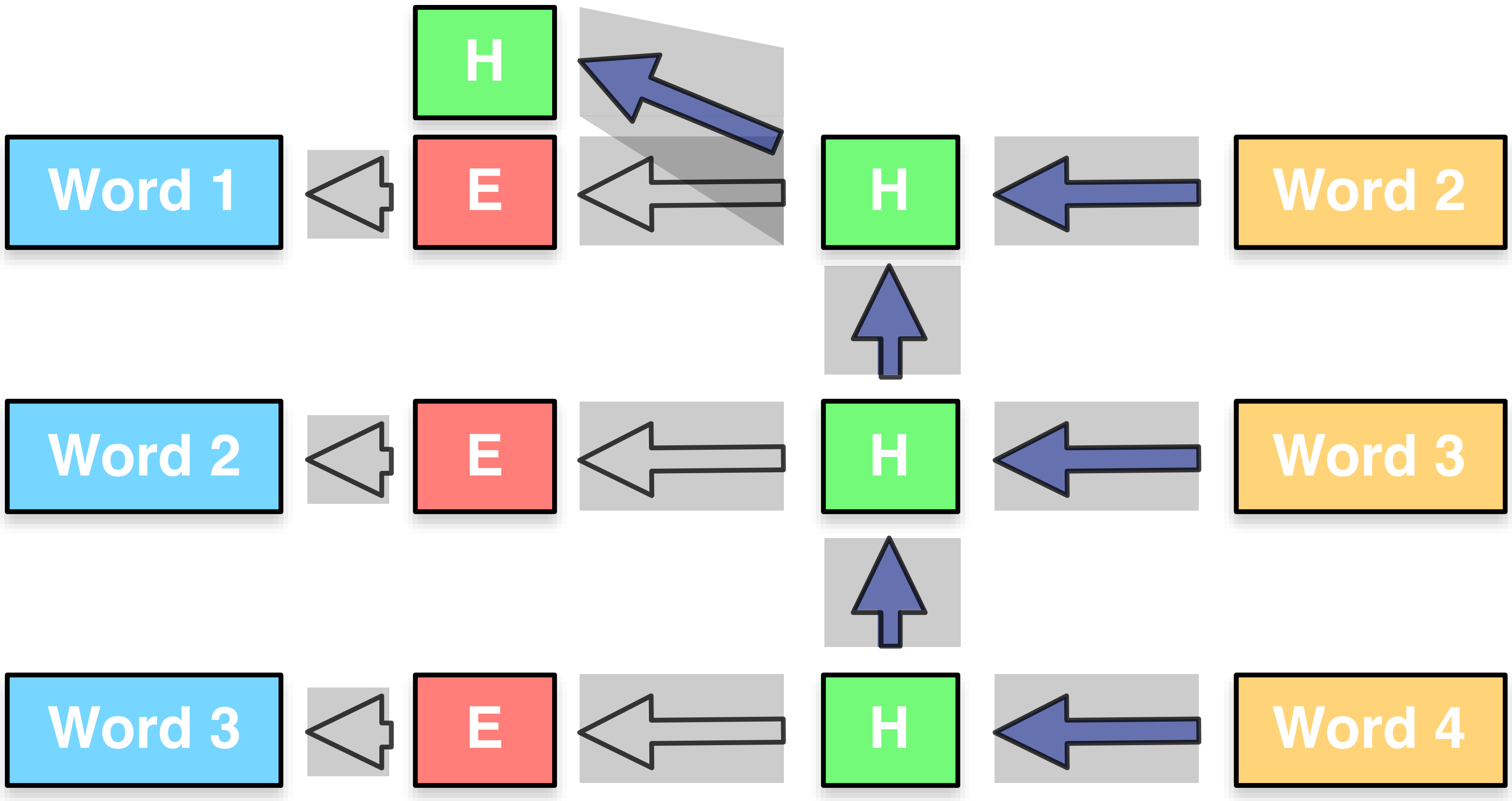
Training



- Process second training example
- Update weights with back-propagation
- And so on...

- But: no feedback to previous history

Back-Propagation Through Time



- After processing a few training examples,
  update through the unfolded recurrent neural network

Back-Propagation Through Time

- Carry out back-propagation though time (BPTT) after each training example
  - 5 time steps seems to be sufficient
  - network learns to store information for more than 5 time steps

long short term memory

Vanishing Gradients

- Error is propagated to previous steps
- Updates consider
  - prediction at that time step
  - impact on future time steps
- Vanishing gradient: propagated error disappears

Recent vs. Early History

- Hidden layer plays double duty
  - memory of the network
  - continuous space representation used to predict output words

- Sometimes only recent context important

  After much economic progress over the years, the **country** $\rightarrow$ *has*

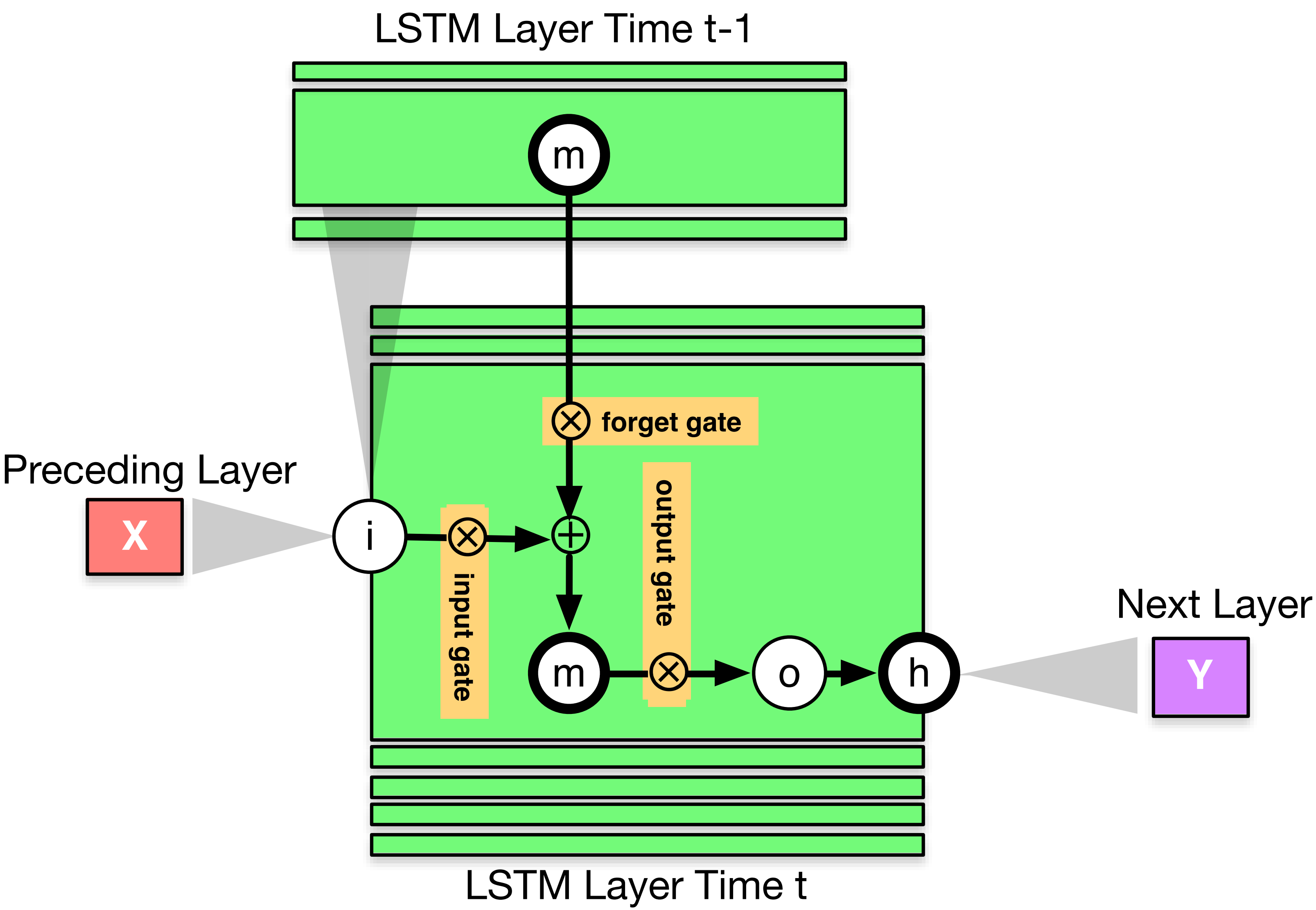
- Sometimes much earlier context important

  The **country** *which has made much economic progress over the years still* $\rightarrow$ *has*

Long Short Term Memory (LSTM)

- Design quite elaborate, although not very complicated to use
- Basic building block: **LSTM cell**
  - similar to a node in a hidden layer
  - but: has a explicit memory state
- Output and memory state change depends on gates
  - **input gate**: how much new input changes memory state
  - **forget gate**: how much of prior memory state is retained
  - **output gate**: how strongly memory state is passed on to next layer.
- Gates can be not just be open (1) and closed (0), but slightly ajar (e.g., 0.2)

LSTM Cell


LSTM Layer Time t-1
m
Preceding Layer
X
i
input gate
forget gate
output gate
m
o
h
Next Layer
Y
LSTM Layer Time t

LSTM Cell (Math)

- Memory and output values at time step t

$$\text{memory}^t = \text{gate}_{\text{input}} \times \text{input}^t + \text{gate}_{\text{forget}} \times \text{memory}^{t-1}$$

$$\text{output}^t = \text{gate}_{\text{output}} \times \text{memory}^t$$

- Hidden node value h^t passed on to next layer applies activation function f

$$h^t = f(\text{output}^t)$$

- Input computed as input to recurrent neural network node
  - given node values for prior layer $\vec{x}^t = (x_1^t, ..., x_X^t)$
  - given values for hidden layer from previous time step $\vec{h}^{t-1} = (h_1^{t-1}, ..., h_H^{t-1})$
  - input value is combination of matrix multiplication with weights w^x and w^h and activation function g

$$\text{input}^t = g\left(\sum_{i=1}^{X} w_i^x x_i^t + \sum_{i=1}^{H} w_i^h h_i^{t-1}\right)$$

Values for Gates

- Gates are very important
- How do we compute their value?

  $\rightarrow$ with a neural network layer!

- For each gate $a \in (\text{input}, \text{forget}, \text{output})$
  - weight matrix W^{xa} to consider node values in previous layer $\vec{x}^t$
  - weight matrix W^{ha} to consider hidden layer $\vec{h}^{t-1}$ at previous time step
  - weight matrix W^{ma} to consider memory at previous time step $\vec{\text{memory}}^{t-1}$
  - activation function h

$$\text{gate}_a = h\left(\sum_{i=1}^{X} w_i^{xa} x_i^t + \sum_{i=1}^{H} w_i^{ha} h_i^{t-1} + \sum_{i=1}^{H} w_i^{ma} \text{memory}_i^{t-1}\right)$$

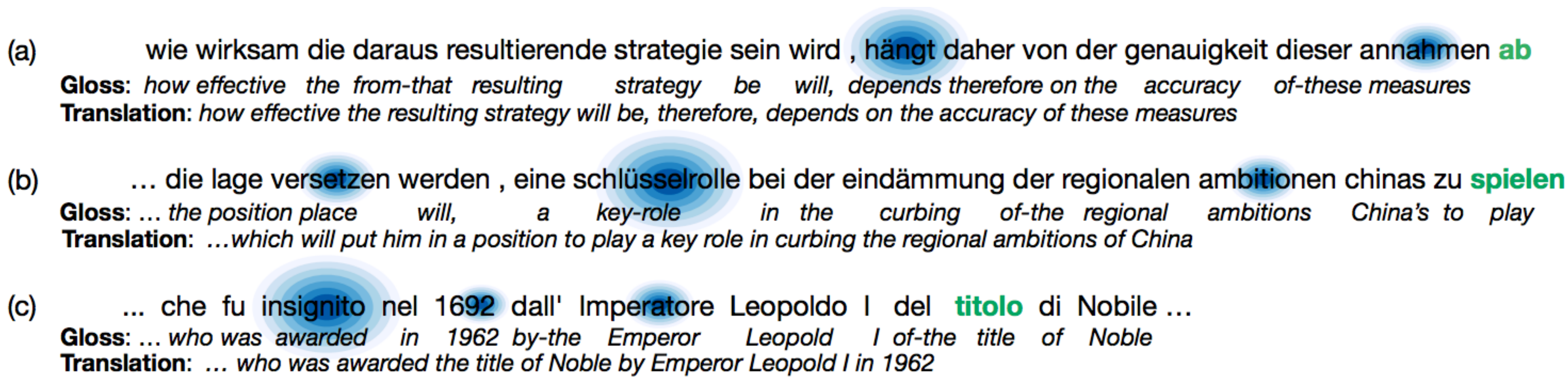
Training

- LSTM are trained the same way as recurrent neural networks
- Back-propagation through time
- This looks all very complex, but:
  - all the operations are still based on
    - matrix multiplications
    - differentiable activation functions

$\rightarrow$ we can compute gradients for objective function with respect to all parameters

$\rightarrow$ we can compute update functions

What is the Point?

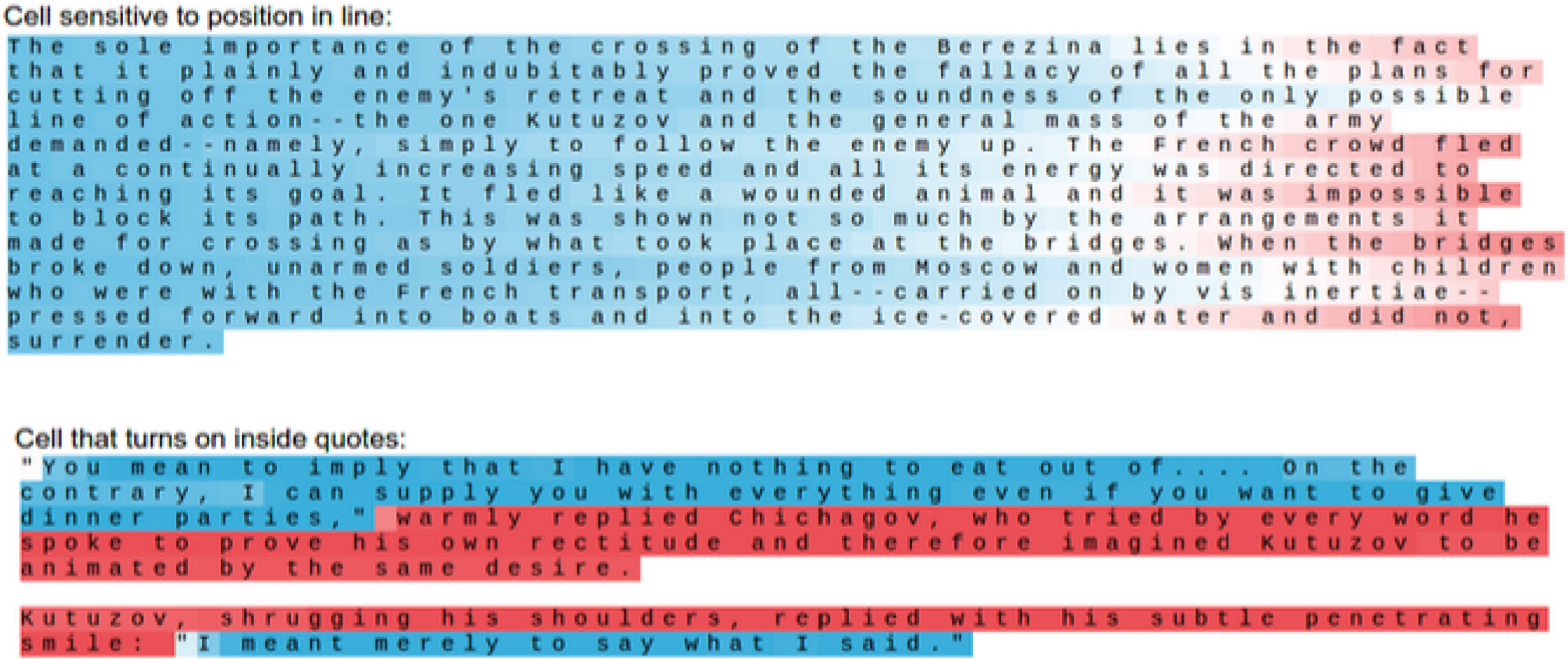


(from Tran, Bisazza, Monz, 2016)

- Each node has memory memory_i independent from current output h_i
- Memory may be carried through unchanged ($\text{gate}^i_{\text{input}} = 0$, $\text{gate}^i_{\text{memory}} = 1$)

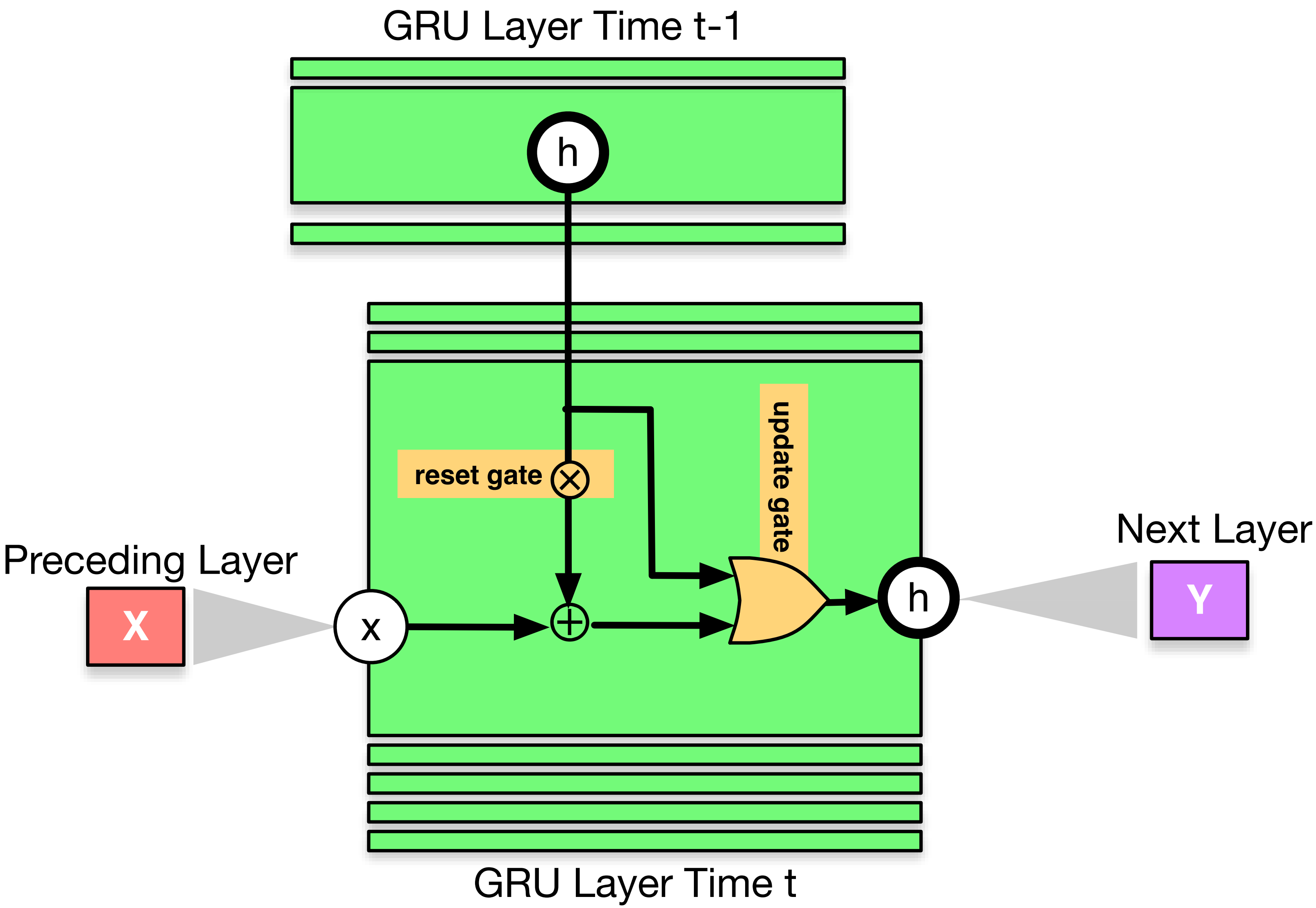
⇒ can remember important features over long time span
(capture long distance dependencies)

Visualizing Individual Cells



Karpathy et al. (2015): "Visualizing and Understanding Recurrent Networks"

Gated Recurrent Unit (GRU)



Gated Recurrent Unit (Math)

- Two Gates

$$\text{update}_t = g(W_{\text{update}}\ \text{input}_t + U_{\text{update}}\ \text{state}_{t-1} + \text{bias}_{\text{update}})$$

$$\text{reset}_t = g(W_{\text{reset}}\ \ \text{input}_t + U_{\text{reset}}\ \ \ \text{state}_{t-1} + \text{bias}_{\text{reset}})$$

- Combination of input and previous state
(similar to traditional recurrent neural network)

$$\text{combination}_t = f(W\ \text{input}_t + U(\text{reset}_t \circ \text{state}_{t-1}))$$

- Interpolation with previous state

$$\begin{aligned} \text{state}_t =& (1 - \text{update}_t) \circ \text{state}_{t-1} + \\ & \text{update}_t \qquad \circ \text{combination}_t) + \text{bias} \end{aligned}$$

Language Models: Quick Summary

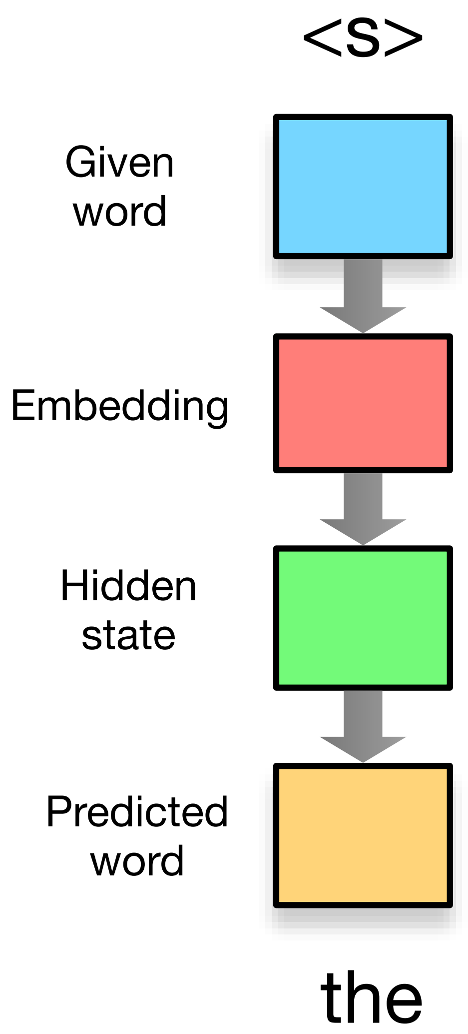
- Modeling variants
  - feed-forward neural network
  - recurrent neural network (LSTM and GRU cells)
- Next: Language modeling on target, but include source information!

Questions?

Outline

- Machine Translation: History & Problem Formulation
- Language Model
- Encoder-Decoder NMT Model
- Training & Inference
- Alternative NMT Models

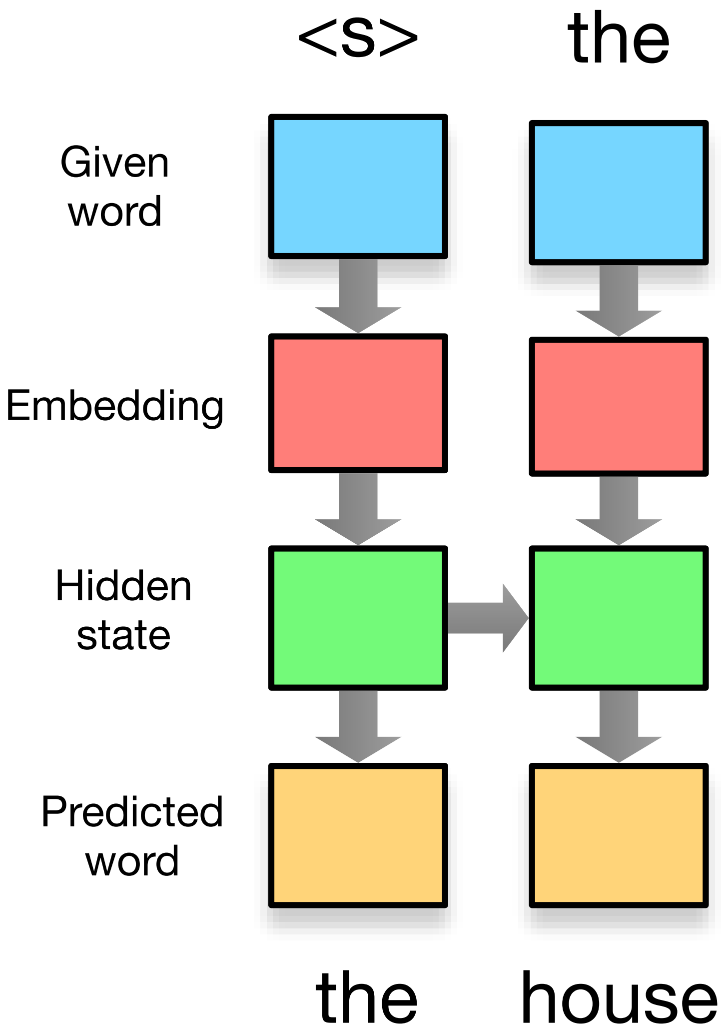
Recurrent Neural Language Model (Again)



Predict
the first word
of a sentence

Same as before,
just drawn top-down

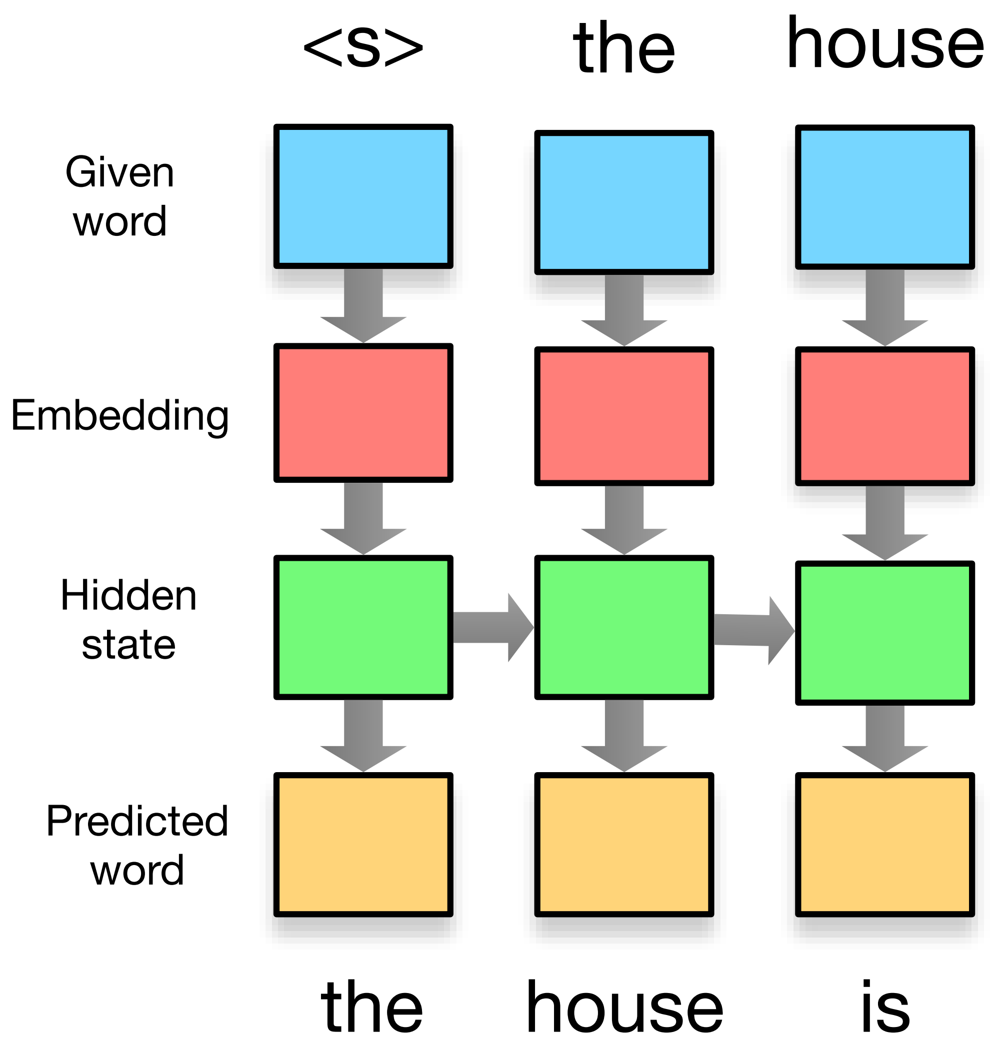
Recurrent Neural Language Model (Again)



Predict
the second word
of a sentence

Re-use hidden state
from
first word prediction

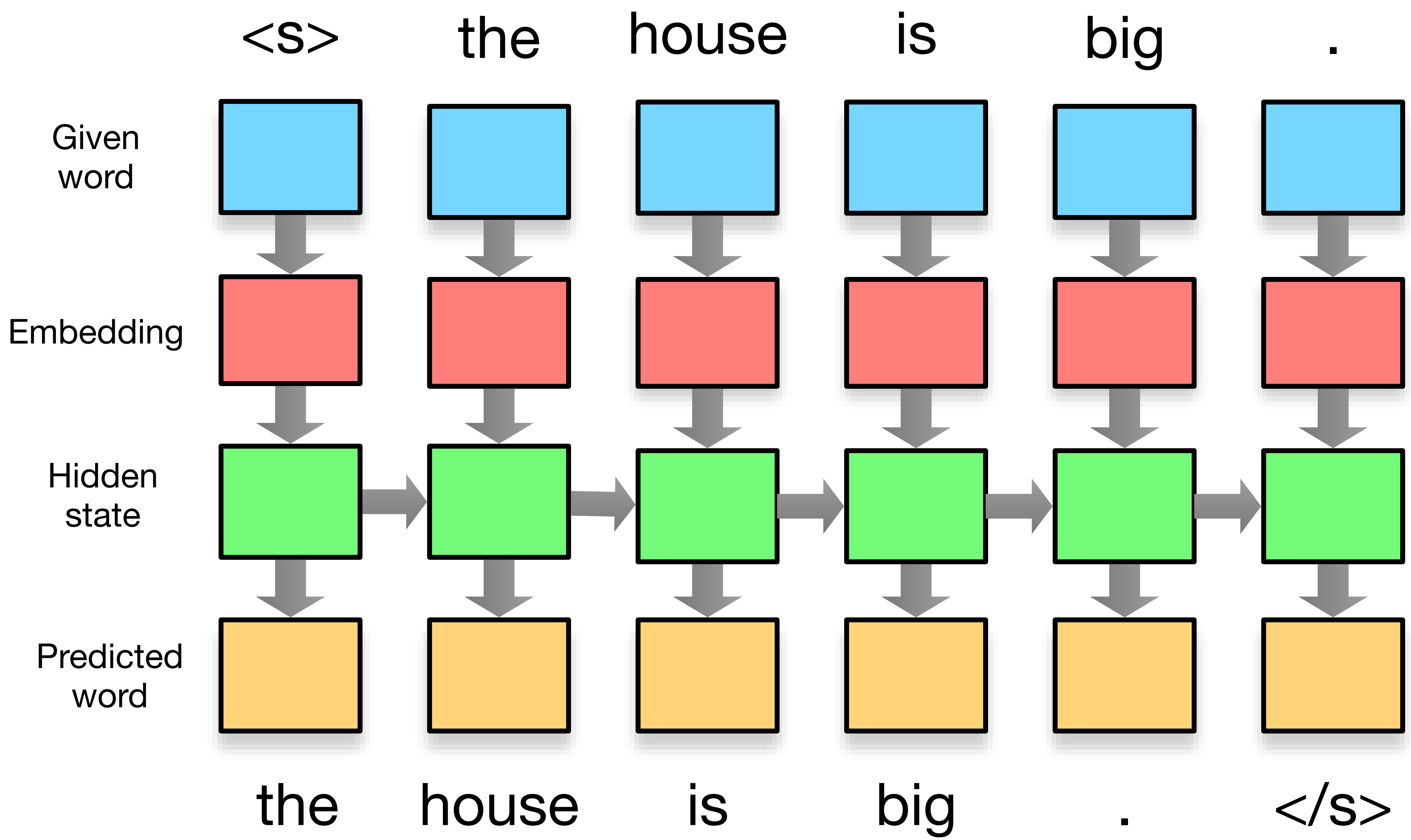
Recurrent Neural Language Model



Predict
the third word
of a sentence

... and so on

Recurrent Neural Language Model

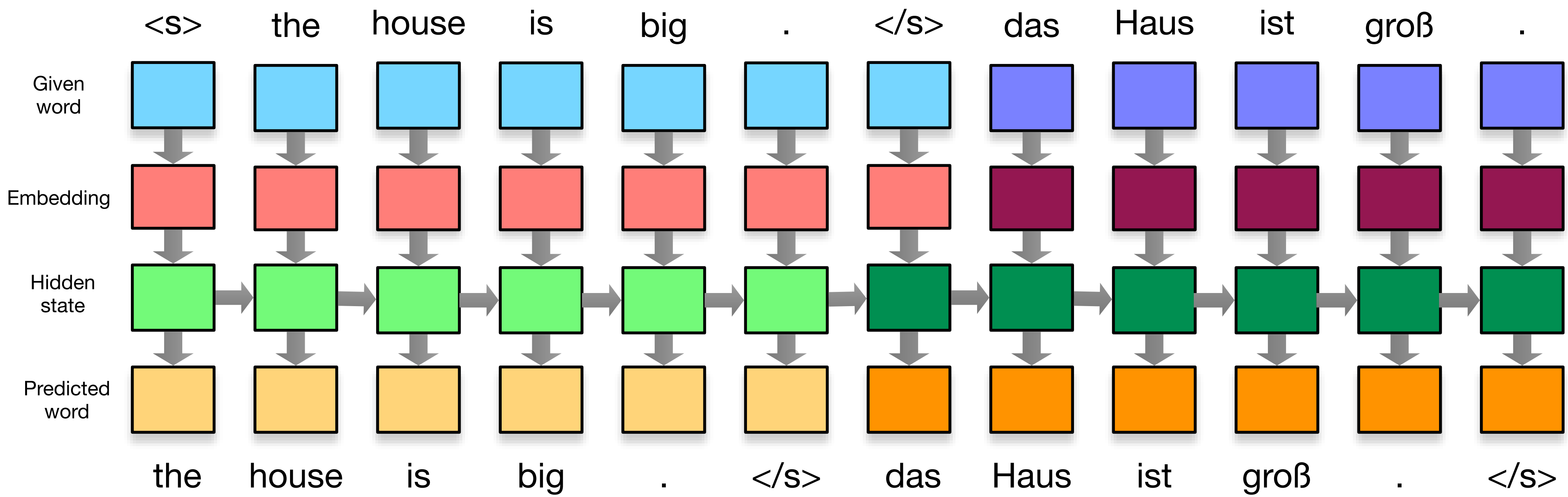


Recurrent Neural Translation Model

- We predicted the words of a sentence

- Why not also predict their translations?

Encoder-Decoder Model



- Obviously madness
- Proposed by Google (Sutskever et al. 2014)

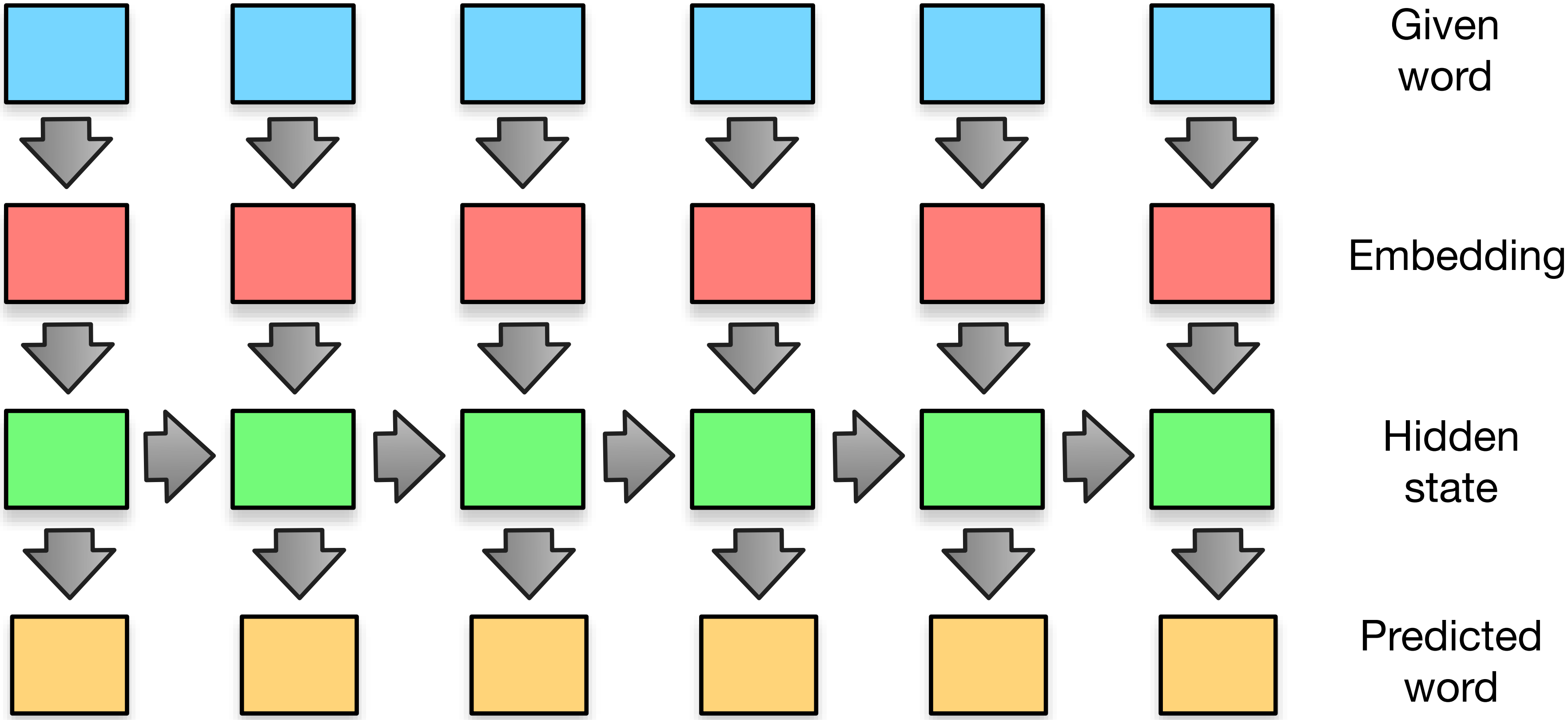
What is missing?

- Alignment of input words to output words

$\Rightarrow$ Solution: attention mechanism

neural translation model with attention

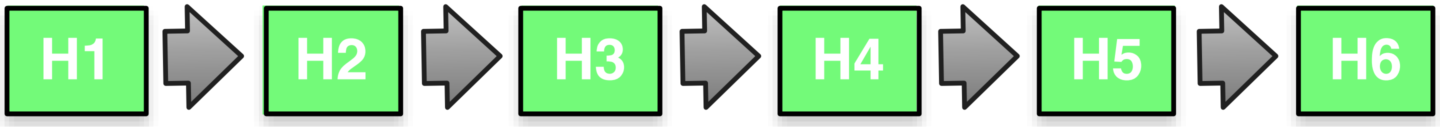
Input Encoding



- Inspiration: recurrent neural network language model on the input side

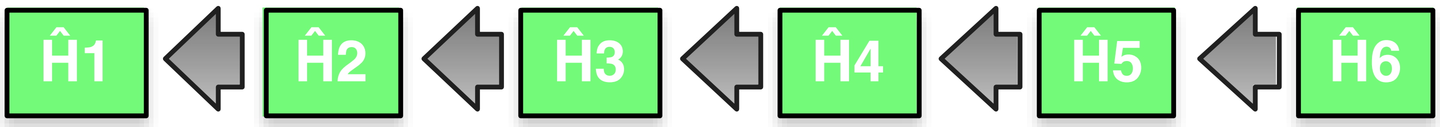
Hidden Language Model States

- This gives us the hidden states

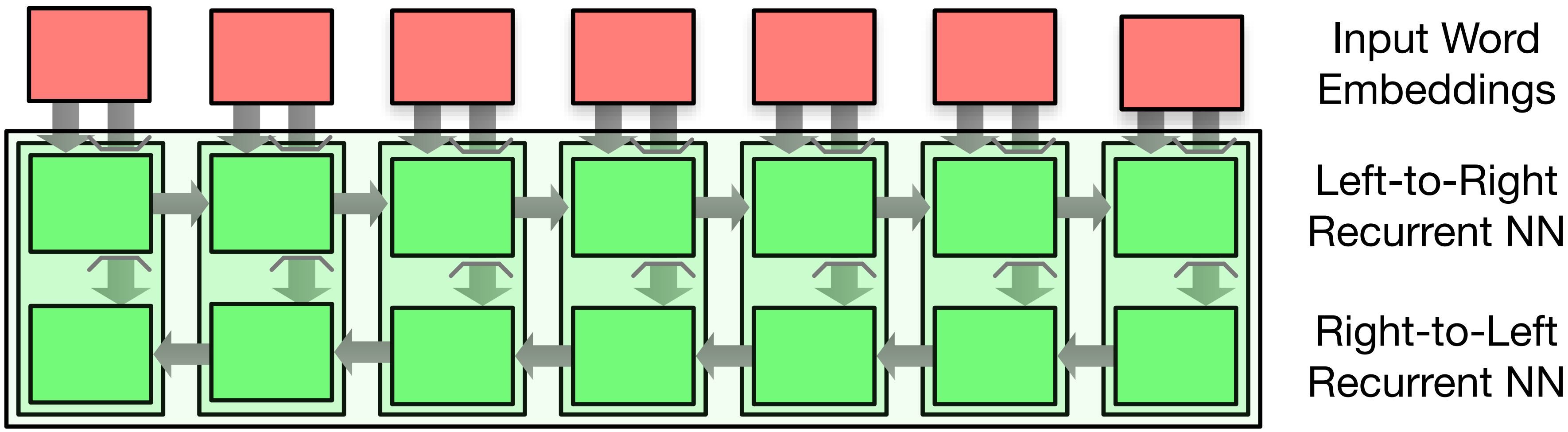


- These encode left context for each word

- Same process in reverse: right context for each word

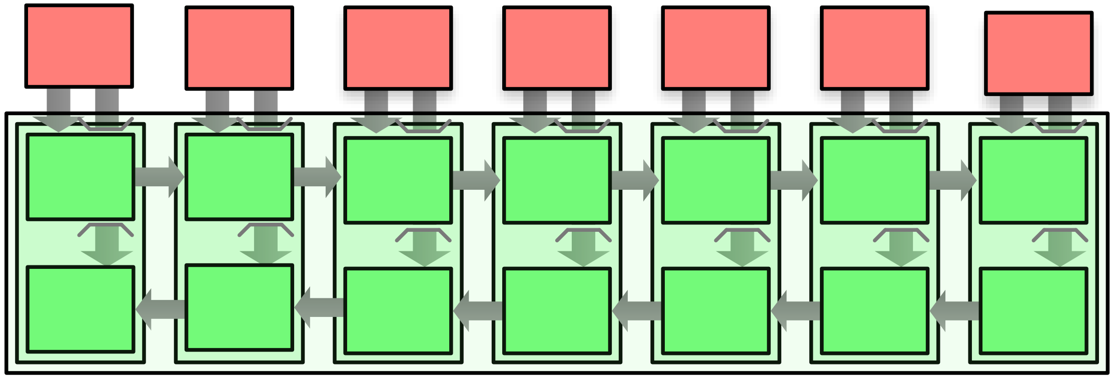


Input Encoder



- Input encoder: concatenate bidirectional RNN states
- Each word representation includes full left and right sentence context

Encoder: Math



- Input is sequence of words x_j, mapped into embedding space $\bar{E}\ x_j$

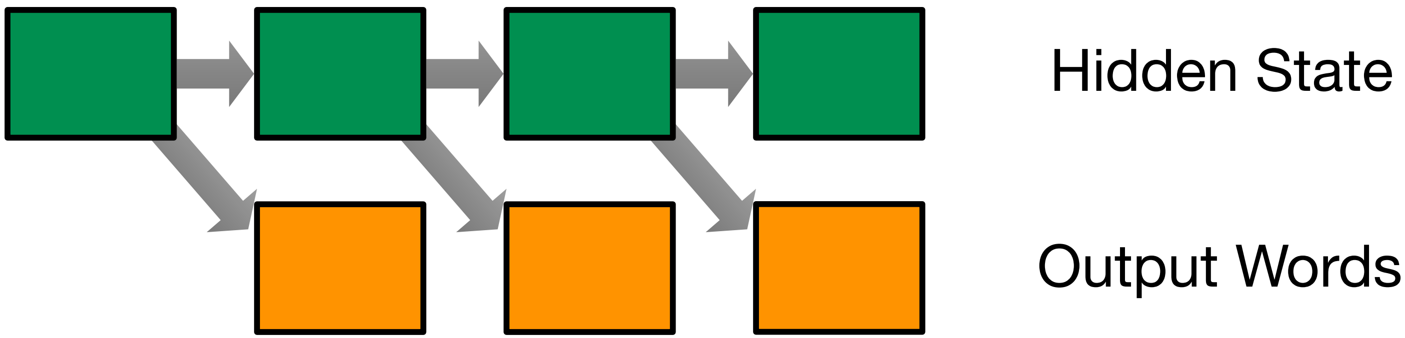
- Bidirectional recurrent neural networks

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

- Various choices for the function $f()$: feed-forward layer, GRU, LSTM, ...

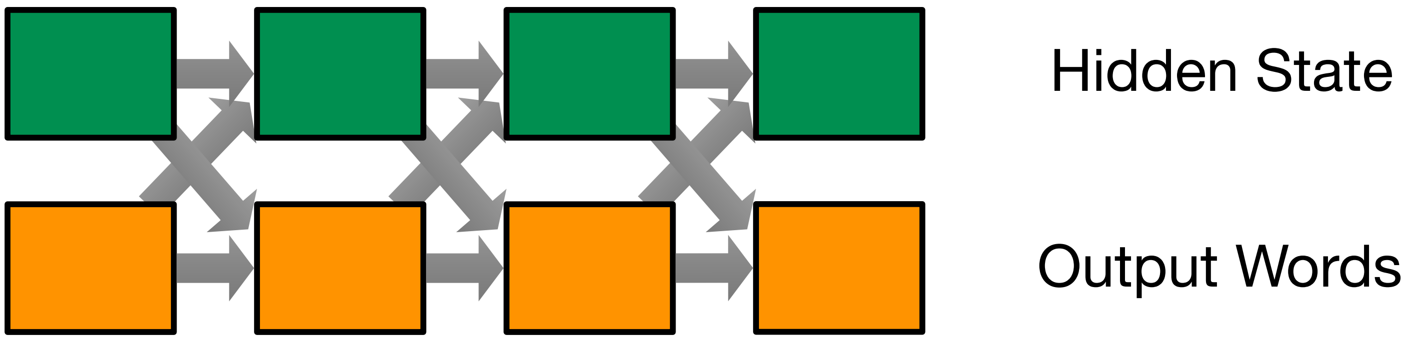
Decoder

- We want to have a recurrent neural network predicting output words



Decoder

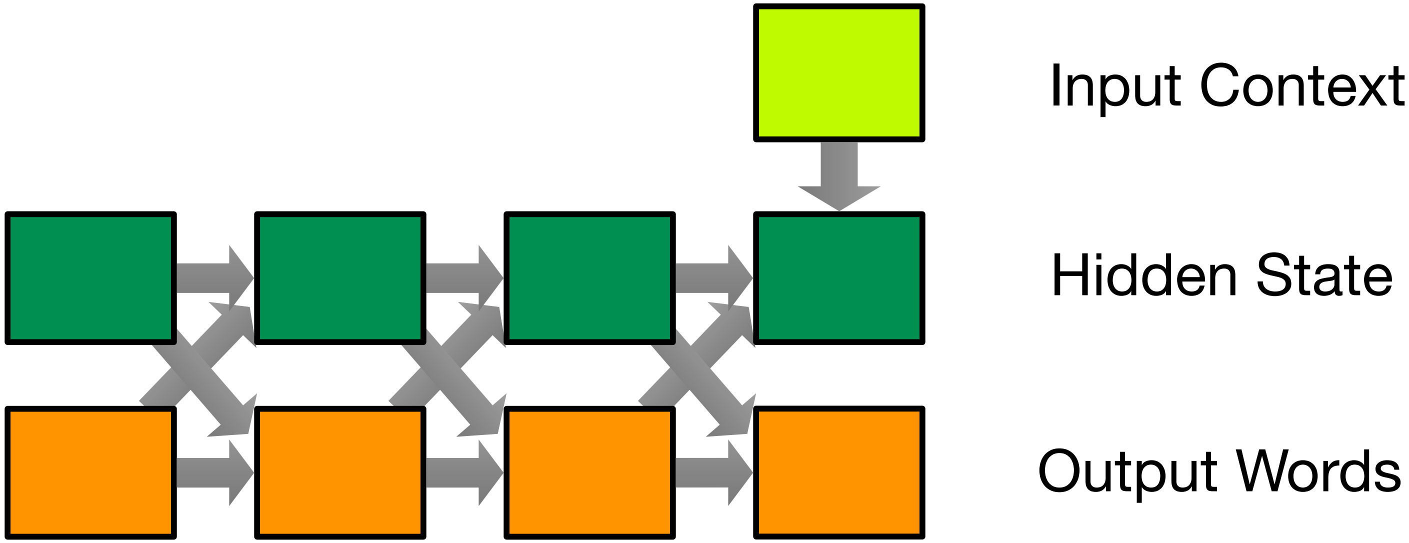
- We want to have a recurrent neural network predicting output words



- We feed decisions on output words back into the decoder state

Decoder

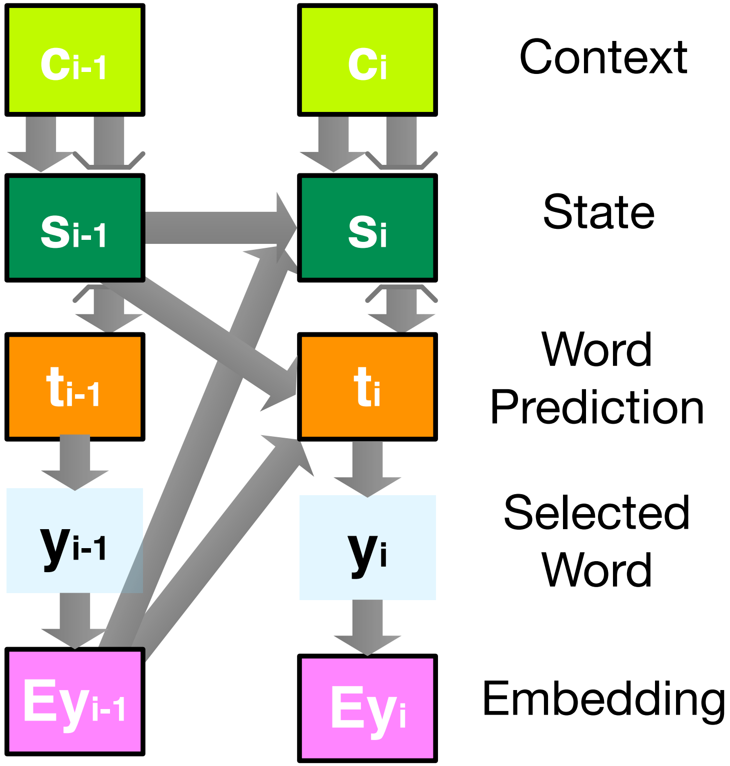
- We want to have a recurrent neural network predicting output words



- We feed decisions on output words back into the decoder state

- Decoder state is also informed by the input context

More Detail



- Decoder is also recurrent neural network over sequence of hidden states s_i

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

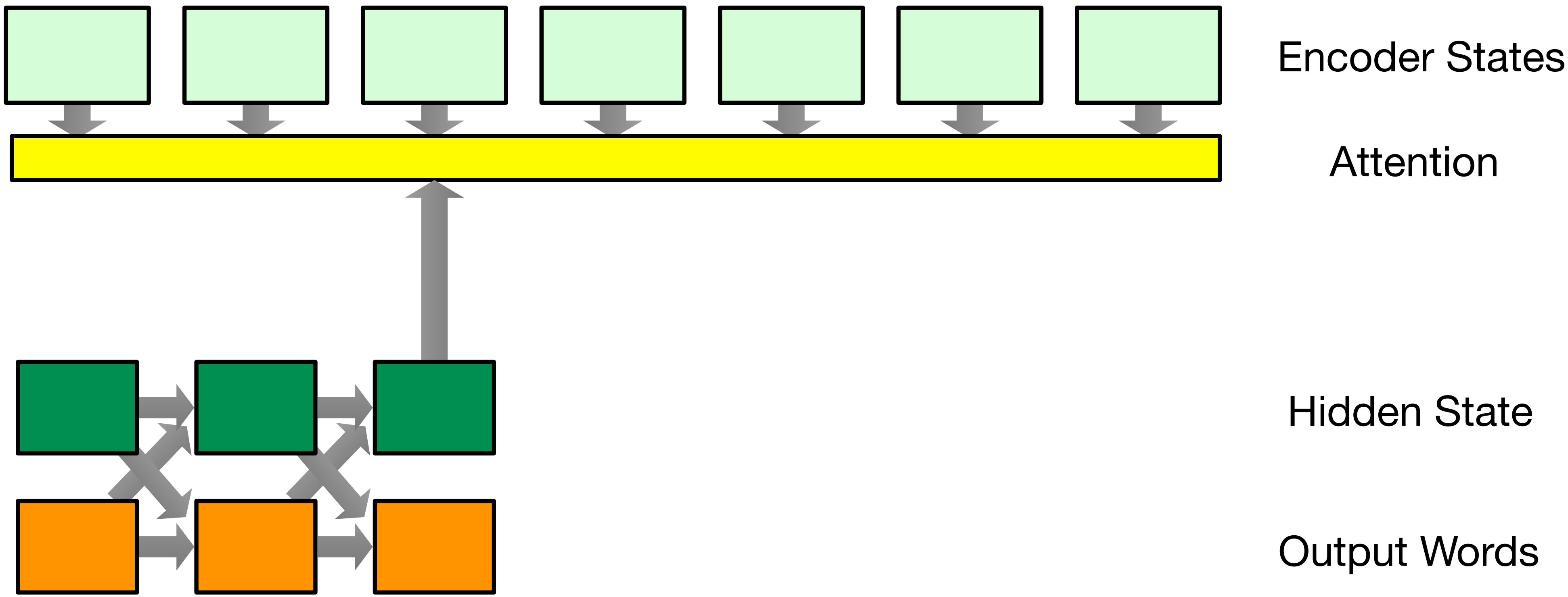
- Again, various choices for the function $f()$: feed-forward layer, GRU, LSTM, ...
- Output word y_i is selected by computing a vector t_i (same size as vocabulary)

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

  then finding the highest value in vector t_i

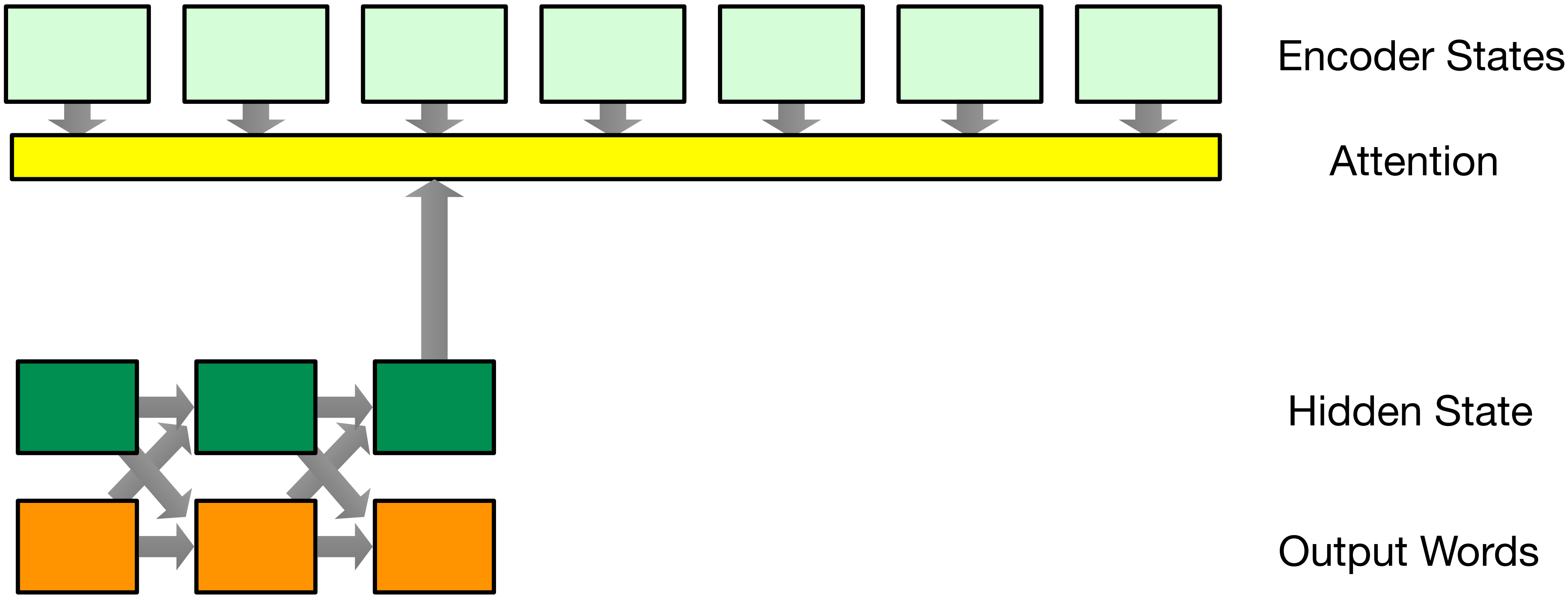
- If we normalize t_i, we can view it as a probability distribution over words
- Ey_i is the embedding of the output word y_i

Attention



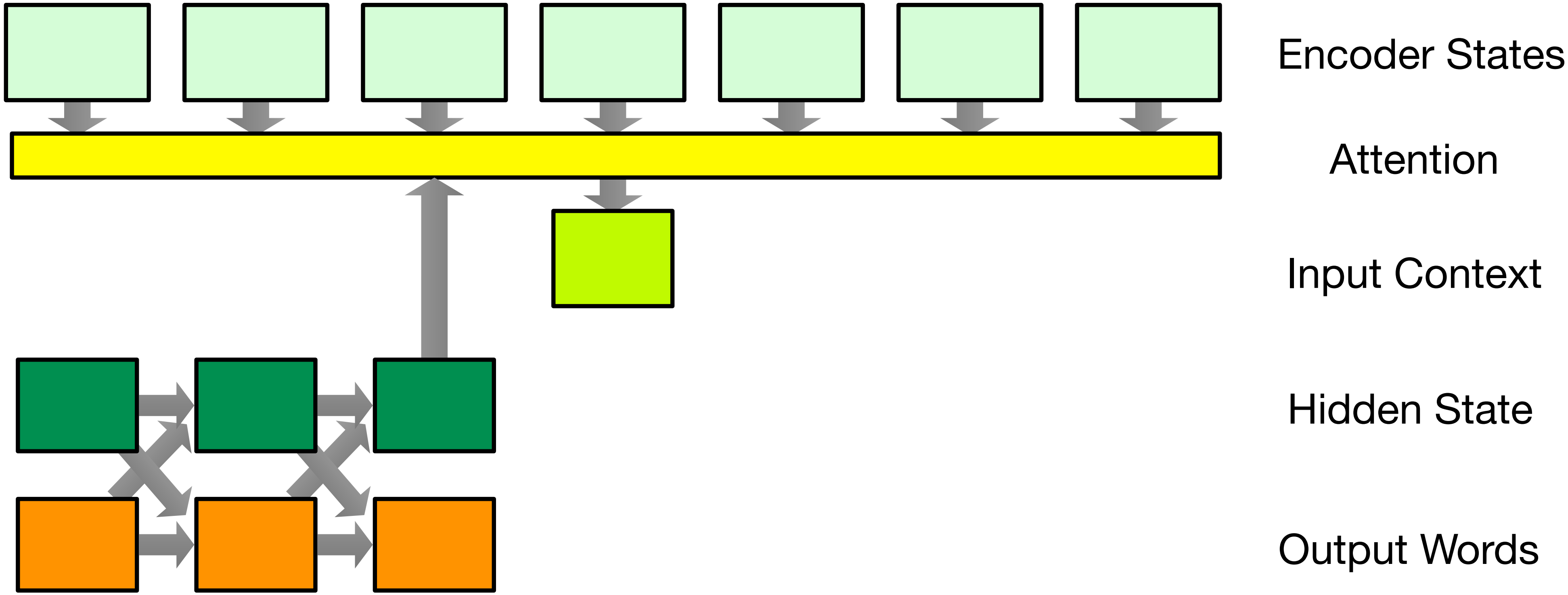
- Given what we have generated so far (decoder hidden state)
- ... which words in the input should we pay attention to (encoder states)?

Attention



- Given: – the previous hidden state of the decoder s_{i-1}
  – the representation of input words $h_j = (\overleftarrow{h_j}, \overrightarrow{h_j})$
- Predict an alignment probability $a(s_{i-1}, h_j)$ to each input word j (modeled with with a feed-forward neural network layer)

Attention

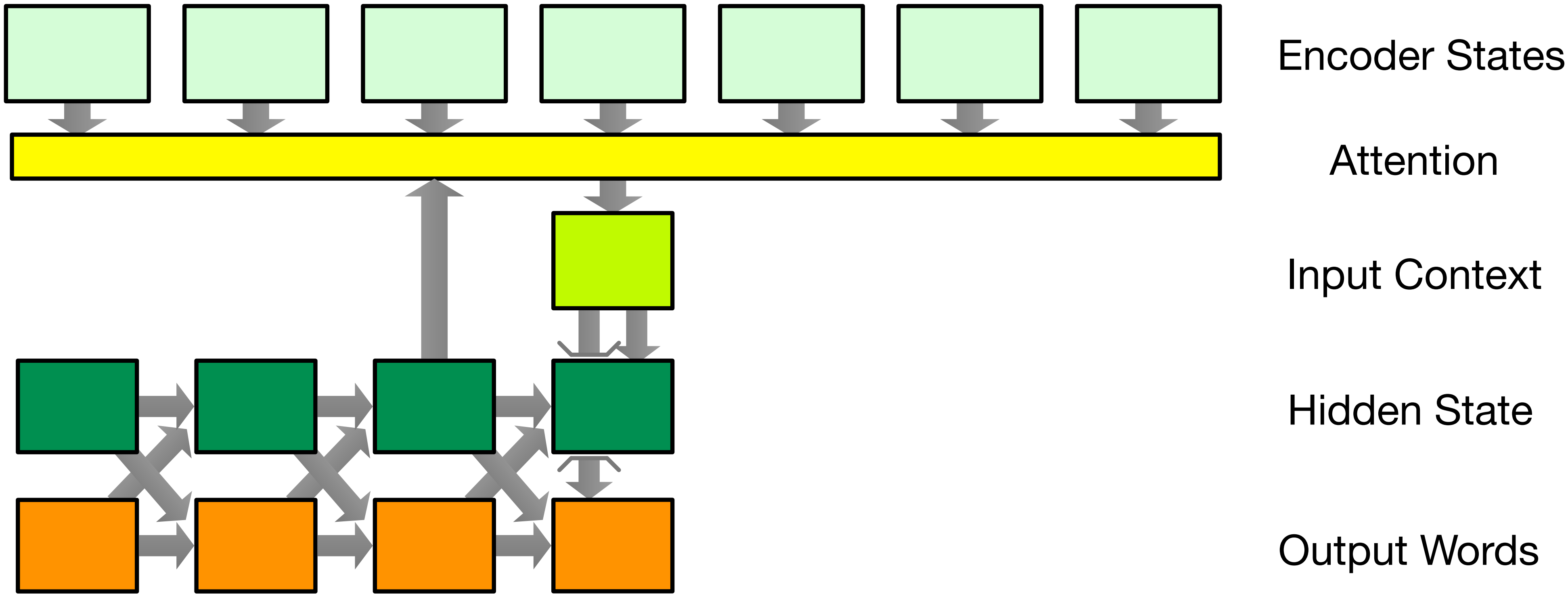


- Normalize attention (softmax)

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

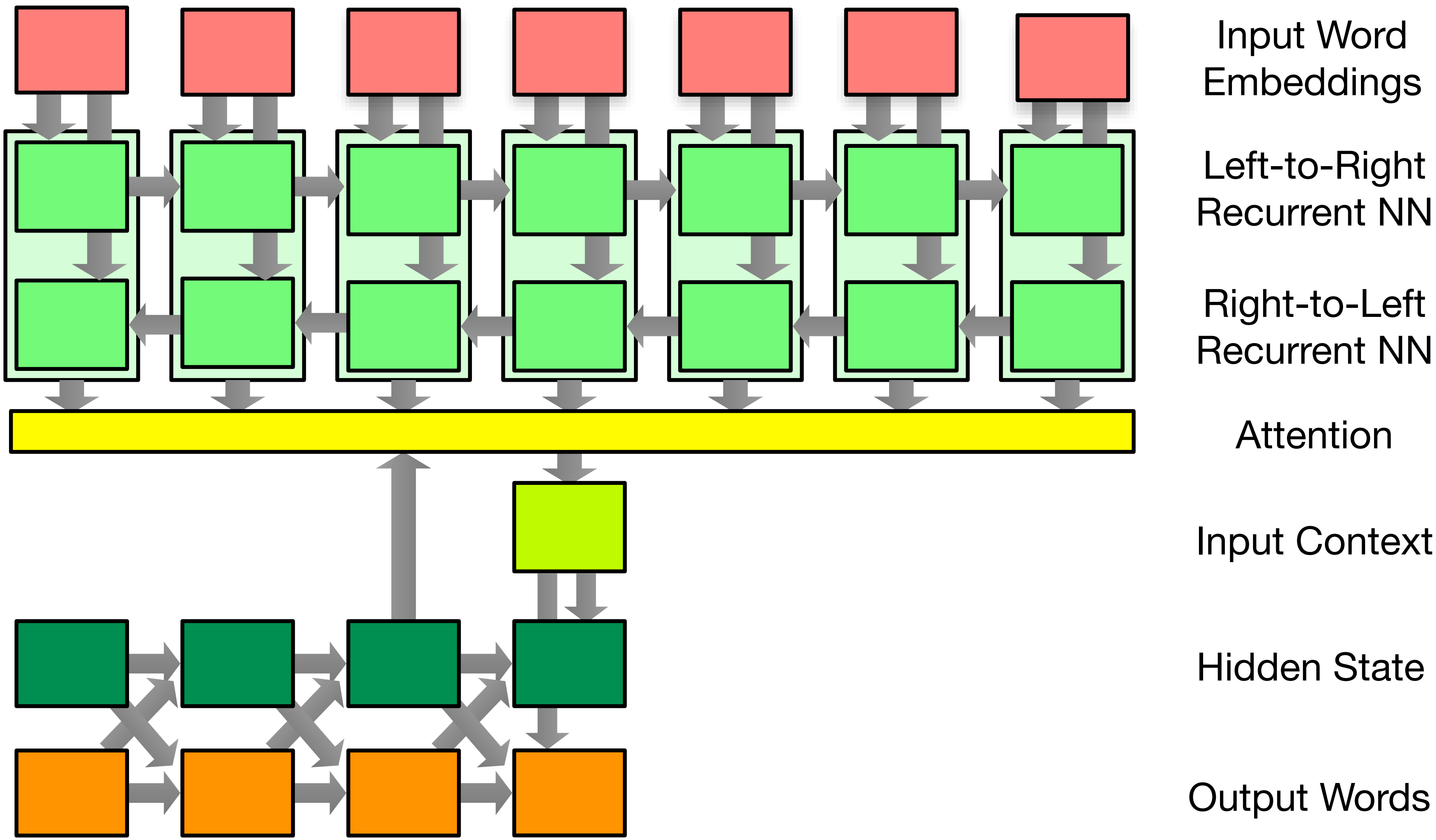
- Relevant input context: weigh input words according to attention: $c_i = \sum_j \alpha_{ij} h_j$

Attention



- Use context to predict next hidden state and output word

Encoder-Decoder with Attention



Questions?

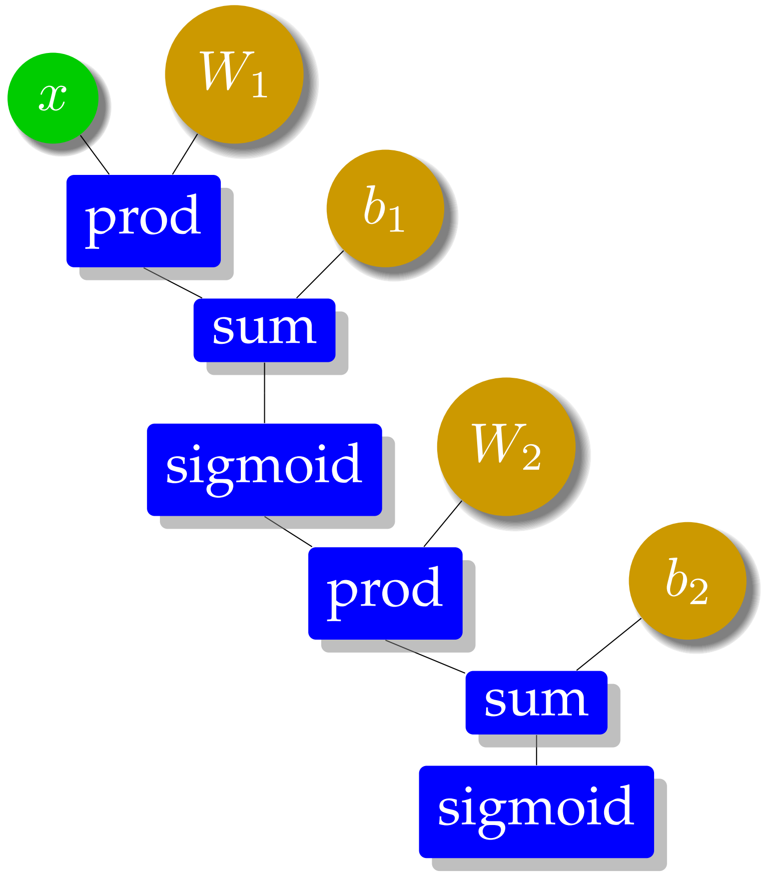
Outline

- Machine Translation: History & Problem Formulation
- Language Model
- Encoder-Decoder NMT Model
- Training & Inference
- Alternative NMT Models

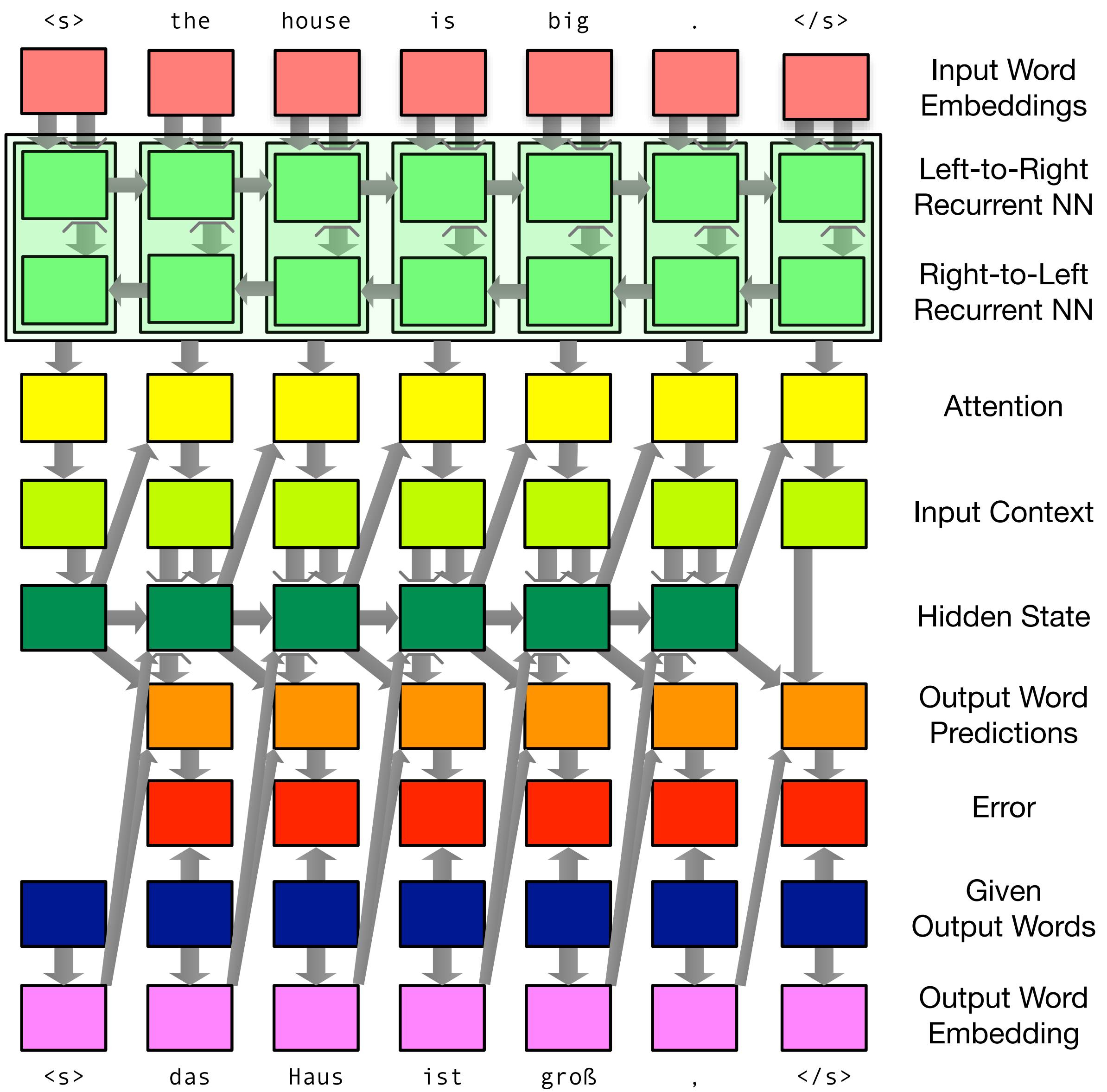
training

Computation Graph

- Math behind neural machine translation defines a computation graph
- Forward and backward computation to compute gradients for model training



Unrolled Computation Graph

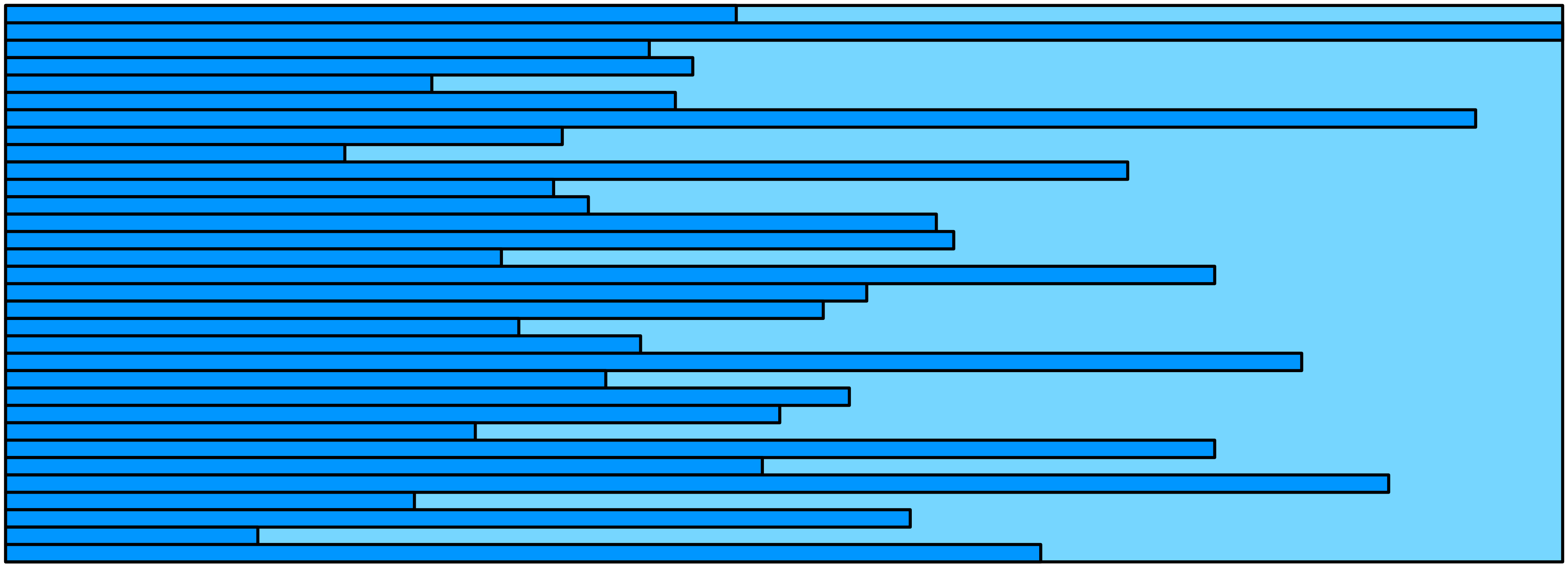


Batching

- Already large degree of parallelism
  - most computations on vectors, matrices
  - efficient implementations for CPU and GPU
- Further parallelism by batching
  - processing several sentence pairs at once
  - scalar operation $\rightarrow$ vector operation
  - vector operation $\rightarrow$ matrix operation
  - matrix operation $\rightarrow$ 3d tensor operation
- Typical batch sizes 50–100 sentence pairs

Batches

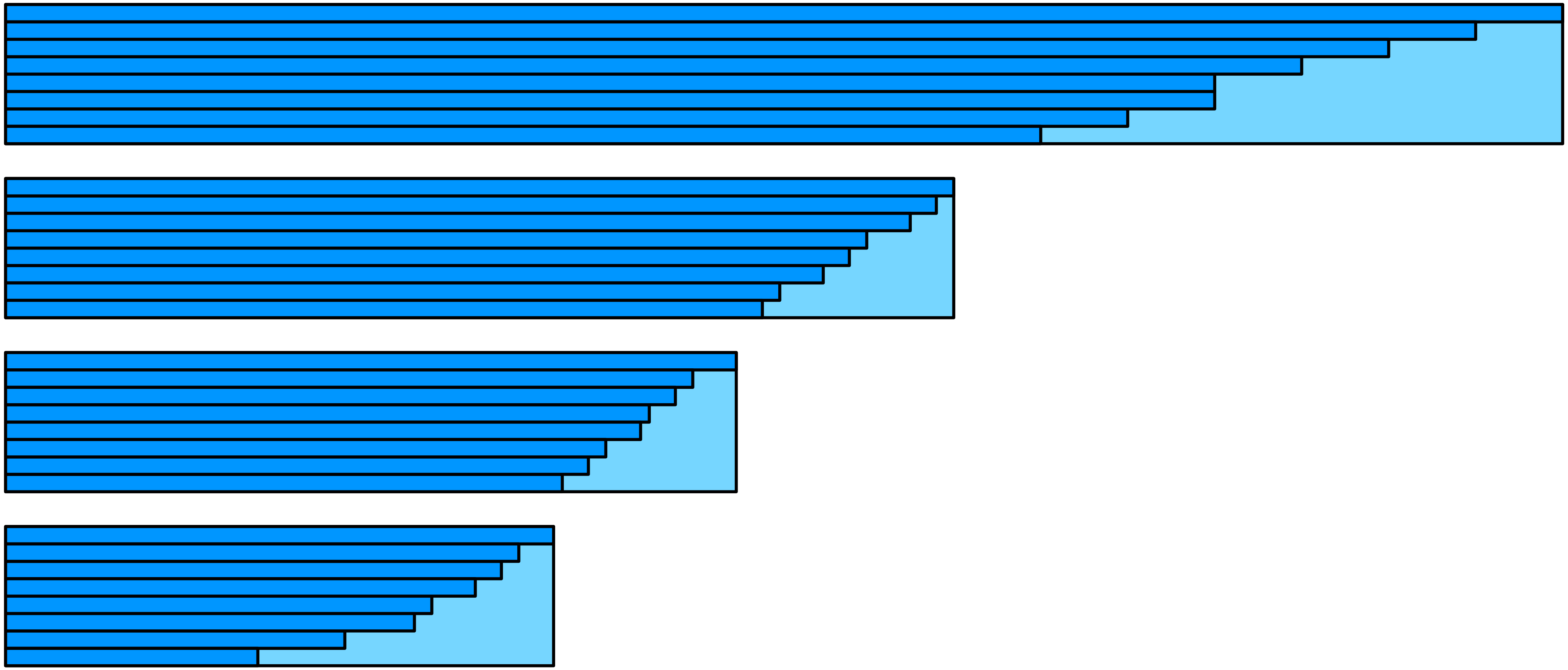
- Sentences have different length
- When batching, fill up unneeded cells in tensors



⇒ A lot of wasted computations

Mini-Batches

- Sort sentences by length, break up into mini-batches



- Example: Maxi-batch 1600 sentence pairs, mini-batch 80 sentence pairs

Overall Organization of Training

- Shuffle corpus
- Break into maxi-batches
- Break up each maxi-batch into mini-batches
- Process mini-batch, update parameters
- Once done, repeat
- Typically 5-15 epochs needed (passes through entire training corpus)

inference

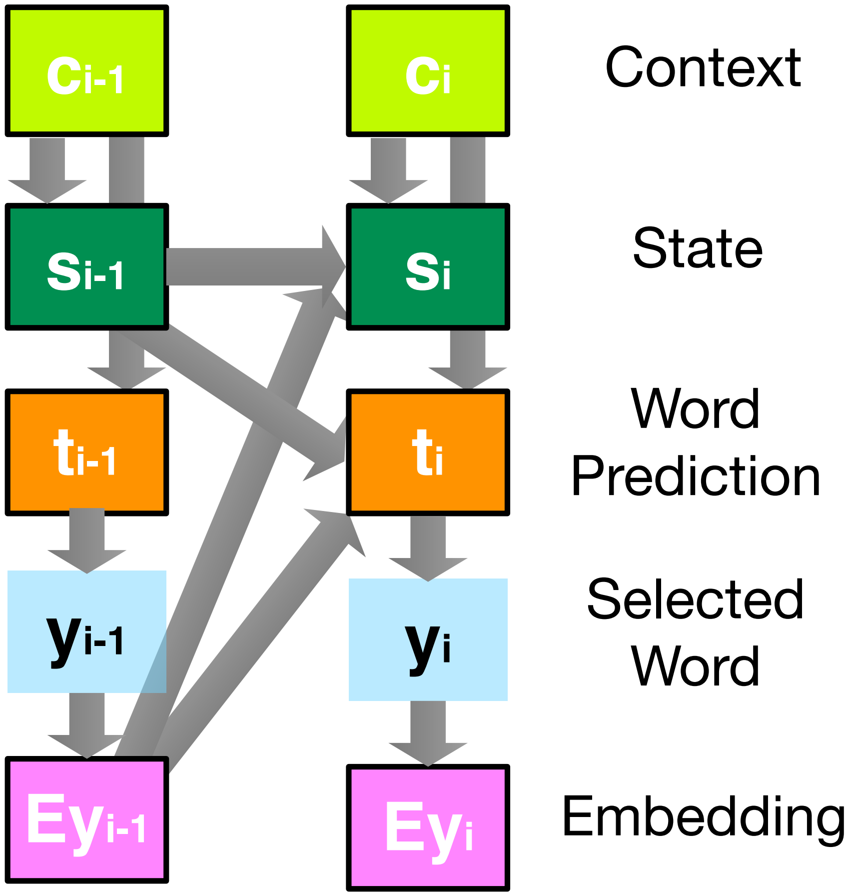
Inference

- Given a trained model

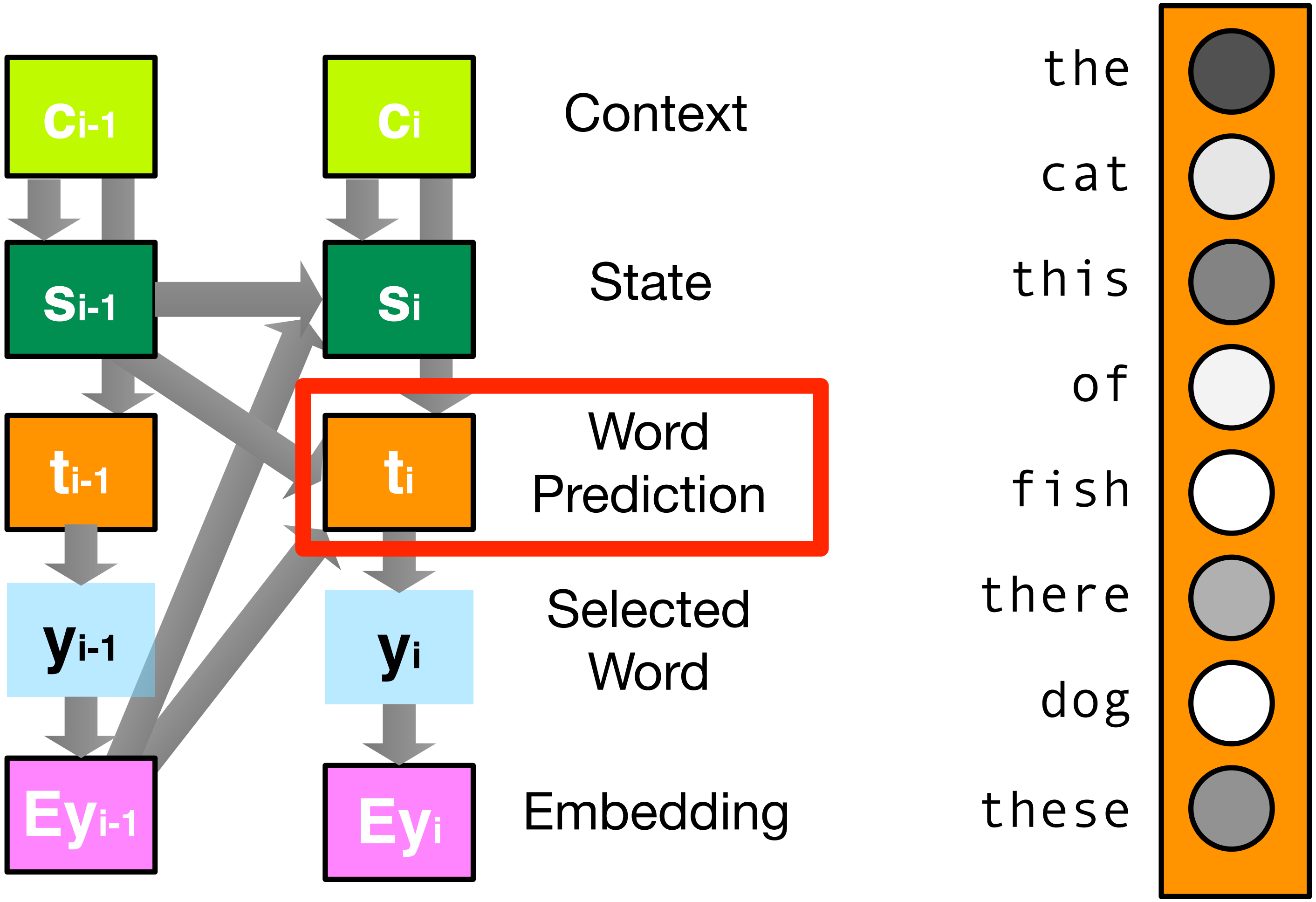
  ... we now want to translate test sentences

- We only need execute the ”forward” step in the computation graph

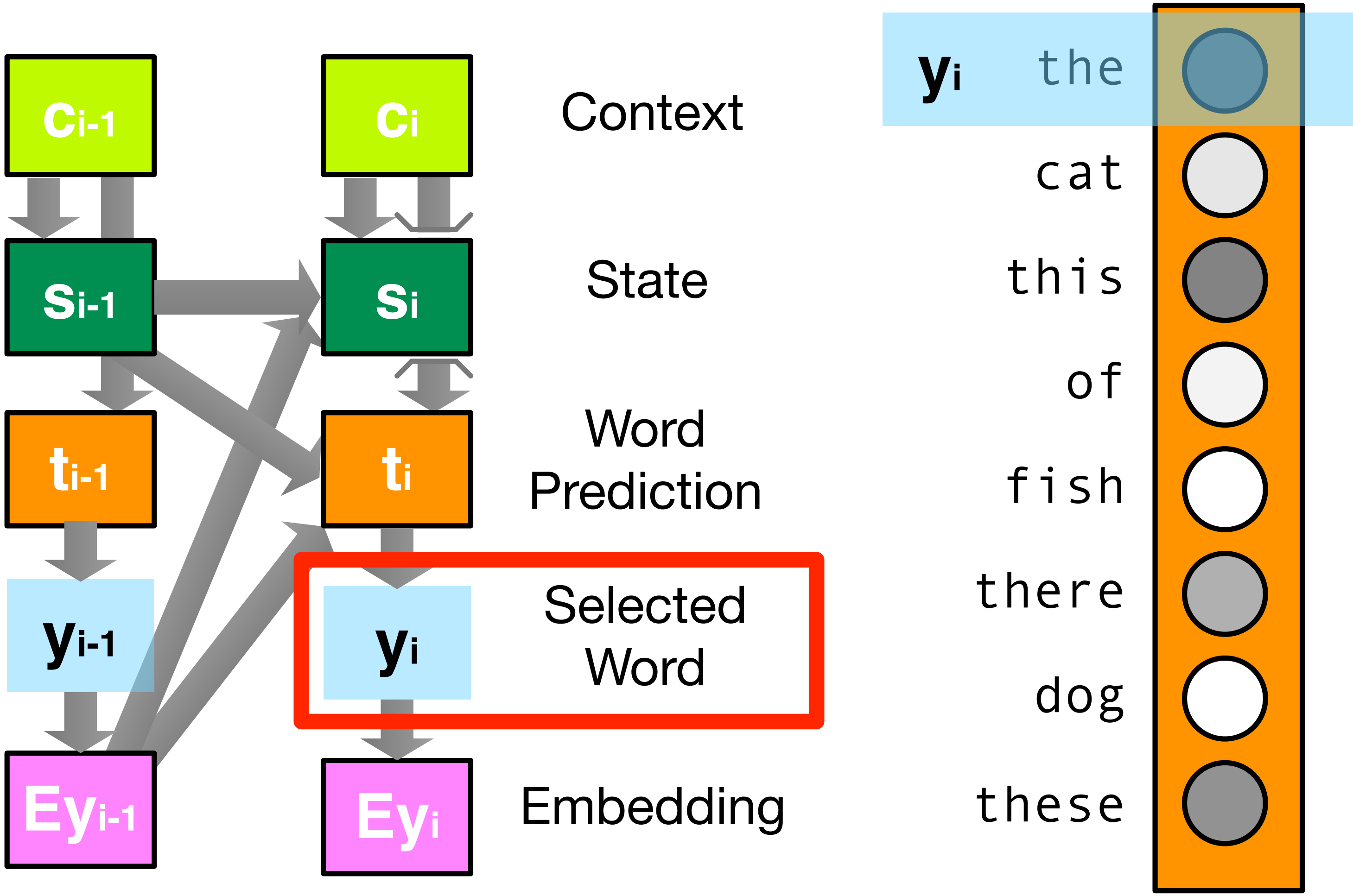
Word Prediction



Selected Word

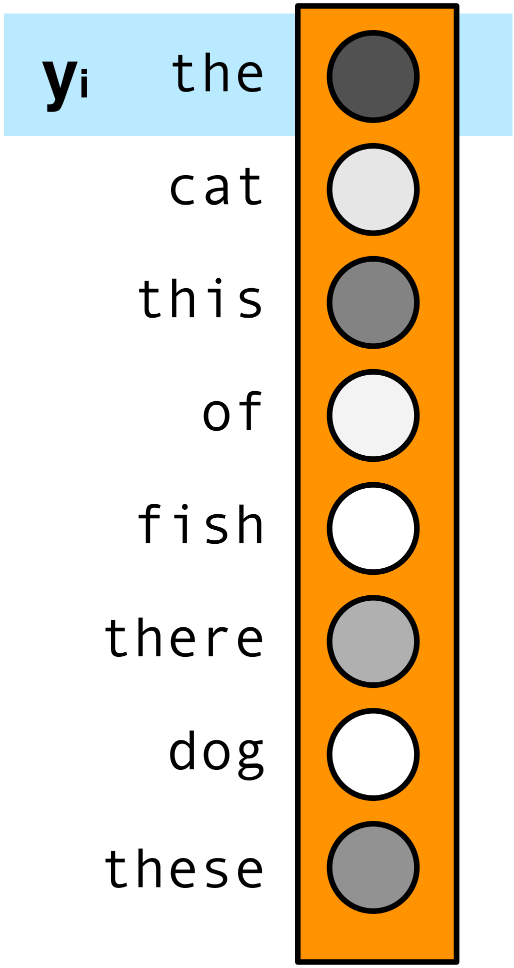


Embedding

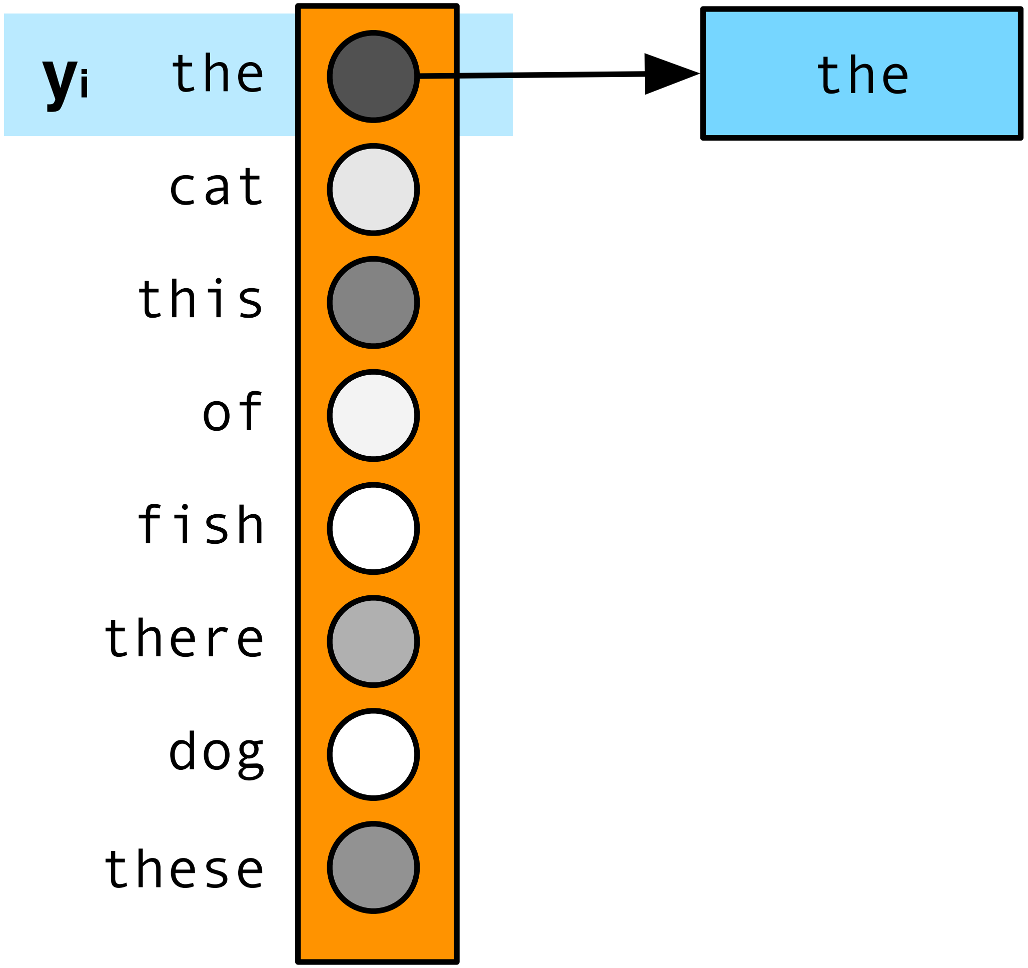


Ci-1
Ci
Context
Si-1
Si
State
ti-1
ti
Word Prediction
yi-1
yi
Selected Word
Eyi-1
Eyi
Embedding
yi
the
cat
this
of
fish
there
dog
these

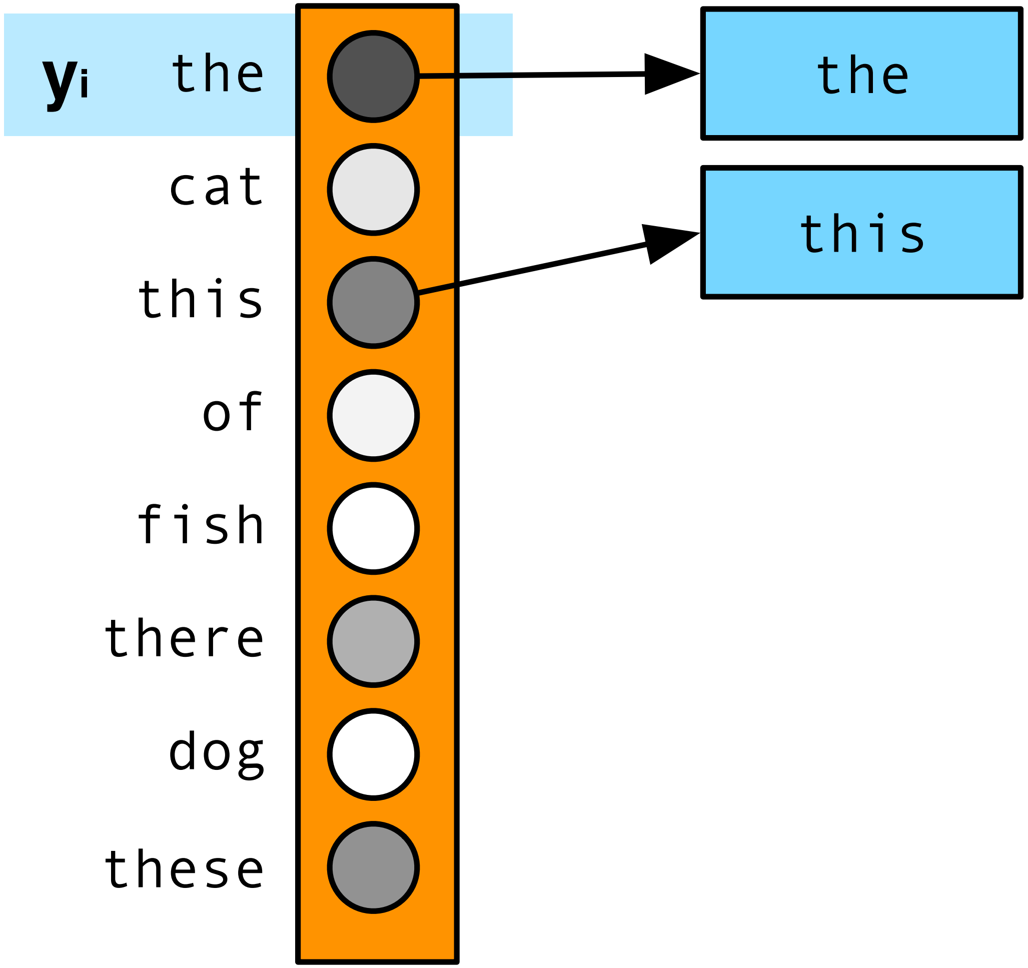
Distribution of Word Predictions



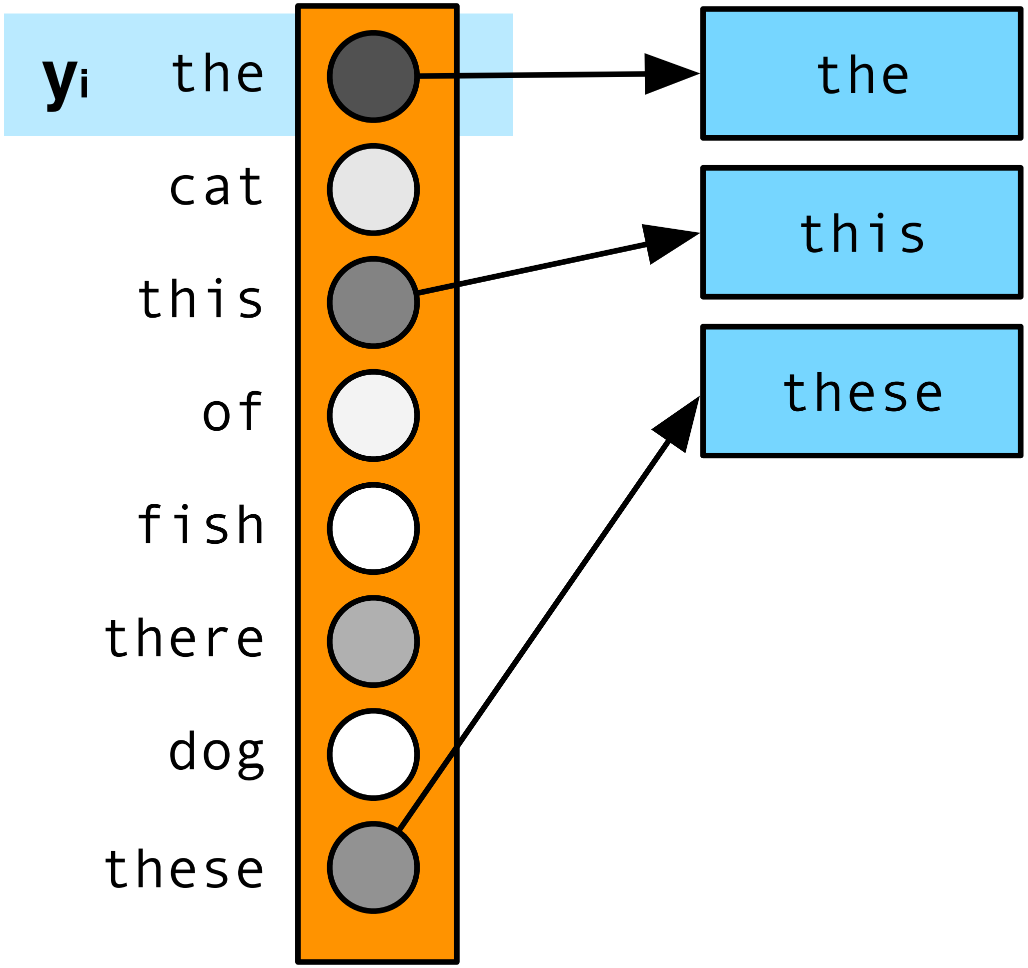
Select Best Word



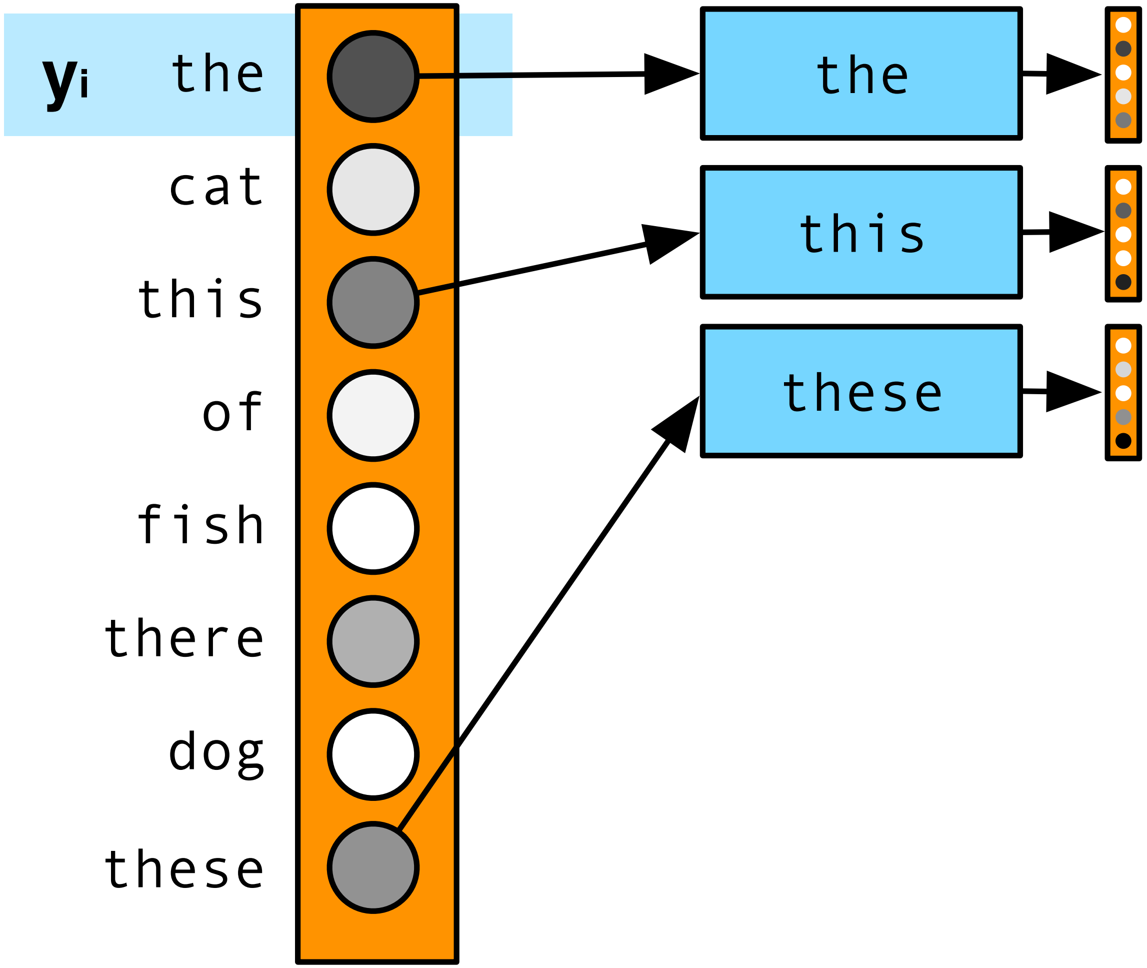
Select Second Best Word



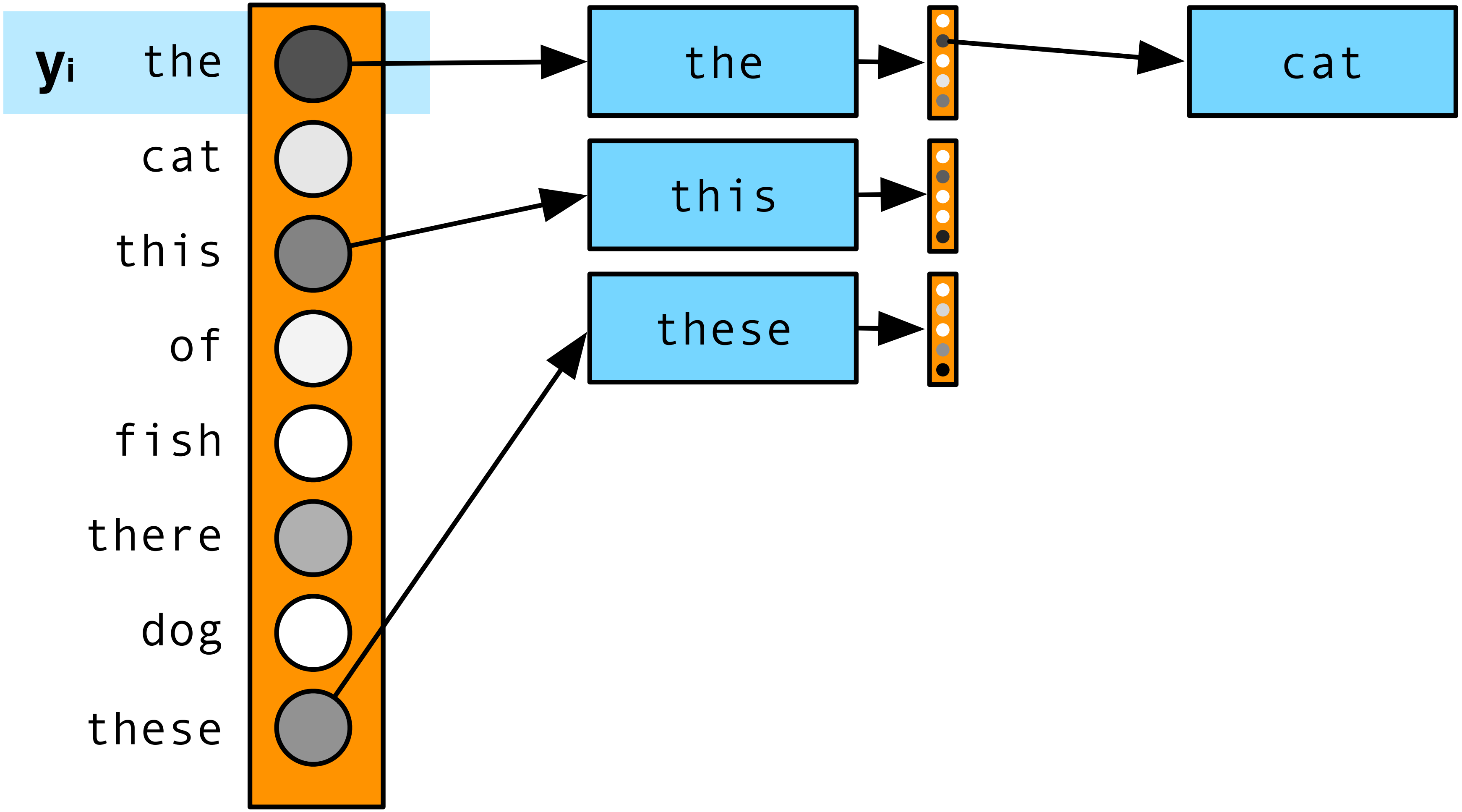
Select Third Best Word



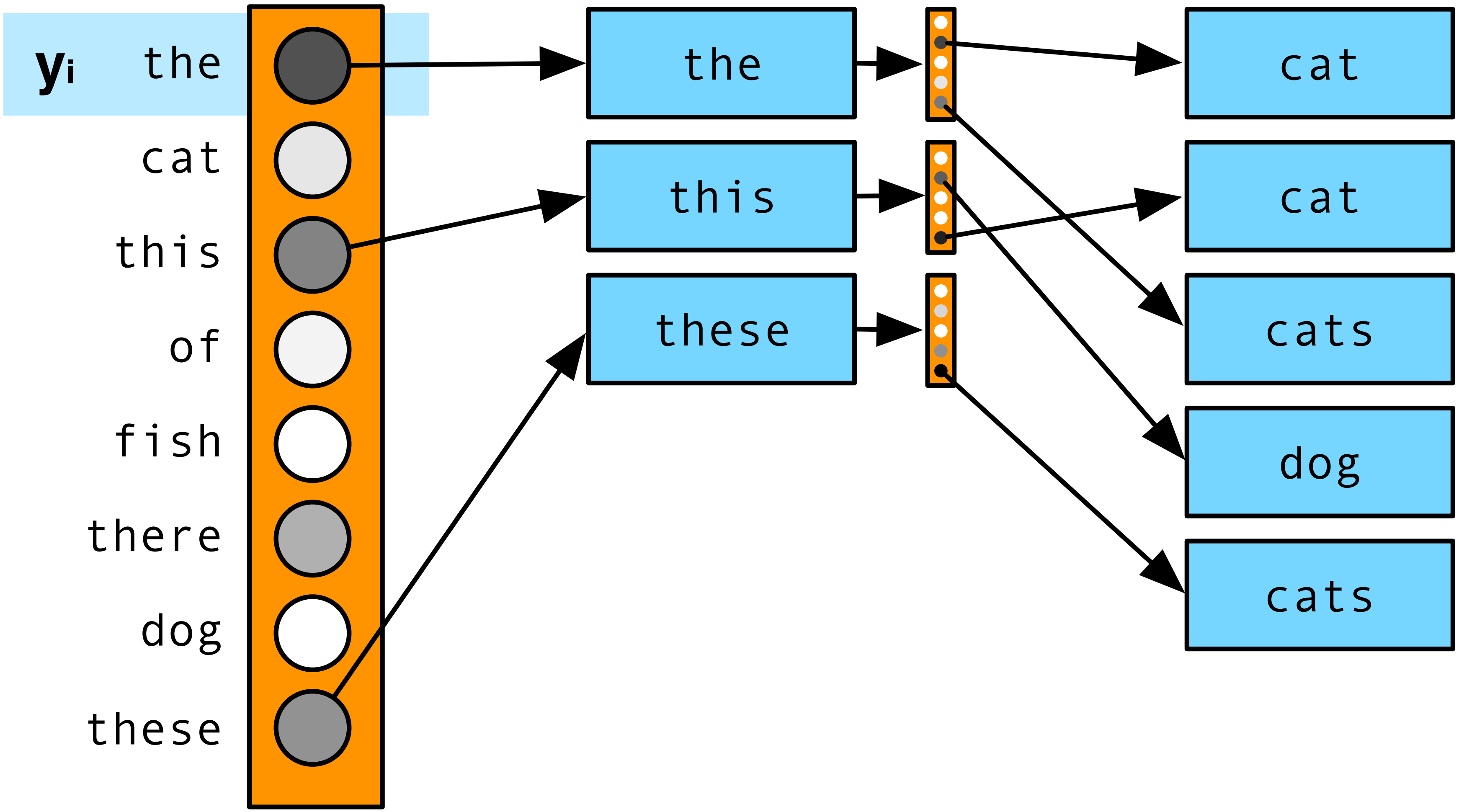
Use Selected Word for Next Predictions



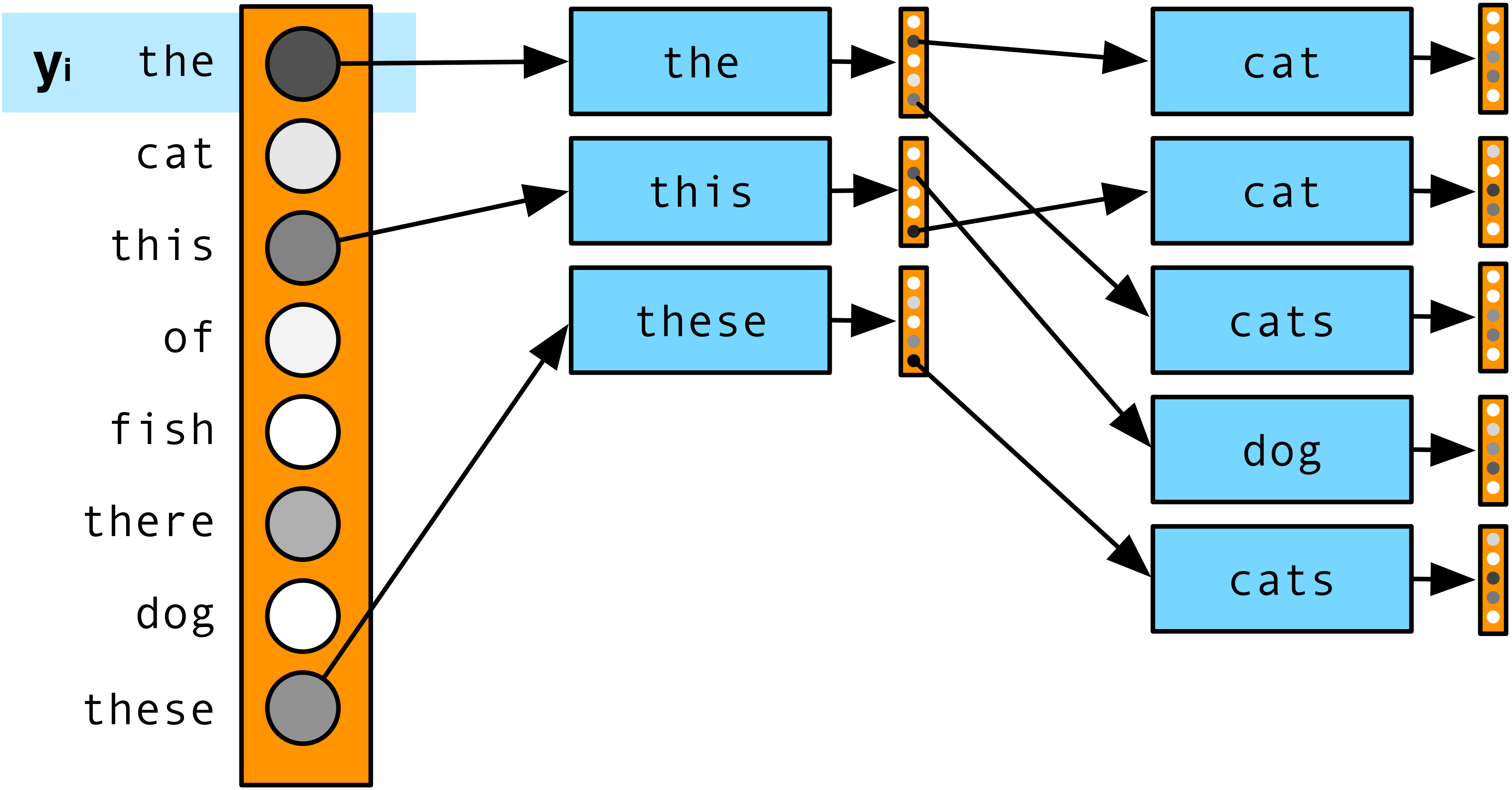
Select Best Continuation



Select Next Best Continuations

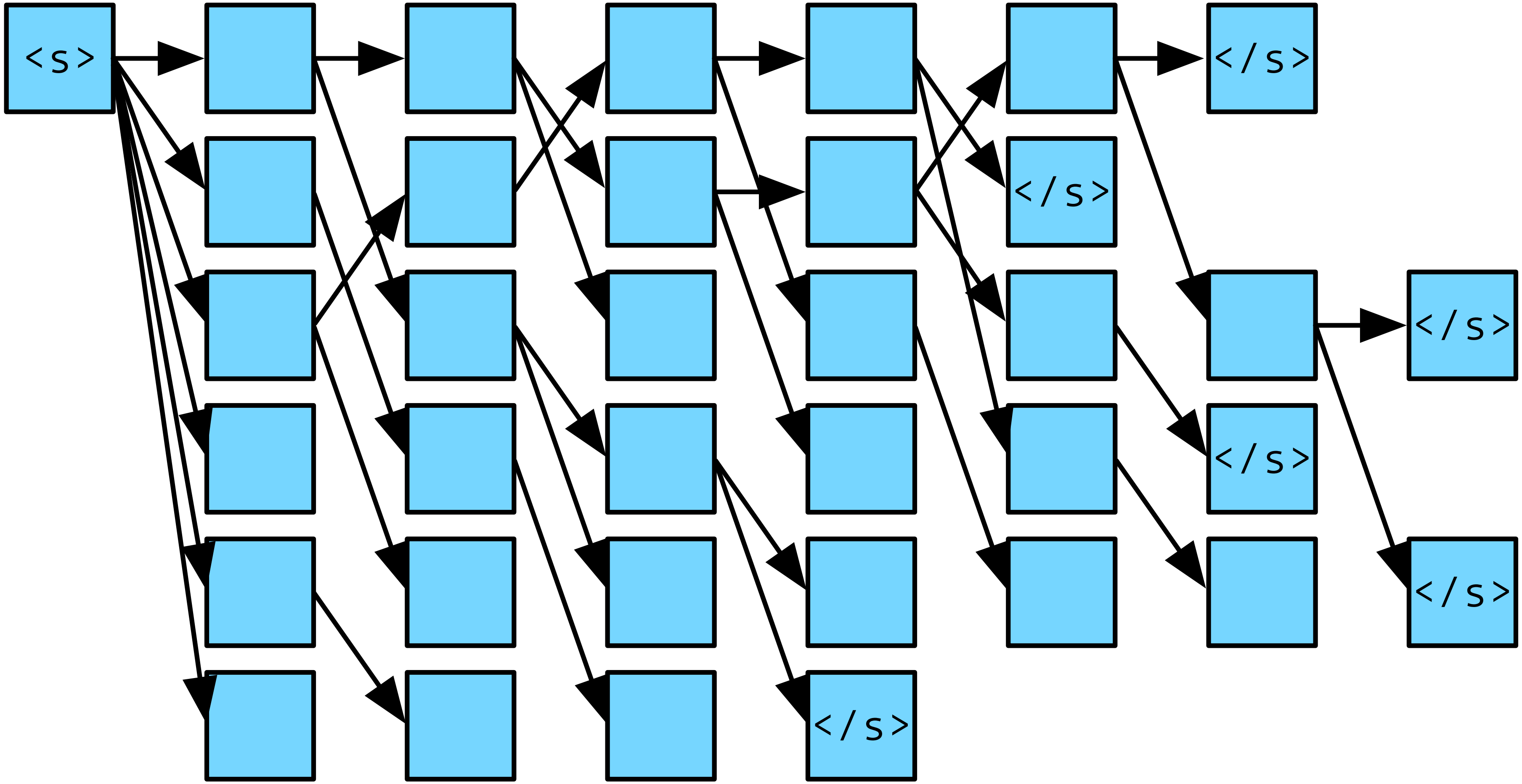


Continue...

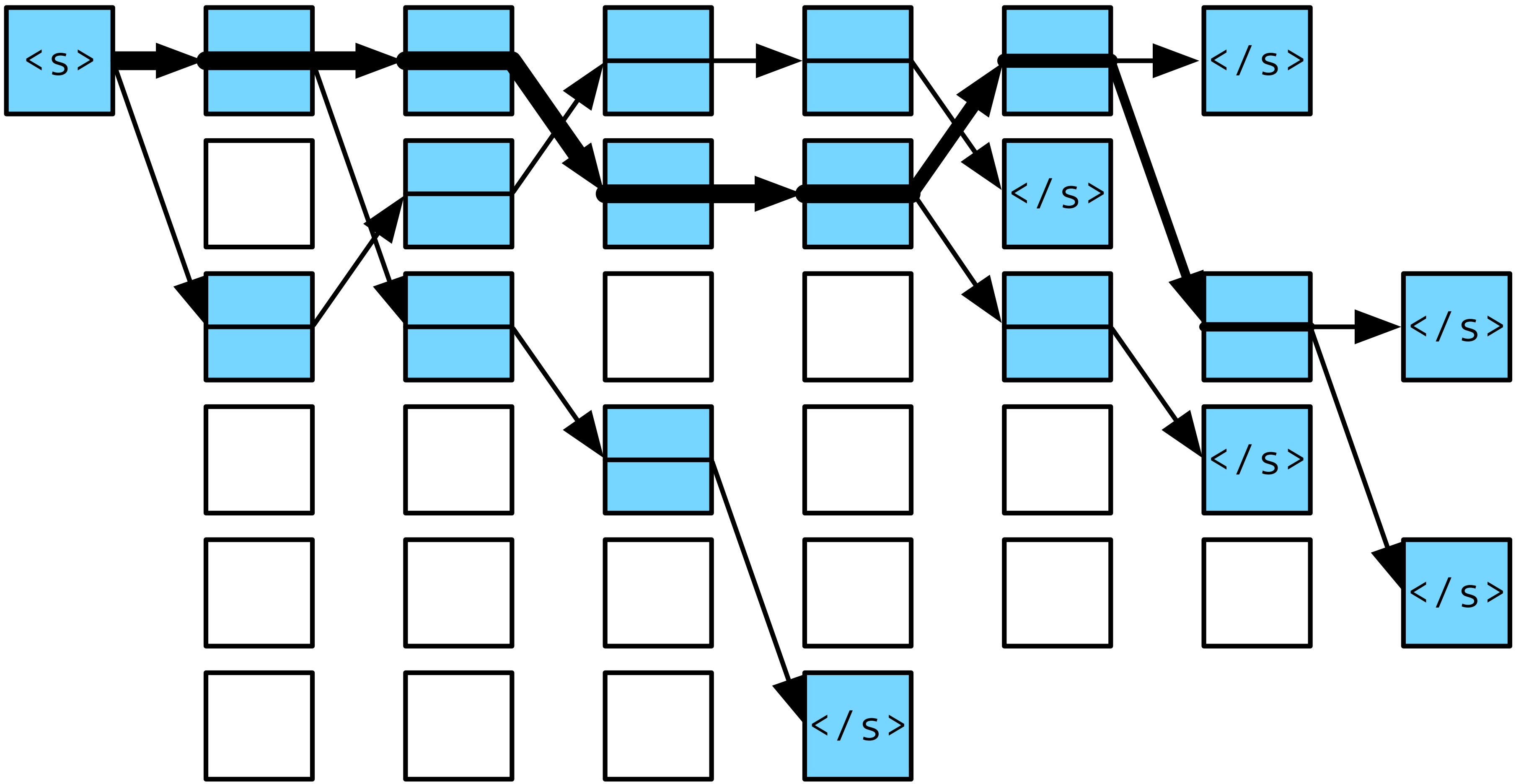


y_i
the
cat
this
of
fish
there
dog
these
the
this
these
cat
cat
cats
dog
cats

Beam Search



Best Paths



Beam Search Details

- Normalize score by length
- No recombination (paths cannot be merged)

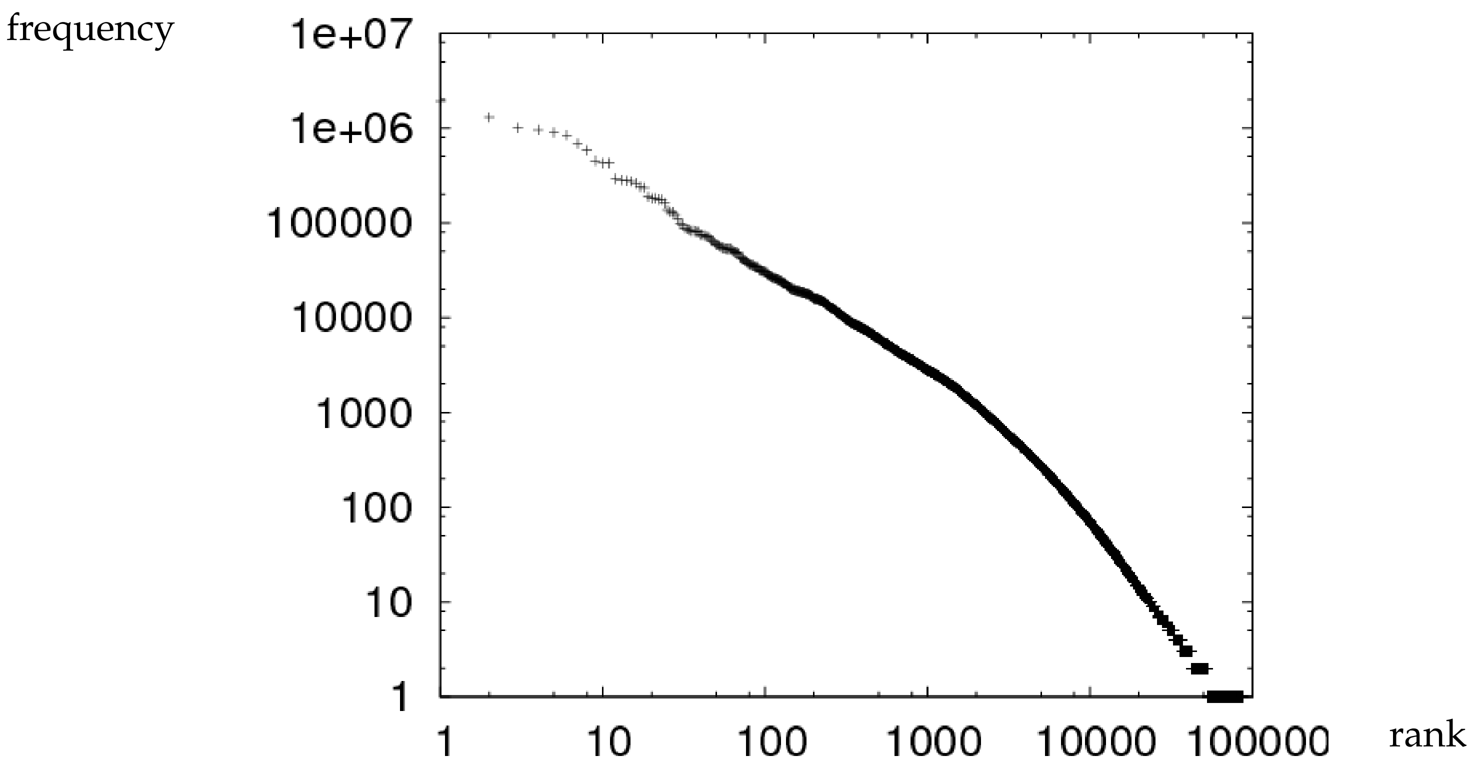
Output Word Predictions

Input Sentence: *ich glaube aber auch , er ist clever genug um seine Aussagen vage genug zu halten , so dass sie auf verschiedene Art und Weise interpretiert werden können .*

Best		Alternatives
but	(42.1%)	however (25.3%), I (20.4%), yet (1.9%), and (0.8%), nor (0.8%), ...
I	(80.4%)	also (6.0%), , (4.7%), it (1.2%), in (0.7%), nor (0.5%), he (0.4%), ...
also	(85.2%)	think (4.2%), do (3.1%), believe (2.9%), , (0.8%), too (0.5%), ...
believe	(68.4%)	think (28.6%), feel (1.6%), do (0.8%), ...
he	(90.4%)	that (6.7%), it (2.2%), him (0.2%), ...
is	(74.7%)	's (24.4%), has (0.3%), was (0.1%), ...
clever	(99.1%)	smart (0.6%), ...
enough	(99.9%)	
to	(95.5%)	about (1.2%), for (1.1%), in (1.0%), of (0.3%), around (0.1%), ...
keep	(69.8%)	maintain (4.5%), hold (4.4%), be (4.2%), have (1.1%), make (1.0%), ...
his	(86.2%)	its (2.1%), statements (1.5%), what (1.0%), out (0.6%), the (0.6%), ...
statements	(91.9%)	testimony (1.5%), messages (0.7%), comments (0.6%), ...
vague	(96.2%)	v@@ (1.2%), in (0.6%), ambiguous (0.3%), ...
enough	(98.9%)	and (0.2%), ...
so	(51.1%)	, (44.3%), to (1.2%), in (0.6%), and (0.5%), just (0.2%), that (0.2%), ...
they	(55.2%)	that (35.3%), it (2.5%), can (1.6%), you (0.8%), we (0.4%), to (0.3%), ...
can	(93.2%)	may (2.7%), could (1.6%), are (0.8%), will (0.6%), might (0.5%), ...
be	(98.4%)	have (0.3%), interpret (0.2%), get (0.2%), ...
interpreted	(99.1%)	interpre@@ (0.1%), constru@@ (0.1%), ...
in	(96.5%)	on (0.9%), differently (0.5%), as (0.3%), to (0.2%), for (0.2%), by (0.1%), ...
different	(41.5%)	a (25.2%), various (22.7%), several (3.6%), ways (2.4%), some (1.7%), ...
ways	(99.3%)	way (0.2%), manner (0.2%), ...
.	(99.2%)	</s> (0.2%), , (0.1%), ...
</s>	(100.0%)	

large vocabularies

Zipf's Law: Many Rare Words



frequency $\times$ rank = constant

Many Problems

- Sparse data
  - words that occur once or twice have unreliable statistics

- Computation cost
  - input word embedding matrix: $|V| \times 1000$
  - outout word prediction matrix: $1000 \times |V|$

Some Causes for Large Vocabularies

- Morphology

  tweet, tweets, tweeted, tweeting, retweet, ...

  → morphological analysis?

- Compounding

  homework, website, ...

  → compound splitting?

- Names

  Netanyahu, Jones, Macron, Hoboken, ...

  → transliteration?

⇒ Breaking up words into **subwords** may be a good idea

Byte Pair Encoding

- Start by breaking up words into characters

`t h e ␣ f a t ␣ c a t ␣ i s ␣ i n ␣ t h e ␣ t h i n ␣ b a g`

- Merge frequent pairs

```
t h→th     th e ␣ f a t ␣ c a t ␣ i s ␣ i n ␣ th e ␣ th i n ␣ b a g
a t→at     th e ␣ f at ␣ c at ␣ i s ␣ i n ␣ th e ␣ th i n ␣ b a g
i n→in     th e ␣ f at ␣ c at ␣ i s ␣ in ␣ th e ␣ th in ␣ b a g
th e→the   the ␣ f at ␣ c at ␣ i s ␣ in ␣ the ␣ th in ␣ b a g
```

- Each merge operation increases the vocabulary size
  - starting with the size of the character set (maybe 100 for Latin script)
  - stopping at, say, 50,000

Example: 49,500 BPE Operations

Obama receives Net@@ any@@ ahu

the relationship between Obama and Net@@ any@@ ahu is not exactly friendly . the two wanted to talk about the implementation of the international agreement and about Teheran 's destabil@@ ising activities in the Middle East . the meeting was also planned to cover the conflict with the Palestinians and the disputed two state solution . relations between Obama and Net@@ any@@ ahu have been stra@@ ined for years . Washington critic@@ ises the continuous building of settlements in Israel and acc@@ uses Net@@ any@@ ahu of a lack of initiative in the peace process . the relationship between the two has further deteriorated because of the deal that Obama negotiated on Iran 's atomic programme . in March , at the invitation of the Republic@@ ans , Net@@ any@@ ahu made a controversial speech to the US Congress , which was partly seen as an aff@@ ront to Obama . the speech had not been agreed with Obama , who had rejected a meeting with reference to the election that was at that time im@@ pending in Israel .

Questions?

Outline

- Machine Translation: History & Problem Formulation
- Language Model
- Encoder-Decoder NMT Model
- Training & Inference
- Alternative NMT Models

Many variants to the standard Encoder-Decoder

- Ensembles
- Coverage and Alignment
- Linguistic Annotation
- Alternative architectures (beyond recurrent architectures)

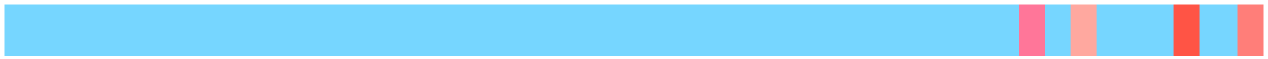
Ensembles

Ensembling

- Train multiple models

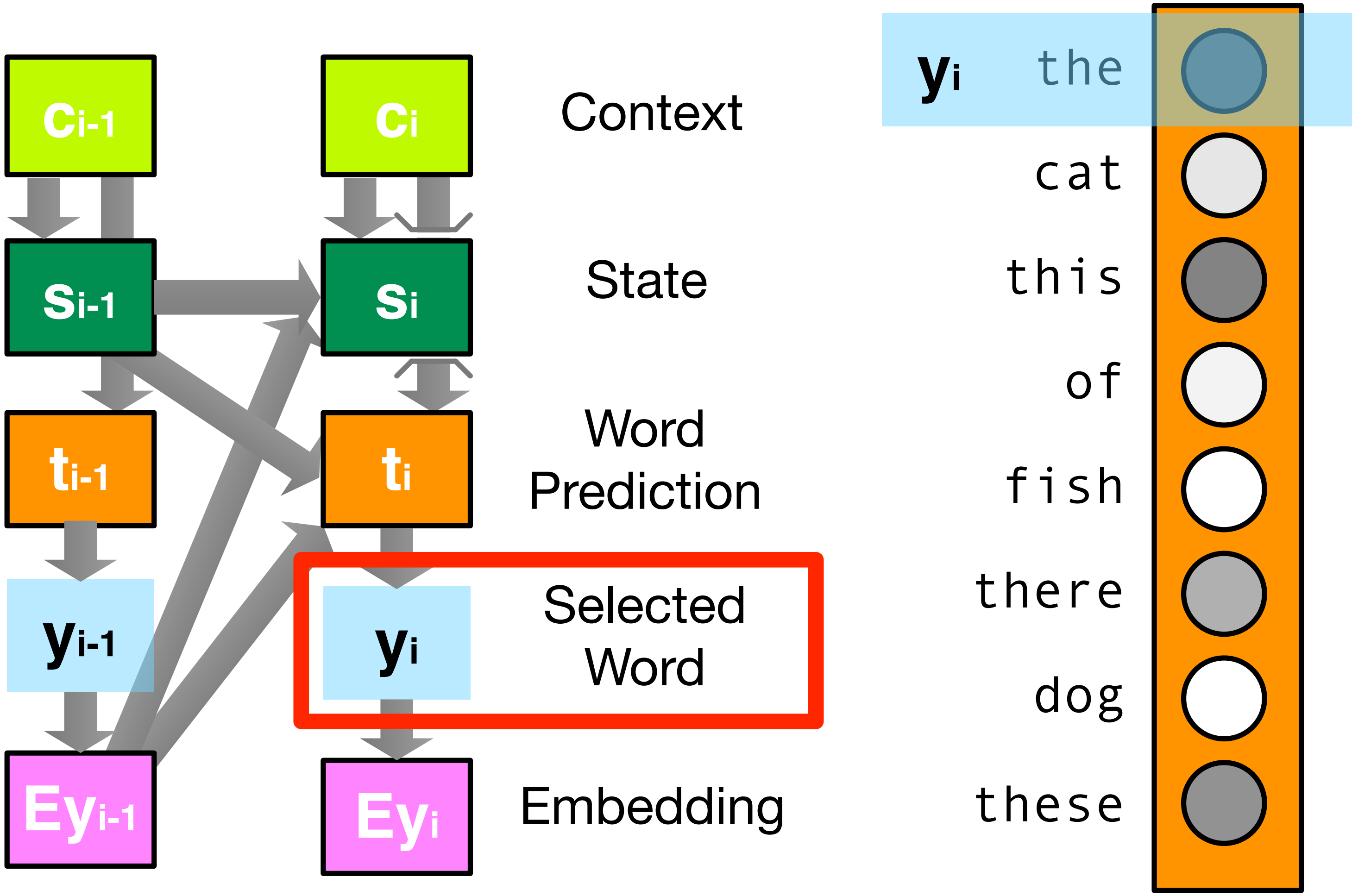
- Say, by different random initializations

- Or, by using model dumps from earlier iterations

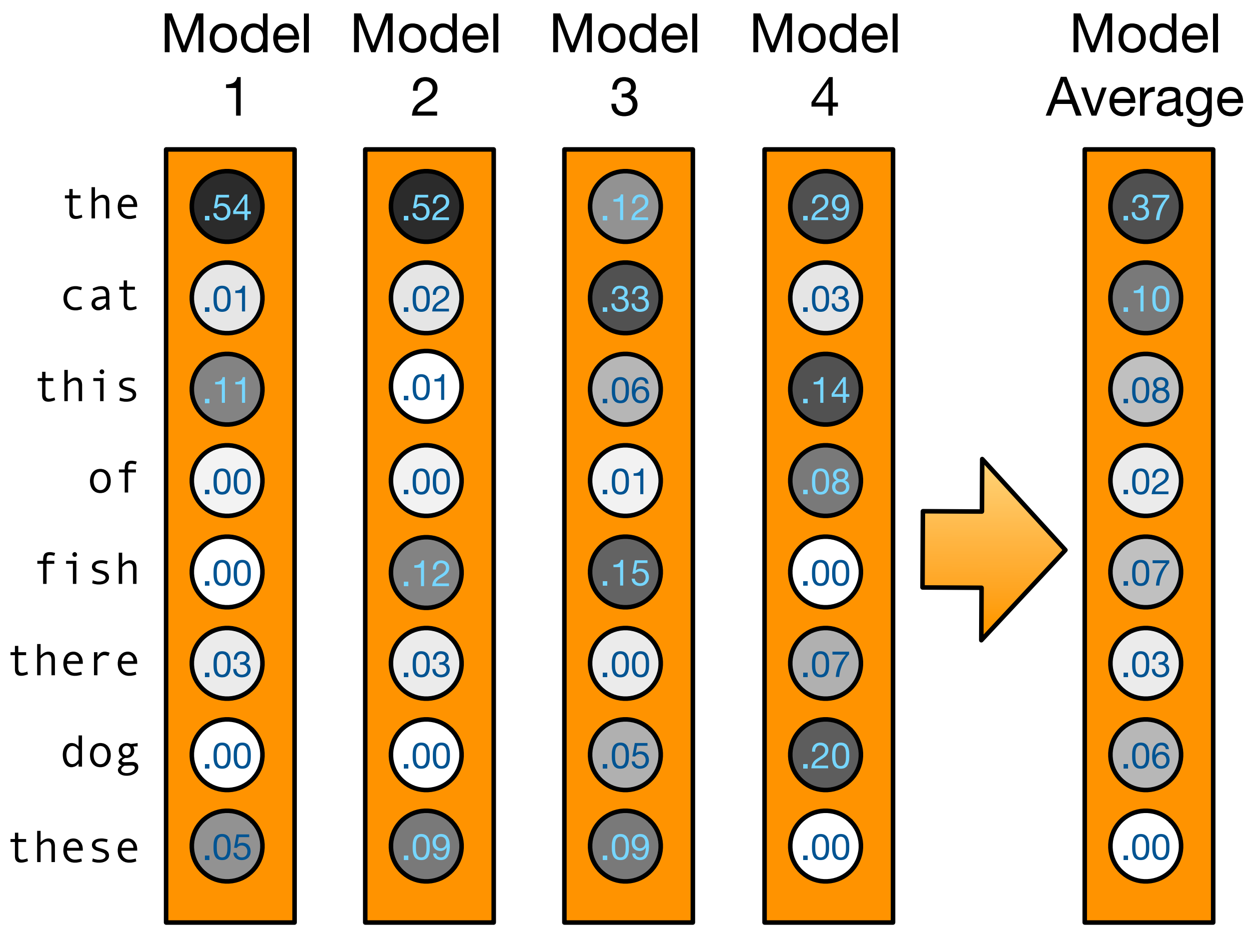


(most recent, or interim models with highest validation score)

Decoding with Single Model



Combine Predictions



Ensembling

- Surprisingly reliable method in machine learning
- Long history, many variants:
  bagging, ensemble, model averaging, system combination, ...
- Works because errors are random, but correct decisions unique

alignment and coverage

Attention vs. Alignment

	relations	between	Obama	and	Netanyahu	have	been	strained	for	years	.
die	56		16								
Beziehungen	89										
zwischen		72	26								
Obama			96								
und				79							
Netanjahu					98						
sind						42	11	38			
seit								22	54	10	
Jahren										98	
angespannt								84			
.						11	14	23			49

Guided Alignment

- Guided alignment training for neural networks
  - traditional objective function: match output words
  - now: also match given word alignments

- Add as cost to objective function
  - given alignment matrix A, with $\sum_j A_{ij} = 1$ (from IBM Models)
  - computed attention α_{ij} (also $\sum_j \alpha_{ij} = 1$ due to softmax)
  - added training objective (cross-entropy)

$$\text{cost}_{\text{CE}} = -\frac{1}{I} \sum_{i=1}^{I} \sum_{j=1}^{J} A_{ij} \log \alpha_{ij}$$

Coverage

	in	order	to	solve	the	problem	,	the	"	Social	Housing	"	alliance	suggests	a	fresh	start	.
um	*37*		*33*					*13*										
das					*84*													
Problem					*10*	*80*												
zu				*63*			*12*											
lösen				*81*														
,							*40*							*30*				
schlägt														*80*				
das								*71*	*18*									
Unternehmen									*86*			*10*						
der									*84*									
Gesellschaft									*80*									
für									*45*	*41*								
soziale									*40*	*44*	*10*							
Bildung											*89*							
vor									*12*			*40*	*10*					
.									*10*			*37*			*11*	*13*		
	43	*7*	*46*	*161*	*108*	*89*	*62*	*112*	*392*	*121*	*110*	*130*	*26*	*132*	*22*	*19*	*6*	*6*

Tracking Coverage

- Neural machine translation may drop or duplicate content
- Track coverage during decoding

$$\text{coverage}(j) = \sum_i \alpha_{i,j}$$

$$\text{over-generation} = \max\Big(0, \sum_j \text{coverage}(j) - 1\Big)$$

$$\text{under-generation} = \min\Big(1, \sum_j \text{coverage}(j)\Big)$$

- Add as cost to hypotheses

Coverage Models

- Use as information for state progression

$$a(s_{i-1}, h_j) = W^a s_{i-1} + U^a h_j + V^a \text{coverage}(j) + b^a$$

- Add to objective function

$$\log \sum_i P(y_i|x) + \lambda \sum_j (1 - \text{coverage}(j))^2$$

- May also model fertility
  - some words are typically dropped
  - some words produce multiple output words

linguistic annotation

Example

Words	*the*	*girl*	*watched*	*attentively*	*the*	*beautiful*	*fireflies*
Part of speech	DET	NN	VFIN	ADV	DET	JJ	NNS
Lemma	*the*	*girl*	*watch*	*attentive*	*the*	*beautiful*	*firefly*
Morphology	-	SING.	PAST	-	-	-	PLURAL
Noun phrase	BEGIN	CONT	OTHER	OTHER	BEGIN	CONT	CONT
Verb phrase	OTHER	OTHER	BEGIN	CONT	CONT	CONT	CONT
Synt. dependency	*girl*	*watched*	-	*watched*	*fireflies*	*fireflies*	*watched*
Depend. relation	DET	SUBJ	-	ADV	DET	ADJ	OBJ
Semantic role	-	ACTOR	-	MANNER	-	MOD	PATIENT
Semantic type	-	HUMAN	VIEW	-	-	-	ANIMATE

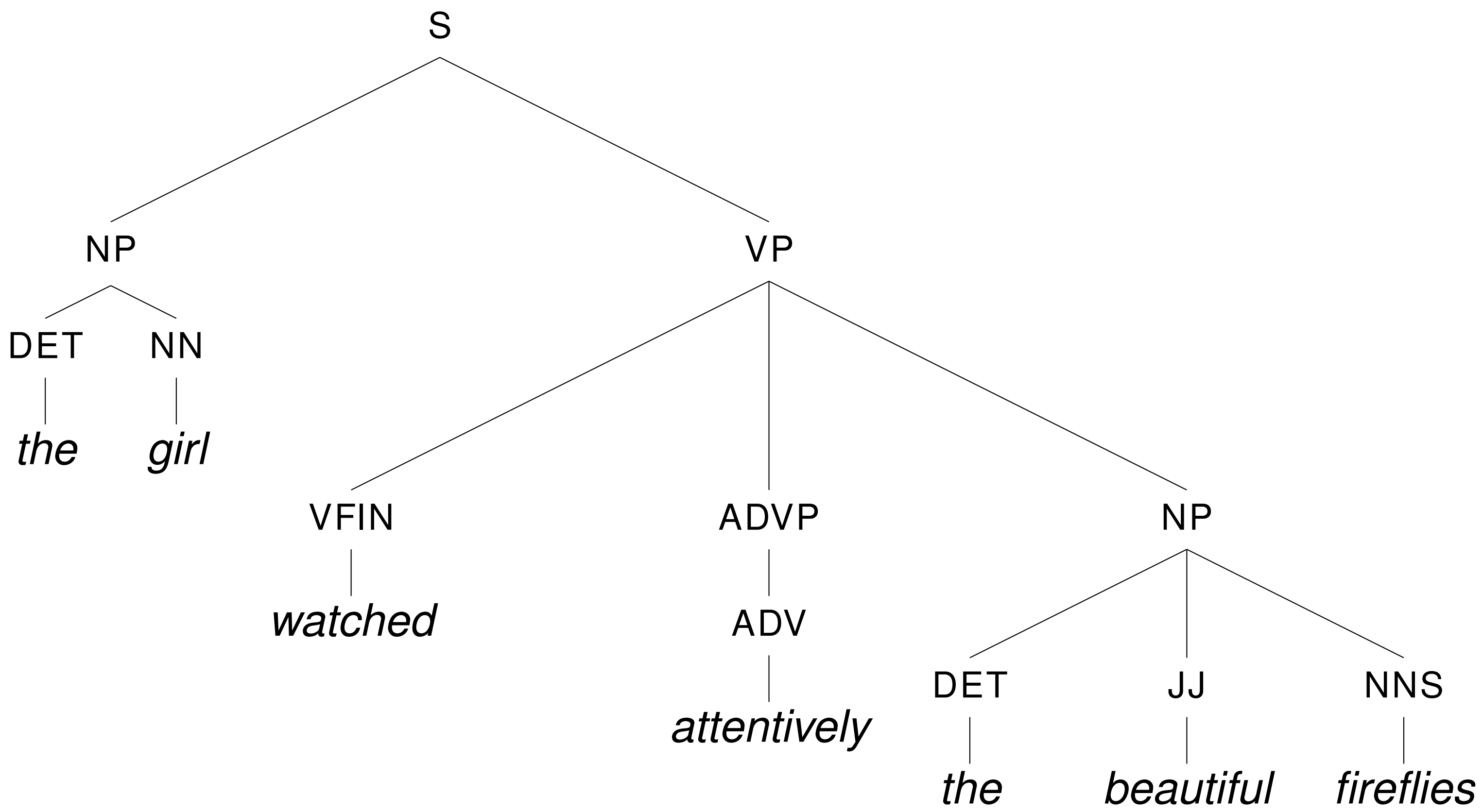
Input Annotation

- Input words are encoded in one-hot vectors
- Additional linguistic annotation
  - part-of-speech tag
  - morphological features
  - etc.
- Encode each annotation in its own one-hot vector space
- Concatenate one-hot vecors
- Essentially:
  - each annotation maps to embedding
  - embeddings are added

Output Annotation

- Same can be done for output
- Additional output annotation is latent feature
  - ultimately, we do not care if right part-of-speech tag is predicted
  - only right output words matter
- Optimizing for correct output annotation $\rightarrow$ better prediction of output words

Linearized Output Syntax

Sentence	*the girl watched attentively the beautiful fireflies*
Syntax tree	
Linearized	(S (NP (DET *the*) (NN *girl*)) (VP (VFIN *watched*) (ADVP (ADV *attentively*)) (NP (DET *the*) (JJ *beautiful*) (NNS *fireflies*))))

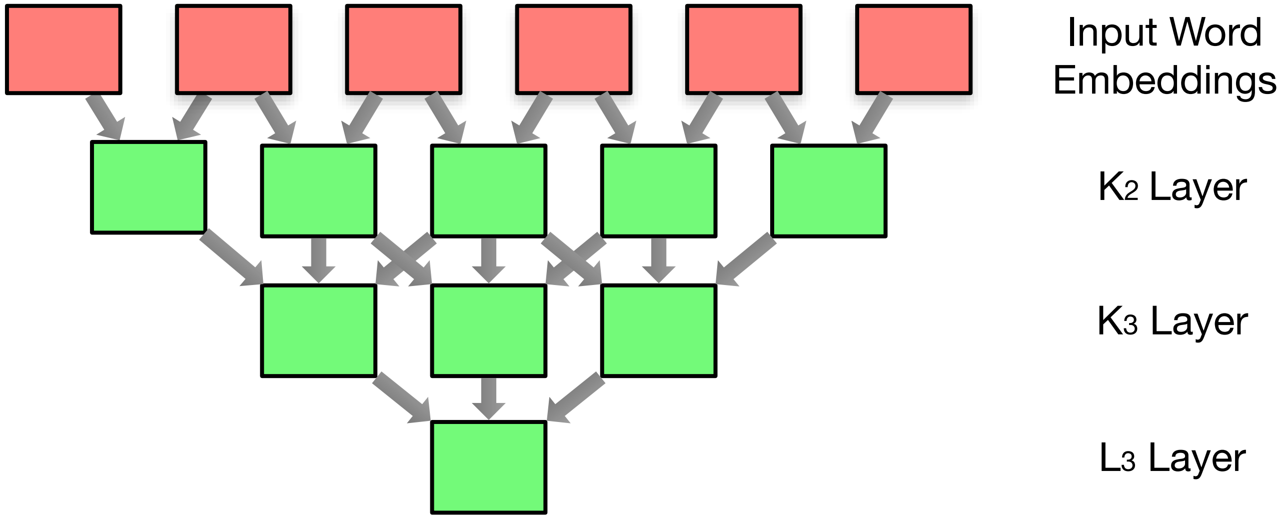
alternative architectures

Beyond Recurrent Neural Networks

- We presented the currently dominant model
  - recurrent neural networks for encoder and decoder
  - attention
- Convolutional neural networks
- Self attention

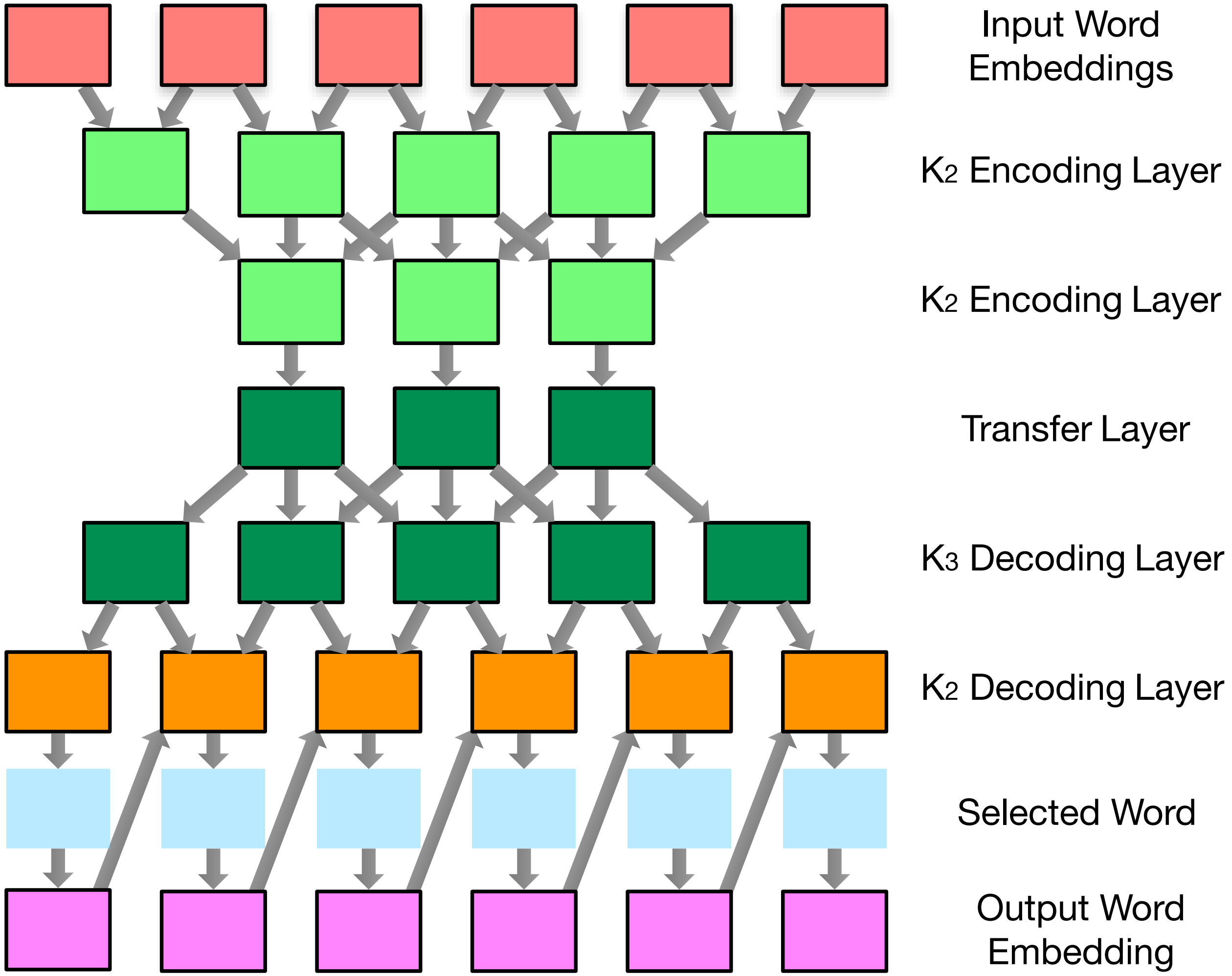
convolutional neural networks

Convolutional Neural Networks



- Build sentence representation bottom-up
  - merge any n neighboring nodes
  - n may be 2, 3, ...

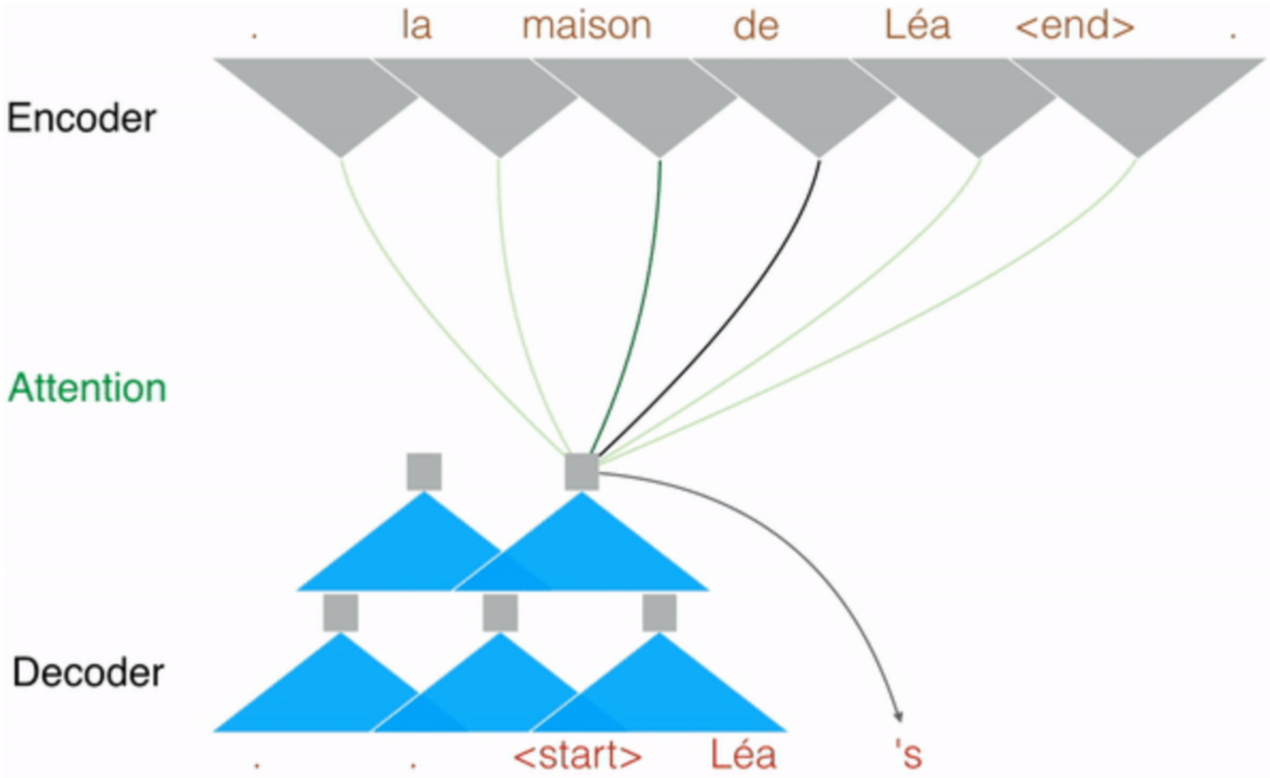
Generation


Input Word Embeddings
K2 Encoding Layer
K2 Encoding Layer
Transfer Layer
K3 Decoding Layer
K2 Decoding Layer
Selected Word
Output Word Embedding

Generation

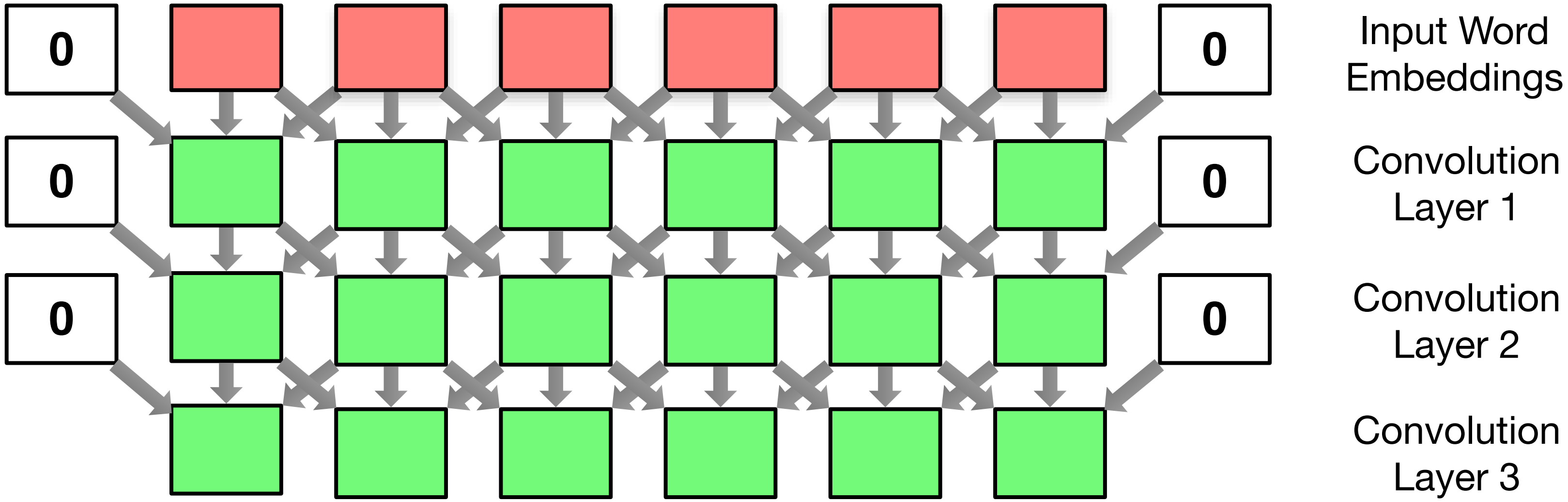
- Encode with convolutional neural network
- Decode with convolutional neural network
- Also include a linear recurrent neural network
- Important: predict length of output sentence
- Does it work?
  used successfully in re-ranking (Cho et al., 2014)

Convolutional Network with Attention



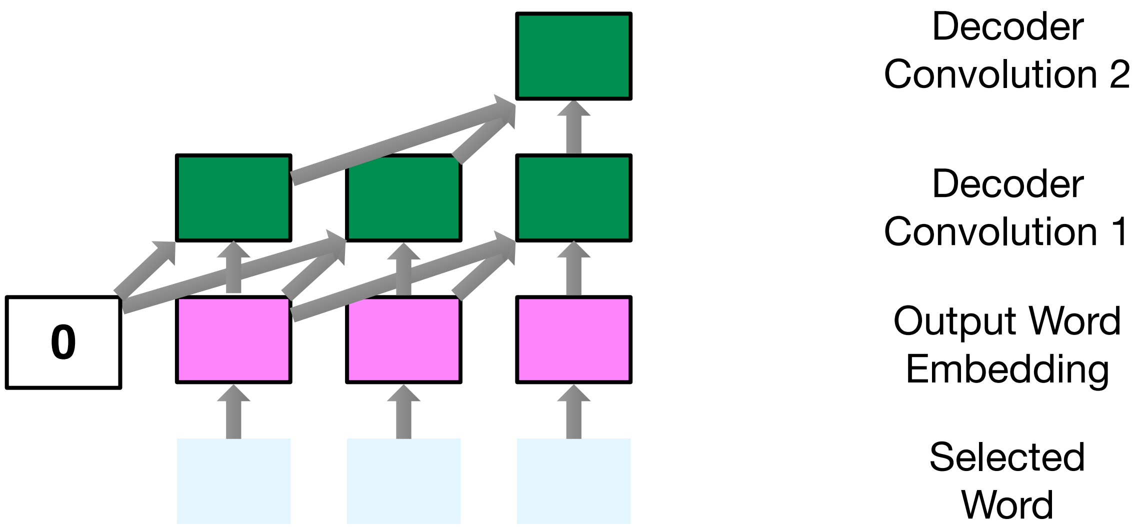
(Facebook, 2017)

Convolutional Encoder



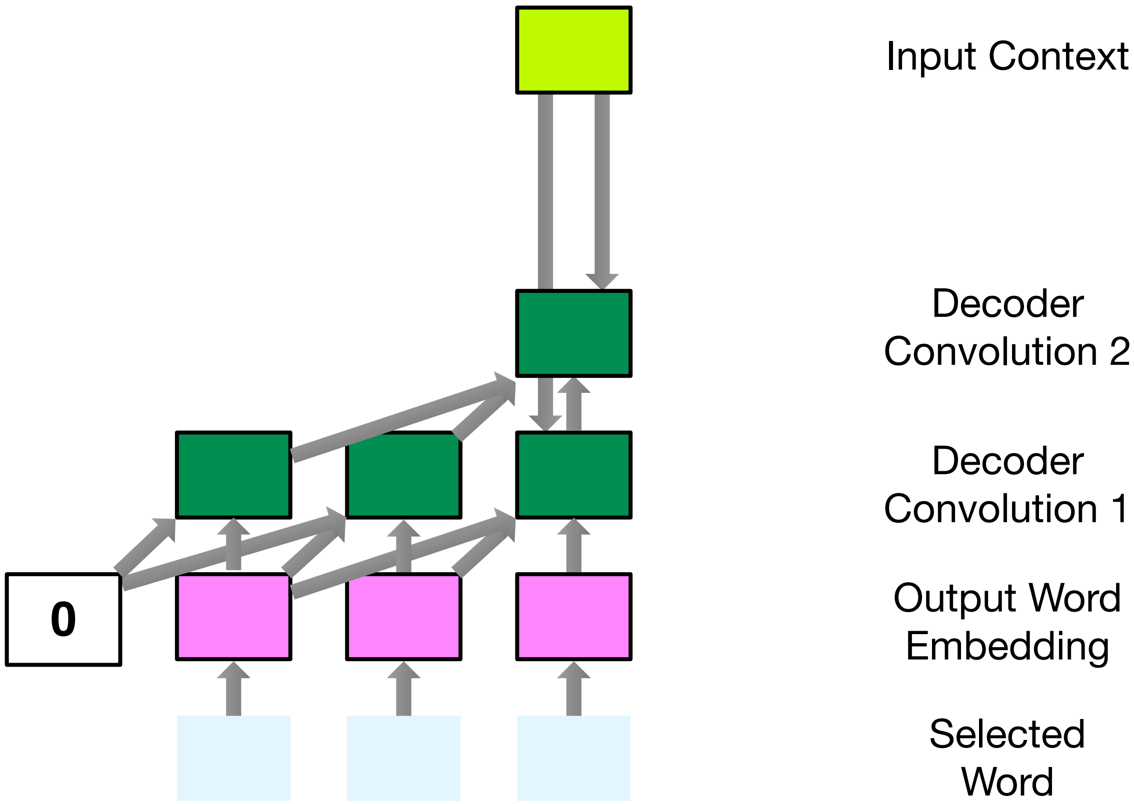
- Similar idea as deep recurrent neural networks
- Good: more parallelizable
- Bad: less context when refining representation of a word

Convolutional Decoder



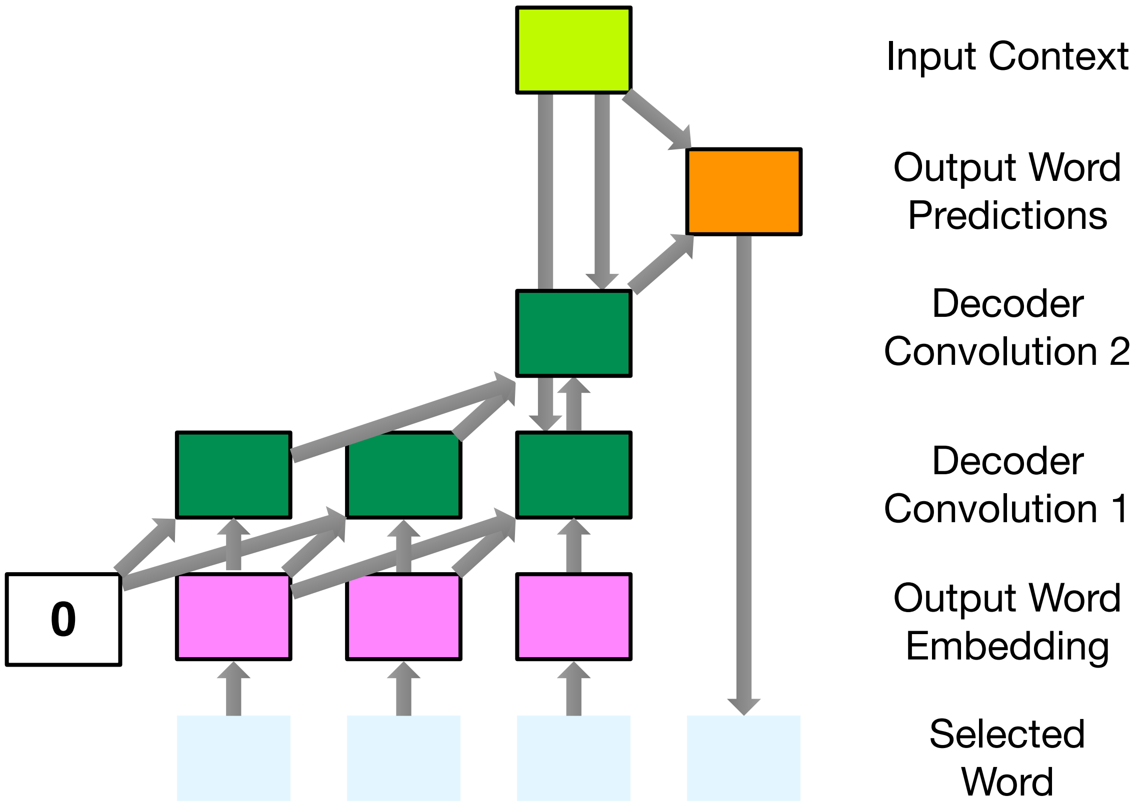
- Convolutions over output words
- Only previously produced output words
  (still left-to-right decoding)

Convolutional Decoder



- Inclusion of Input context
- Context result of attention mechanism (similar to previous)

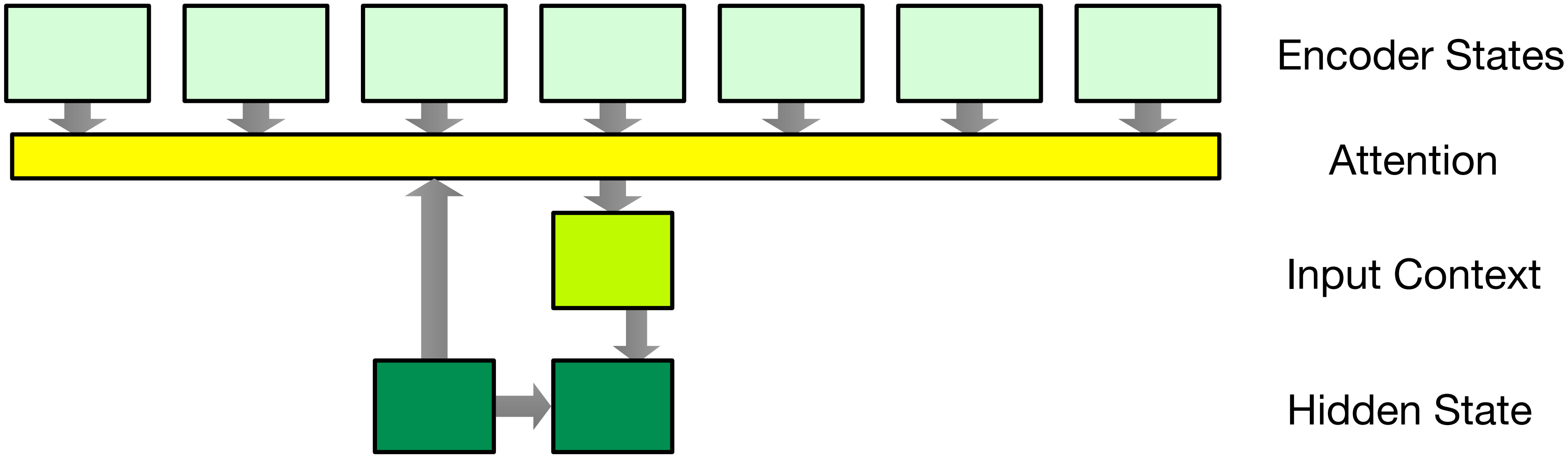
Convolutional Decoder



- Predict output word distribution
- Select output word

self-attention

Attention



- Compute association between last hidden state and encoder states

Attention Math

- Input word representation h_k
- Decoder state s_j
- Computations

$$a_{jk} = \frac{1}{|h|} s_j h_k^T \qquad \text{raw association}$$

$$\alpha_{jk} = \frac{\exp(a_{jk})}{\sum_\kappa \exp(a_{j\kappa})} \qquad \text{normalized association (softmax)}$$

$$\text{self-attention}(h_j) = \sum_k \alpha_{j\kappa} h_k \qquad \text{weighted sum}$$

Self-Attention

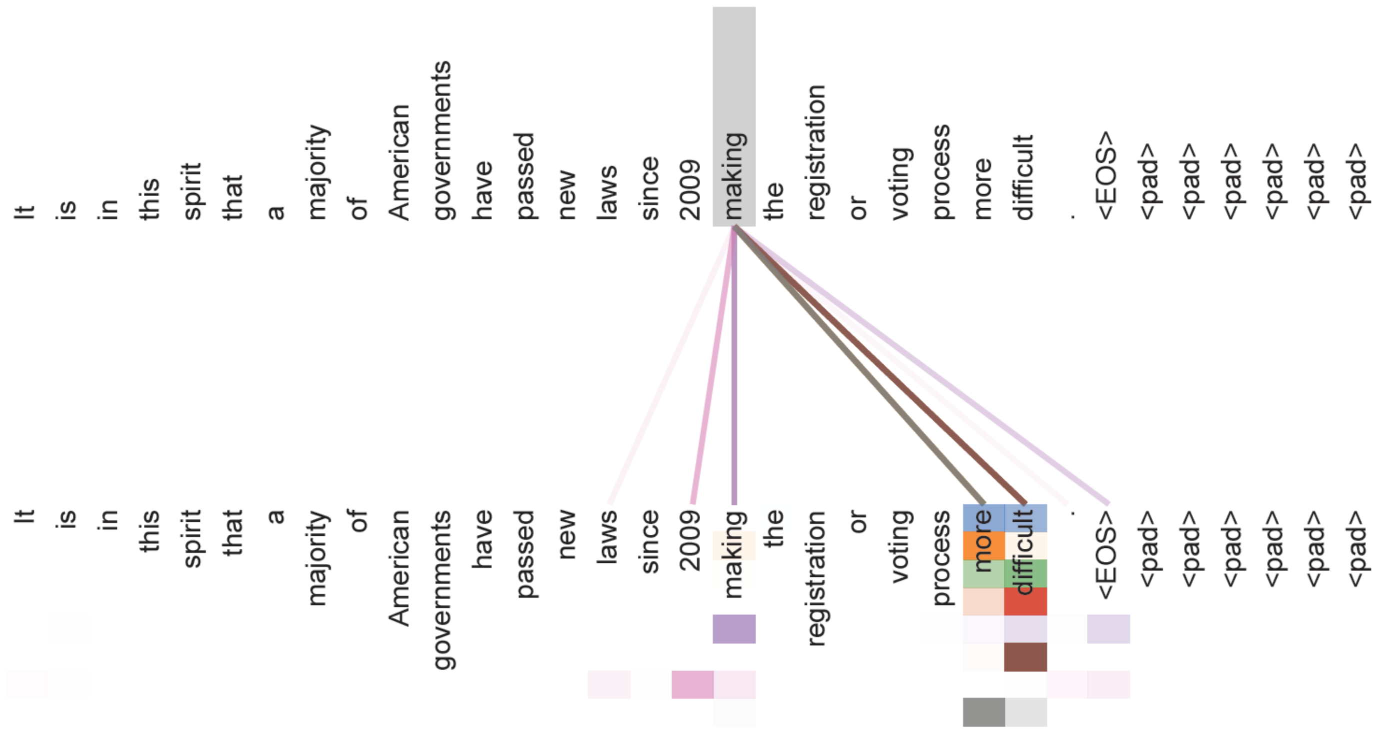
- Attention

$$a_{jk} = \frac{1}{|h|} s_j h_k^T$$

- Self-attention

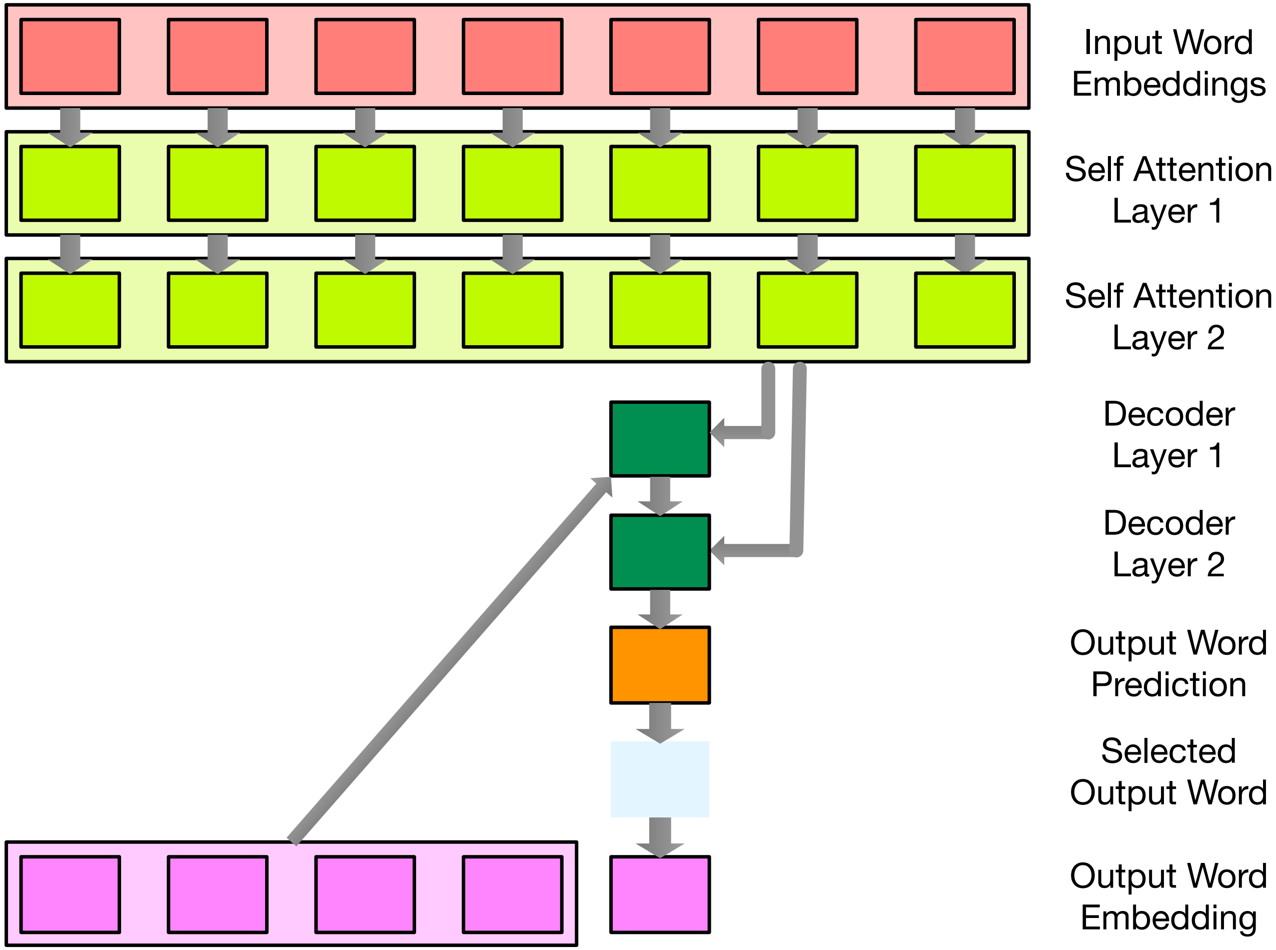
$$a_{jk} = \frac{1}{|h|} h_j h_k^T$$

Why?



- Refine representation of word with related words
  making ... more difficult refines making
- Good: more parallelizable than recurrent neural network
- Good: wide context when refining representation of a word

Stacked Attention in Decoder



Questions?