CS772: Deep Learning for Natural Language Processing (DL-NLP)

Recurrent Neural Networks

Language Modeling & Sequence Labeling

Anoop Kunchukuttan
Microsoft, MT Group, Hyderabad
anoop.kunchukuttan@gmail.com

For CS772 at IIT Bombay, Feb 2022
Course Instructor: Prof. Pushpak Bhattacharyya
Week 9 of 28th Feb, 2022

Language Modeling

Fundamental Task in NLP

The capital of Maharashtra is _____
The capital of ______ is Mumbai

The ability to predict words is a sign of language skill

In statistical NLP, such a capability is at the core of many NLP applications

N-gram Language Models (textbook chapter)

Language Modeling Task

Given a sequence of words ➔ $(x_1\ \ x_2\ \ x_3\ \ x_4 \ldots\ldots x_i)$

Compute the probability distribution of the next word ➔ $P(x_{i+1}|x_1\ \ x_2\ \ x_3\ \ x_4 \ldots\ldots x_i)$

$P(Mumbai|X)$

The capital of Maharashtra is ______

$P(Bihar|X)$

$P(Chennai|X)$

LM can assign probability to a sentence

$P(market|X)$

ngram based Language Modeling

$$P(w|\, The\ capital\ of)$$

$$=$$

$$\frac{count(The\ capital\ of\ w)}{count(The\ capital\ of)}$$

$$P(x_1\ \ x_2\ \ x_3\ \ x_4\ \ldots..x_i, x_{i+1})$$
$$= P(x_{i+1}|x_1\ \ x_2\ \ x_3\ \ x_4\ \ldots..x_i) \times P(x_1\ \ x_2\ \ x_3\ \ x_4\ \ldots..x_i, x_{i+1})$$

$$P(x_{i+1}|x_1\ \ x_2\ \ x_3\ \ x_4\ \ldots..x_i) = P(x_{i+1}|x_i)$$ (bigram LM)

$$P(x_{i+1}|x_1\ \ x_2\ \ x_3\ \ x_4\ \ldots..x_i) = P(x_{i+1}|x_i, x_{i-1})$$ (trigram LM)

Sparsity issues
Storage issues
Limited context

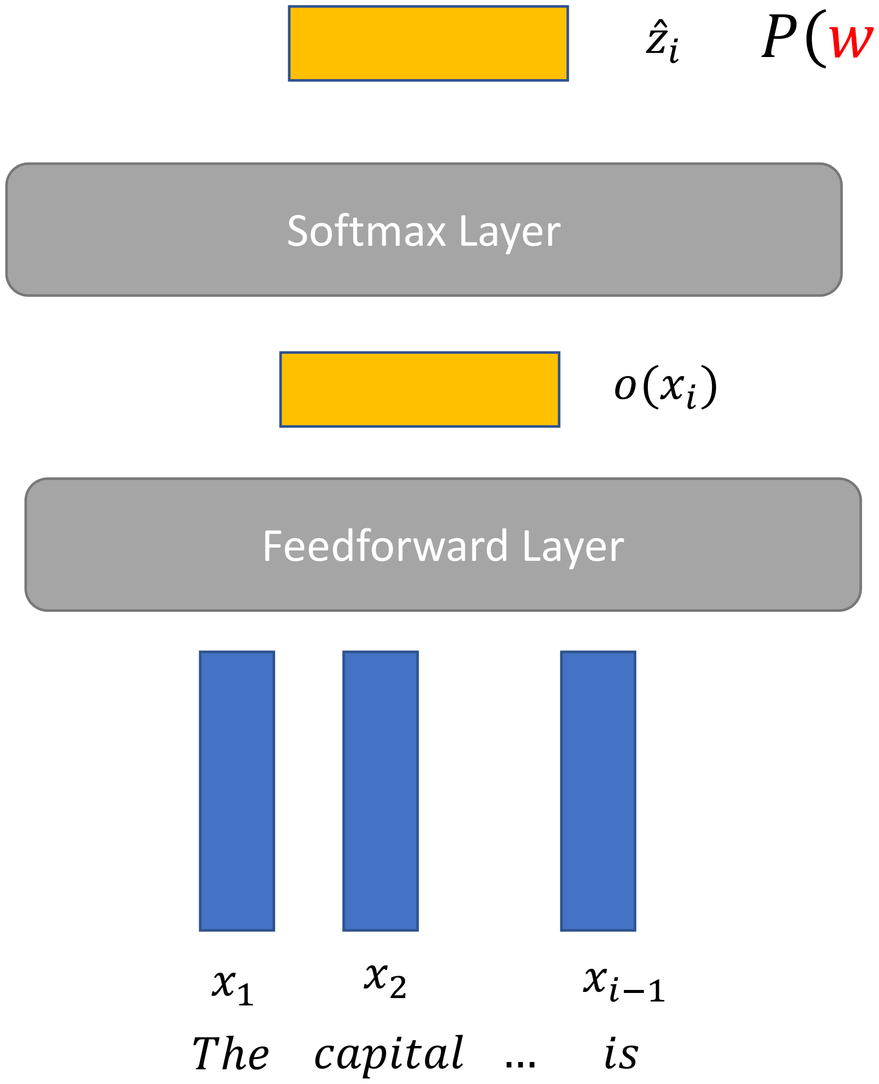
- *Unbounded context not possible* ➔ *Markov assumption & Chain Rule*
- *Estimate based on counting* ➔ *model is a large lookup table of probabilities*
- *Large number of n-grams*
  - *Smoothing, interpolation, etc. to approximate probabilities for rare events*

Smoothing techniques: http://nrs.harvard.edu/urn-3:HUL.InstRepos:25104739

Feedforward NN based LM

Fixed limited context

$P(\textcolor{red}{w} | \, The\ capital\ of\ Maharashtra\ is)$



$$h(x_i) = \sigma(\textcolor{red}{W^x} x_i + b_2)$$

$$o(x_i) = \textcolor{red}{W^o} h(x_i) + b_1$$

$$\hat{z}_i = softmax(o(x_i))$$

$\hat{z}_i$ is the probability distribution of the next word

Word embeddings address sparsity issues

Prob distribution implemented as NN
→ makes the model size reasonably compact

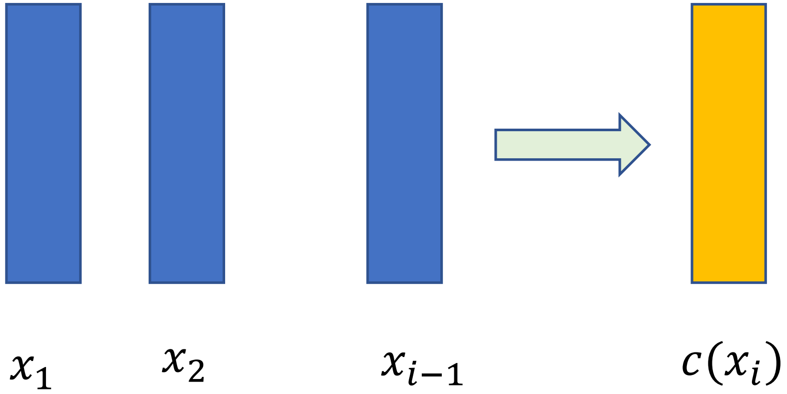
https://www.jmlr.org/papers/volume3/bengio03a/bengio03a.pdf

What do we want?

- LM learning should not run into sparsity issues as context window increases
    - Distributed Representations can help address the problem
    - Low-dimensional representation of context
- Ability to capture long-distance dependency effects
    - Unbounded context

Recurrent Neural Networks: 2 Key Ideas

1. Summarize context information into a single vector



$$c(x_i) = F(x_1, x_2, \ldots, x_{i-1})$$

$$P(x_i | c(x_i))$$

Nature of $\boldsymbol{P}(.)$

n-gram LM: look-up table

FF LM: $c(x_i)$ = $G(x_{i-1}, x_{i-2})$ (trigram LM)

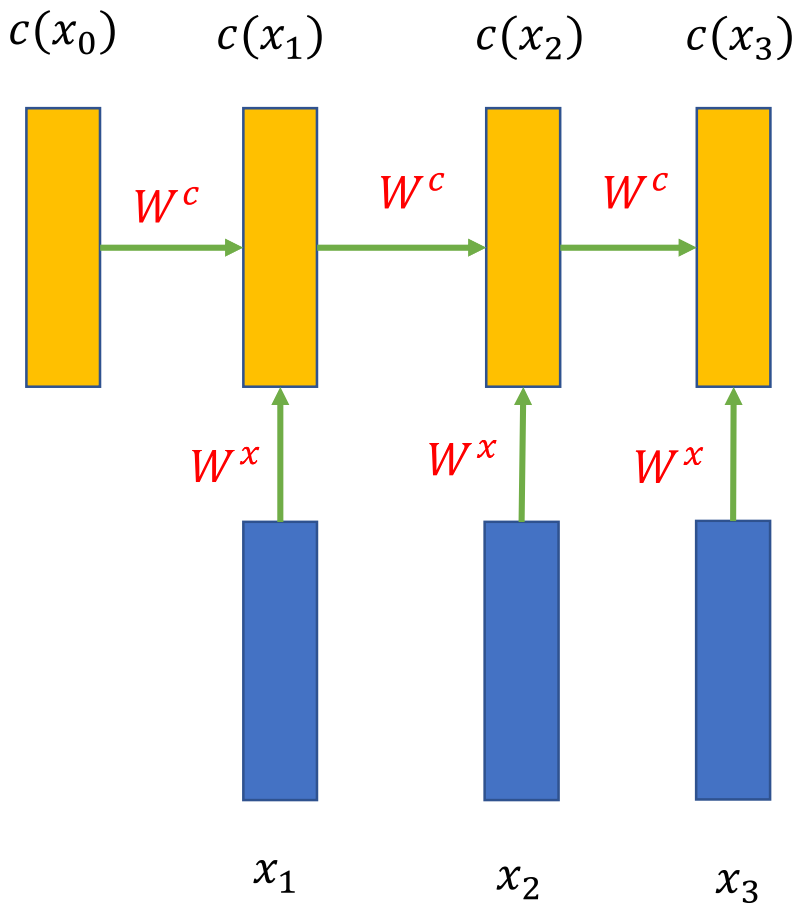
RNN LM: $c(x_i)$ = $F(x_1, x_2, \ldots, x_{i-1})$ (unbounded context)

Function G requires all context inputs at once

How does RNN address this problem?

2 Key Ideas

2. Recursively construct the context



$$c(x_i) = F(c(x_{i-1}), x_i)$$

We just need two inputs to construct the context vector:

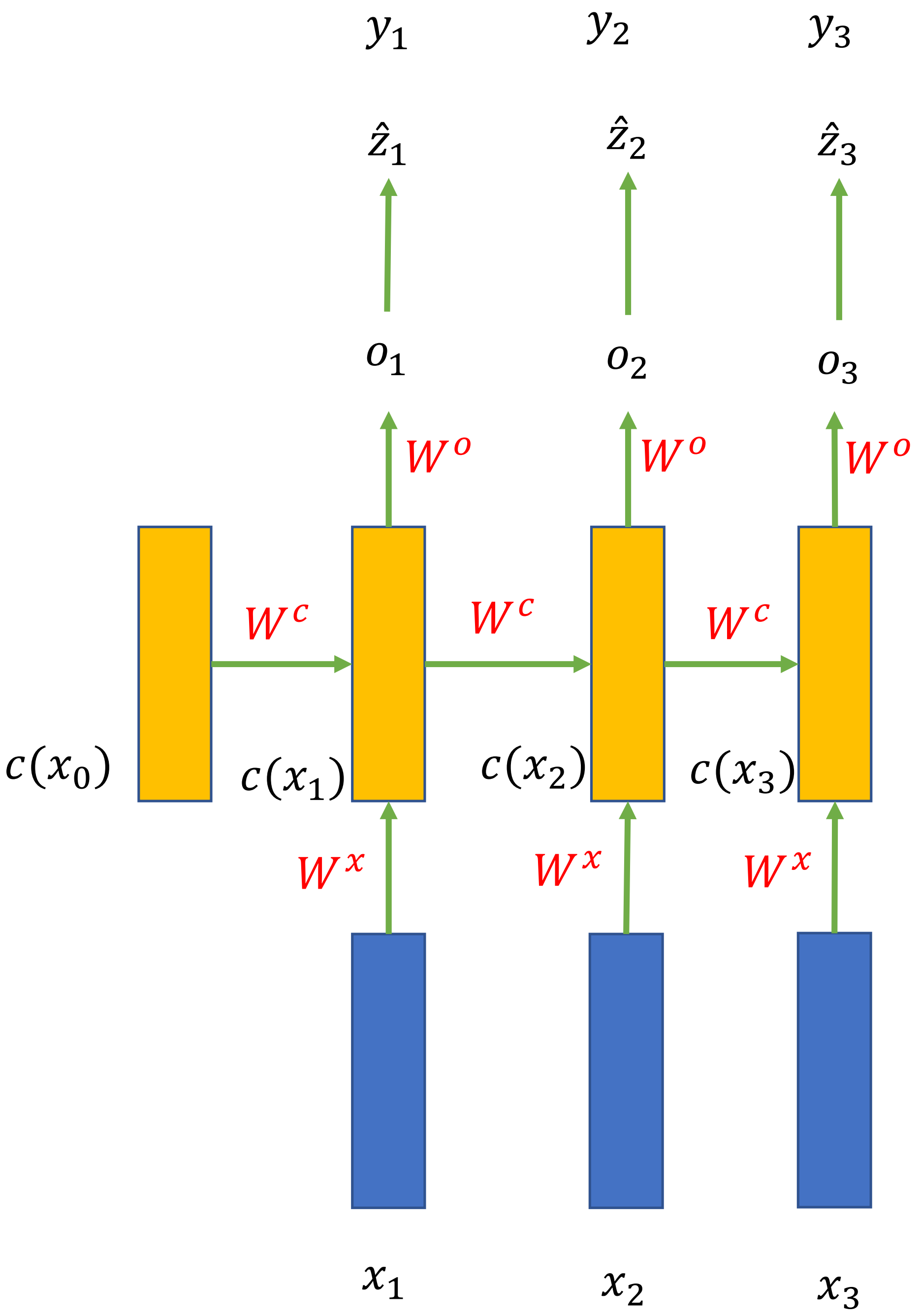
- Context vector of previous timestep
- Current input

The context vector ➔ state/hidden state/contextual representation

$F(.)$ can be implemented as

$$c(x_i) = \sigma(W^c c(x_{i-1}) + W^x x_i + b_1)$$

Like a feed-forward network



Generate output give the current input and state/context

$$o(x_i) = W^o c(x_i) + b_2$$

We are generally interested in categorical outputs

$$\hat{z}_i = softmax(o(x_i)) = P(y_i|ctx(x_i))$$

$$\widehat{z_i^w} = P(y_i = w|ctx(x_i))$$

The same parameters are used at each time-step

Model size does not depend on sequence length

Long range context is modeled

Sequence Labelling Task

Input Sequence: $(x_1 \;\; x_2 \;\; x_3 \;\; x_4 \ldots\ldots x_i \ldots\ldots x_N)$

Output Sequence: $(y_1 \;\; y_2 \;\; y_3 \;\; y_4 \ldots\ldots y_i \ldots\ldots y_N)$

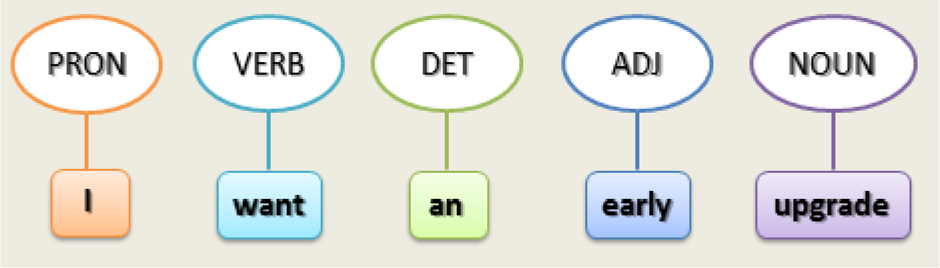
Input and output sequences have the same length

Variable length input

Output contains categorical labels

Output at any time-step typically depends on neighbouring output labels and input elements

Part-of-speech tagging



Recurrent Neural Network is a powerful model to learn sequence labelling tasks

How do we model language modeling as a sequence labeling task?

The	capital	of	Maharashtra	is	Mumbai	<EOS>
<BOS>	The	capital	of	Maharashtra	is	Mumbai

The output sequence is one-time step ahead of the input sequence

Training Language Models

Input: large monolingual corpus

- Each example is a tokenized sentence (sequence of words)
- At each time step, predict the distribution of the next word given all previous words
- Loss Function:
    - Minimize cross-entropy between actual distribution and predicted distribution
    - Equivalently, maximize the likelihood

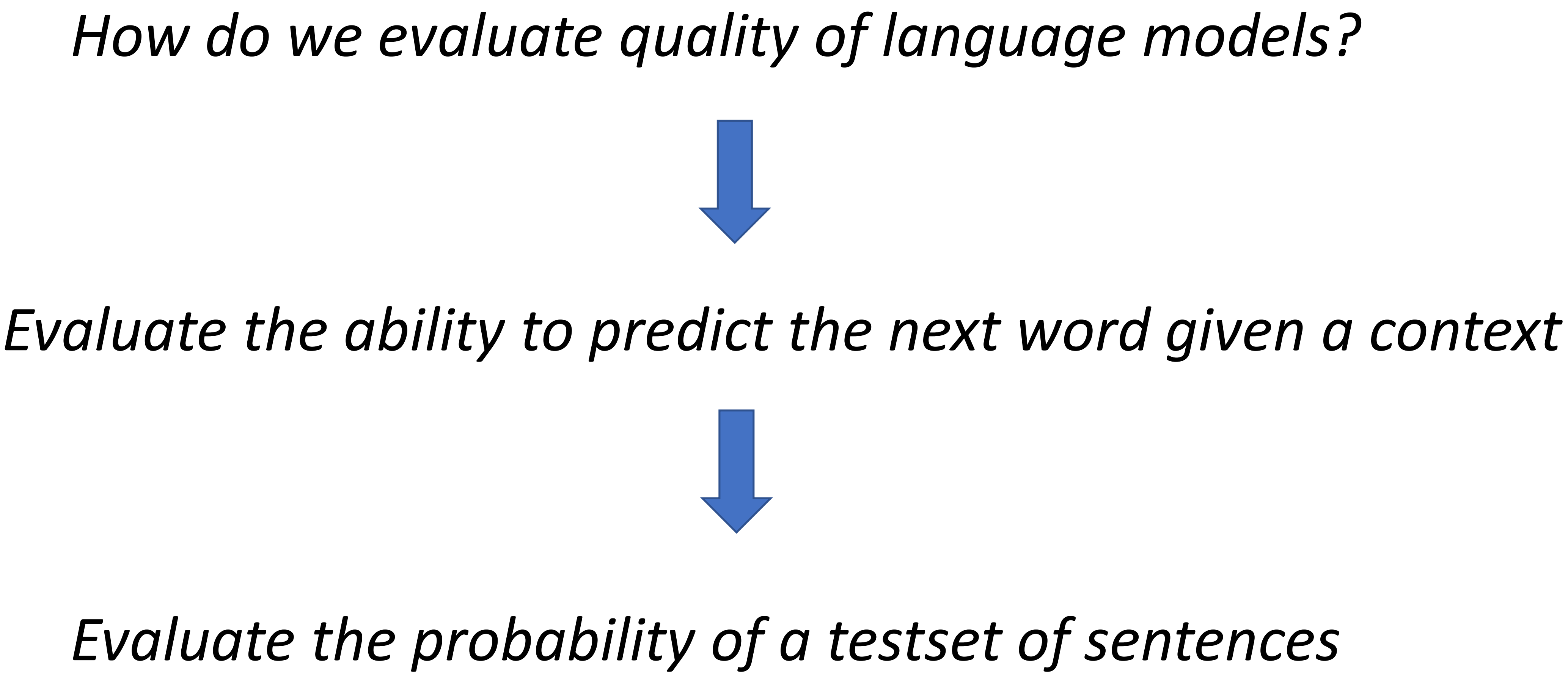
At a single time-step: $J_i(\theta) = CE(z_i, \hat{z}_i) = -\sum_{w \in V} z_i^w \log \widehat{z_i^w} = -\log \widehat{z_i^L}$

where $y_i = L$

Average over time steps for example n: $J^n(\theta) = \frac{1}{T}\sum_{i=1}^{T} J_i(\theta)$

Average over entire corpus: $J(\theta) = \frac{1}{N}\sum_{k=1}^{N} J^n(\theta)$

How do we learn model parameters? More on that later!



How do we evaluate quality of language models?

Evaluate the ability to predict the next word given a context

Evaluate the probability of a testset of sentences

Standard testsets exist for evaluating language models: Penn Treebank, Billion Word Corpus, WikiText

Language Model Perplexity

Perplexity: $\exp\big(J(\theta)\big)$

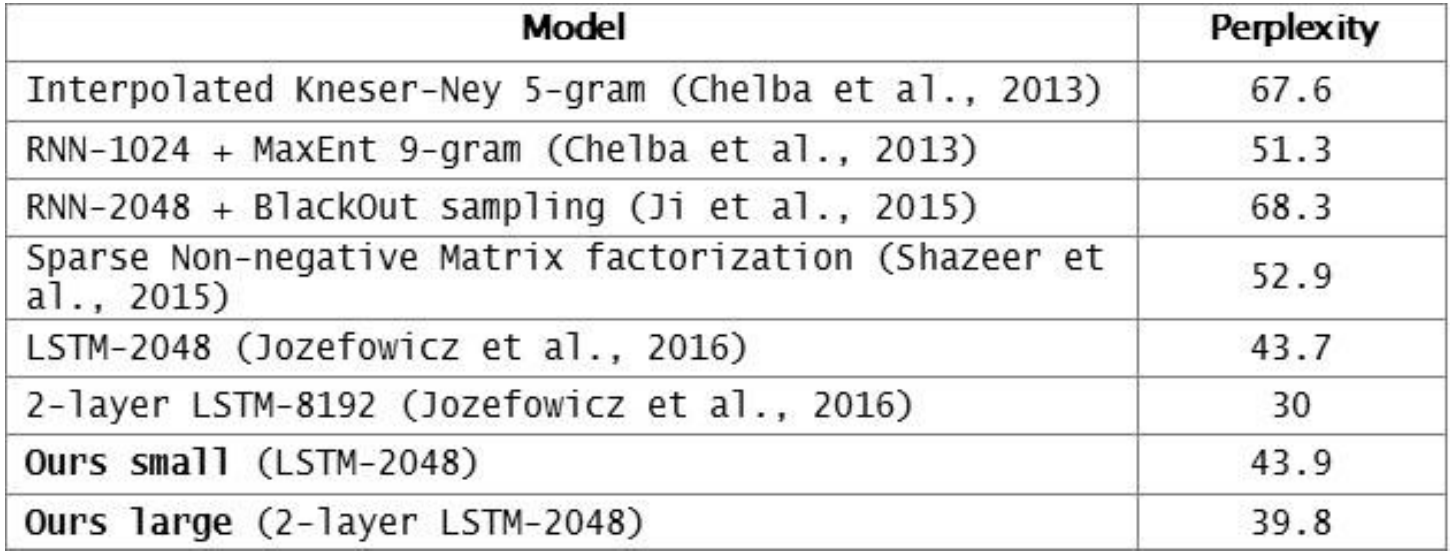
$J(\theta)$ is cross-entropy on the test set

Cross-entropy is measure of difference between actual and predicted distribution

Lower perplexity and cross-entropy is better

Training objective matches evaluation metric *n-gram*

RNN variants



Model	Perplexity
Interpolated Kneser-Ney 5-gram (Chelba et al., 2013)	67.6
RNN-1024 + MaxEnt 9-gram (Chelba et al., 2013)	51.3
RNN-2048 + BlackOut sampling (Ji et al., 2015)	68.3
Sparse Non-negative Matrix factorization (Shazeer et al., 2015)	52.9
LSTM-2048 (Jozefowicz et al., 2016)	43.7
2-layer LSTM-8192 (Jozefowicz et al., 2016)	30
Ours small (LSTM-2048)	43.9
Ours large (2-layer LSTM-2048)	39.8

https://research.fb.com/building-an-efficient-neural-language-model-over-a-billion-words/

RNN models outperform n-gram models

A special kind of RNN network – LSTM does even later ➔ we will see that soon

Generating text from a language model

We know the probability distribution ➔ $P(x_{i+1}|x_1\ \ x_2\ \ x_3\ \ x_4 \ldots\ldots x_i)$

At each step, sample a word as per the distribution

The capital of Maharashtra is ______

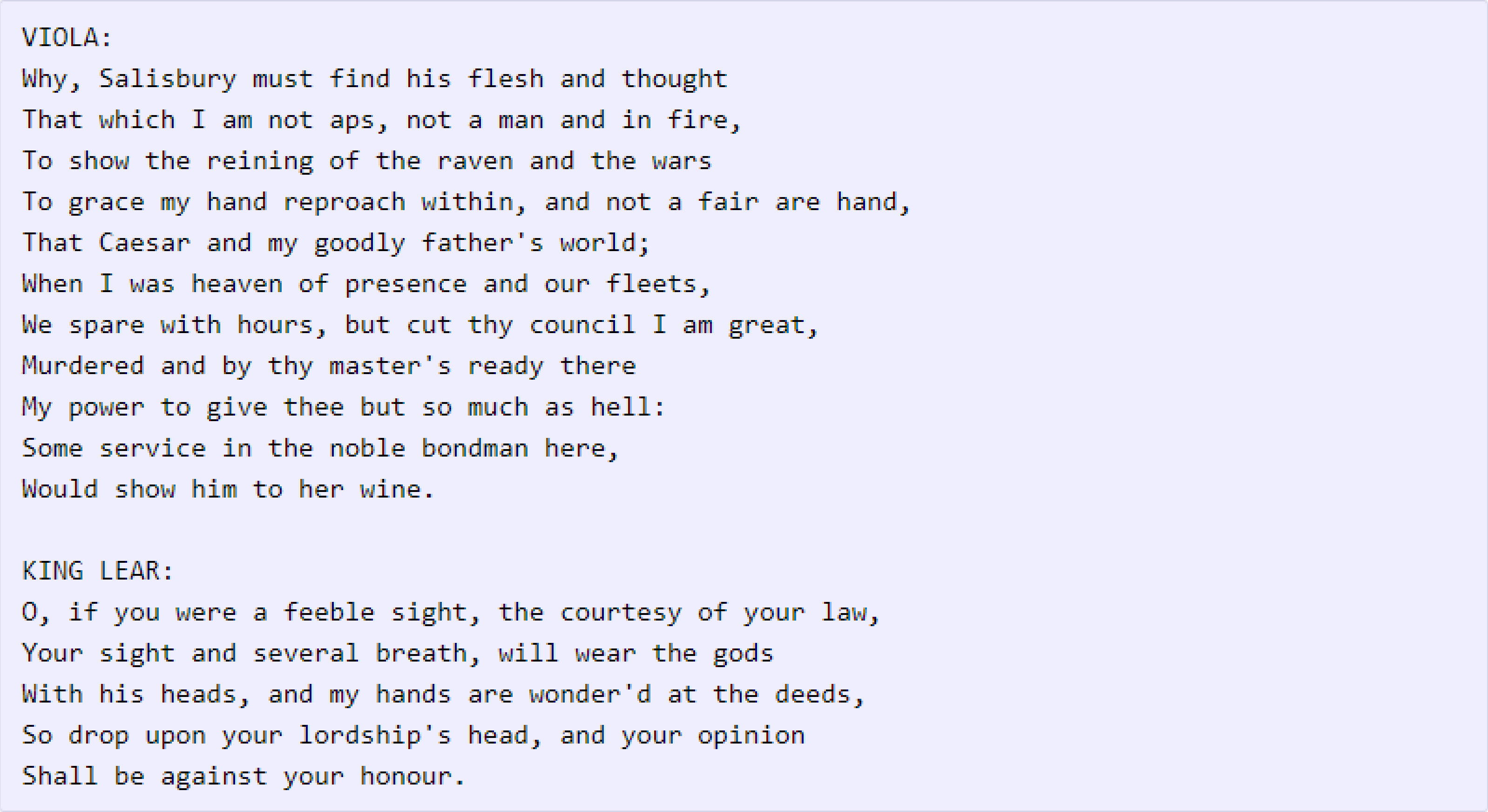
$$P(Mumbai|X) = 0.8$$

$$P(Bihar|X) = 0.1$$

$$P(Chennai|X) = 0.075$$

$$P(market|X) = 0.025$$

Examples of generated text

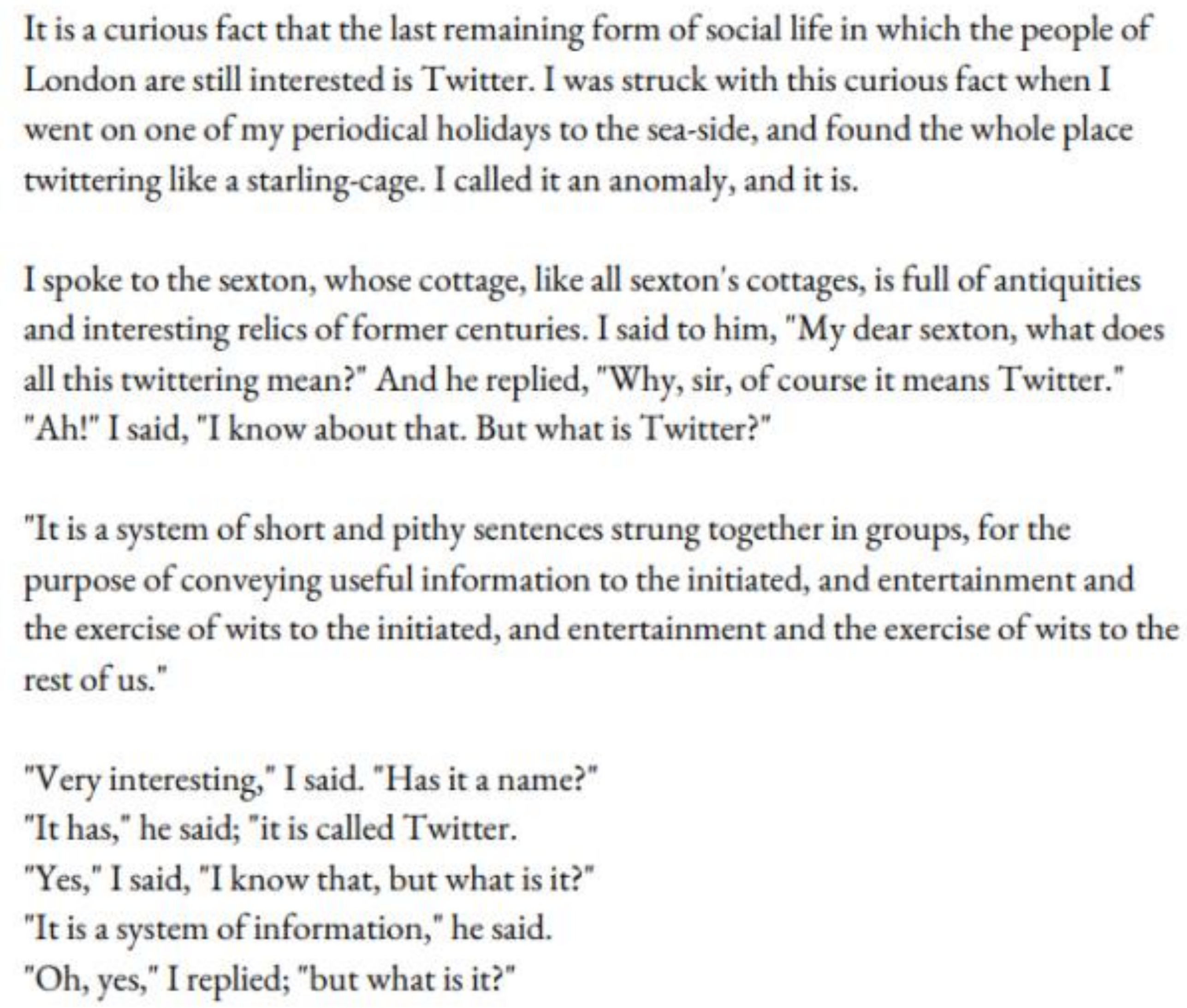


```
VIOLA:
Why, Salisbury must find his flesh and thought
That which I am not aps, not a man and in fire,
To show the reining of the raven and the wars
To grace my hand reproach within, and not a fair are hand,
That Caesar and my goodly father's world;
When I was heaven of presence and our fleets,
We spare with hours, but cut thy council I am great,
Murdered and by thy master's ready there
My power to give thee but so much as hell:
Some service in the noble bondman here,
Would show him to her wine.

KING LEAR:
O, if you were a feeble sight, the courtesy of your law,
Your sight and several breath, will wear the gods
With his heads, and my hands are wonder'd at the deeds,
So drop upon your lordship's head, and your opinion
Shall be against your honour.
```

By a character level RNN-LM trained on Shakespeare’s works

http://karpathy.github.io/2015/05/21/rnn-effectiveness/



It is a curious fact that the last remaining form of social life in which the people of London are still interested is Twitter. I was struck with this curious fact when I went on one of my periodical holidays to the sea-side, and found the whole place twittering like a starling-cage. I called it an anomaly, and it is.

I spoke to the sexton, whose cottage, like all sexton's cottages, is full of antiquities and interesting relics of former centuries. I said to him, "My dear sexton, what does all this twittering mean?" And he replied, "Why, sir, of course it means Twitter."
"Ah!" I said, "I know about that. But what is Twitter?"

"It is a system of short and pithy sentences strung together in groups, for the purpose of conveying useful information to the initiated, and entertainment and the exercise of wits to the initiated, and entertainment and the exercise of wits to the rest of us."

"Very interesting," I said. "Has it a name?"
"It has," he said; "it is called Twitter.
"Yes," I said, "I know that, but what is it?"
"It is a system of information," he said.
"Oh, yes," I replied; "but what is it?"

Generated by GPT-3 after being prompted by the first word

Note: GPT-3 is not an RNN-LM

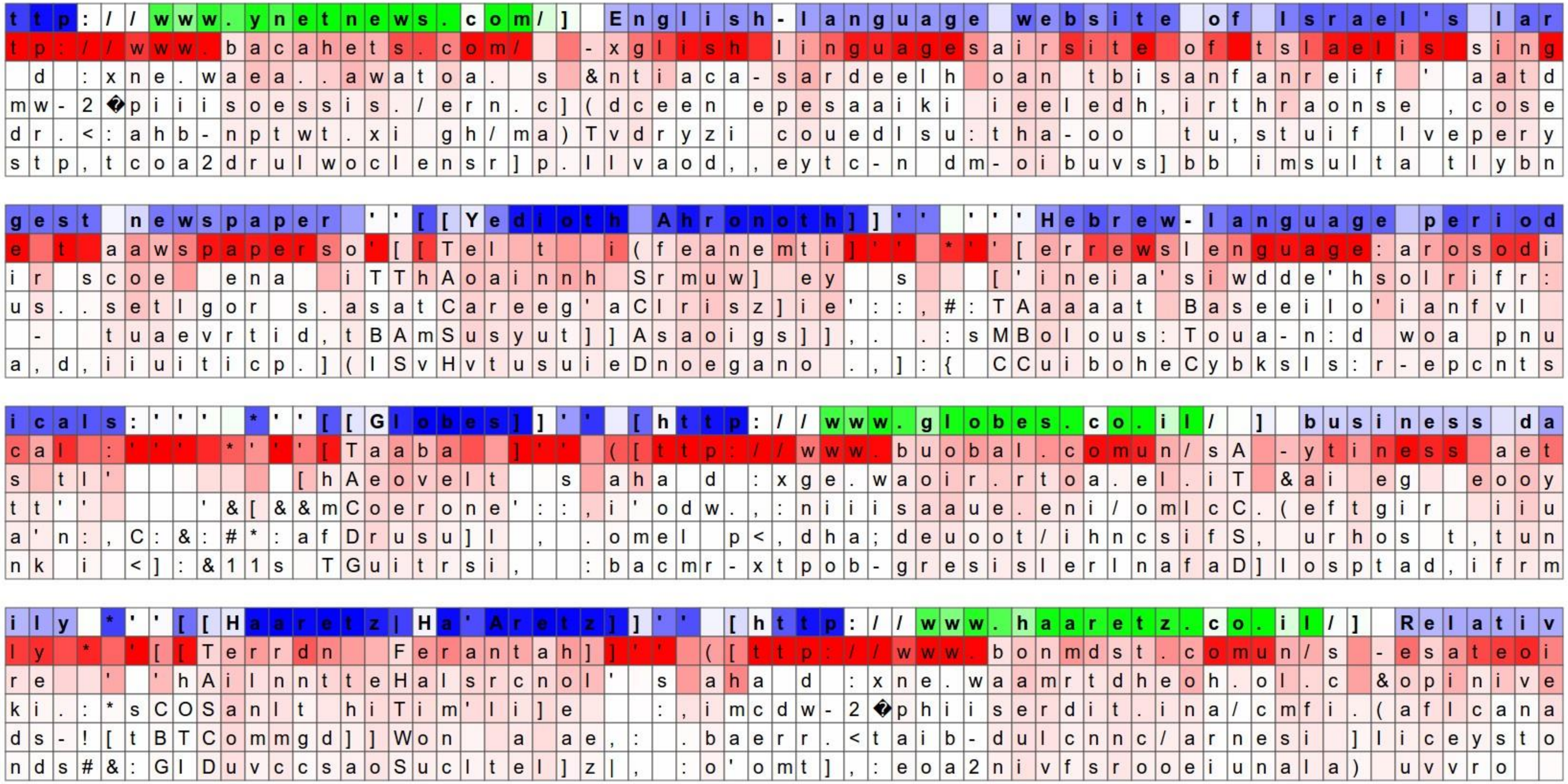
Neural networks have enabled generation of very fluent text

Led to a revolution in natural language generation

Enabled huge advances in natural language understanding

https://www.technologyreview.com/2020/07/20/1005454/openai-machine-learning-language-generator-gpt-3-nlp/

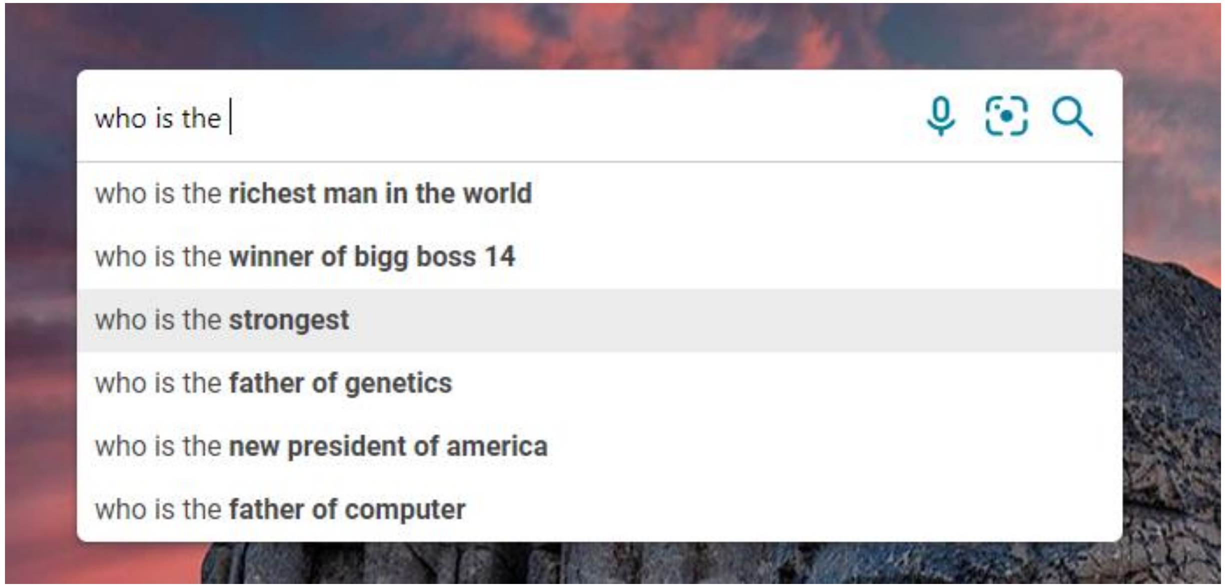
Visualizations



Activation of a neuron in the LSTM hidden layer indicate it fires for URLs

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Language models for auto-suggest



who is the

who is the **richest man in the world**

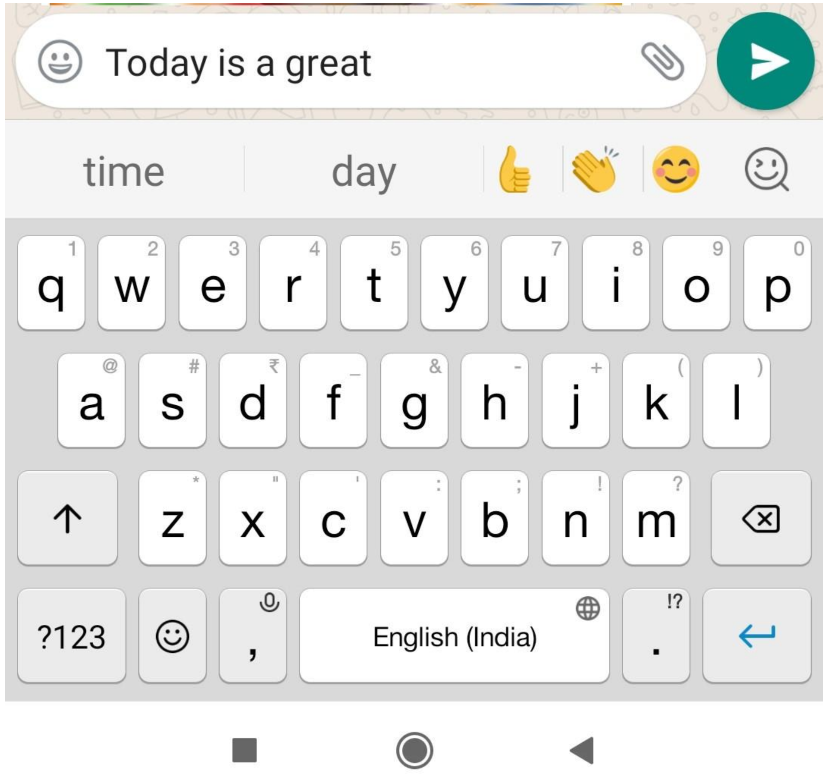
who is the **winner of bigg boss 14**

who is the **strongest**

who is the **father of genetics**

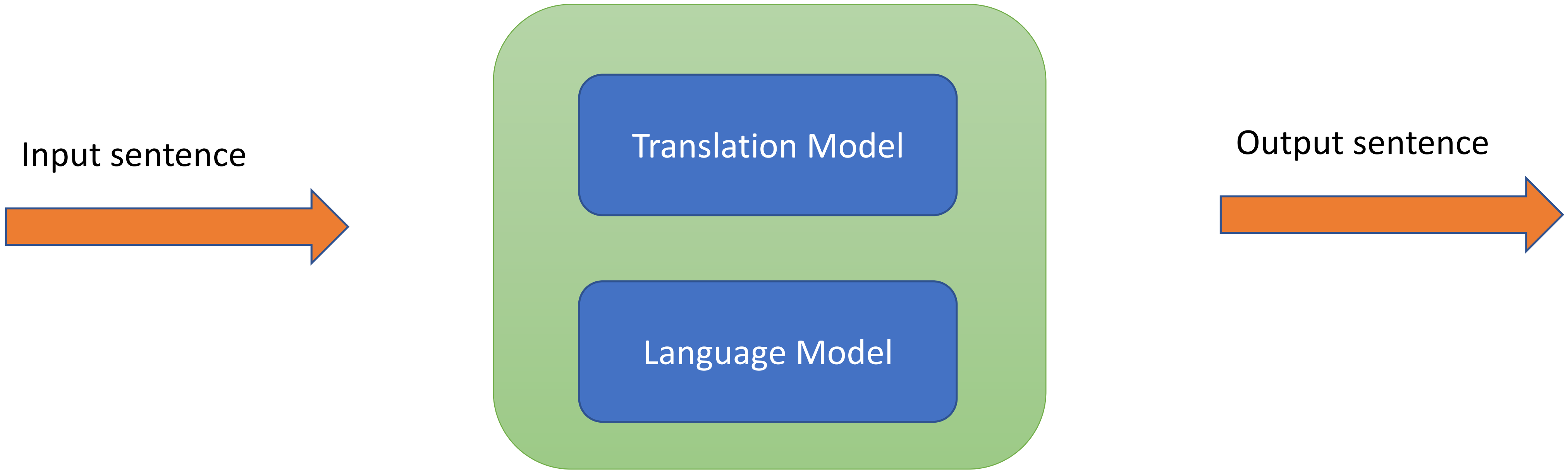
who is the **new president of america**

who is the **father of computer**



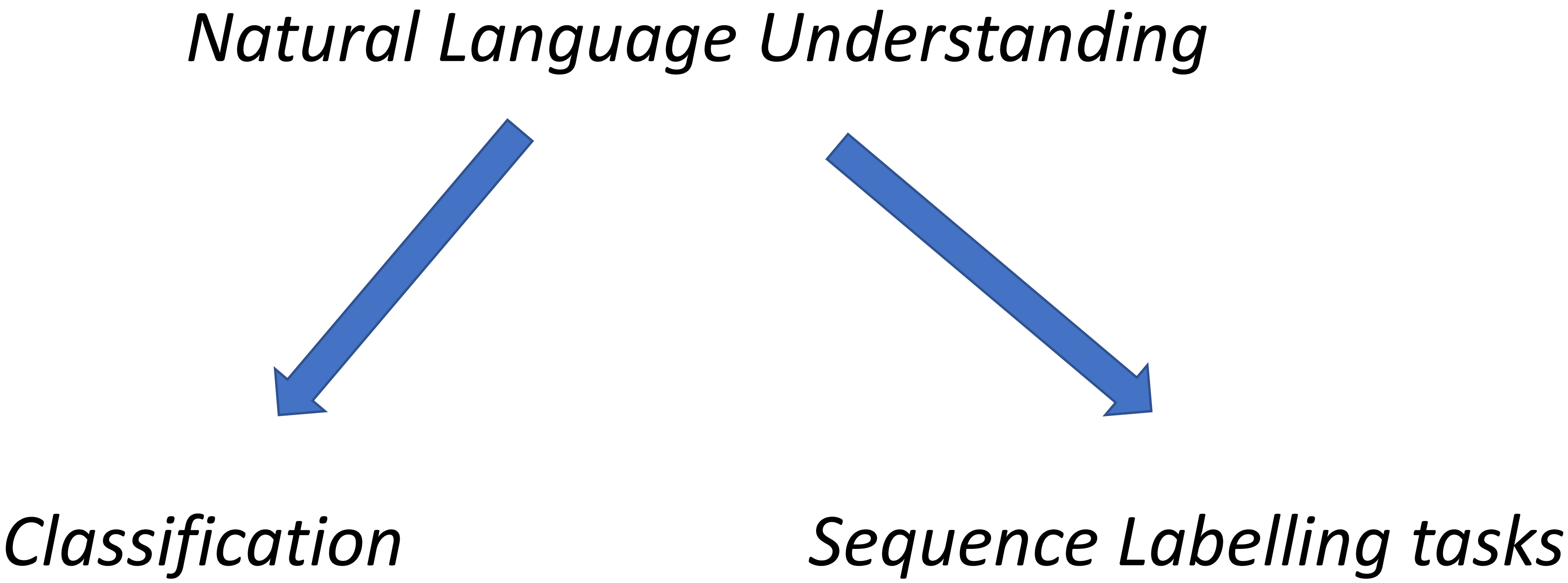
Today is a great

Language models to score outputs from generation systems



Automatic Speech recognition systems combine language models with acoustic models

Transliteration models combine character mapping models with character level language model



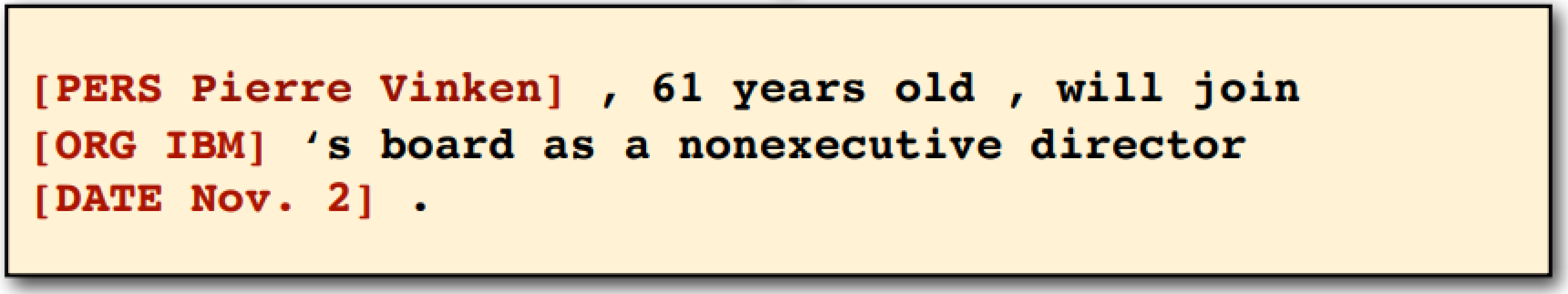
Natural Language Understanding

Classification

Sequence Labelling tasks

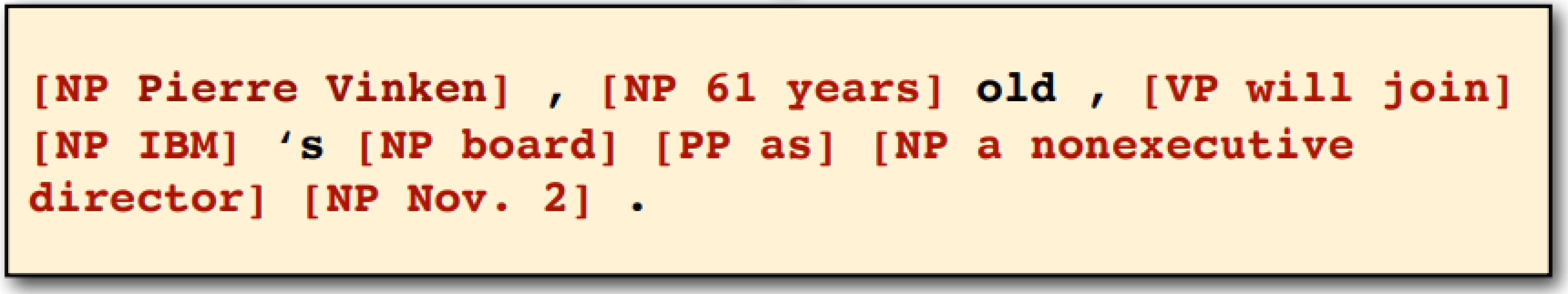
Sequence Labelling Tasks

Named Entity Recognition



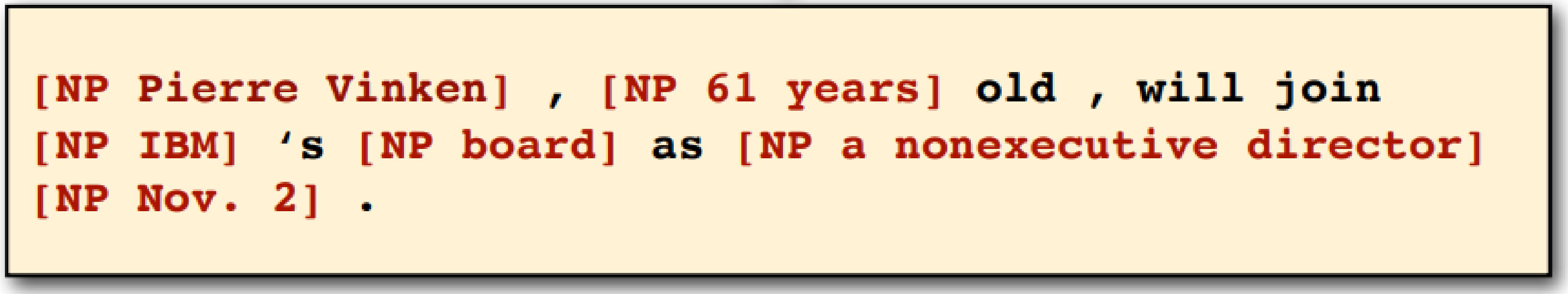
[PERS Pierre Vinken] , 61 years old , will join
[ORG IBM] 's board as a nonexecutive director
[DATE Nov. 2] .

Shallow Parsing



[NP Pierre Vinken] , [NP 61 years] old , [VP will join]
[NP IBM] 's [NP board] [PP as] [NP a nonexecutive
director] [NP Nov. 2] .

NP Chunking



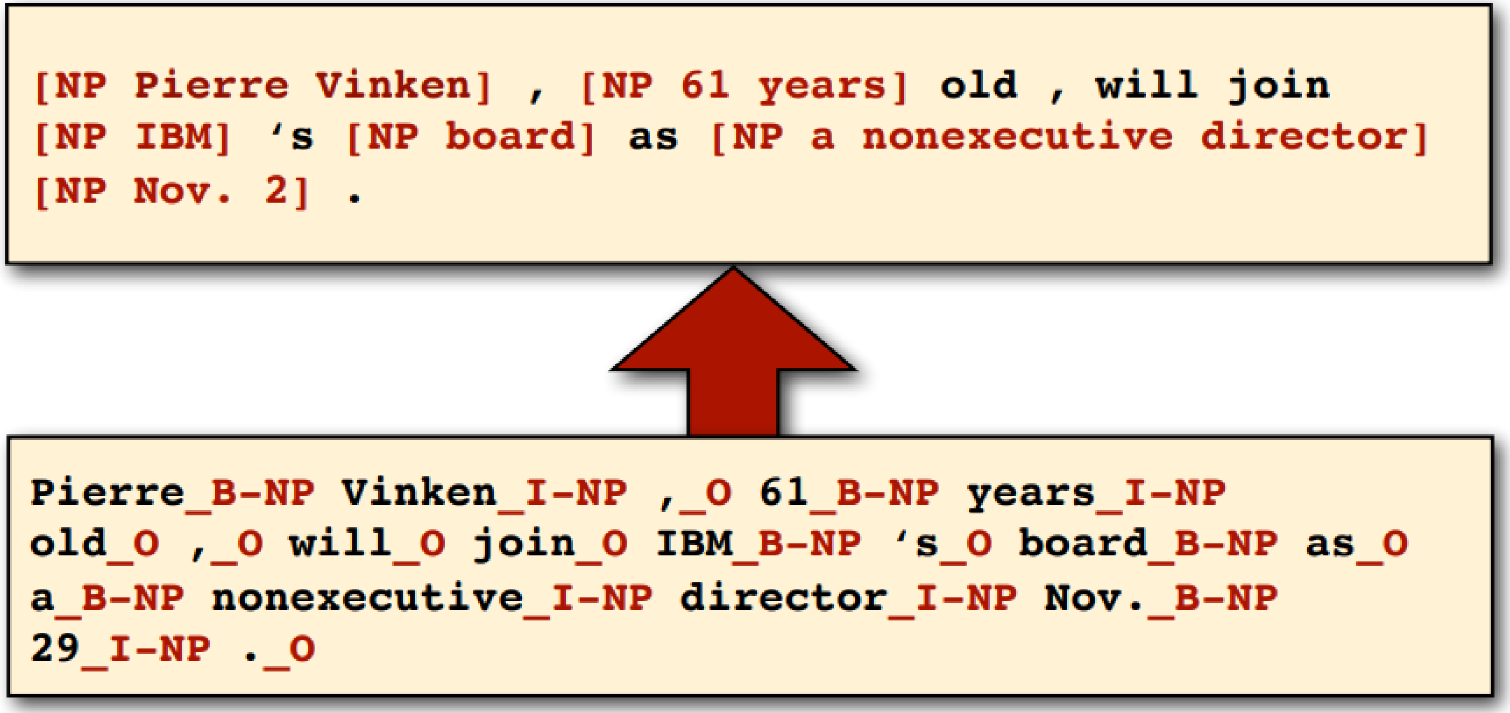
[NP Pierre Vinken] , [NP 61 years] old , will join
[NP IBM] 's [NP board] as [NP a nonexecutive director]
[NP Nov. 2] .

https://courses.engr.illinois.edu/cs447/fa2018/Slides/Lecture07.pdf

The BIO encoding

We define three new tags:

– **B-NP**: beginning of a noun phrase chunk
– **I-NP**: inside of a noun phrase chunk
– **O**: outside of a noun phrase chunk



```
[NP Pierre Vinken] , [NP 61 years] old , will join
[NP IBM] ‘s [NP board] as [NP a nonexecutive director]
[NP Nov. 2] .
```

```
Pierre_B-NP Vinken_I-NP ,_O 61_B-NP years_I-NP
old_O ,_O will_O join_O IBM_B-NP ‘s_O board_B-NP as_O
a_B-NP nonexecutive_I-NP director_I-NP Nov._B-NP
29_I-NP ._O
```

https://courses.engr.illinois.edu/cs447/fa2018/Slides/Lecture07.pdf

Inference in sequence labelling tasks

We know the probability distribution ➔ $P(y_i|x_1 \ \ x_2 \ \ x_3 \ \ x_4 \ldots\ldots x_i)$

At each step, sample a word as per the distribution

$P(LOC|Mumbai) = 0.65$

The capital of Maharashtra is Mumbai

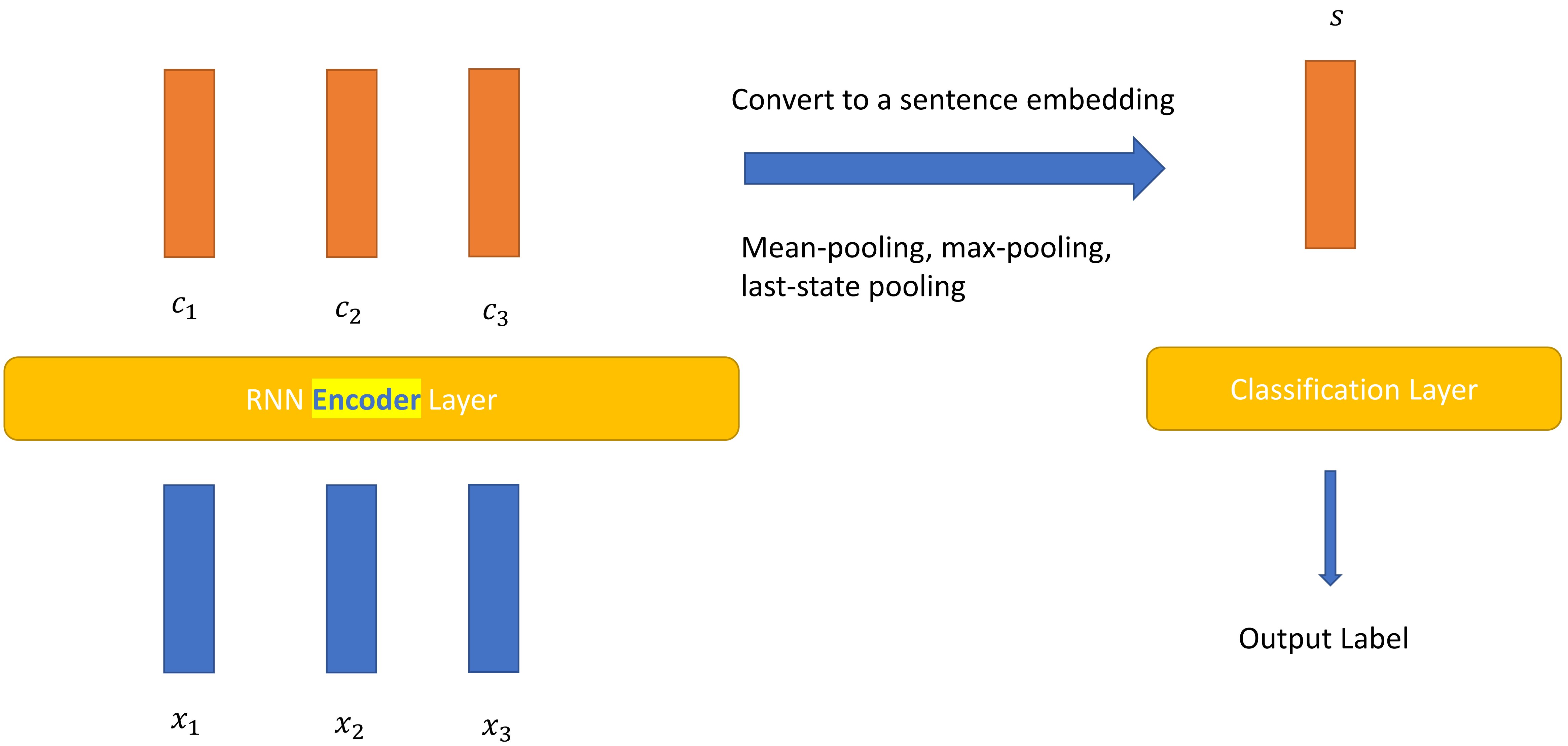
$P(ORG|Mumbai) = 0.3$

$P(PER|Mumbai) = 0.05$

Greedy decoding or beam search decoding ➔ *more on this in the next lecture*

Classification Tasks

Sentiment classification, hate speech detection, etc.



Convert to a sentence embedding

Mean-pooling, max-pooling, last-state pooling

Language Model Pre-training

Do we have to train the RNN parameters for every task like sentiment analysis, POS tagging?

➔ *we may not have enough training data for a task*

➔ *training data for a task will not capture all linguistic knowledge*

Train a large language model on lots of text data ➔ *pre-trained encoder module*

For a particular task ➔ *add classification heads (and optional layers) on top*

Finetune the network for the task

Many models like: ELMO, ULMFit

Sequence to Sequence Tasks

More about these in the next lectures

Input and output sequences of different lengths

Machine Translation, Machine Transliteration, Summarization, etc.

Can be solved with conditional RNNs

Back to Training Recurrent Neural Models

Brief sketch of Backpropagation Through Time (BPTT)

Training Objective revisited

At a single time-step i: $J_i(\theta) =- -\log \widehat{z_i^L}$ where $y_i = L$

Average over time steps for example : $J(\theta) = \sum_{i=1}^{T} J_i(\theta)$

Gradient Computation

$$\frac{\partial J(\theta)}{\partial \theta} = \sum_{i=1}^{T} \frac{\partial J_i(\theta)}{\partial \theta}$$

Model Parameters: $\theta = (W^x, W^c, W^o, b_1, b_2)$

$$\frac{\partial J_i(\theta)}{\partial W^o}$$

$J_i(\theta)$ is a direct function of W^o, so it is easy to compute

Let's focus on gradients w.r.t to W^c

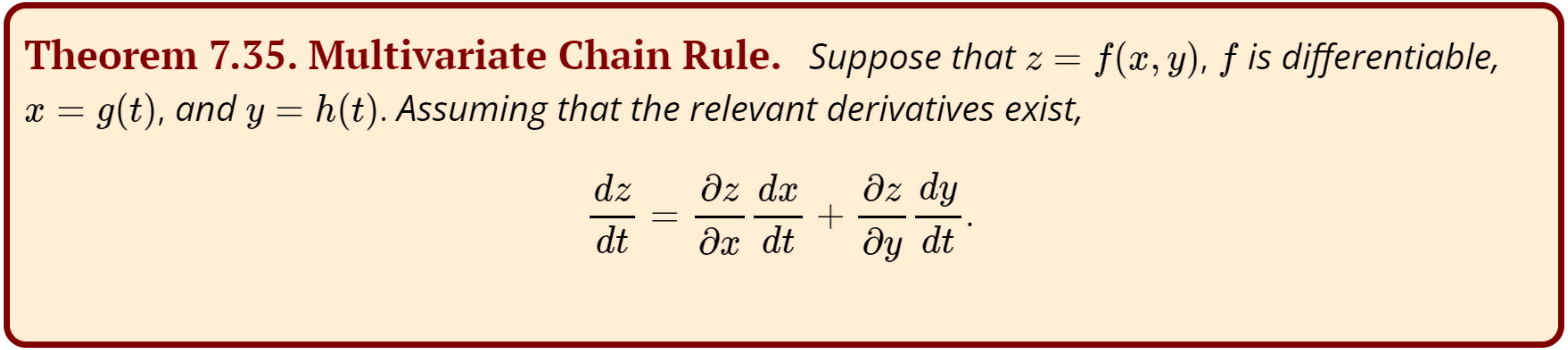
$$\frac{\partial J_i(\theta)}{\partial W^c} = \frac{\partial J_i(\theta)}{\partial c_i} \frac{\partial c_i}{\partial W^c}$$

c_i is a function of $W^c$and c_{i-1}

In turn, c_{i-1} is a function of $W^c$and c_{i-2} and so on ….

$$\frac{\partial c_i}{\partial W^c} = \frac{\partial^+ c_i}{\partial W^c} + \frac{\partial c_i}{\partial c_{i-1}} \frac{\partial c_{i-1}}{\partial W^c}$$

Multivariate chain rule



Theorem 7.35. Multivariate Chain Rule. *Suppose that $z = f(x, y)$, f is differentiable, $x = g(t)$, and $y = h(t)$. Assuming that the relevant derivatives exist,*

$$\frac{dz}{dt} = \frac{\partial z}{\partial x}\frac{dx}{dt} + \frac{\partial z}{\partial y}\frac{dy}{dt}.$$

$c_i = f(W^c, c_{i-1})$ Plug in $z = c_i, x = W^c, y = c_{i-1}$

$$\frac{\partial c_i}{\partial W^c} = \frac{\partial^+ c_i}{\partial W^c} + \frac{\partial c_i}{\partial c_{i-1}} \color{red}{\frac{\partial c_{i-1}}{\partial W^c}}$$

$\frac{\partial^+ c_i}{\partial W^c}$ ➔ explicit derivative term computed assuming C_{i-1} to be constant

$$\frac{\partial c_i}{\partial W^c} = \frac{\partial^+ c_i}{\partial W^c} + \frac{\partial c_i}{\partial c_{i-1}}\left[\frac{\partial^+ c_{i-1}}{\partial W^c} + \frac{\partial c_{i-1}}{\partial c_{i-2}}\frac{\partial c_{i-2}}{\partial W^c}\right]$$

Because of the recurrence, the gradient of a hidden state at time i

depends on gradients of all hidden states before timestep i

$$\frac{\partial c_i}{\partial W^c} = \sum_{k=1}^{i} \frac{\partial c_i}{\partial c_k}\frac{\partial^+ c_k}{\partial W^c}$$

$$\frac{\partial J_i(\theta)}{\partial W^c} = \frac{\partial J_i(\theta)}{\partial c_i}\sum_{k=1}^{i} \frac{\partial c_i}{\partial c_k}\frac{\partial^+ c_k}{\partial W^c}$$

Backpropagation needs to go over all previous time-steps

➔ backpropagation through time (BPTT)

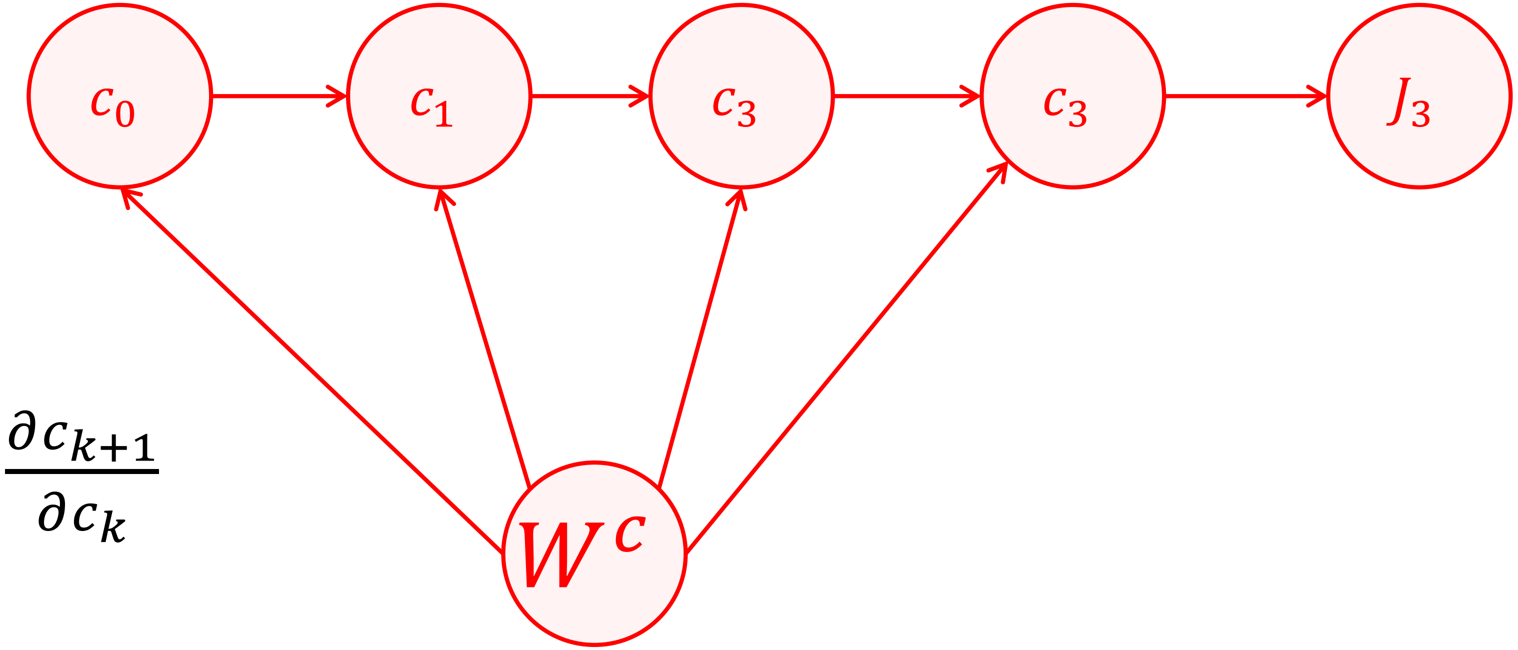
See https://www.cse.iitm.ac.in/~miteshk/CS7015/Slides/Teaching/pdf/Lecture14.pdf for detailed explanation

Vanishing and exploding gradient

Let's look at this particular component of the gradient computation ➔ there is a problem here

$$\frac{\partial c_i}{\partial W^c} = \sum_{k=1}^{i-1} \frac{\partial c_i}{\partial c_k} \frac{\partial^+ c_k}{\partial W^c}$$

$$\frac{\partial c_i}{\partial c_k} = \frac{\partial c_i}{\partial c_{i-1}} \frac{\partial c_{i-1}}{\partial c_{i-2}} \frac{\partial c_{i-2}}{\partial c_{i-3}} \cdots \frac{\partial c_{k+1}}{\partial c_k}$$



Each Path from W^c to J constitutes a term in the summation,
Each edge a path constitutes one element in the product term

https://www.cse.iitm.ac.in/~miteshk/CS7015/Slides/Teaching/pdf/Lecture14.pdf

$$\frac{\partial c_i}{\partial c_k} = \frac{\partial c_i}{\partial c_{i-1}} \frac{\partial c_{i-1}}{\partial c_{i-2}} \frac{\partial c_{i-2}}{\partial c_{i-3}} \cdots \frac{\partial c_{k+1}}{\partial c_k}$$

We can show some bounds on this quantity

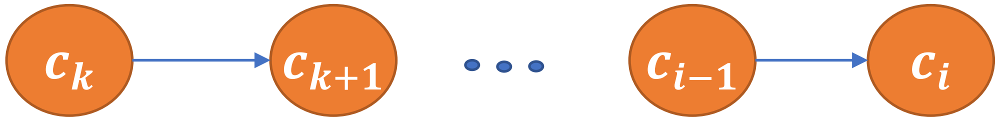
$$\left\|\frac{\partial c_i}{\partial c_{i-1}}\right\| \le \gamma\lambda \quad \Longrightarrow \quad \left\|\frac{\partial c_i}{\partial c_k}\right\| \le (\gamma\lambda)^{i-k}$$

$\gamma\lambda < 1$ ➔ vanishing gradient
$\gamma\lambda > 1$ ➔ exploding gradient

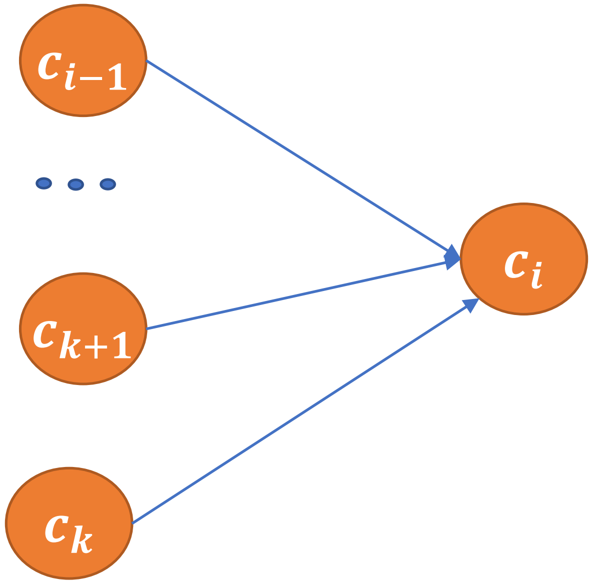
Vanishing Gradient Problem ➔ Intuition

- The product of these quantities can become small if the some of these quantities are small
- Gradient signal can become small for positions that are farther away from a position under consideration ➔ can’t learn long term dependency
- Truncated BPTT ➔ restrict the product to fewer terms
- *Exploding Gradient:* Gradient can also become large ➔ clip the gradient before parameter update

Another way to address the vanishing gradient problem



c_k can only indirectly impact c_i ➔ what if it could have a direct impact?



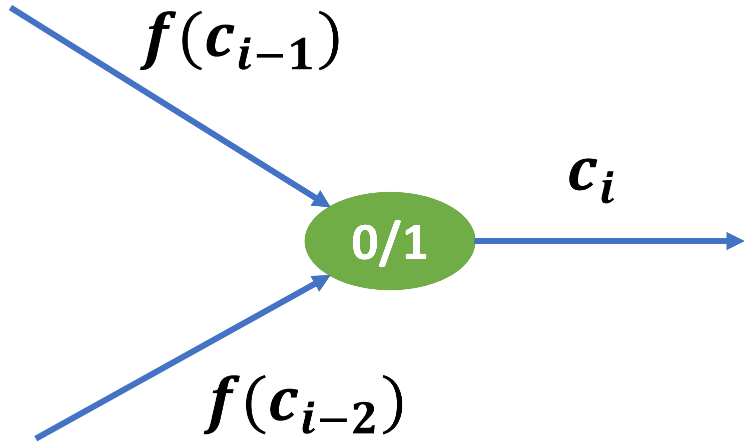
Looks a lot like the feed-forward solution

Cannot handle unbounded context

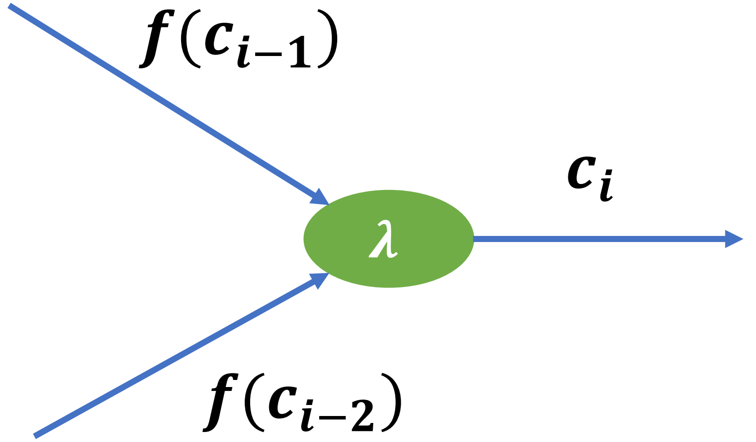
Not a good solution

Selecting hidden states that affect current hidden state

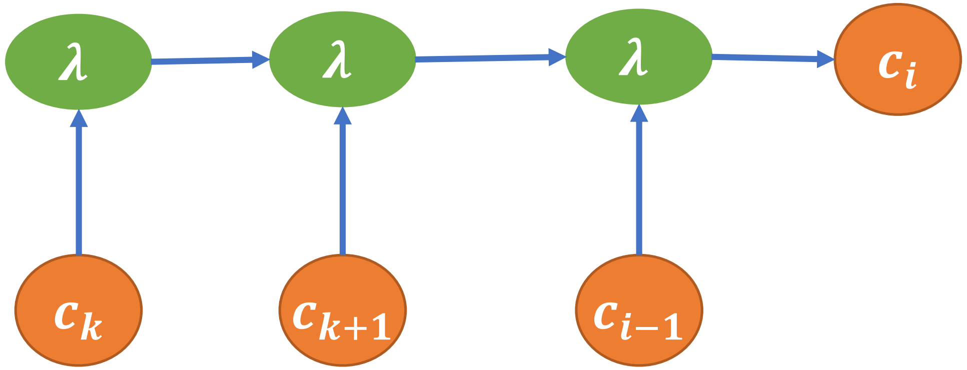
What if there was a switch to select which previous hidden state should impact current hidden state?



Better still, we had a soft-switch



If the switches are set correctly, then distant hidden states can have a major impact on current state



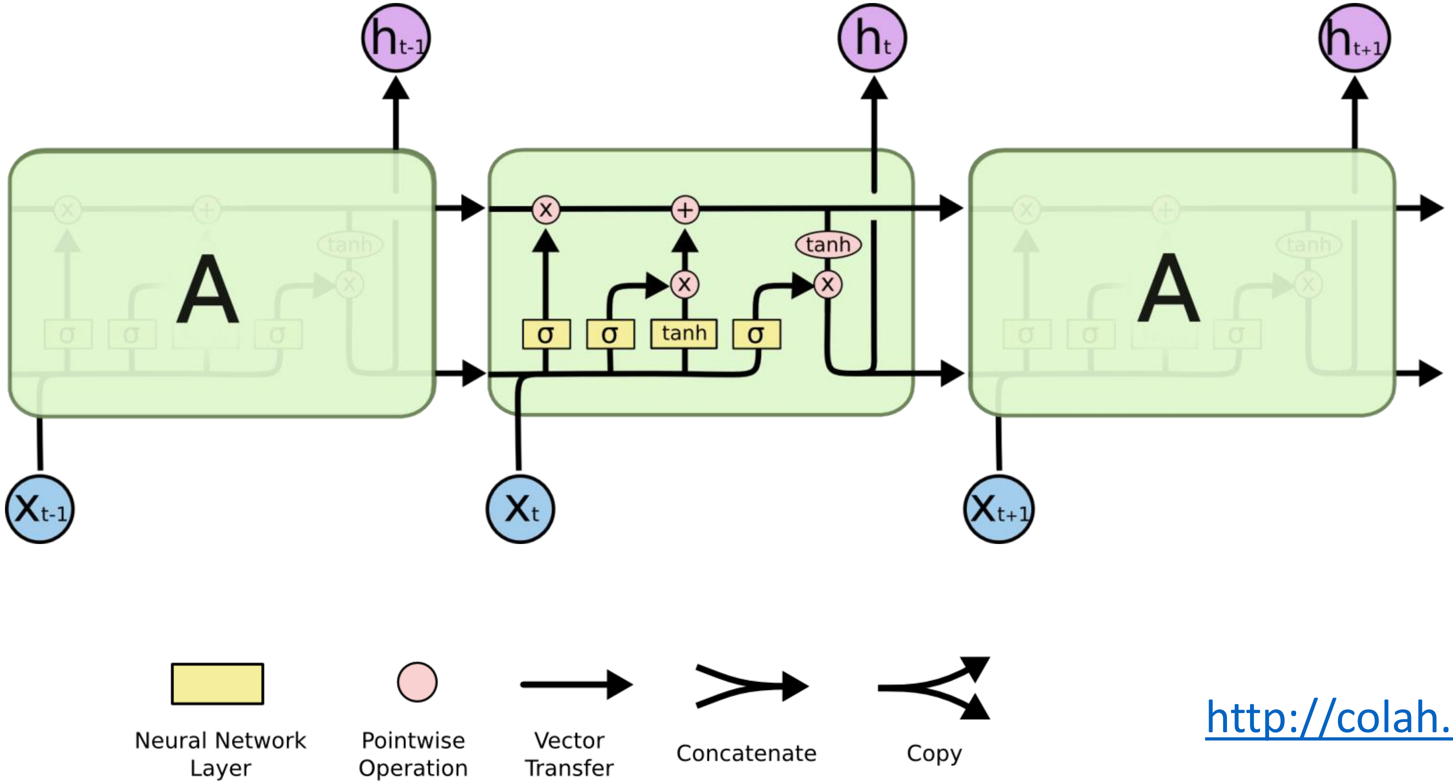
The switches should be set based on the data flowing in.

Switches act a memory control mechanism to decide what affects the output

Long Short-Term Memory (LSTM)

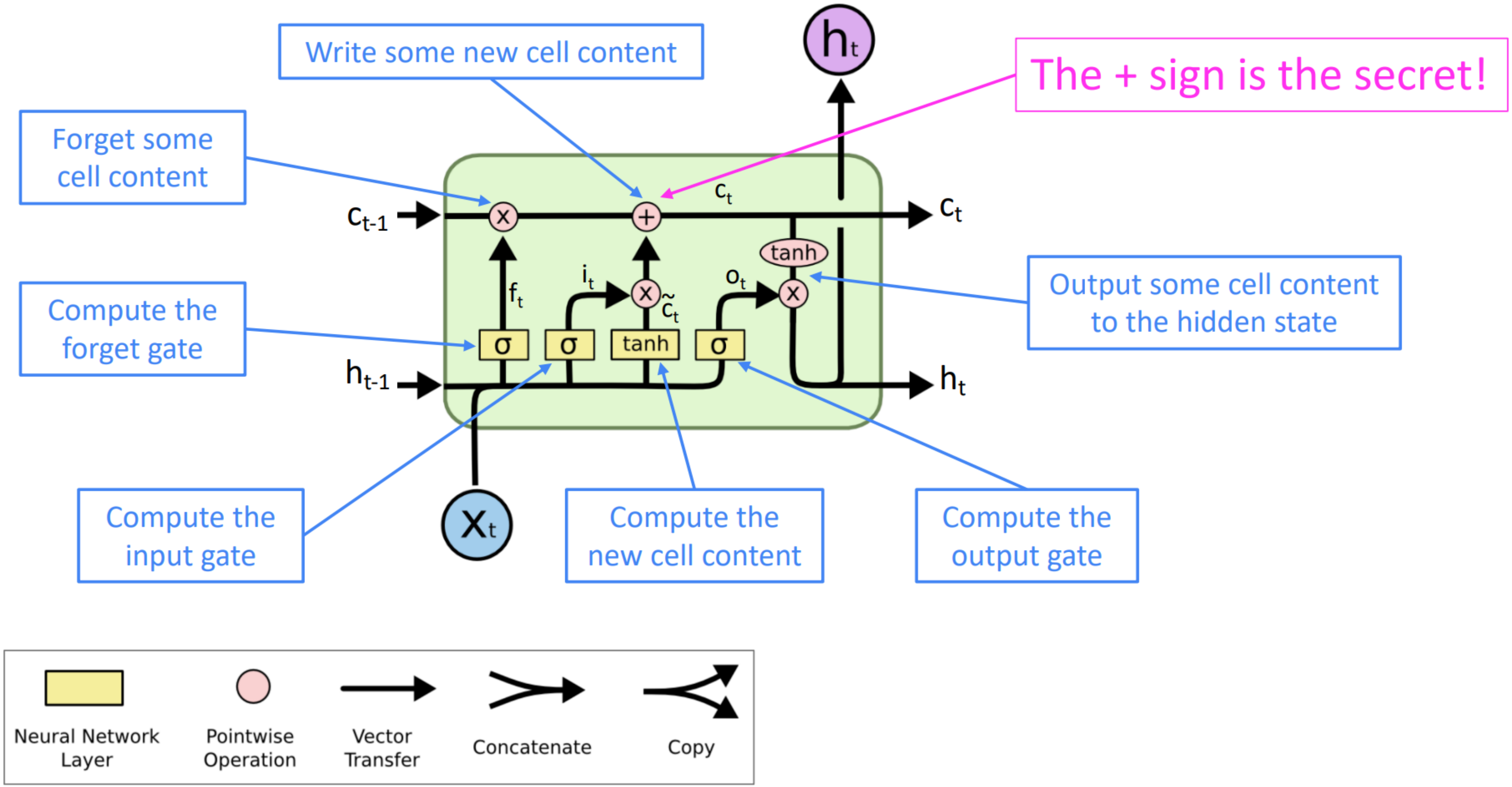
Formalization of the ideas just discussed

A special kind of RNN-cell that enables selective read/write/erasure



Different gates (shown with crosses)control flow of information

http://colah.github.io/posts/2015-08-Understanding-LSTMs



Write some new cell content

The + sign is the secret!

Forget some cell content

Compute the forget gate

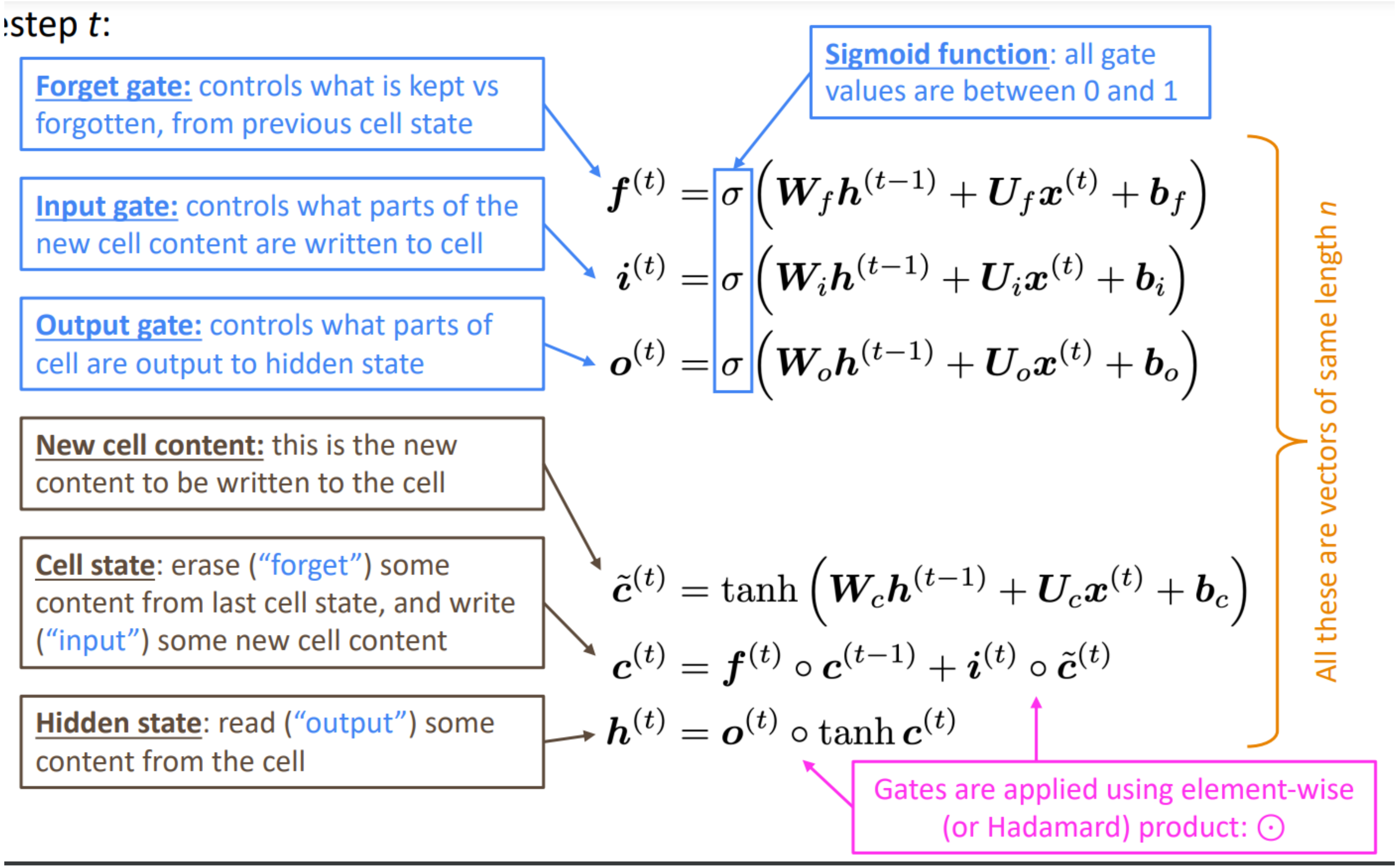
Output some cell content to the hidden state

Compute the input gate

Compute the new cell content

Compute the output gate

http://web.stanford.edu/class/cs224n/slides/cs224n-2021-lecture06-fancy-rnn.pdf



step *t*:

Sigmoid function: all gate values are between 0 and 1

Forget gate: controls what is kept vs forgotten, from previous cell state

Input gate: controls what parts of the new cell content are written to cell

Output gate: controls what parts of cell are output to hidden state

$$\boldsymbol{f}^{(t)} = \boxed{\sigma}\left(\boldsymbol{W}_f\boldsymbol{h}^{(t-1)} + \boldsymbol{U}_f\boldsymbol{x}^{(t)} + \boldsymbol{b}_f\right)$$

$$\boldsymbol{i}^{(t)} = \sigma\left(\boldsymbol{W}_i\boldsymbol{h}^{(t-1)} + \boldsymbol{U}_i\boldsymbol{x}^{(t)} + \boldsymbol{b}_i\right)$$

$$\boldsymbol{o}^{(t)} = \sigma\left(\boldsymbol{W}_o\boldsymbol{h}^{(t-1)} + \boldsymbol{U}_o\boldsymbol{x}^{(t)} + \boldsymbol{b}_o\right)$$

New cell content: this is the new content to be written to the cell

Cell state: erase ("forget") some content from last cell state, and write ("input") some new cell content

Hidden state: read ("output") some content from the cell

$$\tilde{\boldsymbol{c}}^{(t)} = \tanh\left(\boldsymbol{W}_c\boldsymbol{h}^{(t-1)} + \boldsymbol{U}_c\boldsymbol{x}^{(t)} + \boldsymbol{b}_c\right)$$

$$\boldsymbol{c}^{(t)} = \boldsymbol{f}^{(t)} \circ \boldsymbol{c}^{(t-1)} + \boldsymbol{i}^{(t)} \circ \tilde{\boldsymbol{c}}^{(t)}$$

$$\boldsymbol{h}^{(t)} = \boldsymbol{o}^{(t)} \circ \tanh \boldsymbol{c}^{(t)}$$

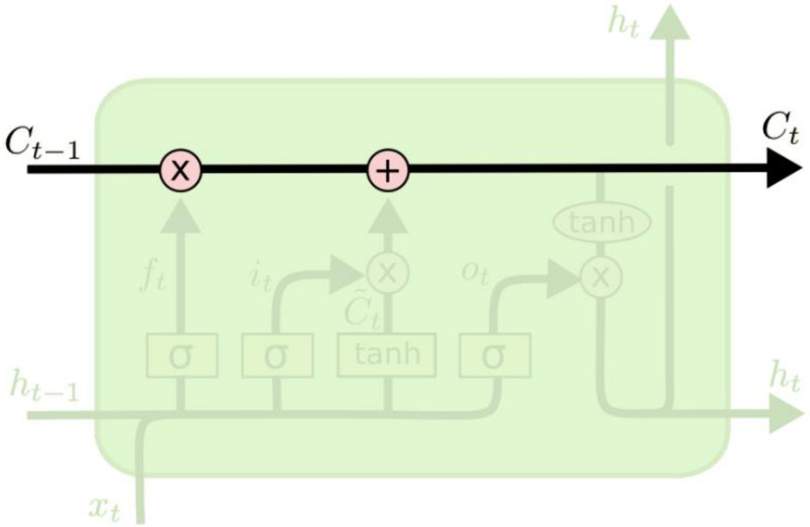
All these are vectors of same length *n*

Gates are applied using element-wise (or Hadamard) product: ⊙

http://web.stanford.edu/class/cs224n/slides/cs224n-2021-lecture06-fancy-rnn.pdf

LSTMs solve vanishing gradient problem

***Intuition**: "The cell state is kind of like a* conveyor belt. It runs straight down the entire chain, with only some minor linear interactions. It's very easy for information to just flow along it unchanged."



You can show that the gradient of loss with respect to cell state depends on forget gate values along the path

For a detailed explanation see this:
http://www.cse.iitm.ac.in/~miteshk/CS7015/Slides/Teaching/pdf/Lecture15.pdf

Other Noteworthy Points

- Bi-directional RNNs
  - All NLU applications typically use this to incorporate context in both directions
  - Run separate RNNs in forward and backward directions
  - Concatenate hidden states at each time step for bidirectional context vector
- Deep RNNs: RNNS layers can be stacked
- Gated Recurrent Units (GRU): a simpler variant of LSTM
- LSTM-CRF network: A LSTM-variant where dependencies between output variables can be modeled

Suggested Reading

- N-gram Language Models (textbook chapter)
- Smoothing techniques: http://nrs.harvard.edu/urn-3:HUL.InstRepos:25104739
- FF-Neural LM: https://www.jmlr.org/papers/volume3/bengio03a/bengio03a.pdf
- The Unreasonable Effectiveness of Recurrent Neural Networks (nice blog post overview)
- Sequence Modeling: Recurrent and Recursive Neural Nets (Sections 10.1 and 10.2)
- On Chomsky and the Two Cultures of Statistical Learning
- Sequence Modeling: Recurrent and Recursive Neural Nets (Sections 10.3, 10.5, 10.7-10.12)
- Learning long-term dependencies with gradient descent is difficult (one of the original vanishing gradient papers)

Suggested Reading

- On the difficulty of training Recurrent Neural Networks (proof of vanishing gradient problem)
- Vanishing Gradients Jupyter Notebook (demo for feedforward networks)
- Understanding LSTM Networks (must read, blog post overview)
- https://r2rt.com/written-memories-understanding-deriving-and-extending-the-lstm.html

Suggested Reading

- Original LSTM papers:
  - http://www.bioinf.jku.at/publications/older/2604.pdf (the original paper)
  - ftp://ftp.idsia.ch/pub/juergen/TimeCount-IJCNN2000.pdf (peephole variant that is most widely used)
- Pretrained Models
  - ELMo
  - ULMFit
- Other lectures on RNNs
  - https://github.com/oxford-cs-deepnlp-2017/lectures (Lectures 3 & 4)
  - https://www.cse.iitm.ac.in/~miteshk/CS7015.html (Lectures 14 & 15)
  - http://web.stanford.edu/class/cs224n/ (Lectures 5 & 6)

Encoder-Decoder Models : Part 1

Machine Translation & Sequence-2-Sequence Tasks

Anoop Kunchukuttan
Microsoft, MT Group, Hyderabad
anoop.kunchukuttan@gmail.com

For CS772 at IIT Bombay, Feb 2022

Course Instructor: Prof. Pushpak Bhattacharyya

Outline

- **Introduction**
- Statistical Machine Translation
- Neural Machine Translation
- Evaluation of Machine Translation
- Multilingual Neural Machine Translation
- Transformers

Automatic conversion of text/speech from one natural language to another

Be the change you want to see in the world

वह परिवर्तन बनो जो संसार में देखना चाहते हो

Microsoft

Google Translate

Government: administrative requirements, education, security.

Enterprise: product manuals, customer support

Social: travel (signboards, food), entertainment (books, movies, videos)

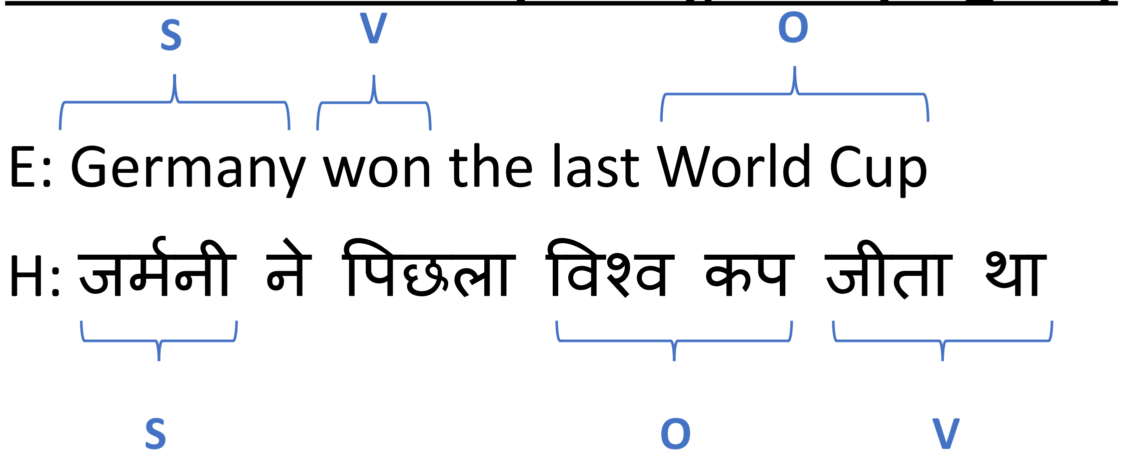
Translation under the hood

- Cross-lingual Search
- Cross-lingual Summarization
- Building multilingual dictionaries

Any multilingual NLP system will involve some kind of machine translation at some level

What is Machine Translation?

Word order: SOV (Hindi), SVO (English)



E: Germany won the last World Cup

H: जर्मनी ने पिछला विश्व कप जीता था

Free (Hindi) vs rigid (English) word order

पिछला विश्व कप जर्मनी ने जीता था *(correct)*

The last World Cup Germany won *(grammatically incorrect)*

The last World Cup won Germany *(meaning changes)*

Language Divergence ➔ the great diversity among languages of the world

The central problem of MT is to bridge this language divergence

Shachi Dave, Jignashu Parikh and Pushpak Bhattacharyya, Interlingua Based English Hindi Machine Translation and Language Divergence, Journal of Machine Translation (JMT), Volume 17, September, 2002.

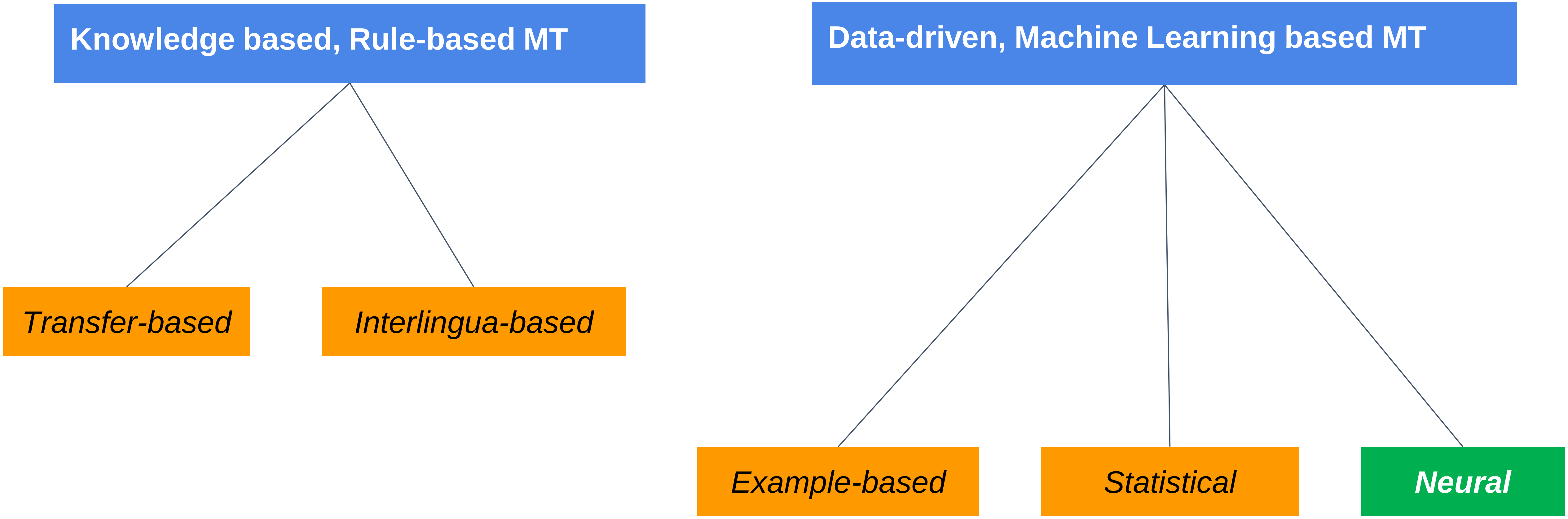
Why is Machine Translation difficult?

- **Ambiguity**
  - Same word, multiple meanings: *मंत्री (minister or chess piece)*
  - Same meaning, multiple words: *जल, पानी, नीर (water)*
- **Word Order**
  - Underlying deeper syntactic structure
  - Phrase structure grammar?
  - Computationally intensive
- **Morphological Richness**
  - Identifying basic units of words
  - *घर ◌ा समोर चा*
  - *That which is in front of the house*

Why should you study Machine Translation?

- One of the most challenging problems in Natural Language Processing
- Pushes the boundaries of NLP
- Involves analysis as well as synthesis
- Involves all layers of NLP: morphology, syntax, semantics, pragmatics, discourse
- *Theory and techniques in MT are applicable to a wide range of other problems like transliteration, speech recognition and synthesis, and other NLP problems.*

Approaches to build MT systems



Knowledge based, Rule-based MT

Transfer-based

Interlingua-based

Data-driven, Machine Learning based MT

Example-based

Statistical

Neural

Outline

- Introduction
- **Statistical Machine Translation**
- Neural Machine Translation
- Evaluation of Machine Translation
- Multilingual Neural Machine Translation
- Summary

Statistical Machine Translation

Let’s formalize the translation process

We will model translation using a ***probabilistic model****. Why?*

- *We would like to have a measure of confidence for the translations we learn*
- *We would like to model uncertainty in translation*

E: target language
F: source language

e: target language sentence
f : source language sentence

Best translation

$$\bar{e} = \arg\max_{e} P(e|f)$$

How do we **model** this quantity?

Model*: a simplified and idealized understanding of a physical process*

We must first explain the process of translation

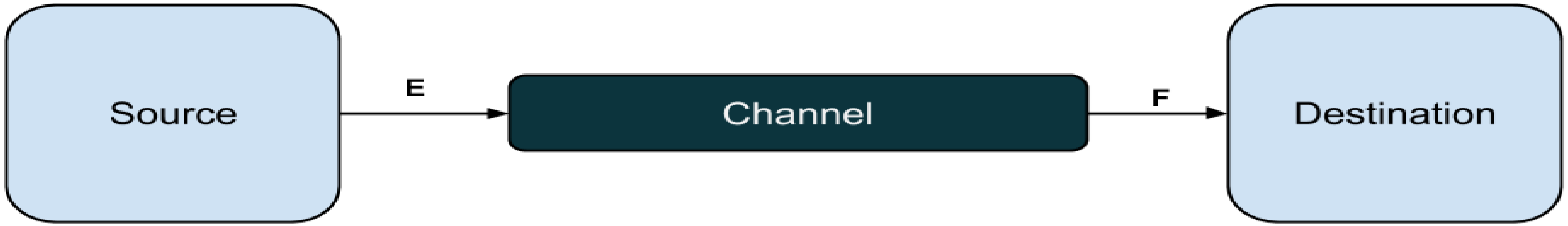
A very general framework for many NLP problems

We explain translation using the Noisy Channel Model

Generate target sentence

Channel corrupts the target

Source sentence is a corruption of the target sentence



Language Model (LM)
P(e)

Captures fluency

Translation Model (TM)
P(f|e)

Captures fidelity

$$P(e|f) = P(e) \times P(f|e)$$

Translation is the process of recovering the original signal given the corrupted signal

Why use this counter-intuitive way of explaining translation?

- Makes it easier to mathematically represent translation and learn probabilities
- **Fidelity** and **Fluency** can be modelled separately

Brown, Peter F., et al. "The mathematics of statistical machine translation: Parameter estimation." *Computational linguistics* 19.2 (1993): 263-311.

We know how to learn n-gram language models

Let's see how to learn the translation model → $P(\boldsymbol{f}|\boldsymbol{e})$

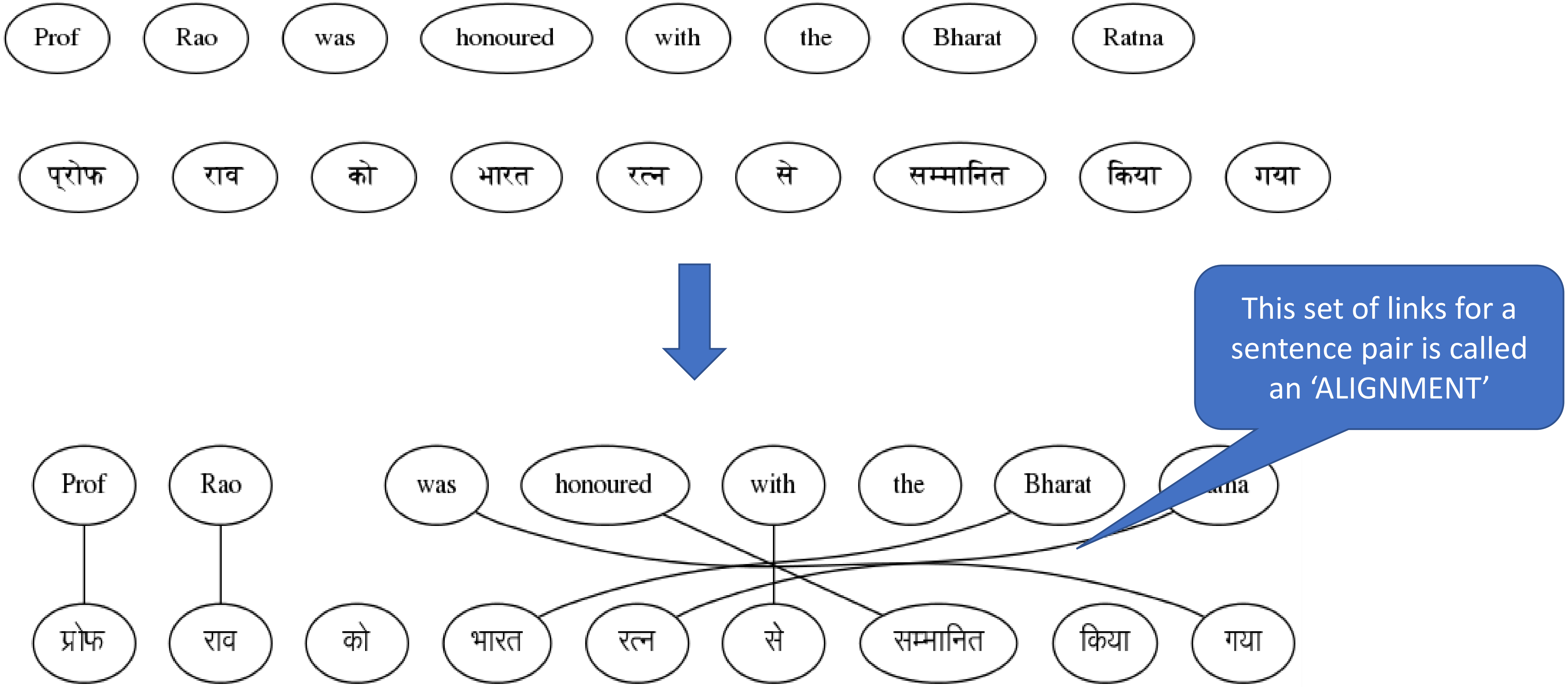
To learn sentence translation probabilities,

→ we first need to learn word-level translation probabilities

That is the task of word alignment

Parallel Corpus	
A boy is sitting in the kitchen	एक लडका रसोई में बैठा है
A boy is playing tennis	एक लडका टेनिस खेल रहा है
A boy is sitting on a round table	एक लडका एक गोल मेज पर बैठा है
Some men are watching tennis	कुछ आदमी टेनिस देख रहे है
A girl is holding a black book	एक लडकी ने एक काली किताब पकडी है
Two men are watching a movie	दो आदमी चलचित्र देख रहे है
A woman is reading a book	एक औरत एक किताब पढ रही है
A woman is sitting in a red car	एक औरत एक काले कार मे बैठी है

Given a parallel sentence pair, find word level correspondences



This set of links for a sentence pair is called an ‘ALIGNMENT’

Brown, Peter F., et al. "The mathematics of statistical machine translation: Parameter estimation." *Computational linguistics* 19.2 (1993): 263-311.

Parallel Corpus	
A boy is **sitting** in the kitchen	एक लडका रसोई में **बैठा** है
A boy is playing **tennis**	एक लडका **टेनिस** खेल रहा है
A boy is **sitting** on a round table	एक लडका एक गोल मेज पर **बैठा** है
Some men **are watching** **tennis**	कुछ आदमी **टेनिस** **देख रहे है**
A girl is holding a black book	एक लडकी ने एक काली किताब पकडी है
Two men **are watching** a movie	दो आदमी चलचित्र **देख रहे है**
A woman is reading a book	एक औरत एक किताब पढ रही है
A woman is **sitting** in a red car	एक औरत एक काले कार मे **बैठा** है

https://www.isi.edu/natural-language/mt/wkbk.rtf

Key Idea

Co-occurrence of translated words

Words which occur together in the parallel sentence are likely to be translations (higher P(f|e))

Phrase Based SMT

Why stop at learning word correspondences?

KEY IDEA ➔ Use “Phrase” (Sequence of Words) as the basic translation unit

Note: the term ‘phrase’ is not used in a linguistic sense

The Prime Minister of India	भारत के प्रधान मंत्री bhArata ke pradhAna maMtrI India of Prime Minister
is running fast	तेज भाग रहा है teja bhAg rahA hai fast run -continuous is
honoured with	से सम्मानित किया se sammanita kiyA with honoured did
Rahul lost the match	राहुल मुकाबला हार गया rAhula mukAbalA hAra gayA Rahul match lost

Koehn, Philipp, Franz J. Och, and Daniel Marcu. Statistical phrase-based translation. UNIVERSITY OF SOUTHERN CALIFORNIA MARINA DEL REY INFORMATION SCIENCES INST, 2003. https://apps.dtic.mil/sti/pdfs/ADA461156.pdf

Benefits of PB-SMT

Local Reordering → Intra-phrase re-ordering can be memorized

The Prime Minister of India	भारत के प्रधान मंत्री bhaarat ke pradhaan maMtrI India of Prime Minister

Sense disambiguation based on local context → Neighbouring words help make the choice

heads towards Pune	पुणे की ओर जा रहे है pune ki or jaa rahe hai Pune towards go –continuous is
heads the committee	समिति की अध्यक्षता करते है Samiti kii adhyakshata karte hai committee of leading -verbalizer is

Benefits of PB-SMT (2)

Handling institutionalized expressions

- Institutionalized expressions, idioms can be learnt as a single unit

hung assembly	त्रिशंकु विधानसभा trishanku vidhaansabha
Home Minister	गृह मंत्री gruh mantrii
Exit poll	चुनाव बाद सर्वेक्षण chunav baad sarvekshana

- Improved Fluency
  - The phrases can be arbitrarily long (even entire sentences)

Parallel Corpus	
A boy is **sitting** in the kitchen	एक लडका रसोई में **बैठा** है
A boy is playing **tennis**	एक लडका **टेनिस** खेल रहा है
A boy is **sitting** on a round table	एक लडका एक गोल मेज पर **बैठा** है
Some men **are watching tennis**	कुछ आदमी **टेनिस देख रहे है**
A girl is holding a black book	एक लडकी ने एक काली किताब पकडी है
Two men **are watching** a movie	दो आदमी चलचित्र **देख रहे है**
A woman is reading a book	एक औरत एक किताब पढ रही है
A woman is **sitting** in a red car	एक औरत एक काले कार मे **बैठा** है

Mathematical Model

Let's revisit the decision rule for SMT model

$$\mathbf{e}_{\text{best}} = \text{argmax}_{\mathbf{e}}\, p(\mathbf{e}|\mathbf{f}) = \text{argmax}_{\mathbf{e}}\, p(\mathbf{f}|\mathbf{e})\, p_{\text{LM}}(\mathbf{e})$$

Let's revisit the translation model *p*(**f|e**)

- Source sentence can be segmented in **I** phrases
- Then, *p*(**f|e**) can be decomposed as:

$$p(\bar{f}_1^I|\bar{e}_1^I) = \prod_{i=1}^{I} \phi(\bar{f}_i|\bar{e}_i)\, d(\text{start}_i - \text{end}_{i-1} - 1)$$

Distortion probability

Phrase Translation Probability

start_i :start position in **f** of ith phrase of **e**
end_i :end position in **f** of ith phrase of **e**

Learning The Phrase Translation Model

Involves Structure + Parameter Learning:

- Learn the **Phrase Table**: the central data structure in PB-SMT

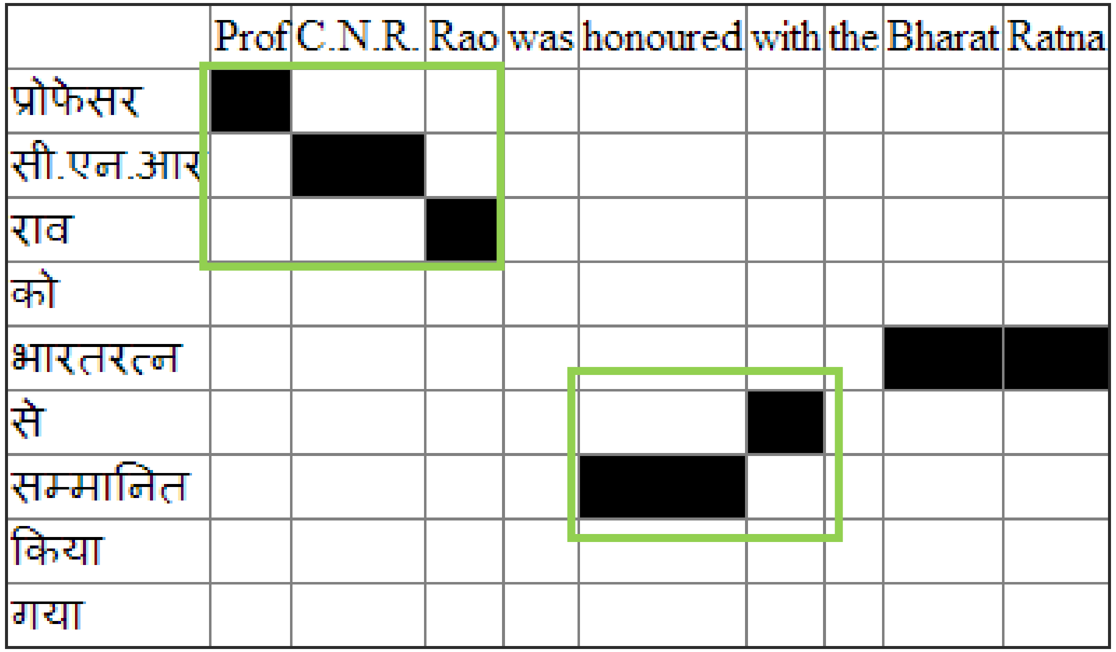
The Prime Minister of India	भारत के प्रधान मंत्री
is running fast	तेज भाग रहा है
the boy with the telescope	दूरबीन से लड़के को
Rahul lost the match	राहुल मुकाबला हार गया

- Learn the **Phrase Translation Probabilities**

Prime Minister of India	भारत के प्रधान मंत्री India of Prime Minister	0.75
Prime Minister of India	भारत के भूतपूर्व प्रधान मंत्री India of former Prime Minister	0.02
Prime Minister of India	प्रधान मंत्री Prime Minister	0.23

Learning Phrase Tables from Word Alignments

- Start with word alignments
- Word Alignment : reliable input for phrase table learning
  - high accuracy reported for many language pairs
- Central Idea: A consecutive sequence of aligned words constitutes a "phrase pair"



	Prof	C.N.R.	Rao	was	honoured	with	the	Bharat	Ratna
प्रोफेसर	■								
सी.एन.आर		■							
राव			■						
को									
भारतरत्न								■	■
से						■			
सम्मानित					■				
किया									
गया									

Which phrase pairs to include in the phrase table?

Discriminative Training of PB-SMT

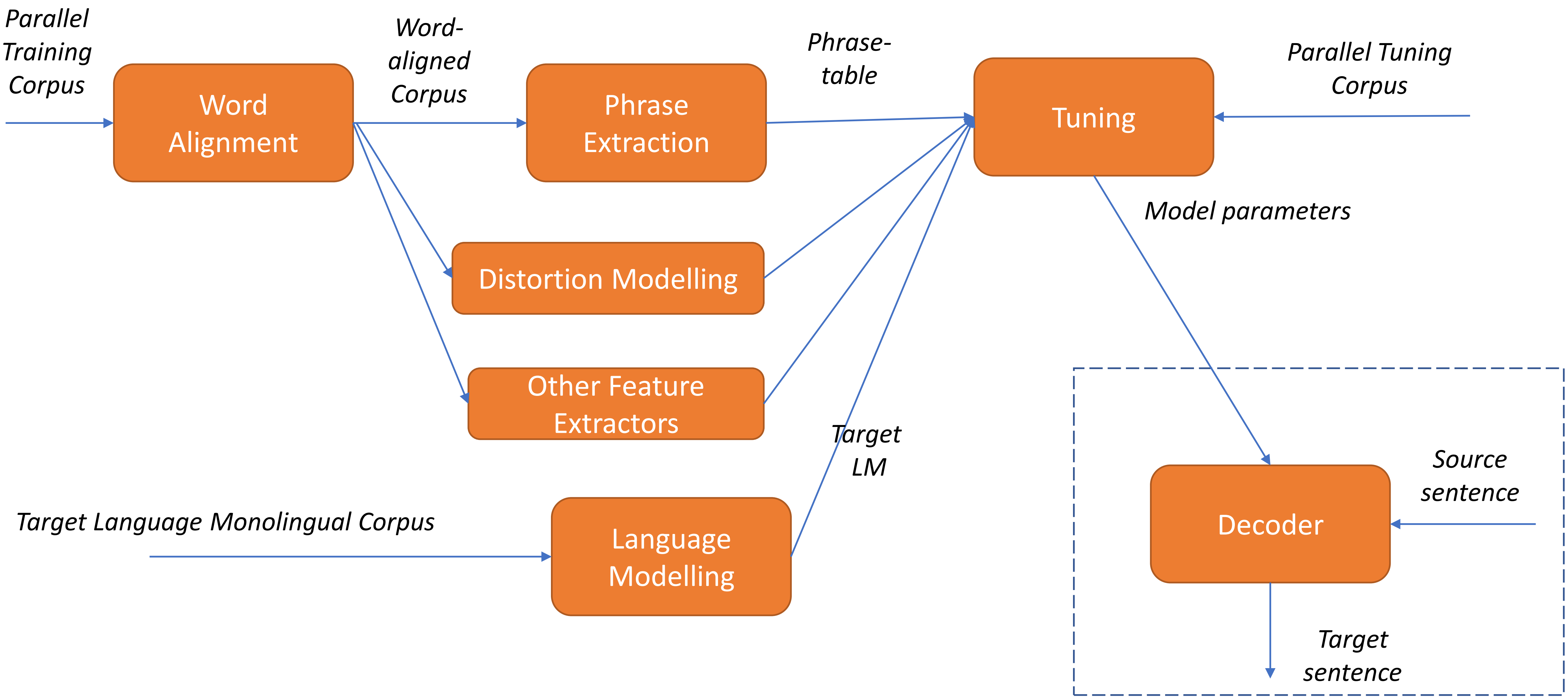
- Directly model the posterior probability p**(e|f)**
- Use the Maximum Entropy framework

$$P(\mathbf{e}|\mathbf{f}) = \exp\left(\sum_i \lambda_i h_i(f_1^I, e_1^J)\right)$$

$$e^* = \arg\max_{e_i} \sum_i \lambda_i h_i(f_1^I, e_1^J)$$

  - h_i**(f,e)** are feature functions , λ_i's are feature weights
- Benefits:
  - *Can add arbitrary features to score the translations*
  - Can assign different weight for each features
  - Assumptions of generative model may be incorrect
  - Feature weights λ_i are learnt during tuning

Typical SMT Pipeline



We have looked at a basic phrase-based SMT system

This system can learn word and phrase translations from parallel corpora

But many important linguistic phenomena need to be handled

- Divergent Word Order
- Rich morphology
- Named Entities and Out-of-Vocabulary words

Limitations of SMT

- *No end-to-end optimization*
    - *Separately developed complex components strung together*
- *Divergent word-order is a big challenge*
- *n-gram LM not the best way to score translation fluency*
- *Model size is a function of the data size*

Outline

- Introduction
- Statistical Machine Translation
- **Neural Machine Translation**
- Evaluation of Machine Translation
- Multilingual Neural Machine Translation
- Summary

Neural Machine Translation

Topics

- *Why NMT?*
- *Encoder-Decoder Models*
- *Attention Mechanism*
- *Backtranslation*
- *Subword-level Models*

Topics

- *Why NMT?*
- *Encoder-Decoder Models*
- *Attention Mechanism*
- *Backtranslation*
- *Subword-level Models*

Sequence to Sequence Task

Input Sequence: $(x_1 \;\; x_2 \;\; x_3 \;\; x_4 \ldots\ldots x_i \ldots\ldots x_N)$

Output Sequence: $(y_1 \;\; y_2 \;\; y_3 \;\; y_4 \ldots y_k \ldots y_M)$

Input and output sequences have different lengths

Variable length input

Output contains categorical labels

Output at any time-step typically depends on neighbouring output labels and input elements

Machine Translation

Encoder-decoder model is a general framework for sequence to sequence tasks

Many tasks as Sequence to Sequence transformations

- *Summarization: Article ⇒ Summary*
- *Question answering: Question ⇒ Answer*
- *Dialogue: Previous utterance ⇒ next utterance*
- *Transliteration: character sequence ⇒ character sequence*
- *Grammar Correction: Incorrect sentence ⇒ Correct Sentence*
- *Translation Postediting: Incorrect translation ⇒ Correct translation*
- *Image labelling: Image ⇒ Label*

We are interested in modeling $P(\boldsymbol{Y}|\boldsymbol{X})$

Conditional Language Modelling task ➔

Learning Target LM conditioned on the source sentence

$$P(\boldsymbol{Y}|\boldsymbol{X}) = \prod_{j=1}^{M} P(y_j|\underbrace{y_1, y_2, \ldots y_{j-1}}_{\textit{Target language RNN}}, \boldsymbol{X})$$

Additional conditioning on the source sentence

How do we model this?

LM for generating the target sequence

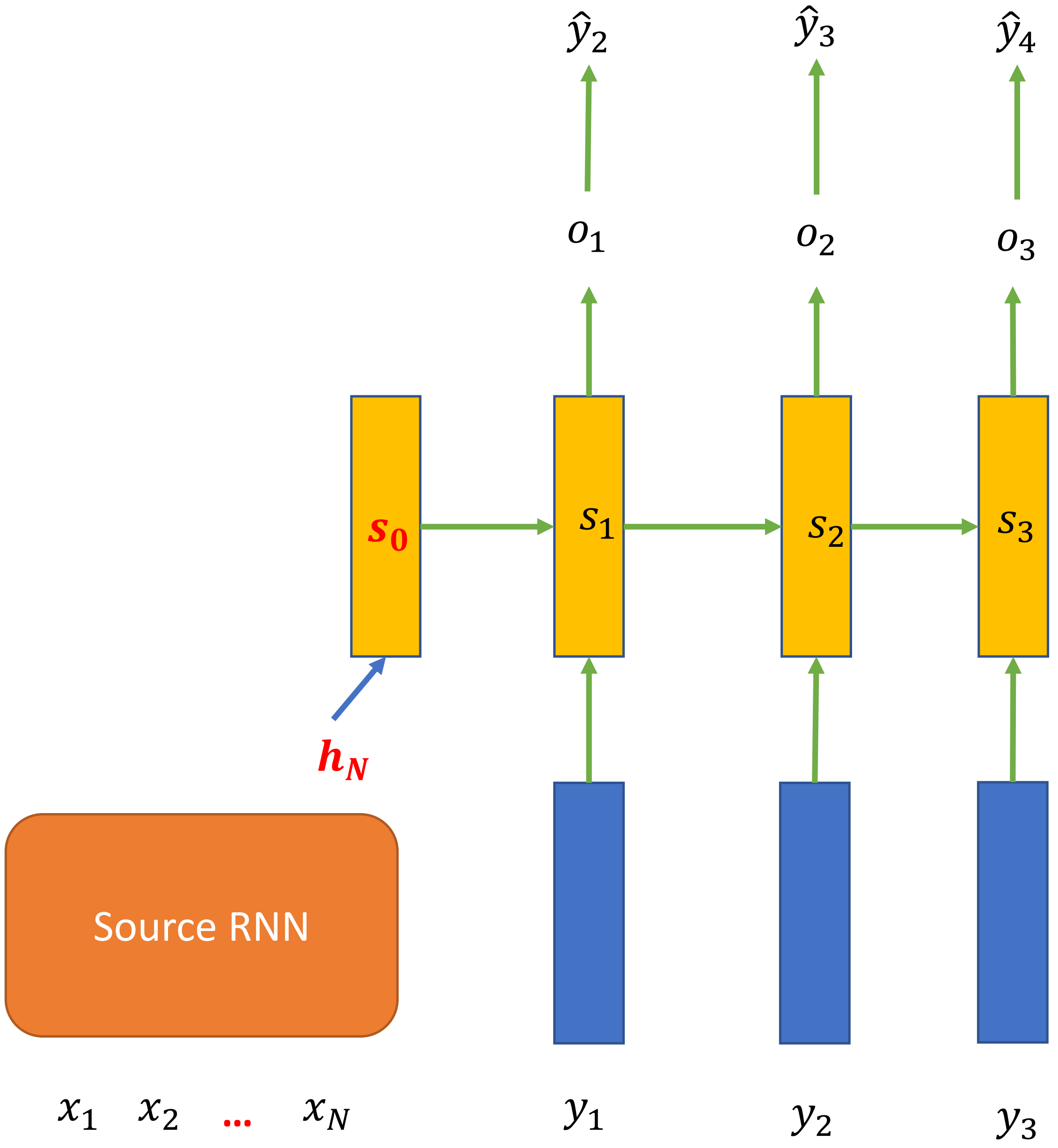
s_0 ➔ Initial state of target language RNN

Set s_0 = a vector representation of source

*We have our **conditional** LM*

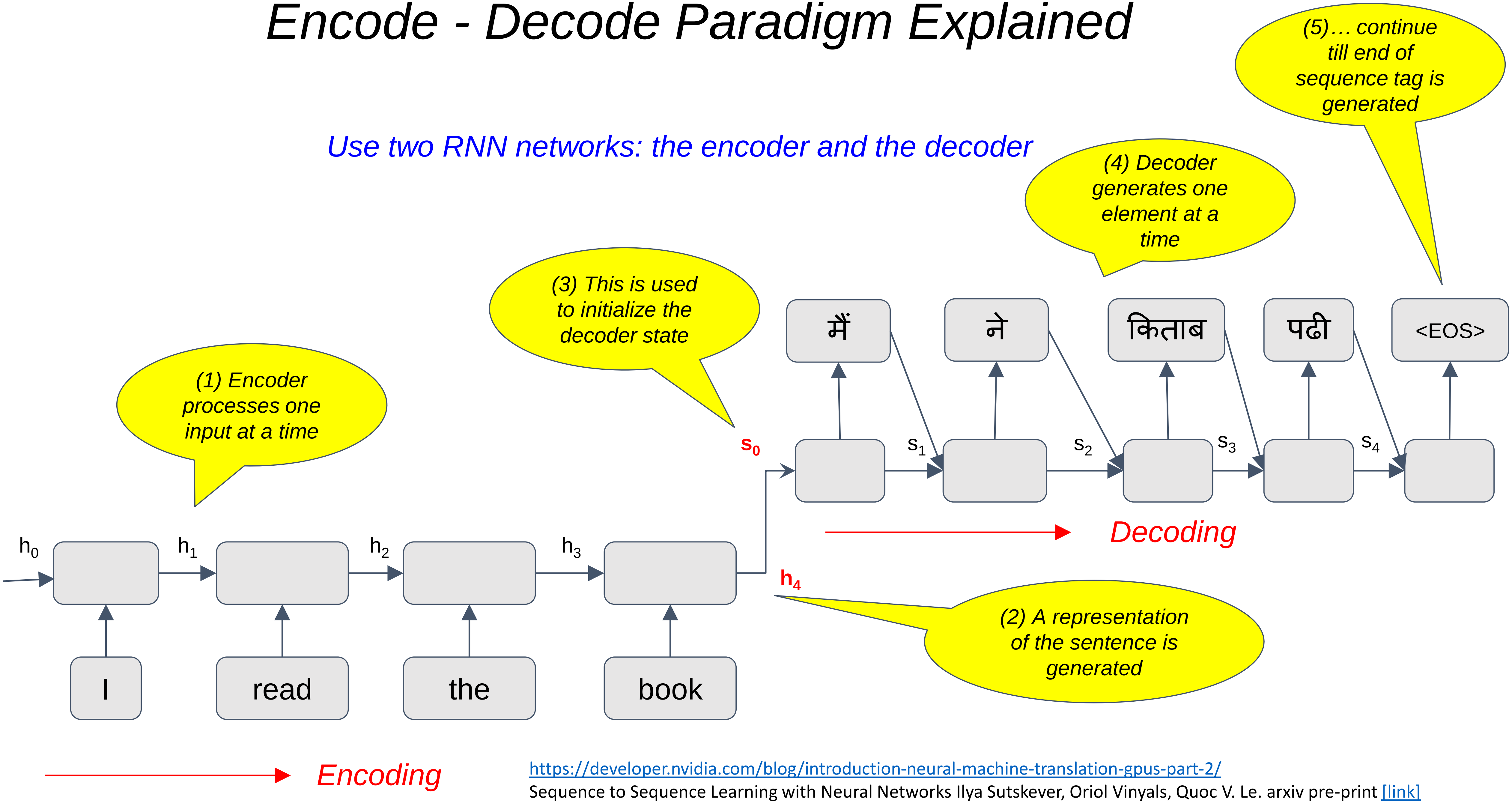
Source Vector Representation ➔
last state of source sentence RNN

$$s_0 = h_N$$



Encode - Decode Paradigm Explained

Use two RNN networks: the encoder and the decoder

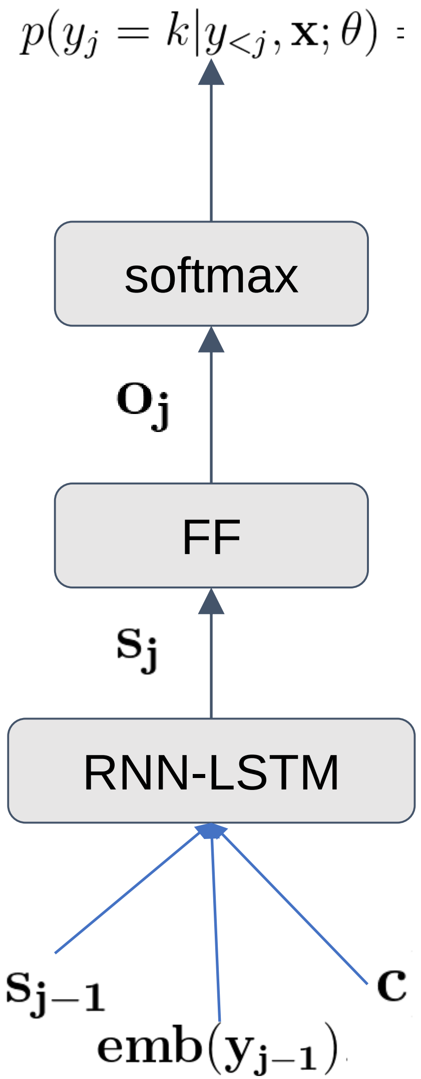


(5)… continue till end of sequence tag is generated
(4) Decoder generates one element at a time
(3) This is used to initialize the decoder state
(1) Encoder processes one input at a time
मैं
ने
किताब
पढी
<EOS>
s_0
s_1
s_2
s_3
s_4
Decoding
h_0
h_1
h_2
h_3
h_4
(2) A representation of the sentence is generated
I
read
the
book
Encoding

https://developer.nvidia.com/blog/introduction-neural-machine-translation-gpus-part-2/

Sequence to Sequence Learning with Neural Networks Ilya Sutskever, Oriol Vinyals, Quoc V. Le. arxiv pre-print [link]

What is the decoder doing at each time-step?



$p(y_j = k | y_{<j}, \mathbf{x}; \theta) =$

softmax

$\mathbf{o_j}$

FF

$\mathbf{s_j}$

RNN-LSTM

$\mathbf{s_{j-1}}$

$\mathbf{emb(y_{j-1})}$

$\mathbf{c}$

This captures $y_{<j}$

This captures x, $c=h_4$

$$softmax(o_{jk}) = \frac{\exp(o_{jk})}{\sum_{m=0}^{m=T} \exp(o_{jm})}$$

$$\mathbf{o_j} = FF(s_j)$$

$$\mathbf{s_j} = g(\mathbf{s_{j-1}}, \mathbf{emb(y_{j-1})}, \mathbf{c})$$

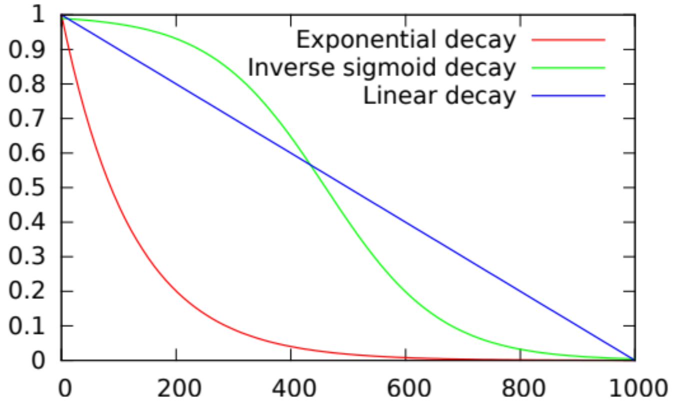
Training an NMT Model

$$p(\mathbf{y}|\mathbf{x};\theta) = \prod_{j=1}^{m} p(y_j|y_{<j},\mathbf{x};\theta) \qquad p(y_j = k|y_{<j},\mathbf{x};\theta) = softmax(o_{jk})$$

$$\mathcal{L}_\theta = \sum_{(\mathbf{x},\mathbf{y})\in\mathbf{C}} \log p(\mathbf{y}|\mathbf{x};\theta)$$

Maximum
Likelihood
Estimation

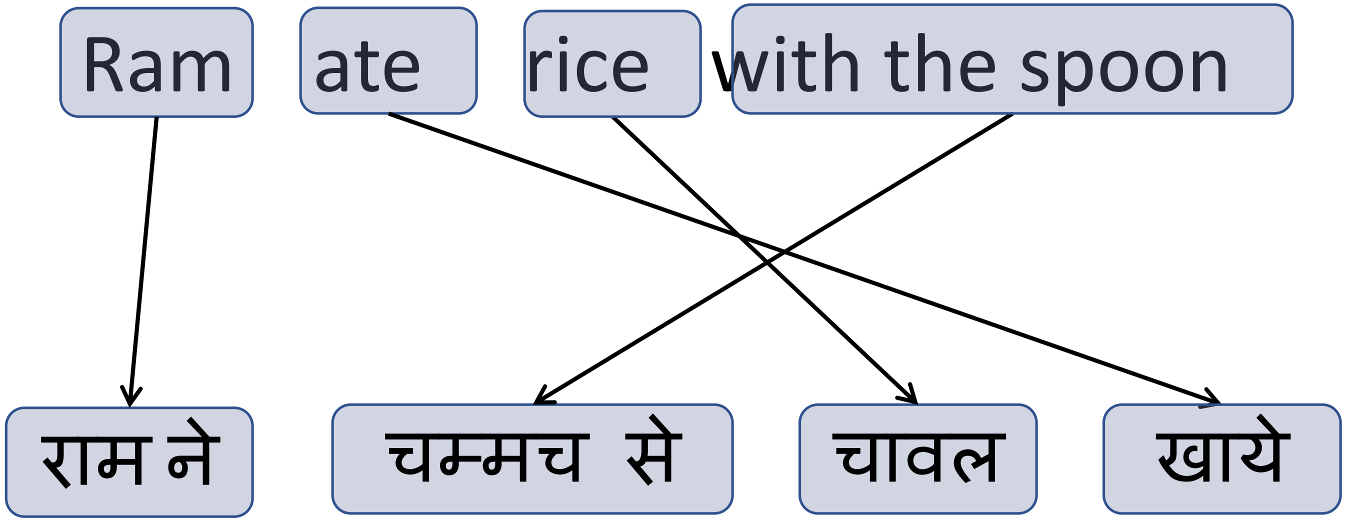
- *At each time decoder step:*
  - Feed model output from previous time step ➔ degrades performance
  - Feed ground-truth output from previous time step ➔ **teacher forcing**
- *Discrepancy in train and test scenarios* ➔ ***Exposure bias***
  - Solution → scheduled sampling
  - Sample from ground truth or predicted label
  - Sampling probability is varied: prefer ground truth earlier in training



Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent Neural networks. NeurIPS 2015.

Decoding

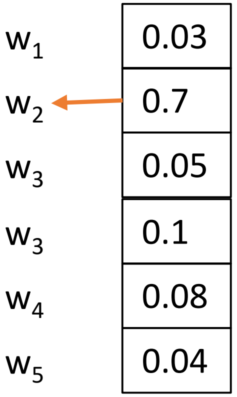
Searching for the best translations in the space of all translations



Decoding Strategies

- Exhaustive Search: *Score each and every possible translation – Forget it!* ➔ $O(V^N)$
- Sampling ➔ $O(NV)$
- Greedy ➔ $O(NV)$
- Beam Search ➔ $O(kNV)$

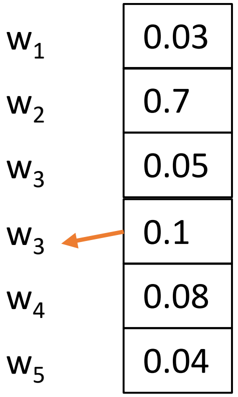
Greedy Decoding



w_1	0.03
w_2	0.7
w_3	0.05
w_3	0.1
w_4	0.08
w_5	0.04

Select best word using the distribution $P(y_j | y_{<j}, \boldsymbol{x})$

Sampling Decoding



w_1	0.03
w_2	0.7
w_3	0.05
w_3	0.1
w_4	0.08
w_5	0.04

Sample next word using the distribution $P(y_j | y_{<j}, \boldsymbol{x})$

Generate one word at a time sequentially

Not used to find best translation, but these methods have their uses ➔ for efficiency reasons

Greedy Search is not optimal

w_1	**0.5**	w_1	0.1
w_2	0.4	w_2	0.2
w_3	0.05	w_3	**0.3**
w_3	0.02	w_3	0.1
w_4	0.01	w_4	0.1
w_5	0.02	w_5	0.2

Probability of best sequence w_1w_3 *=0.15*

w_1	0.5	w_1	0.1
w_2	**0.4**	w_2	0.45
w_3	0.05	w_3	0.2
w_3	0.02	w_3	0.15
w_4	0.01	w_4	0.08
w_5	0.02	w_5	0.02
	t_1		t_2

Probability of best sequence w_2w_2 *=0.18*

Beam Search

A compromise solution between greedy decoding and exhaustive search

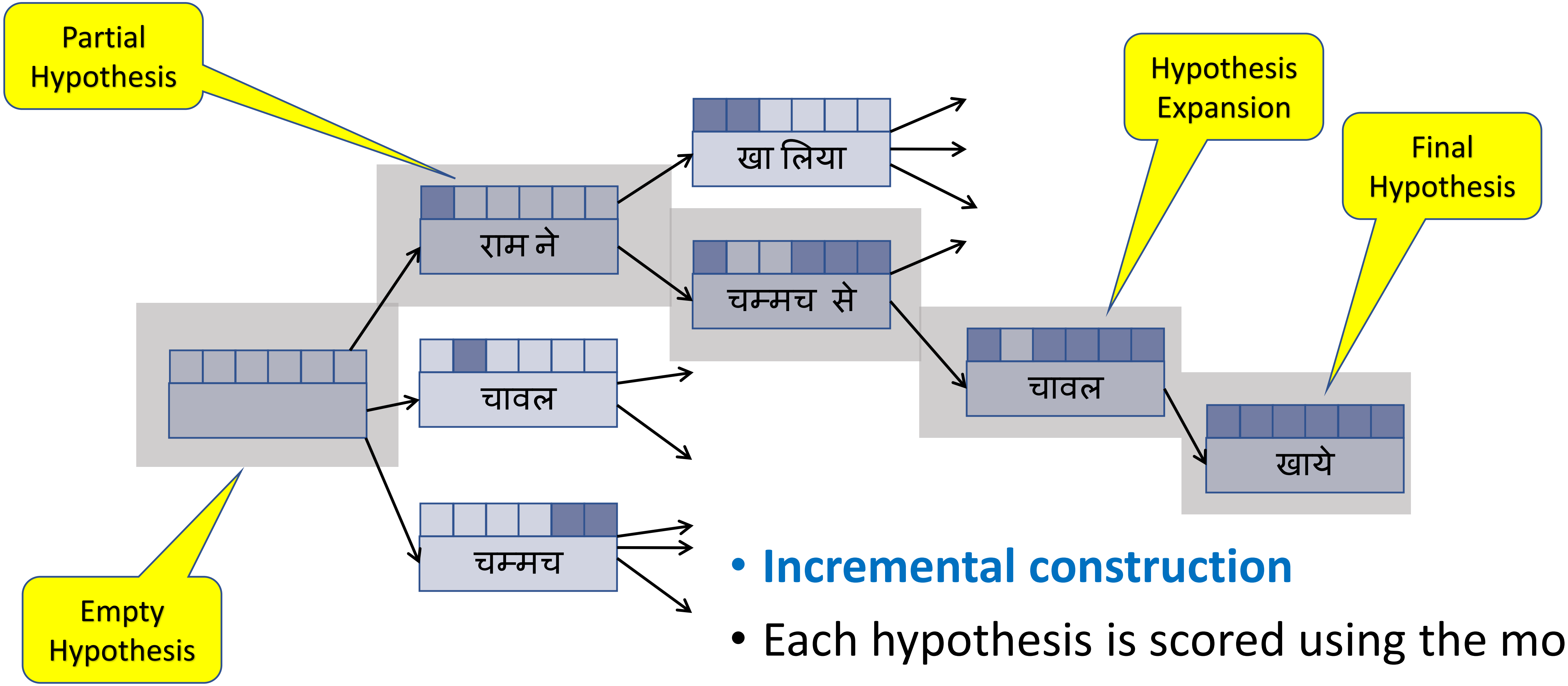
- *Explores more translation candidates than greedy search*
- *More efficient than exhaustive search*

2 Core Ideas:

- ***Incremental*** *construction & scoring of translation candidate (one decoder time step at a time)*
- *At each decoder time step,* ***keep the k-most probable partial translations***
  - ➔ *these will be used for candidates expansion*

- ***Not guaranteed to find optimal solution***

http://www.phontron.com/slides/nlp-programming-en-13-search.pdf



- **Incremental construction**
- Each hypothesis is scored using the model
- Hypotheses are maintained in a priority queue

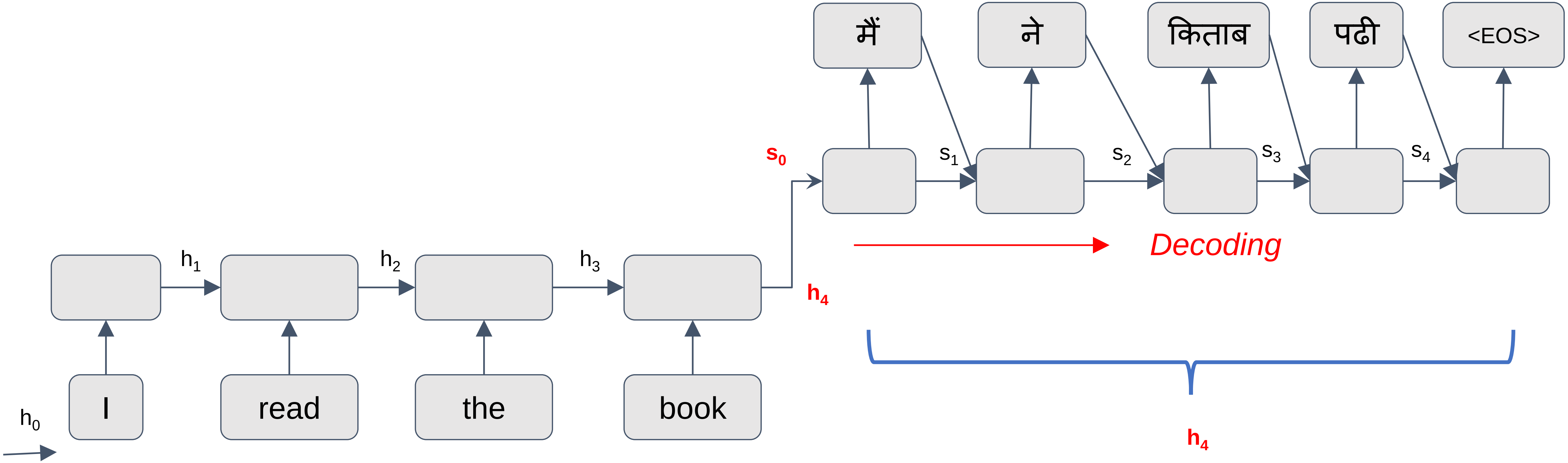
Topics

- *Why NMT?*
- *Encoder-Decoder Models*
- *Attention Mechanism*
- *Backtranslation*
- *Subword-level Models*

The entire source sentence is represented by a single vector

Problems

- Insufficient to represent to capture all the syntactic and semantic complexities
  - *Solution: Use a richer representation for the sentences*
- Long-term dependencies: Source sentence representation not useful after few decoder time steps
  - *Solution: Make source sentence information when making the next prediction*



Feed the encoder state as input at each decoder timestep

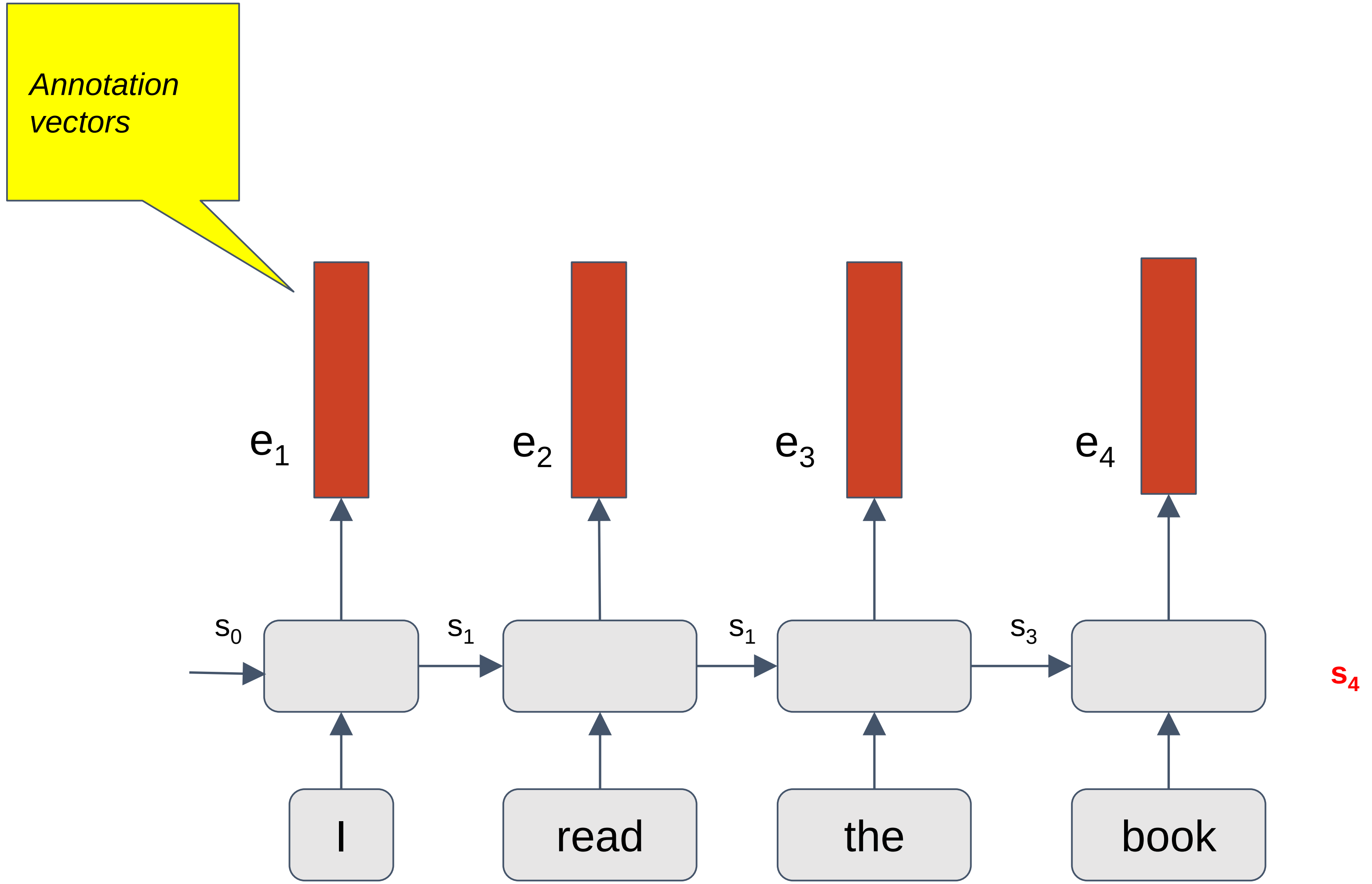
The entire source sentence is represented by a single vector

Problems

- Insufficient to represent to capture all the syntactic and semantic complexities
  - *Solution: Use a richer representation for the sentences*
- Long-term dependencies: Source sentence representation not useful after few decoder time steps
  - *Solution: Make source sentence information when making the next prediction*
  - *Even better, make **RELEVANT** source sentence information available*

These solutions motivate the next paradigm

Encode - Attend - Decode Paradigm



Represent the source sentence by the **set of output vectors** from the encoder

Each output vector at time *t* is a contextual representation of the input at time *t*

Let’s call these encoder output vectors ***annotation vectors***

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." *ICLR 2015.*

https://developer.nvidia.com/blog/introduction-neural-machine-translation-gpus-part-3/

How can the annotation vectors help predicting the next output?

Key Insight:

(1) Not all annotation vectors are equally important for prediction of the next element

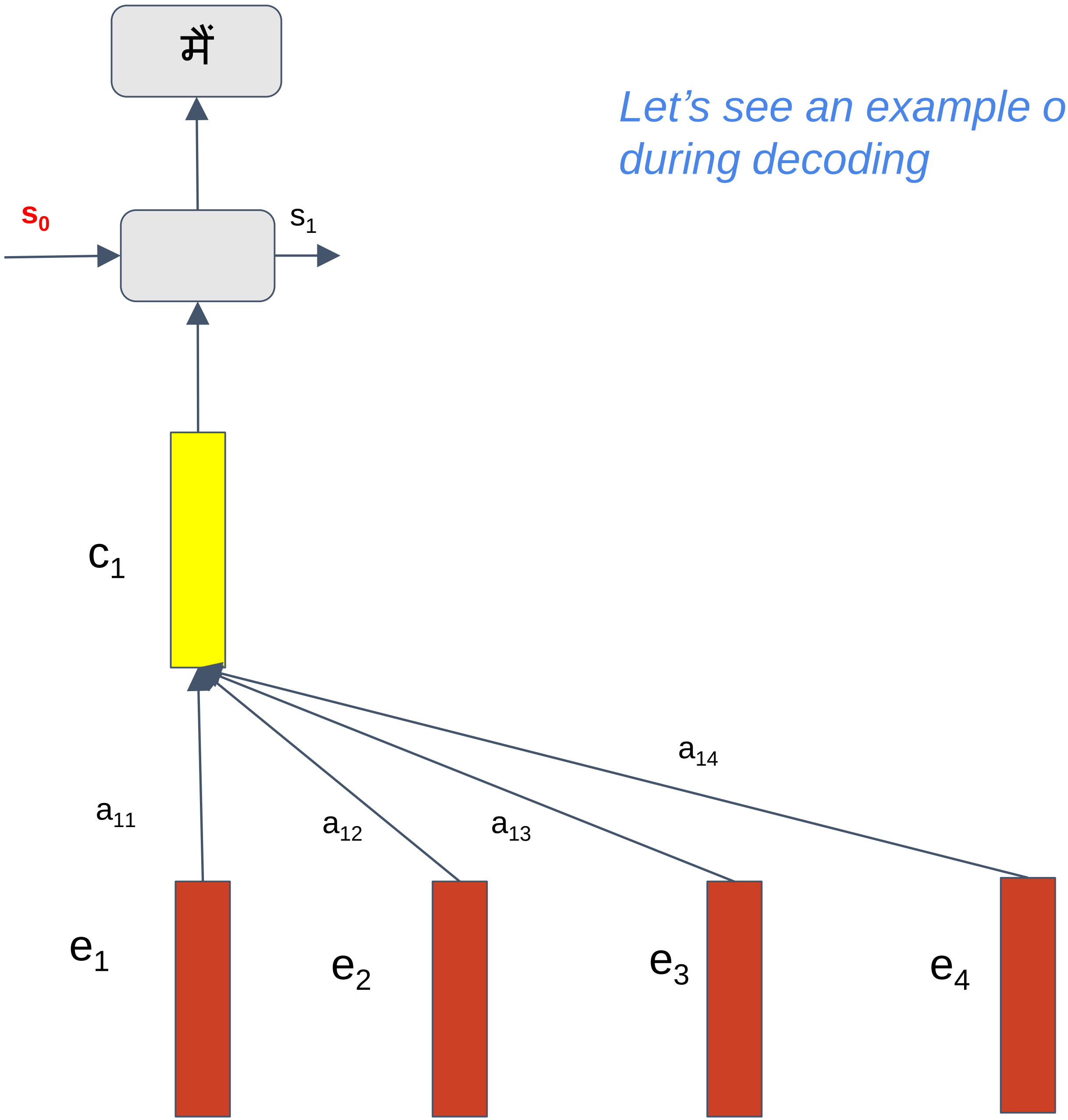
(2) The annotation vector to use next depends on what has been generated so far by the decoder

eg. To generate the 3rd target word, the 3rd source word is most important

Context vector = weighted average of the annotation vectors

More weight to annotation vectors which need more **focus or attention**

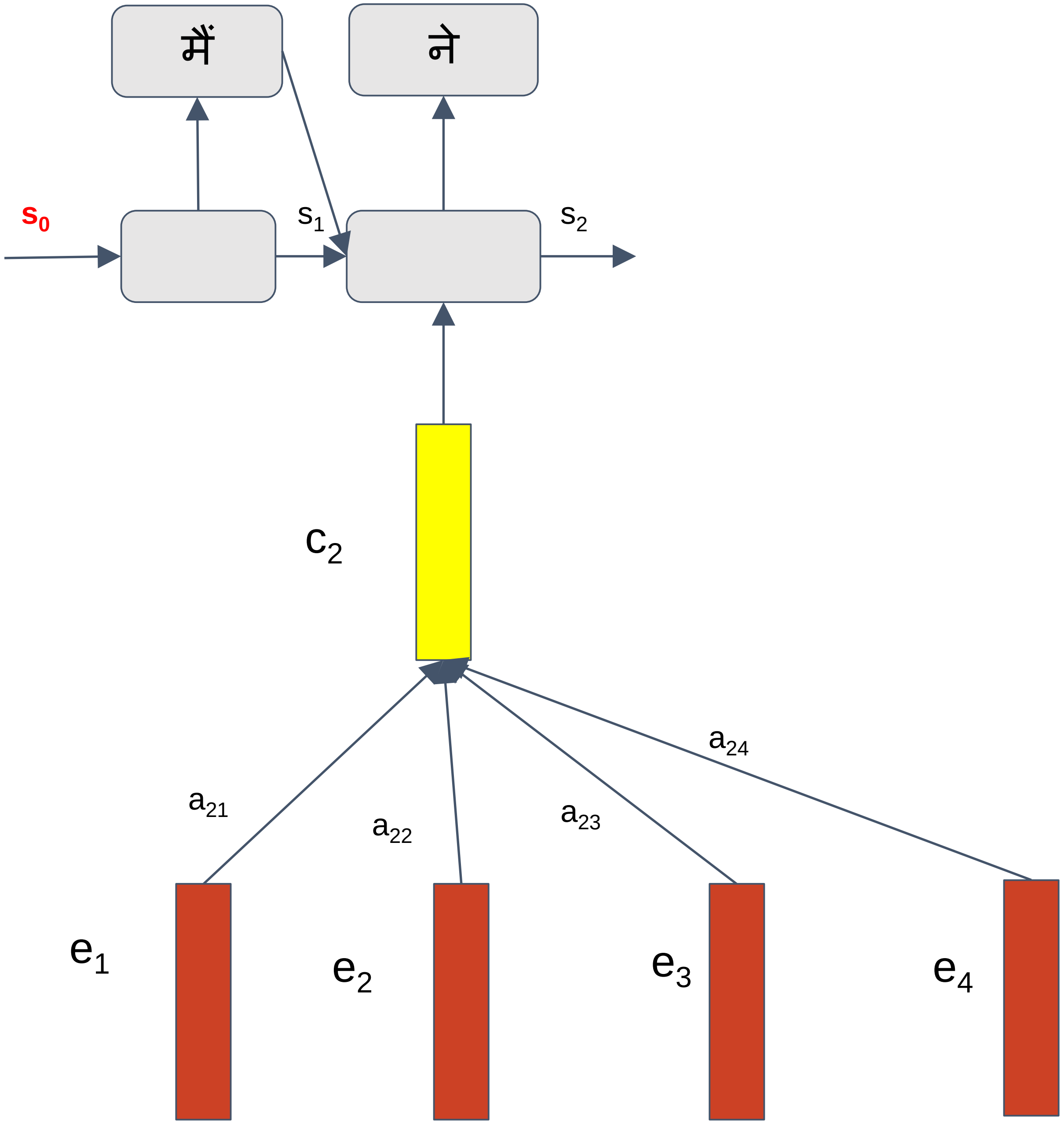
This averaged ***context vector*** is an input to the decoder

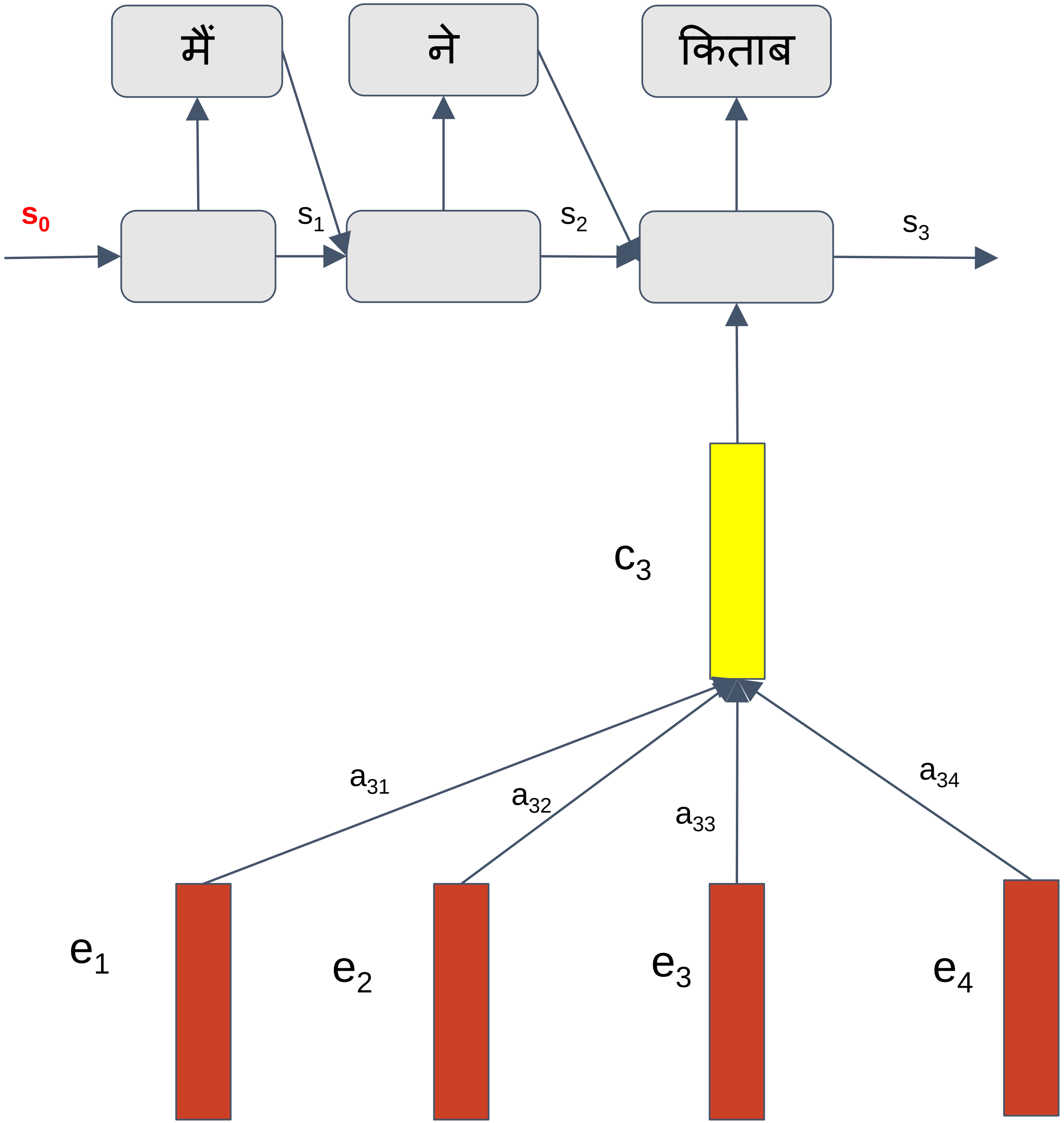


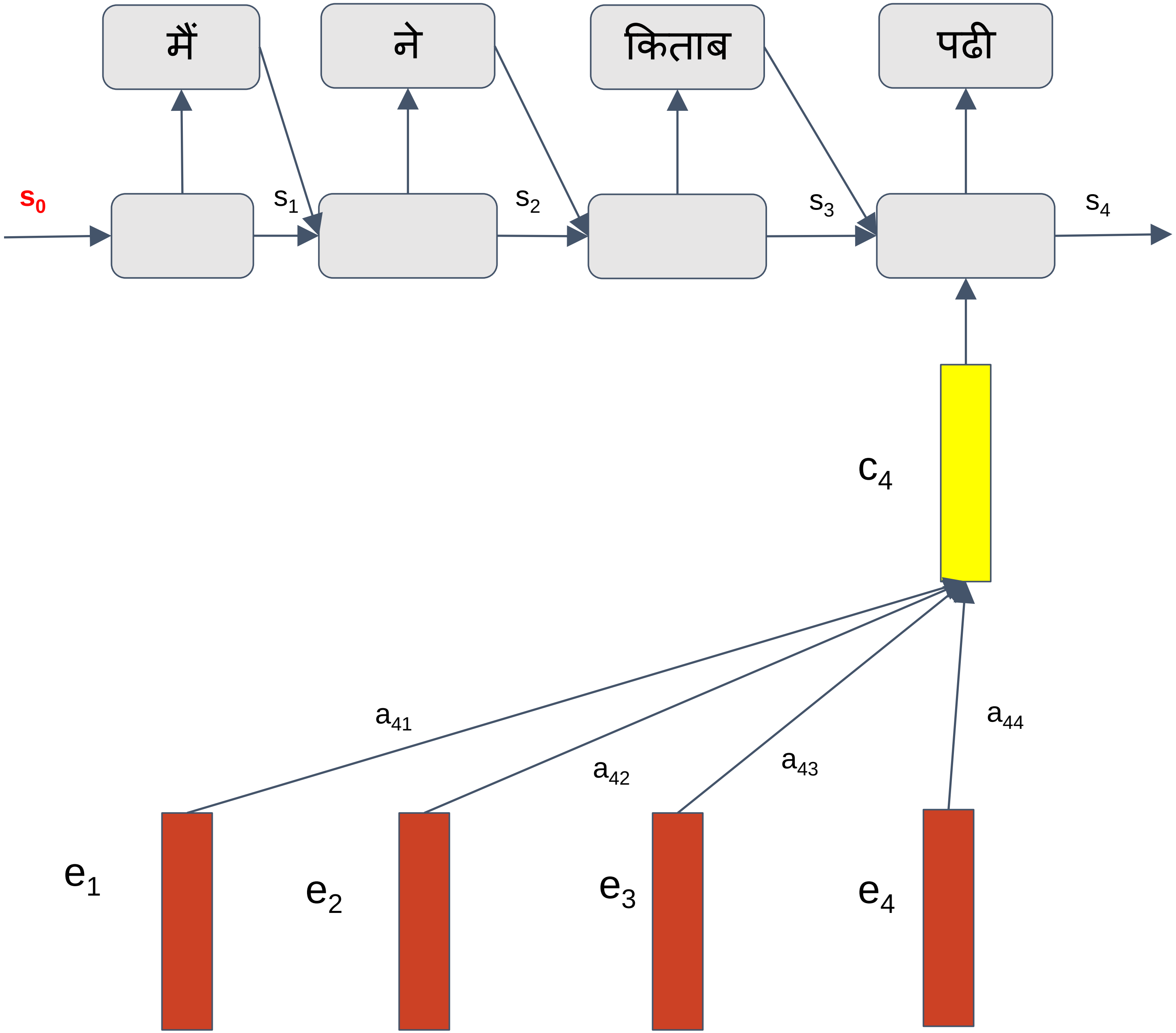
Let's see an example of how the ***attention mechanism*** *works during decoding*

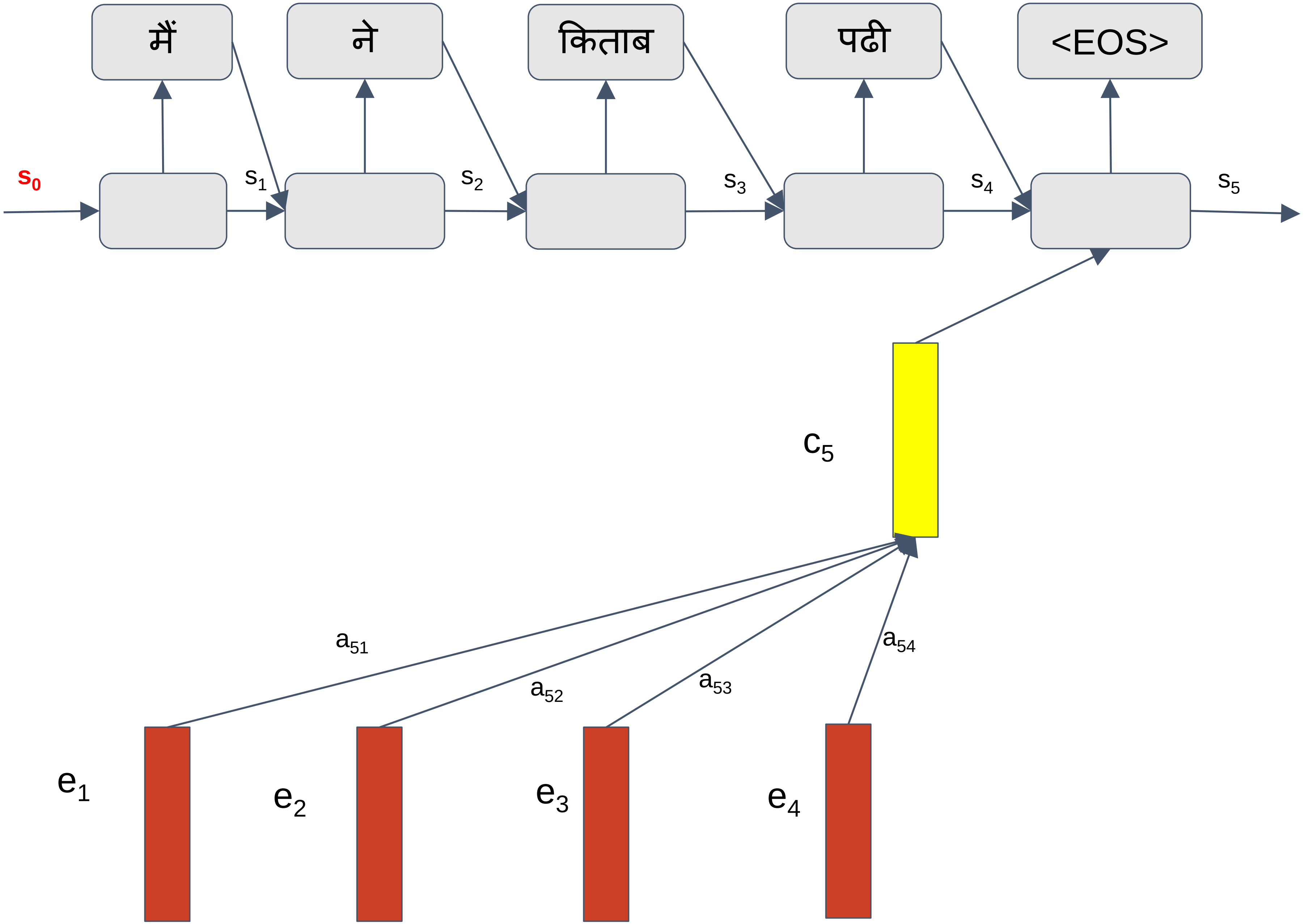
$$c_i = \sum_{j=1}^{n} a_{ij} e_j$$

For generation of ith *output character:*
c_i : context vector
a_{ij} : annotation weight for the jth *annotation vector*
o_j: jth *annotation vector*


मैं
ने
s_0
s_1
s_2
c_2
a_{21}
a_{22}
a_{23}
a_{24}
e_1
e_2
e_3
e_4


मैं
ने
किताब
s_0
s_1
s_2
s_3
c_3
a_{31}
a_{32}
a_{33}
a_{34}
e_1
e_2
e_3
e_4


मैं
ने
किताब
पढी
s_0
s_1
s_2
s_3
s_4
c_4
a_{41}
a_{42}
a_{43}
a_{44}
e_1
e_2
e_3
e_4


मैं
ने
किताब
पढी
<EOS>
s_0
s_1
s_2
s_3
s_4
s_5
c_5
a_{51}
a_{52}
a_{53}
a_{54}
e_1
e_2
e_3
e_4

How do we find the attention weights?

Let the training data help you decide!!

Idea: Pick the attention weights that maximize the overall translation likelihood accuracy

Scoring function **g** to match the encoder and decoder states

$$\alpha_{ij} = g(s_{j-1}, e_i, \mathbf{emb}(y_{j-1}))$$

$$a_{ij} = \frac{\exp(\alpha_{ij})}{\sum_{i=1}^{i=N} \exp(\alpha_{kj})}$$

$$c_j = \sum_{i=1}^{i=N} a_{ij} e_i$$

How do we find the attention weights?

Let the training data help you decide!!

Idea: Pick the attention weights that maximize the overall translation likelihood accuracy

$$\alpha_{ij} = g(s_{j-1}, e_i, \mathbf{emb}(y_{j-1}))$$

g can be a feedforward network or a similarity metric like dot product

$$a_{ij} = \frac{\exp(\alpha_{ij})}{\sum_{i=1}^{i=N} \exp(\alpha_{kj})}$$

$$c_j = \sum_{i=1}^{i=N} a_{ij} e_i$$

How do we find the attention weights?

Let the training data help you decide!!

Idea: Pick the attention weights that maximize the overall translation likelihood accuracy

$$\alpha_{ij} = g(s_{j-1}, e_i, \mathbf{emb}(y_{j-1}))$$

Normalize score to obtain attention weights

$$a_{ij} = \frac{\exp(\alpha_{ij})}{\sum_{i=1}^{i=N} \exp(\alpha_{kj})}$$

$$c_j = \sum_{i=1}^{i=N} a_{ij} e_i$$

How do we find the attention weights?

Let the training data help you decide!!

Idea: Pick the attention weights that maximize the overall translation likelihood accuracy

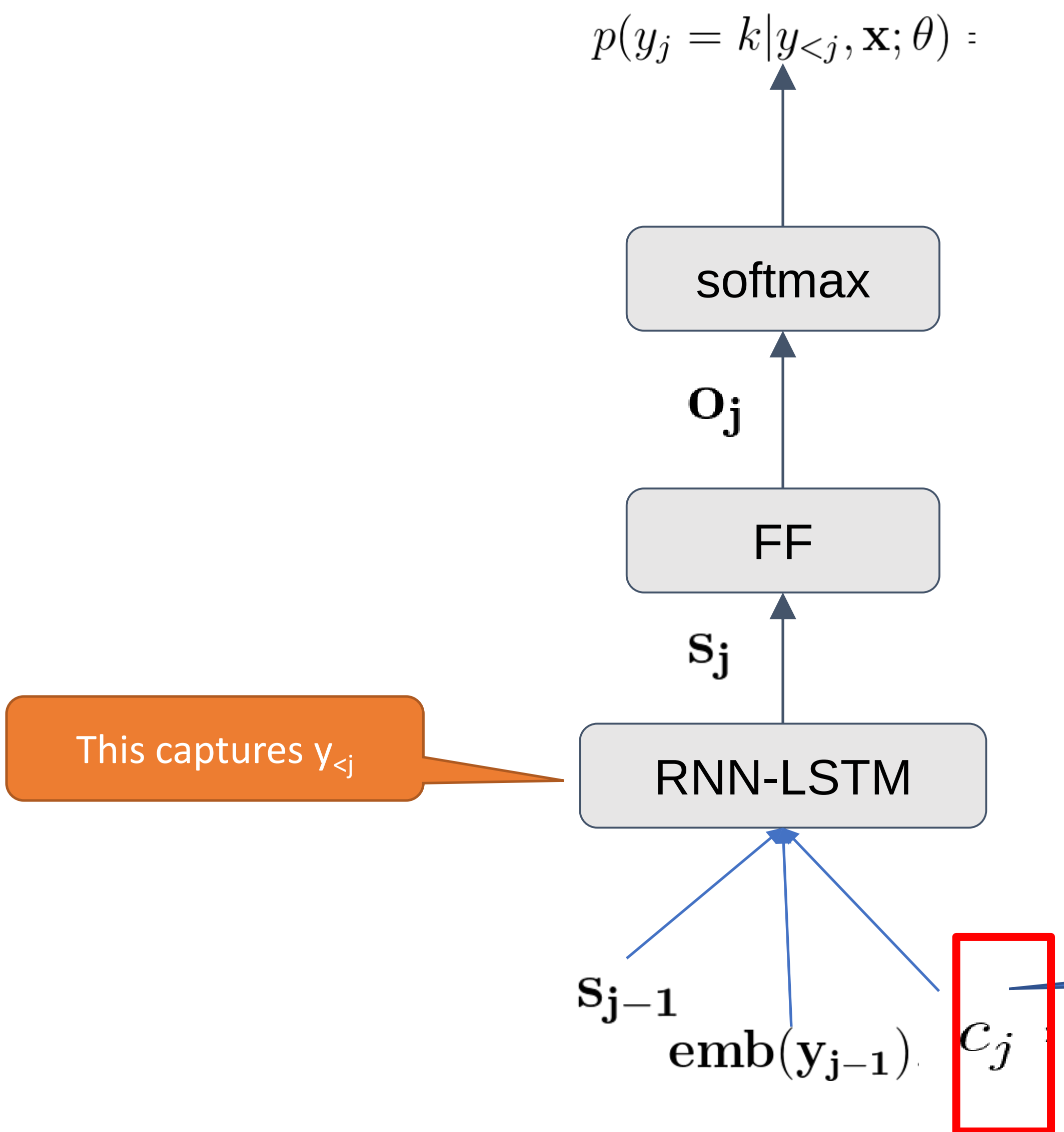
$$\alpha_{ij} = g(s_{j-1}, e_i, \mathbf{emb}(y_{j-1}))$$

$$a_{ij} = \frac{\exp(\alpha_{ij})}{\sum_{i=1}^{i=N} \exp(\alpha_{kj})}$$

$$c_j = \sum_{i=1}^{i=N} a_{ij} e_i$$

Final context vector is weighted average of encoder outputs

Let us revisit what the decoder does at time step t



This captures x

$$softmax(o_{jk}) = \frac{\exp(o_{jk})}{\sum_{m=0}^{m=T} \exp(o_{jm})}$$

$$\mathbf{o_j} = FF(s_j)$$

$$\mathbf{s_j} = g(\mathbf{s_{j-1}}, \mathbf{emb(y_{j-1})}, \mathbf{c})$$

Choice of Attention Scoring Function

Feedforward	: $\alpha_{ij} = \boldsymbol{W_a}[e_j; s_i]$
Dot Product	: $\alpha_{ij} = s_i^T e_j$
Scaled Dot Product	: $\alpha_{ij} = \frac{s_i^T e_j}{\sqrt{\lvert e_j \rvert}}$
Multiplicative Attention	: $\alpha_{ij} = s_i^T \boldsymbol{W_a} e_j$
Additive Attention	: $\alpha_{ij} = \boldsymbol{W_1} s_i + \boldsymbol{W_2} e_j$

Effective Approaches to Attention-based Neural Machine Translation. Thang Luong, Hieu Pham, Christopher D. Manning. EMNLP 2015.

Attention is a general and important concept in Deep learning

Given a set of VALUES → select a summary of the values that is relevant to a QUERY

Each VALUE represented by a KEY → the QUERY is matched to the KEY (content similarity)

Select a summary with different focus on different values → Weighted average

Associative memory read + selection

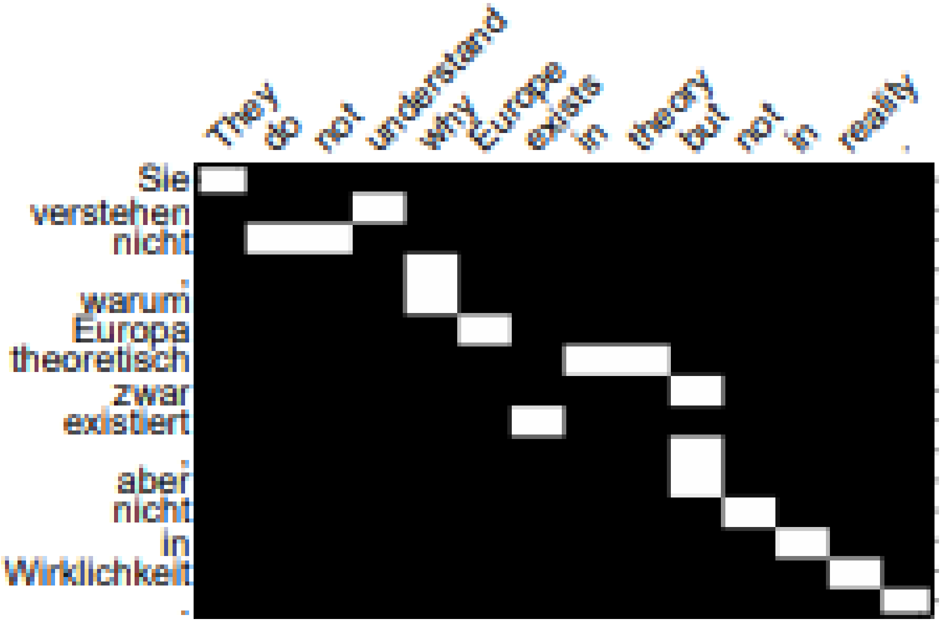
For MT

QUERY: decoder state

VALUE, KEY: encoder annotation vector

Benefits of Attention

- Significant improves in NMT quality
  - Performs better on long sentences
  - Word-order is no longer a major issue for
  - Used in all NMT systems
- Attention provides some interpretability
  - Attention!=Alignment
- There is more to attention



https://arxiv.org/pdf/1508.04025.pdf

A lot of interesting work with attention

- Pointer Networks
- Pointer Generator Networks
- Modeling Coverage
- Learning word-alignments

Benefits of Neural MT

~~Limitations of SMT~~

- *No end-to-end optimization* ➔ *Single optimization objective that accounts for alignment, word reordering*
- *Divergent word-order is a big challenge* ➔ *Attention mechanism*
- *n-gram LM not the best way to score translation fluency*

  ➔ *Target conditional-RNN LM (no standalone LM)*
- *Model size is a function of the data size*

  ➔ *Give architecture, model size is fixed*

Benefits of Neural MT

Opens-up new possibilities

- *Multi-source translation models*
- *Transfer Learning*
- *Cross-lingual/Multilingual MNMT*
- *Unsupervised NMT*
- *Multimodal translation*
- *End-to-End Speech-to-Speech Translation*
- *Document-level Translation*

More Reading Material

This was a small introduction, you can find mode elaborate presentations, books and further references below:

SMT Tutorials & Books

- *Machine Learning for Machine Translation (An Introduction to Statistical Machine Translation)*. **Tutorial at ICON 2013** [slides]
- *Machine Translation: Basics and Phrase-based SMT*. **Talk at the Ninth IIIT-H Advanced Summer School on NLP (IASNLP 2018), IIIT Hyderabad** . [pdf]
- Statistical Machine Translation. Philip Koehn. Cambridge University Press. 2008. [site]
- Machine Translation. Pushpak Bhattacharyya. CRC Press. 2015. [site]

NMT Tutorials & Books

- Neural Machine Translation and Sequence-to-sequence Models: A Tutorial. Graham Neubig. 2017. [pdf]
- CMU CS 11-731, Fall 2019 - Machine Translation and Sequence-to-Sequence Models. [link]
- Neural Machine Translation: A Review and Survey. Felix Stahlberg. JAIR. 2020.
- Raj Dabre, Chenhui Chu, and Anoop Kunchukuttan. 2020. A Survey of Multilingual Neural Machine Translation. ACM Computing Surveys. [COLING 2020 Tutorial Material]

Other Lectures

- https://github.com/oxford-cs-deepnlp-2017/lectures (Lectures 7 & 8)
- https://www.cse.iitm.ac.in/~miteshk/CS6910.html (Lectures 16)
- http://web.stanford.edu/class/cs224n/ (Lectures 7)

Tools

- **moses**: A production-quality open source package for SMT
- **fairseq**: Modular and high-performance NMT system based on PyTorch
- **openNMT-pytorch**: Modular NMT system based on PyTorch
- **marian**: High-performance NMT system written in C++
- **subword-nmt**: BPE tokenizer
- **sentencepiece**: Subword tokenizer implementing BPE and word-piece
- **indic-nlp-library**: Python library for processing Indian language datasets
- **sacrebleu**: MT evaluation tool

Datasets

- Workshop on Machine Translation datasets
- Workshop on Asian Translation datasets
- Samanatar Parallel Corpus
- IITB English-Hindi Parallel Corpus
- ILCI parallel corpus
- WAT-Indic Languages Multilingual Parallel

More parallel corpora and resources for Indian languages can be found here:

https://github.com/indicnlpweb/indicnlp_catalog

Thank You!

anoop.kunchukuttan@gmail.com

http://anoopk.in

Extra Study Material

Phrase-based SMT Enhancements

We have looked at a basic phrase-based SMT system

This system can learn word and phrase translations from parallel corpora

But many important linguistic phenomena need to be handled

- **Divergent Word Order**
- Rich morphology
- Named Entities and Out-of-Vocabulary words

Getting word order right

Phrase based MT is not good at learning word ordering

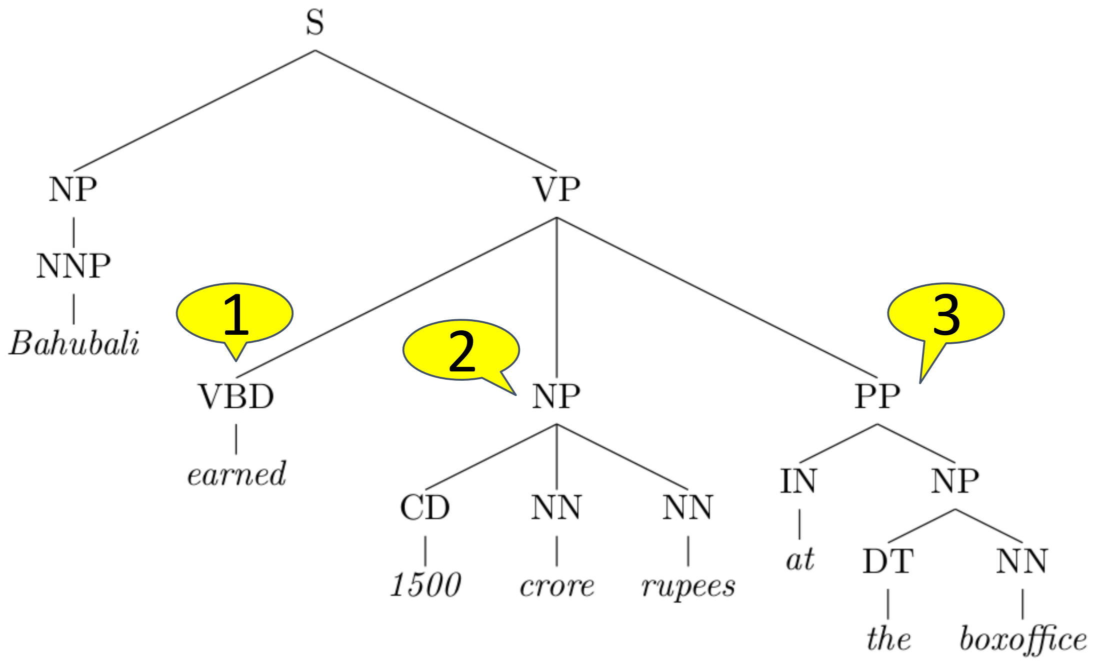
Solution: Let’s help PB-SMT with some preprocessing of the input

Change order of words in input sentence to match order of the words in the target language

Let’s take an example

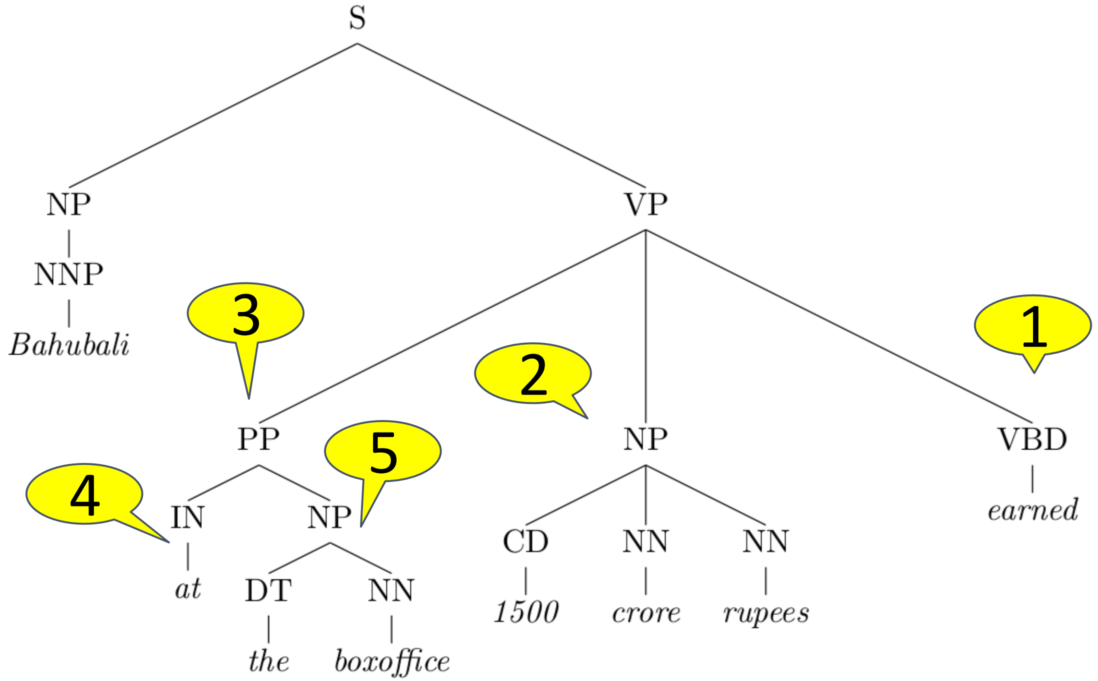
Bahubali earned more than 1500 crore rupees at the boxoffice

Parse the sentence to understand its syntactic structure



Apply rules to transform the tree

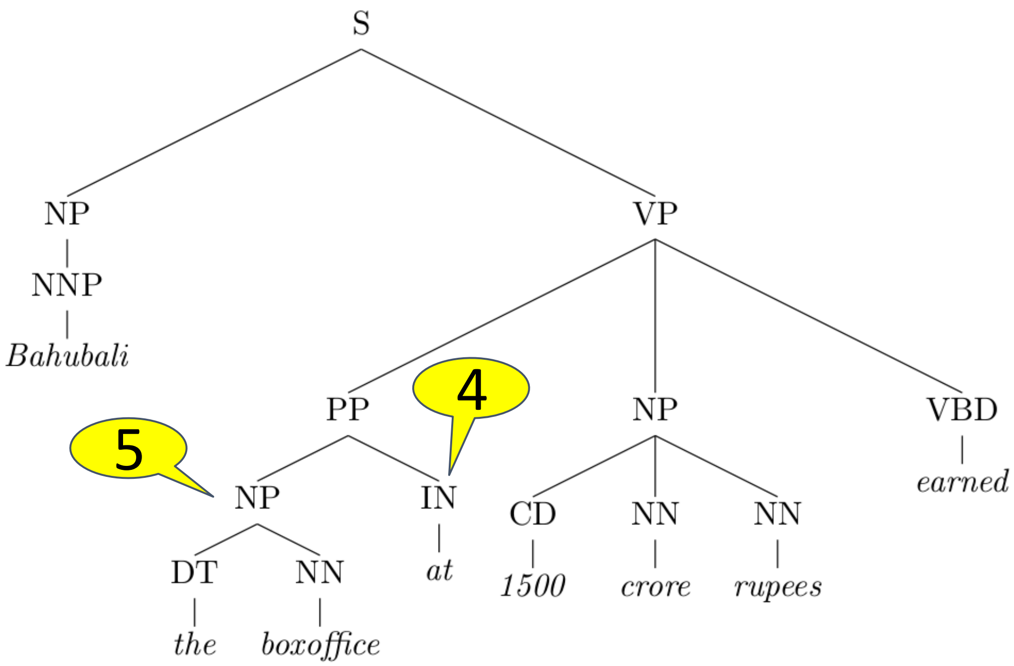
VP → VBD NP PP ⇒ VP → PP NP VBD



This rule captures Subject-Verb-Object to Subject-Object-Verb divergence

Prepositions in English become postpositions in Hindi

PP → IN NP ⇒ PP → NP IN



The new input to the machine translation system is

Bahubali the boxoffice at 1500 crore rupees earned

Now we can translate with little reordering

बाहुबली ने बॉक्सओफिस पर 1500 करोड रुपए कमाए

These rules can be written manually or learnt from parse trees

Addressing Rich Morphology

Inflectional forms of the Marathi word घर

घर	house
घरात	in the house
घरावरती	on the house
घराखाली	below the house
घरामध्ये	in the house
घरामागे	behind the house
घराचा	of the house
घरामागचा	that which is behind the house
घरासमोर	in front of the house
घरासमोरचा	that which is in front of the house
घरांसमोर	in front of the houses

Hindi words with the suffix वाद

साम्यवाद	communism
समाजवाद	socialism
पूंजीवाद	capitalism
जातीवाद	casteism
साम्राज्यवाद	imperialism

The corpus should contains all variants to learn translations

This is infeasible!

Language is very productive, you can combine words to generate new words

Addressing Rich Morphology

Inflectional forms of the Marathi word घर

घर	house
घर ा त	in the house
घर ा वरती	on the house
घर ा खाली	below the house
घर ा मध्ये	in the house
घर ा मागे	behind the house
घर ा चा	of the house
घर ा माग चा	that which is behind the house
घर ा समोर	in front of the house
घर ा समोर चा	that which is in front of the house
घर ा ं समोर	in front of the houses

Hindi words with the suffix वाद

साम्य वाद	communism
समाज वाद	socialism
पूंजी वाद	capitalism
जाती वाद	casteism
साम्राज्य वाद	imperialism

- *Break the words into its component morphemes*
- *Learn translations for the morphemes*
- *Far more likely to find morphemes in the corpus*

Handling Names and OOVs

Some words not seen during train will be seen at test time
These are out-of-vocabulary (OOV) words

***Names** are one of the most important category of OOVs*
⇒ There will always be names not seen during training

How do we translate names like Sachin Tendulkar to Hindi?
What we want to do is map the Roman characters to Devanagari to they sound the same when read → सचिन तेंदुलकर
➔ *We call this process **'transliteration'***

Can be seen as a simple translation problem at character level with no re-ordering

s a c h i n →स च ि न

Encoder-Decoder Models: Part 2

Machine Translation & Sequence-2-Sequence Tasks

Anoop Kunchukuttan
Microsoft, MT Group, Hyderabad
anoop.kunchukuttan@gmail.com

For CS772 at IIT Bombay, Feb 2022

Course Instructor: Prof. Pushpak Bhattacharyya

Questions?

Outline

- Introduction
- Statistical Machine Translation
- **Neural Machine Translation**
- Evaluation of Machine Translation
- Transformers
- Multilingual Neural Machine Translation

The models discussed so far do not use monolingual data

Can monolingual data help improve NMT models?

Backtranslation

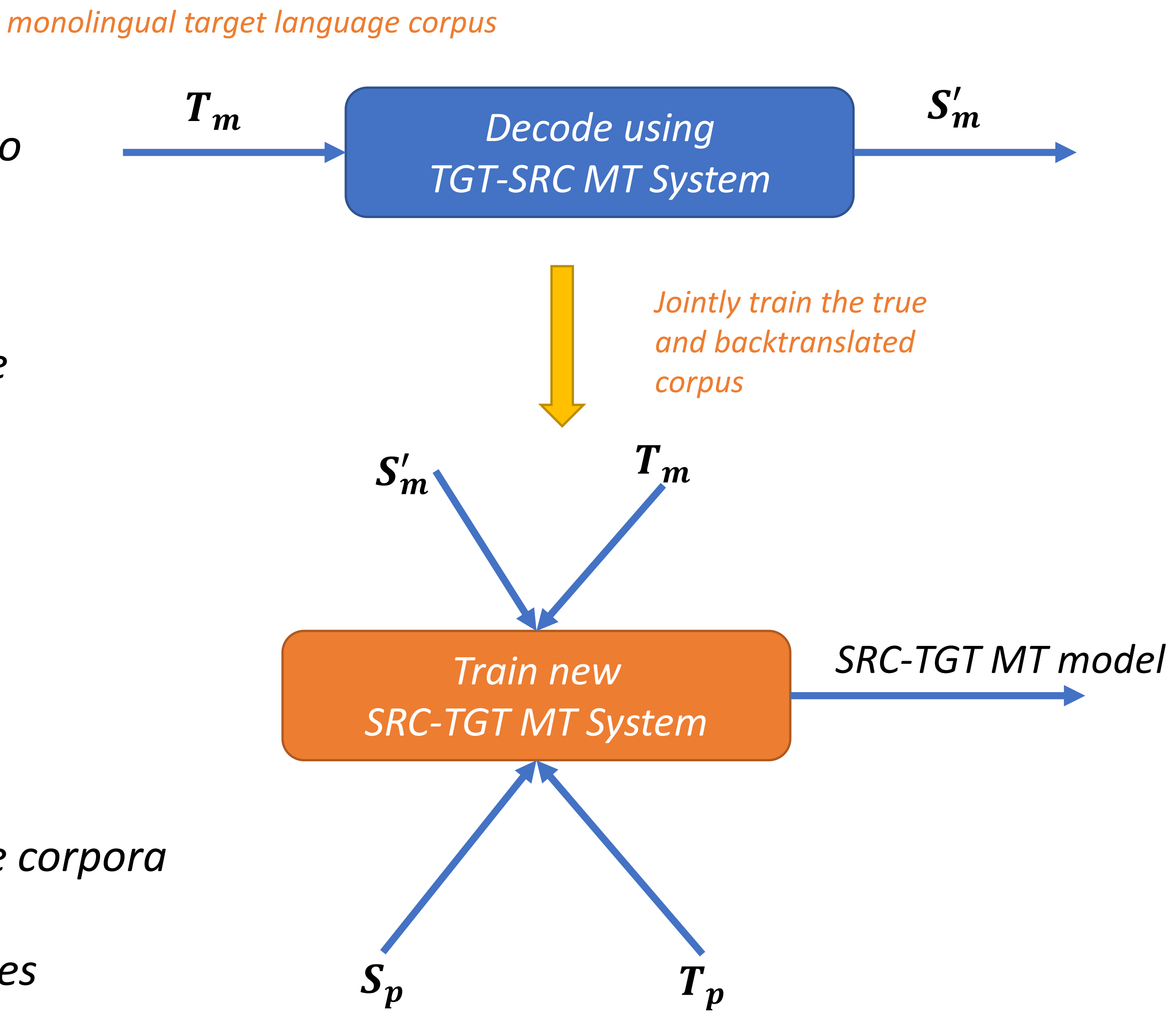
Create pseudo-parallel corpus using Target to source model (Backtranslated corpus)

Need to find the right balance between true and backtranslated corpus

Why is backtranslation useful?

- *Target side language model improves*
  - *target side is clean*
- *Adaptation to target language domain*
- *Prevent overfitting by exposure to diverse corpora*

Particularly useful for low-resource languages



monolingual target language corpus

T_m

Decode using TGT-SRC MT System

S'_m

Jointly train the true and backtranslated corpus

S'_m

T_m

Train new SRC-TGT MT System

SRC-TGT MT model

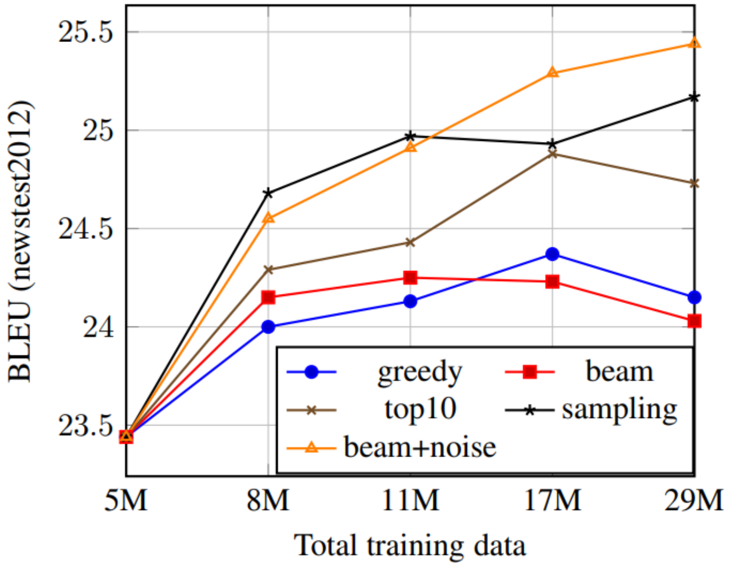
S_p

T_p

Sennrich, Rico, Barry Haddow, and Alexandra Birch. "Improving neural machine translation models with monolingual data." ACL 2016

Make backtranslation more diverse

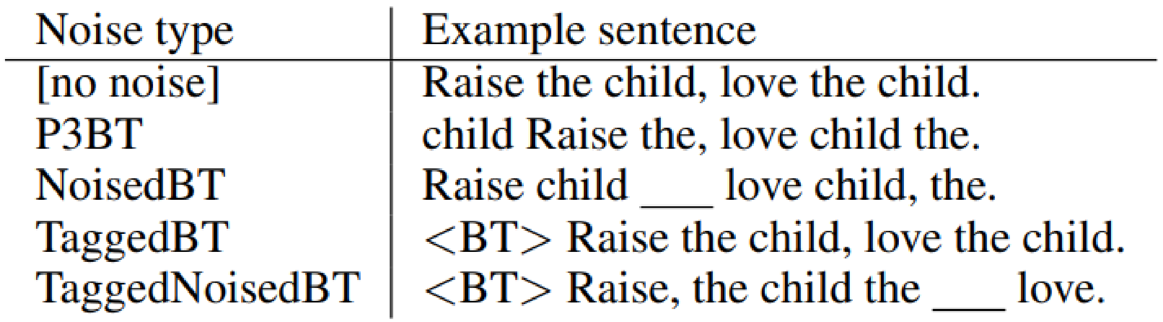
- *Sampling*
- *Restricted Sampling*
- *Beam+noising*



Make it easy for the model to distinguish between natural & synthetic input

Tagged Backtranslation ➔

add a special token indicating that the input is synthetic



Noise type	Example sentence
[no noise]	Raise the child, love the child.
P3BT	child Raise the, love child the.
NoisedBT	Raise child ___ love child, the.
TaggedBT	<BT> Raise the child, love the child.
TaggedNoisedBT	<BT> Raise, the child the ___ love.

Tagged BT and Noised BT serve the same purpose ➔ distinguishing inputs

Sergey Edunov, Myle Ott, Michael Auli, David Grangier . Understanding Back-Translation at Scale. EMNLP 2018.
Isaac Caswell, Ciprian Chelba, David Grangier. Tagged Back-Translation. WMT 2019

Self Training

Create pseudo-parallel corpus using initial source to target model (Forward translated corpus)

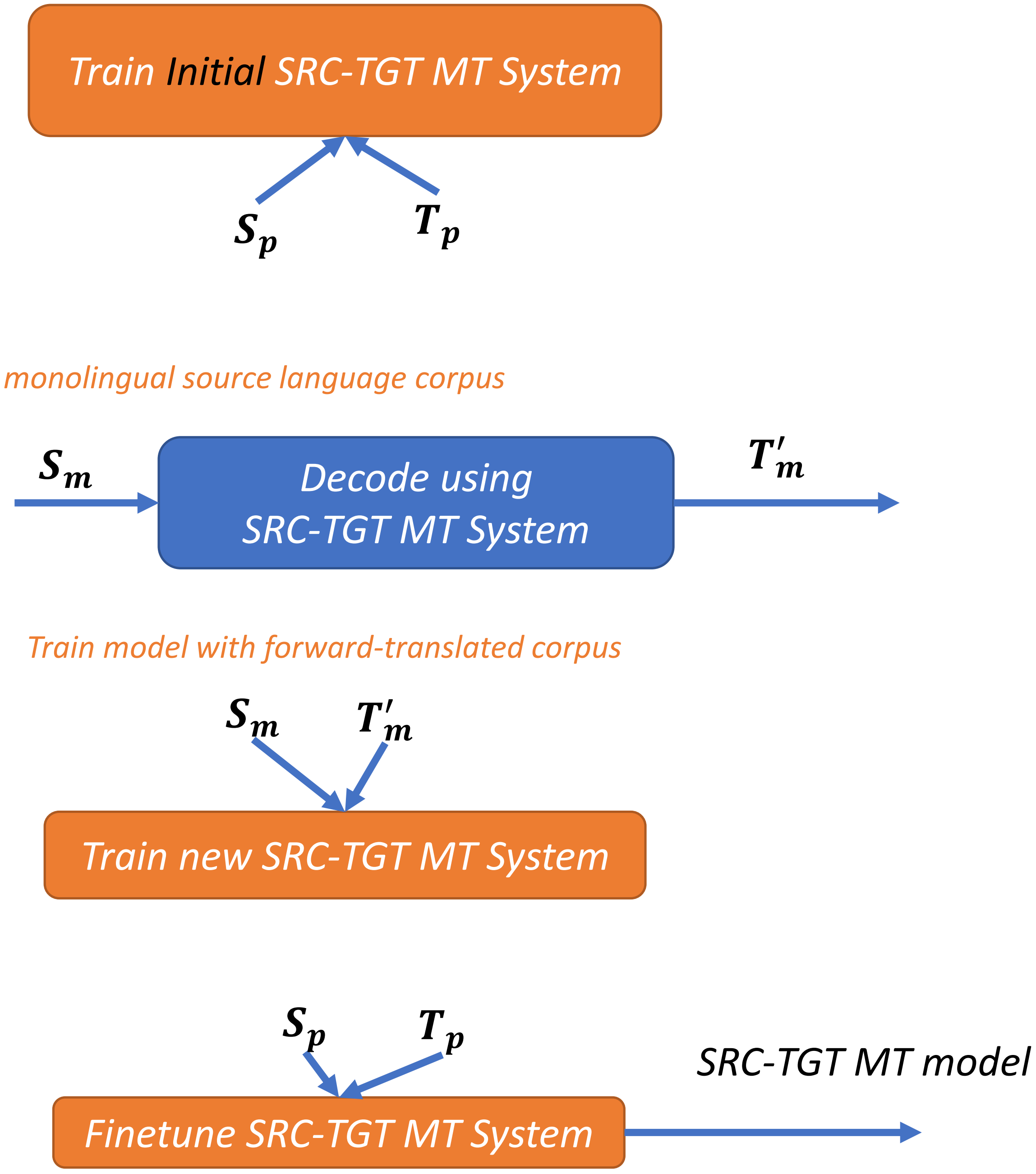
Target side of pseudo-parallel corpus is noisy

- *Train the S-T mode on pseudo-parallel corpora*
- *Tune on true parallel corpora*
- *(Noising the input helps, use beam search)*

Why is self-training useful?

- *Noise plays an important role*
- *Adaptation to source language domain*
- *Prevent overfitting by exposure to diverse corpora*

Works well if the initial model is reasonably good


Train Initial SRC-TGT MT System
S_p
T_p
monolingual source language corpus
S_m
Decode using SRC-TGT MT System
T'_m
Train model with forward-translated corpus
S_m
T'_m
Train new SRC-TGT MT System
S_p
T_p
SRC-TGT MT model
Finetune SRC-TGT MT System

Junxian He, Jiatao Gu, Jiajun Shen, Marc'Aurelio Ranzato. Revisiting Self-Training for Neural Sequence Generation. ICLR 2020.

Topics

- *Why NMT?*
- *Encoder-Decoder Models*
- *Attention Mechanism*
- *Backtranslation*
- *Subword-level Models*

The Vocabulary Problem

- **The input & output embedding layers are finite**
  - How to handle an open vocabulary?
  - How to translate named entities?

- **Softmax computation at the output layer is expensive**
  - Proportional to the vocabulary size

$$softmax(o_{jk}) = \frac{\exp(o_{jk})}{\sum\limits_{m=0}^{m=T} \exp(o_{jm})}$$

Subword-level Translation

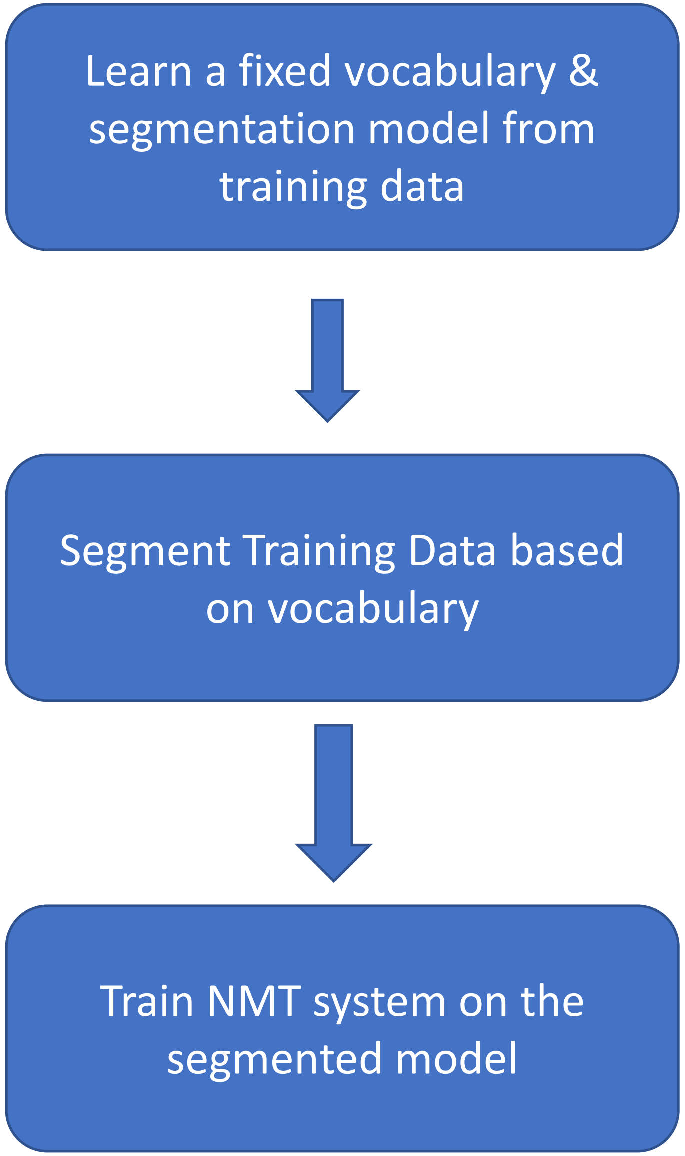
Original sentence: प्रयागराज में 43 दिनों तक चलने वाला माघ मेला आज से शुरू हो गया है

Possible inputs to NMT system:

- प्रयाग @@राज में 43 दि @@नों तक चल @@ने वाला माघ मेला आज से शुरू हो गया है
- प्र या ग रा ज _में _43 _दि नों _त क _च ल ने _ वा ला _मा घ मे ला _आज _से _शुरू _हो _गया _है

Obvious Choices: Character, Character n-gram, Morphemes ➔ They all have their flaws!

The New Subword Representations: Byte-Pair Encoding, Unigram (implemented in SentencePiece package)



Learn a fixed vocabulary & segmentation model from training data

Segment Training Data based on vocabulary

Train NMT system on the segmented model

{प्रयाग, राज, में दि, नों, तक, चल, ने}

vocabulary

{प्रयाग राज}
{च ल}
{चल, ने}

Segmentation model

प्रयाग@@राज में 43 दि@@नों तक चल@@ने वाला माघ मेला आज से शुरू हो गया है

- Every word can be expressed as a concatenation of subwords
- A small subword vocabulary has good representative power
    - 4k to 64k depending on the size of the parallel corpus
- Most frequent words should not be segmented

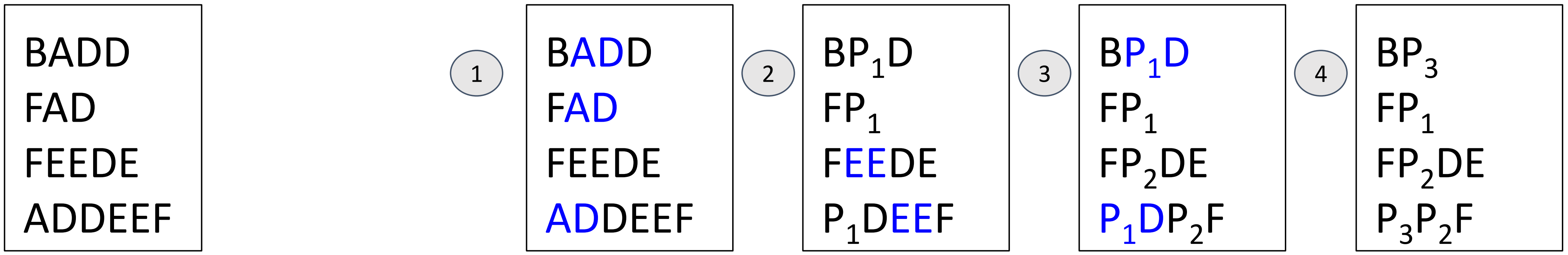
Byte Pair Encoding

Byte Pair Encoding is a greedy compression technique (Gage, 1994)

Number of BPE merge operations=3
Vocab: A B C D E F

P_1=AD P_2=EE P_3=P_1D

Words to encode *Iterations*



BADD
FAD
FEEDE
ADDEEF

1

BADD
FAD
FEEDE
ADDEEF

2

BP_1D
FP_1
FEEDE
P_1DEEF

3

BP_1D
FP_1
FP_2DE
P_1DP_2F

4

BP_3
FP_1
FP_2DE
P_3P_2F

Data-dependent segmentation

- Inspired from compression theory
- MDL Principle *(Rissansen, 1978)* ⇒ Select segmentation which maximizes data likelihood

Problems with subword level translation

Unwanted splits:

नाराज़ → ना राज़ ➔ no secret

Problem is exacerbated for:

- Named Entities
- Rare Words
- Numbers

Explore multiple subword segmentations

- BPE dropout
- Unigram + subword-regularization

Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates. Taku Kudo. ACL 2018.
BPE-Dropout: Simple and Effective Subword Regularization Ivan Provilkov, Dmitrii Emelianenko, Elena Voita. ACL 2020

Outline

- Introduction
- Statistical Machine Translation
- Neural Machine Translation
- **Evaluation of Machine Translation**
- Transformers
- Multilingual Neural Machine Translation

Evaluation of Machine Translation

Evaluation of MT output

- How do we judge a good translation?
- Can a machine do this?
- Why should a machine do this?
  - Because human evaluation is time-consuming and expensive!
  - Not suitable for rapid iteration of feature improvements

What is a good translation?

Evaluate the quality with respect to:

- **Adequacy**: How good the output is in terms of preserving content of the source text
- **Fluency**: How good the output is as a well-formed target language entity

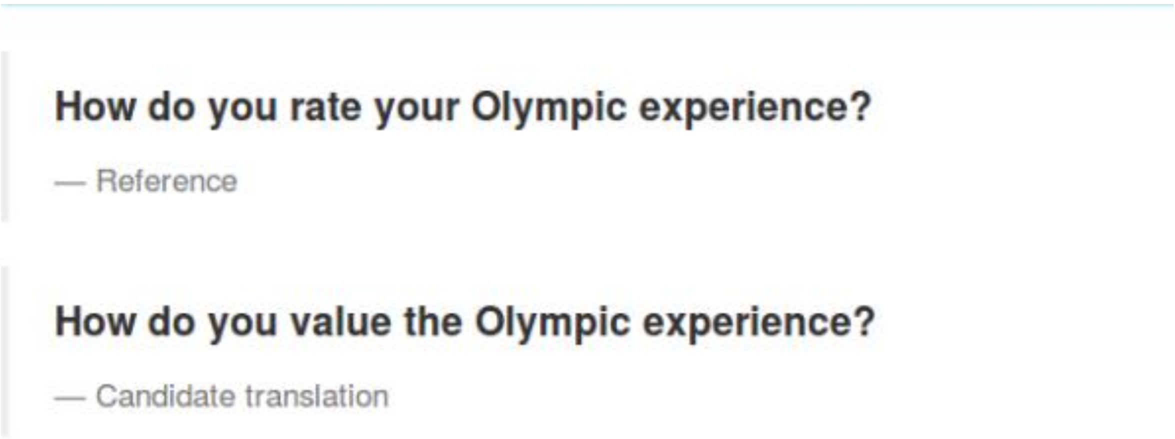
For example, I am attending a lecture

मैं एक व्याख्यान बैठा हूँ
Main ek vyaakhyan baitha hoon
I a lecture sit (Present-first person)
I sit a lecture : Adequate but not fluent

मैं व्याख्यान हूँ
Main vyakhyan hoon
I lecture am
I am lecture: Fluent but not adequate.

Human Evaluation

Direct Assessment



How do you rate your Olympic experience?

— Reference

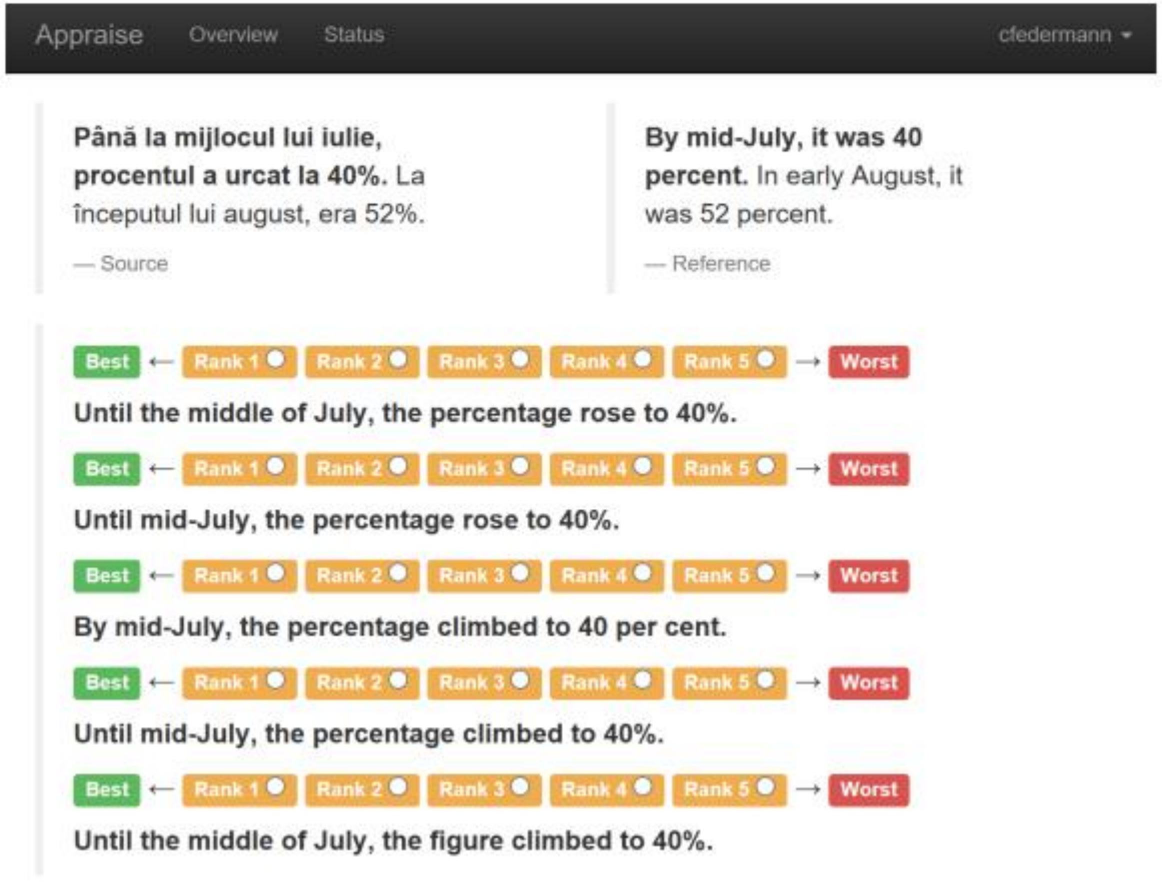
How do you value the Olympic experience?

— Candidate translation

Adequacy: Is the meaning translated correctly?
Fluency: Is the sentence grammatically valid?

Adequacy	**Fluency**
5 = All	5 = Flawless
4 = Most	4 = Good
3 = Much	3 = Non-native
2 = Little	2 = Disfluent
1 = None	1 = Incomprehensible

Ranking Translations



Până la mijlocul lui iulie, procentul a urcat la 40%. La începutul lui august, era 52%.

— Source

By mid-July, it was 40 percent. In early August, it was 52 percent.

— Reference

Best ← Rank 1 ○ Rank 2 ○ Rank 3 ○ Rank 4 ○ Rank 5 ○ → Worst

Until the middle of July, the percentage rose to 40%.

Best ← Rank 1 ○ Rank 2 ○ Rank 3 ○ Rank 4 ○ Rank 5 ○ → Worst

Until mid-July, the percentage rose to 40%.

Best ← Rank 1 ○ Rank 2 ○ Rank 3 ○ Rank 4 ○ Rank 5 ○ → Worst

By mid-July, the percentage climbed to 40 per cent.

Best ← Rank 1 ○ Rank 2 ○ Rank 3 ○ Rank 4 ○ Rank 5 ○ → Worst

Until mid-July, the percentage climbed to 40%.

Best ← Rank 1 ○ Rank 2 ○ Rank 3 ○ Rank 4 ○ Rank 5 ○ → Worst

Until the middle of July, the figure climbed to 40%.

$$\text{score}(S_i) = \frac{1}{|\{S\}|} \sum_{S_j \neq S_i} \frac{\text{wins}(S_i, S_j)}{\text{wins}(S_i, S_j) + \text{wins}(S_j, S_i)}$$

Automatic Evaluation

Human evaluation is not feasible in the development cycle

Key idea of Automatic evaluation:

The closer a machine translation is to a professional human translation, the better it is.

- Given: A corpus of good quality human reference translations
- Output: A numerical “translation closeness” metric
- Given (ref,sys) pair, score = f(ref,sys) ➔ $\mathbb{R}$

  where,
  sys (candidate Translation): Translation returned by an MT system
  ref (reference Translation): ‘Perfect’ translation by humans

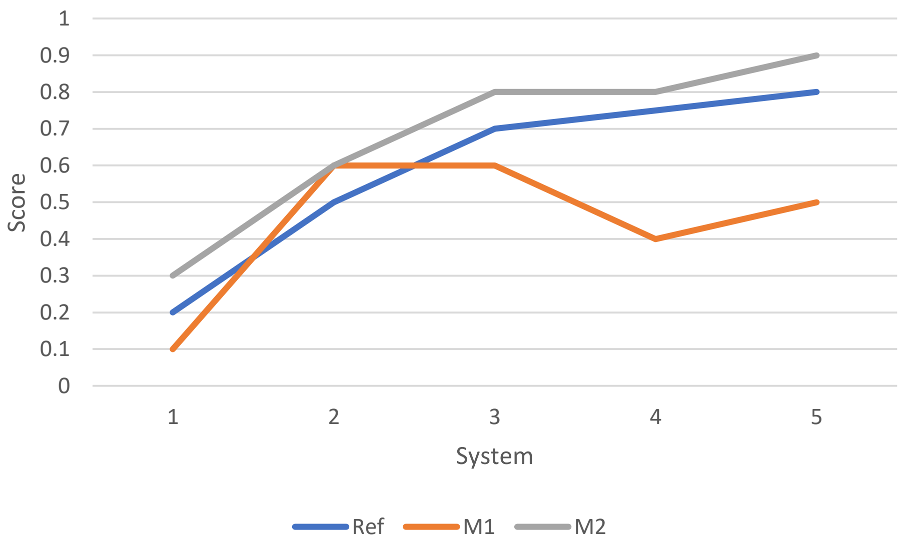
Multiple references are better

Some popular automatic evaluation metrics

- BLEU (Bilingual Evaluation Understudy)
- TER (Translation Edit Rate)
- METEOR (Metric for Evaluation of Translation with Explicit Ordering)

How good is an automatic metric?

How well does it correlate with human judgment?



BLEU

- Most popular MT evaluation metric
- Requires only reference translations
  - No additional resources required
- Precision-oriented measure
- Difficult to interpret absolute values
- Useful to compare two systems

Reference 1: मैने अभी खाना खाया

maine abhi khana khaya

I now food ate

I ate food now.

Candidate 1: **मैने** अब **खाना खाया**

maine ab khana khaya

I *now food ate*

I ate food now

matching unigrams: 3, precision=3/4

matching bigrams: 1, precision=1/3

Weighted average of n-gram precision + Brevity penalty

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a Method for Automatic Evaluation of Machine Translation. ACL 2002.

chrF

- character n-gram F-score
- No additional resources required
- Can address morph-syntactic phenomena
- β controls precision-recall tradeoff ➔ β =2 is widely used

$$\text{CHRF}\beta = (1 + \beta^2) \frac{\text{CHRP} \cdot \text{CHRR}}{\beta^2 \cdot \text{CHRP} + \text{CHRR}}$$

Maja Popović. chrF: character n-gram F-score for automatic MT evaluation. WMT 2015.

Outline

- Introduction
- Statistical Machine Translation
- Neural Machine Translation
- Evaluation of Machine Translation
- **Transformer Architecture**
- Multilingual Neural Machine Translation

Transformer Architecture

Limitations of Recurrent Architectures

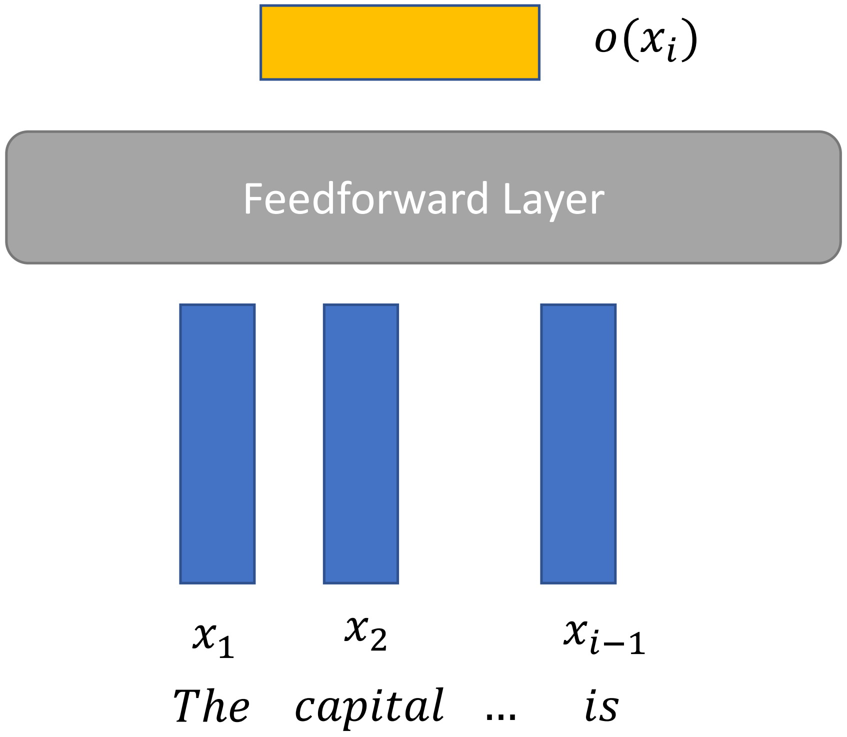
- Elements of a sequence have to processed serially
  - Pro: Number of computations linear in the length of the sequence
  - Con: Encoding time ∝ Length of sequence
- Not effective at modeling long-term dependencies

Revisiting idea → Compare every elements with all other elements

We have already seen this …

Feedforward Network can handle only a limited context

Can we approach this problem differently?

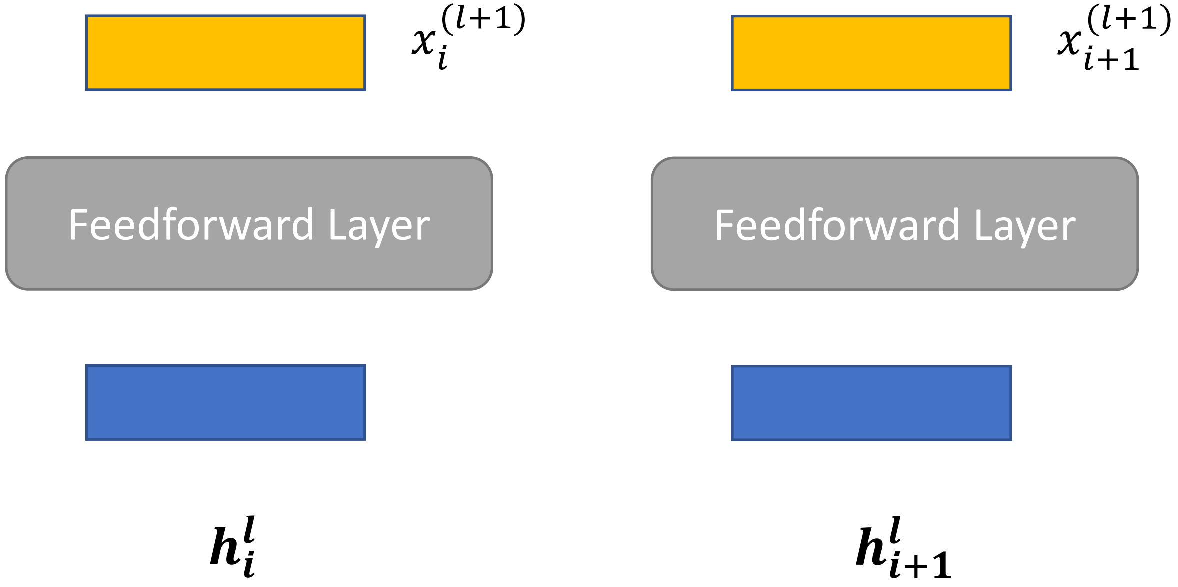


Revisiting idea → Compare every elements with all other elements

Represent the input context as a weighted average of input word embeddings

$$\boldsymbol{h}_i^l = \sum_{i=1}^{N} \boldsymbol{w}_i x_i$$

$$\boldsymbol{x}_i^{l+1} = \boldsymbol{FF}(h_i^l)$$



How do we compute weights → Attention!

Non-recurrent → this operation can be applied in parallel to all elements in the sequence

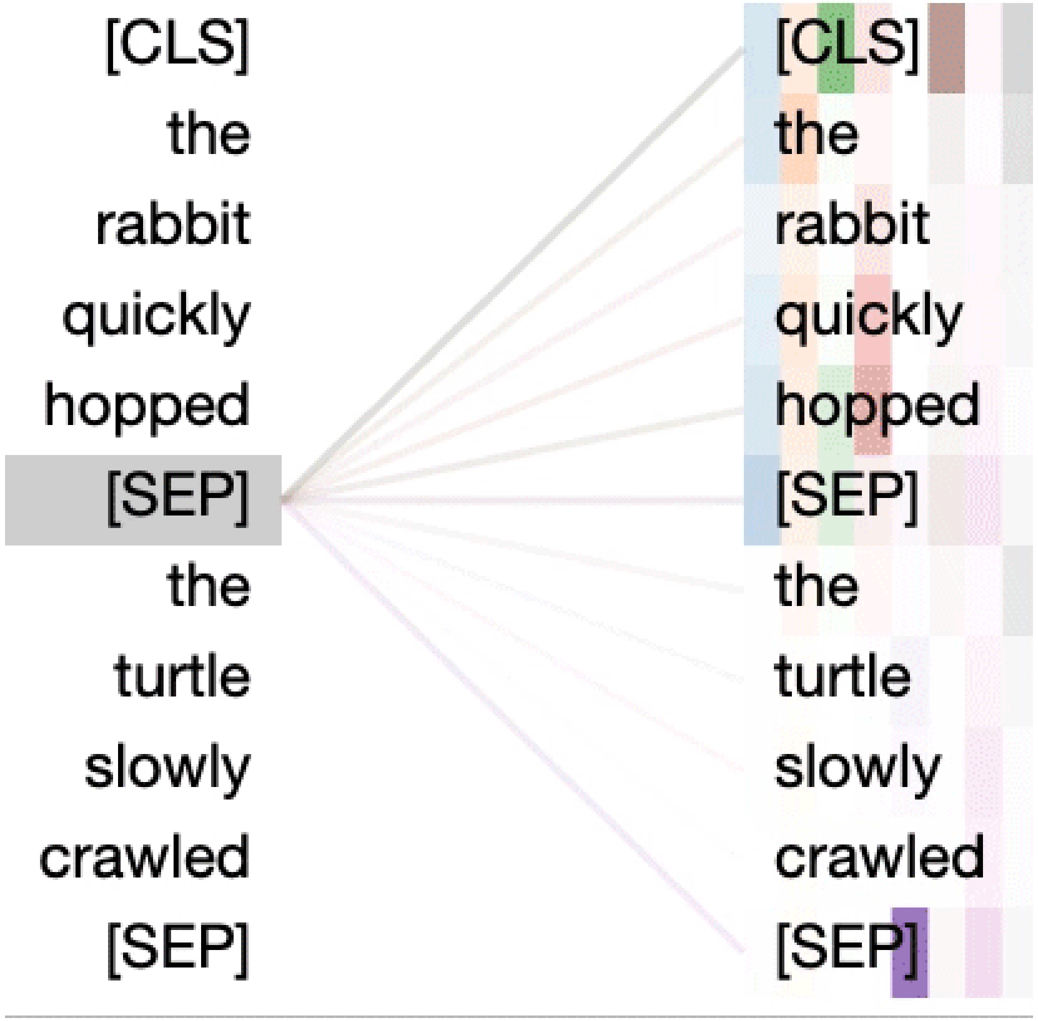
Self-Attention

Every word is compared to every other word in the same sentence

x_i ➔ query

$x_1, x_2\ x_3 \ldots x_n$ ➔ values

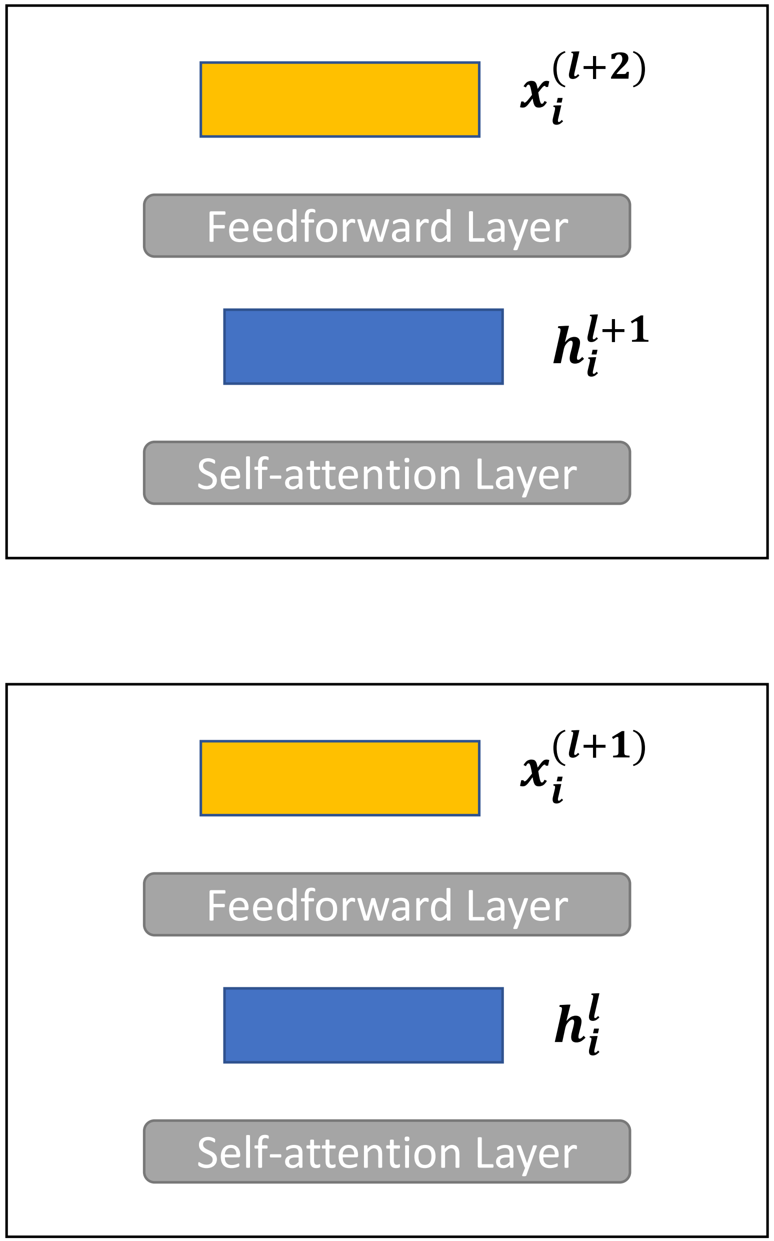
Direct comparison between arbitrary words ➔
long-range dependencies can be better modelled



More computations than Recurrent models: $O(n^2)$

Transformer Architecture

Stack self-attention blocks to create deep networks



Positional Embeddings

The ICICI bank branch is the bank of the river

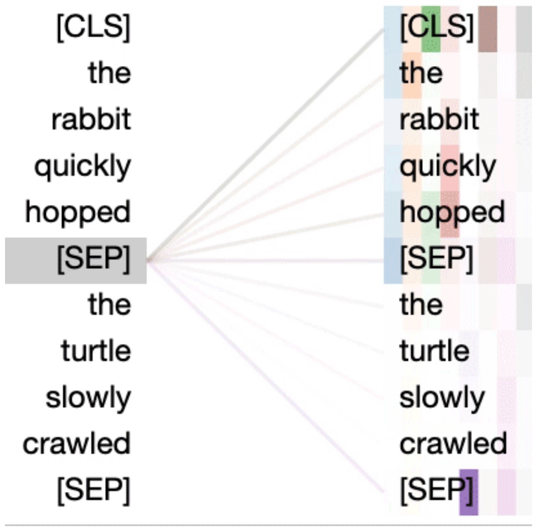
The self-attention model has no notion of position,
➔ same words will have same representations irrespective of their position/syntactic role in the sentence

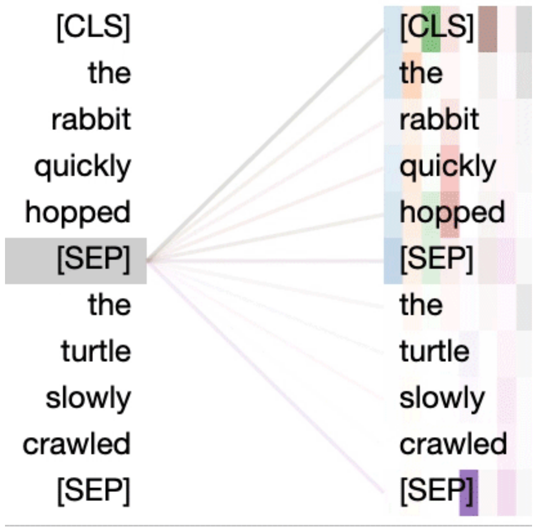
Create positional embeddings that uniquely and deterministically identify a position

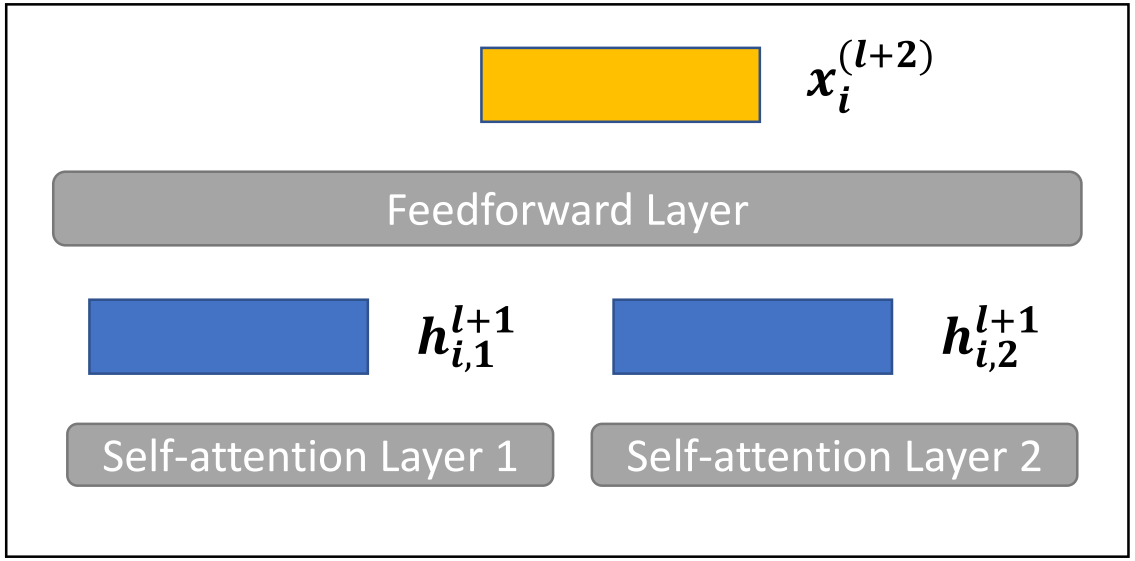
Add it to the word embedding at the bottom layer

https://kazemnejad.com/blog/transformer_architecture_positional_encoding/

Multiple self-attention heads

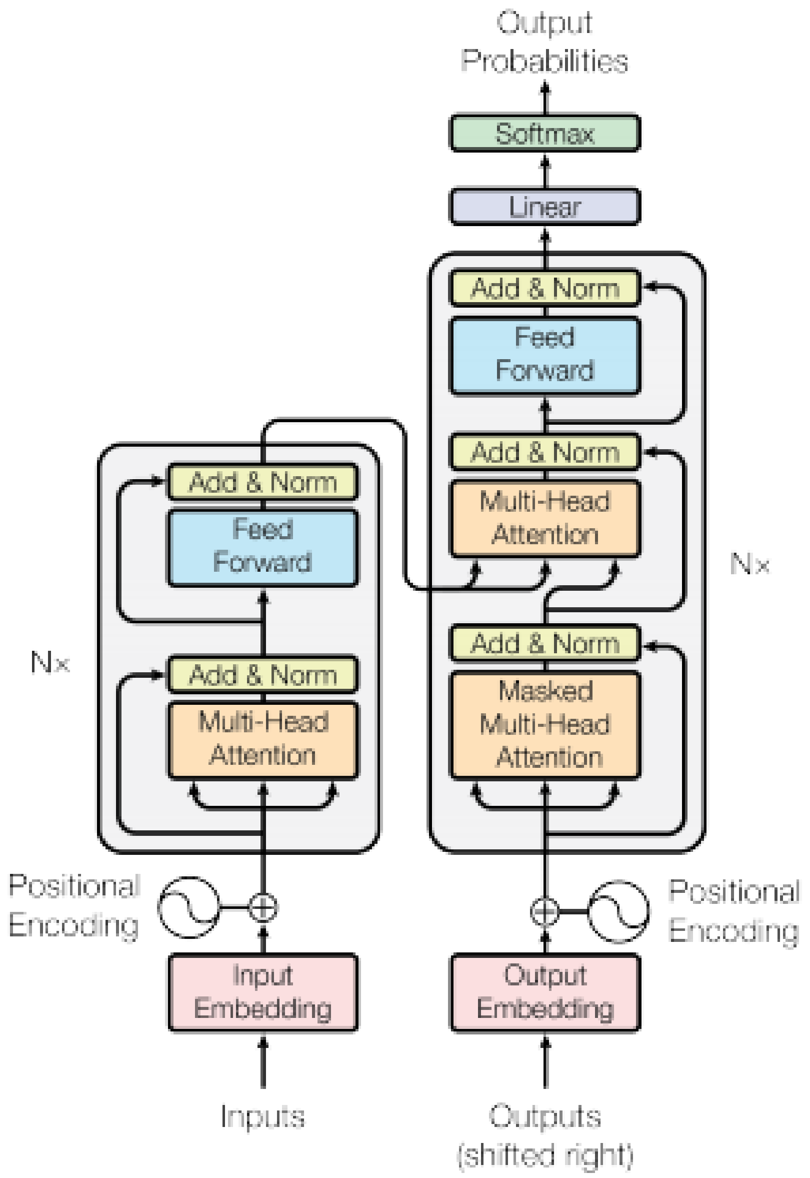






Multiple self-attention networks at each layer

Each head learns different kinds of dependencies



Putting it all together

Decoder layer also has a cross-attention layer

Decoder ➔ masking for future time-steps while computing self-attention

There are residual connections & layer-normalization between layers

http://nlp.seas.harvard.edu/2018/04/03/attention.html
http://jalammar.github.io/illustrated-transformer/

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. "Attention is all you need." NeurIPS (2017).

Transformer has led to tremendous advances in MT

Encoder architectures like BERT based on Transformer have yielded large improvements in NLU tasks

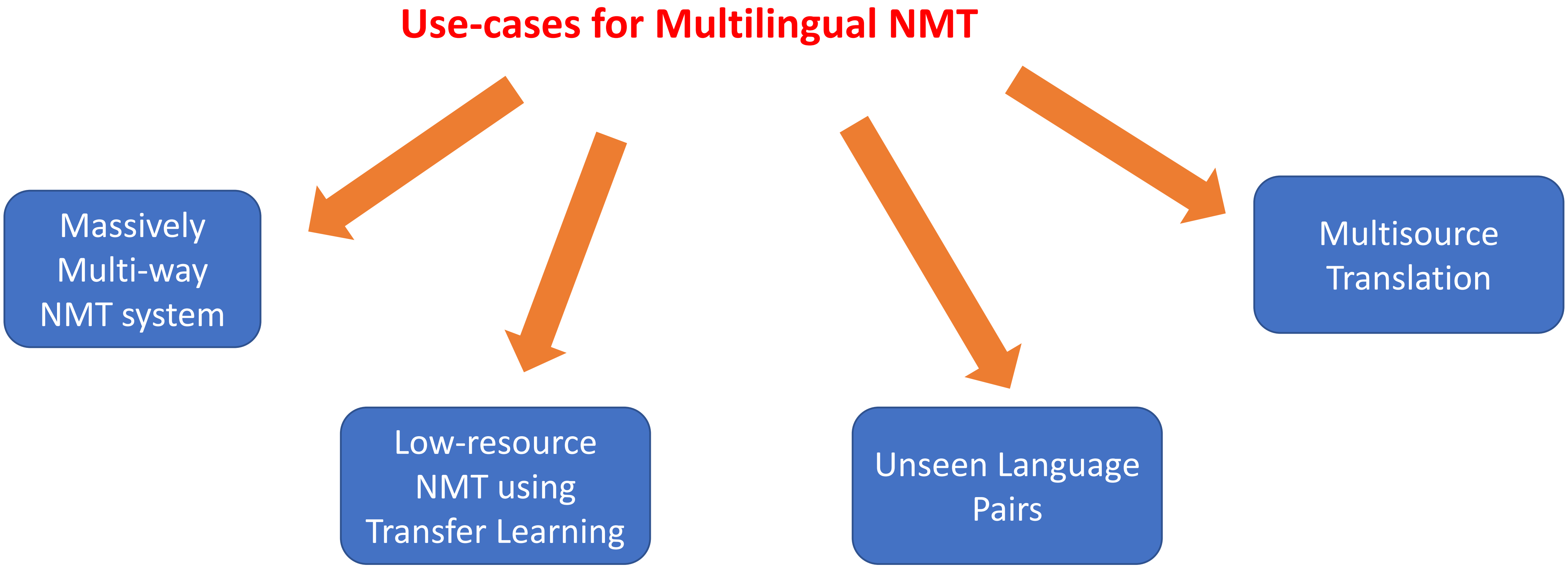
Transformer models are the de-facto standard models for many NLP tasks

Outline

- Introduction
- Statistical Machine Translation
- Neural Machine Translation
- Evaluation of Machine Translation
- Transformer Architecture
- **Multilingual Neural Machine Translation**

Multilingual Neural Machine Translation

NMT Models involving more than two languages



Use-cases for Multilingual NMT

Massively Multi-way NMT system

Low-resource NMT using Transfer Learning

Unseen Language Pairs

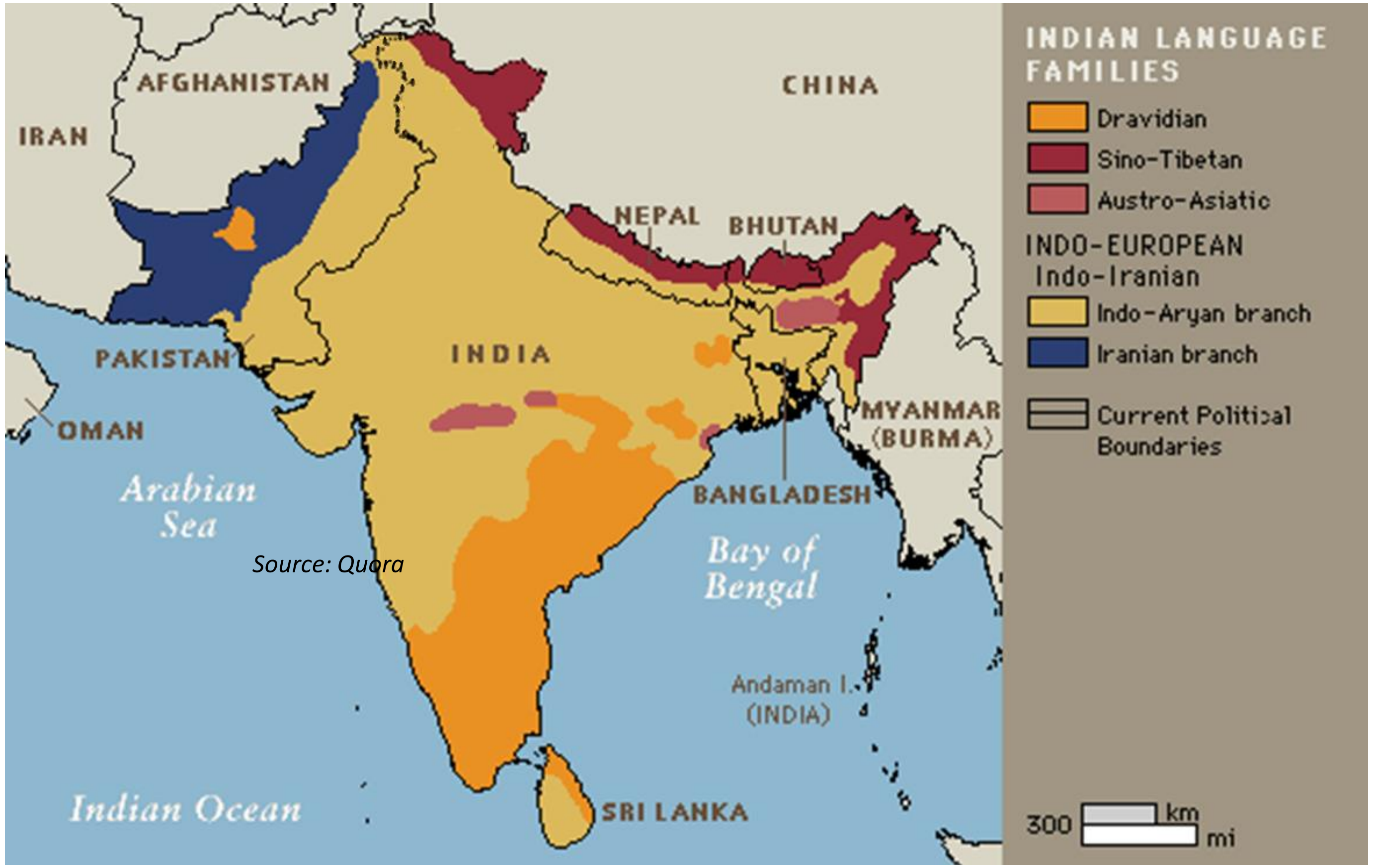
Multisource Translation

Raj Dabre, Chenhui Chu, Anoop Kunchukuttan. *A Comprehensive Survey of Multilingual Neural Machine Translation.* ACM Computing Surveys. 2020.

Diversity of Indian Languages

Highly multilingual country

Greenberg Diversity Index 0.9



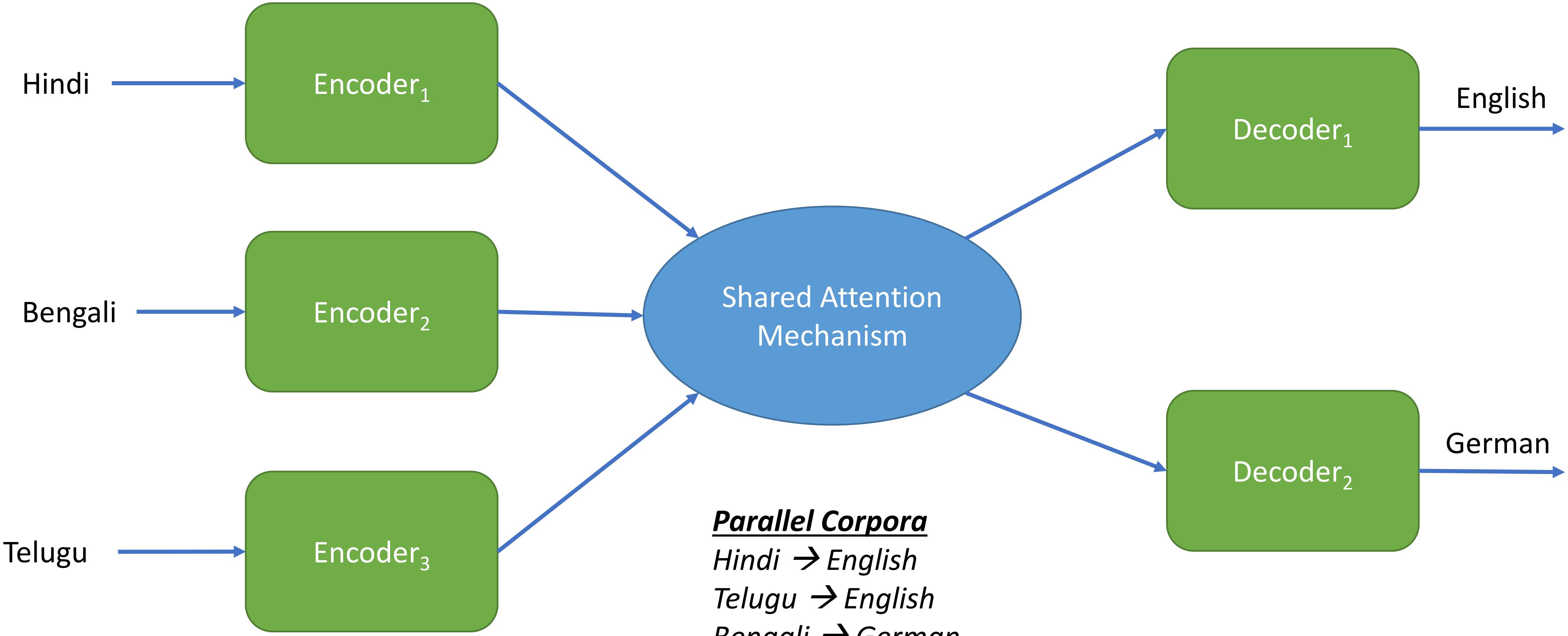
Source: Quora

- *4 major language families*
- *1600 dialects*
- *22 scheduled languages*
- *125 million English speakers*
- *8 languages in the world’s top 20 languages*
- *11 languages with more than 25 million speakers*
- *30 languages with more than 1 million speakers*

Sources: Wikipedia, Census of India 2011

General Multilingual Neural Translation

(Firat et al., 2016)



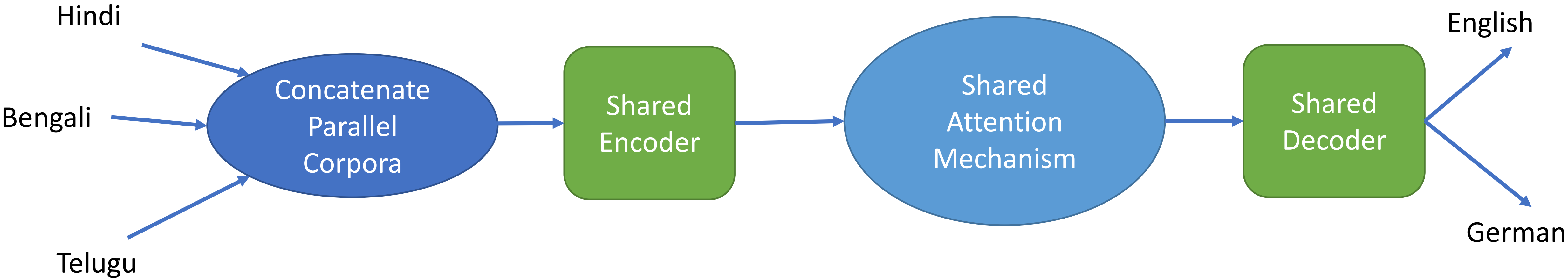
Parallel Corpora

Hindi → English
Telugu → English
Bengali → German

Firat, Orhan, Kyunghyun Cho and Yoshua Bengio. "Multi-way, multilingual neural machine translation with a shared attention mechanism." *NAACL. 2016.*

Compact Multilingual NMT

(Johnson et al., 2017)



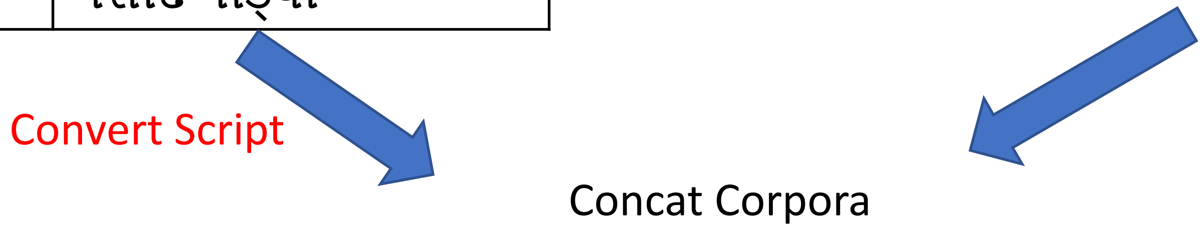
Johnson, Melvin, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat et al. "Google’s multilingual neural machine translation system: Enabling zero-shot translation." TACL (2017).

Combine Corpora from different languages

(Nguyen and Chang, 2017)

I am going home	હુ ઘરે જવ છૂ
It rained last week	છેલ્લા આઠવડિયા મા વર્સાદ પાડ્યો

It is cold in Pune	पुण्यात थंड आहे
My home is near the market	माझा घर बाजाराजवळ आहे



Convert Script

Concat Corpora

I am going home	हु घरे जव छू
It rained last week	छेल्ला आठवडिया मा वर्साद पाड्यो
It is cold in Pune	पुण्यात थंड आहे
My home is near the market	माझा घर बाजाराजवळ आहे

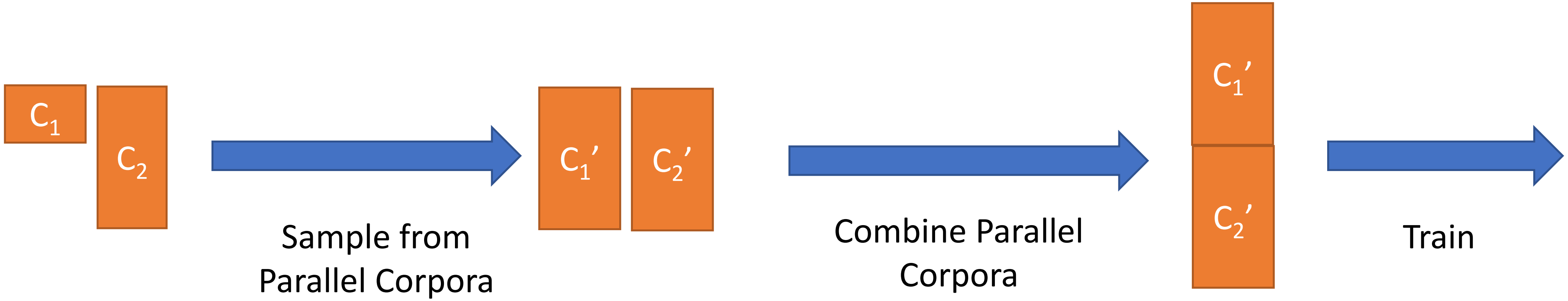
There is only one decoder, how do we generate multiple languages?

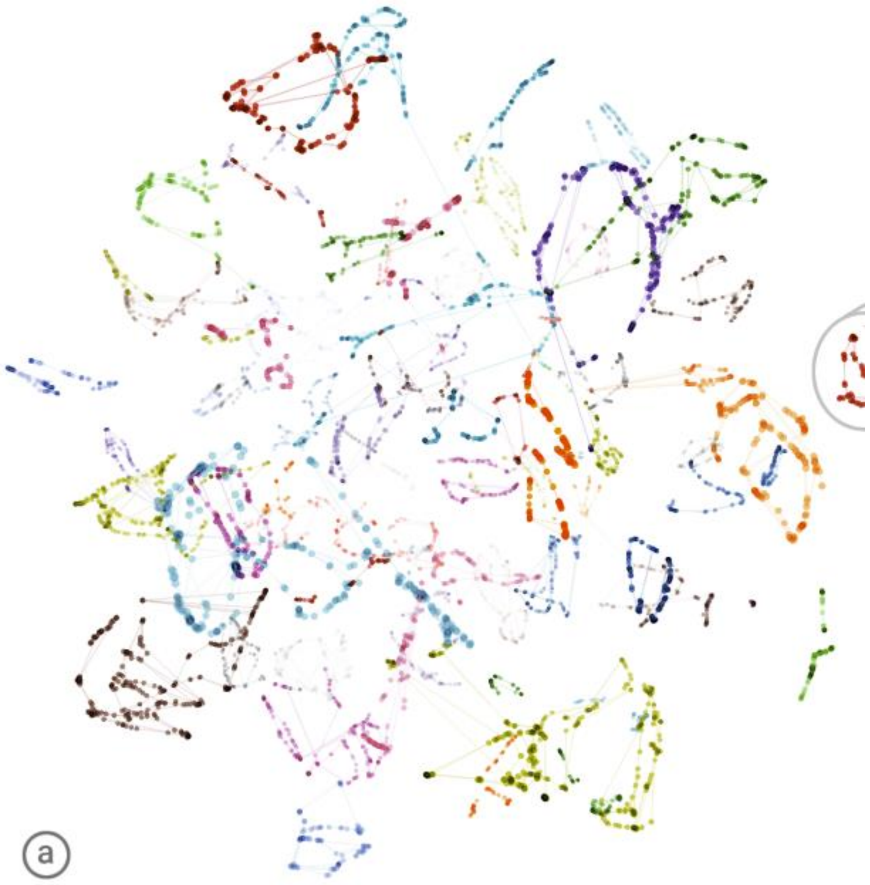
Language Tag Trick → Special token in input to indicate target language

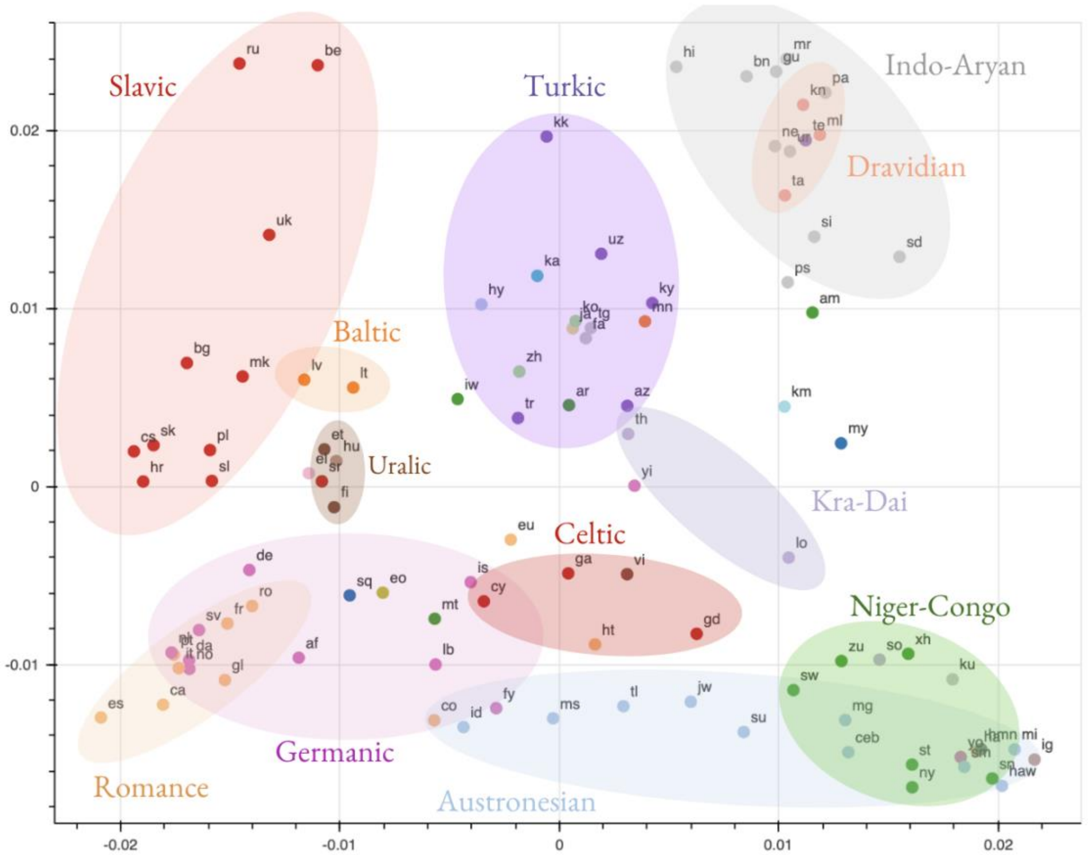
Original Input: *मकर संक्रांति भगवान सूर्य के मकर में आने का पर्व है*

Modified Input: *मकर संक्रांति भगवान सूर्य के मकर में आने का पर्व है*<eng>

Joint Training







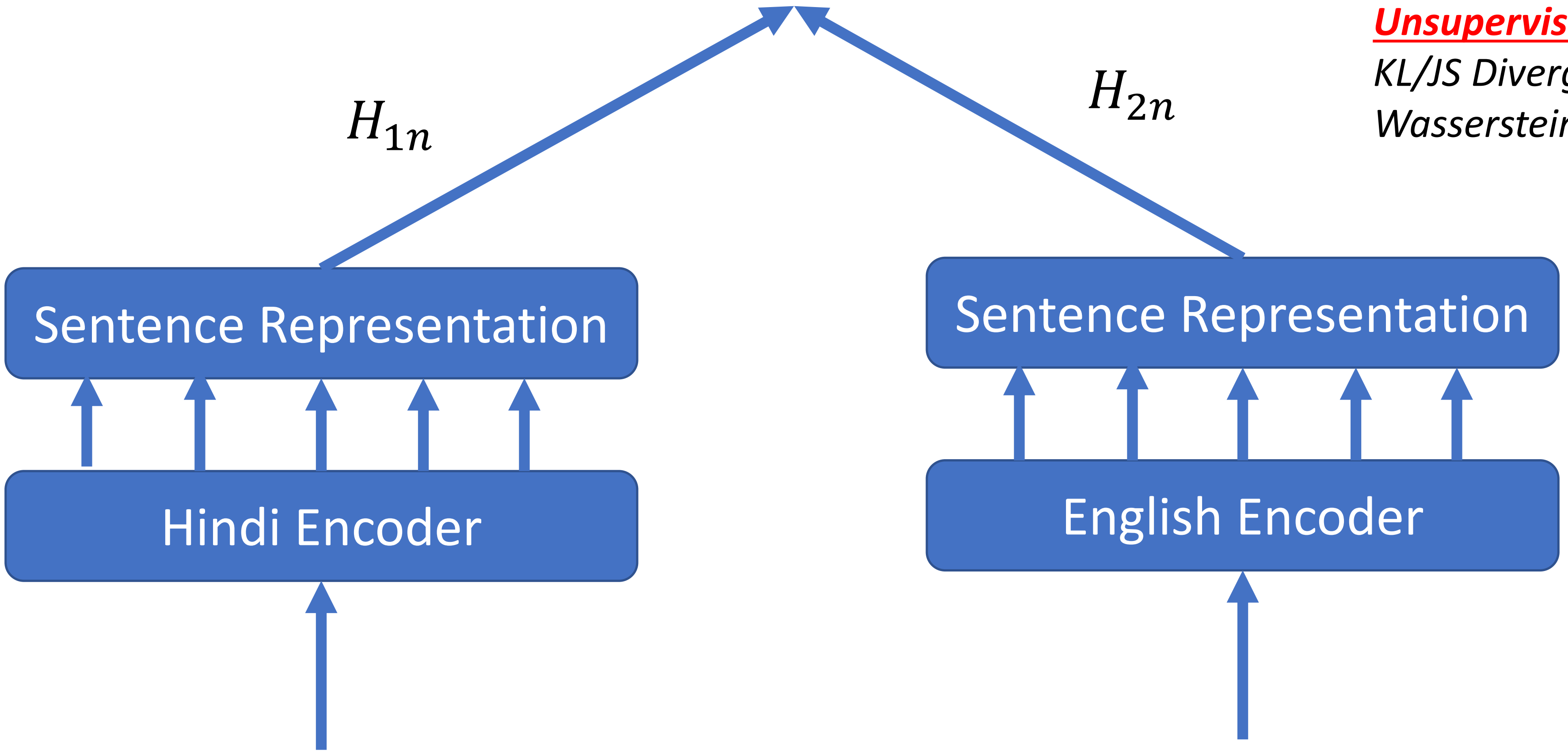
Similar sentences have similar encoder representations

But the multilingual representation is not perfect

Learning common representations across languages is one of the central problems for multilingual NMT

Aligning Encoder Representations

$$\min_{\theta} \sum_{n=1}^{N} dist(H_{1n}(\theta), H_{2n}(\theta))$$



Supervised Distance Functions

Cosine

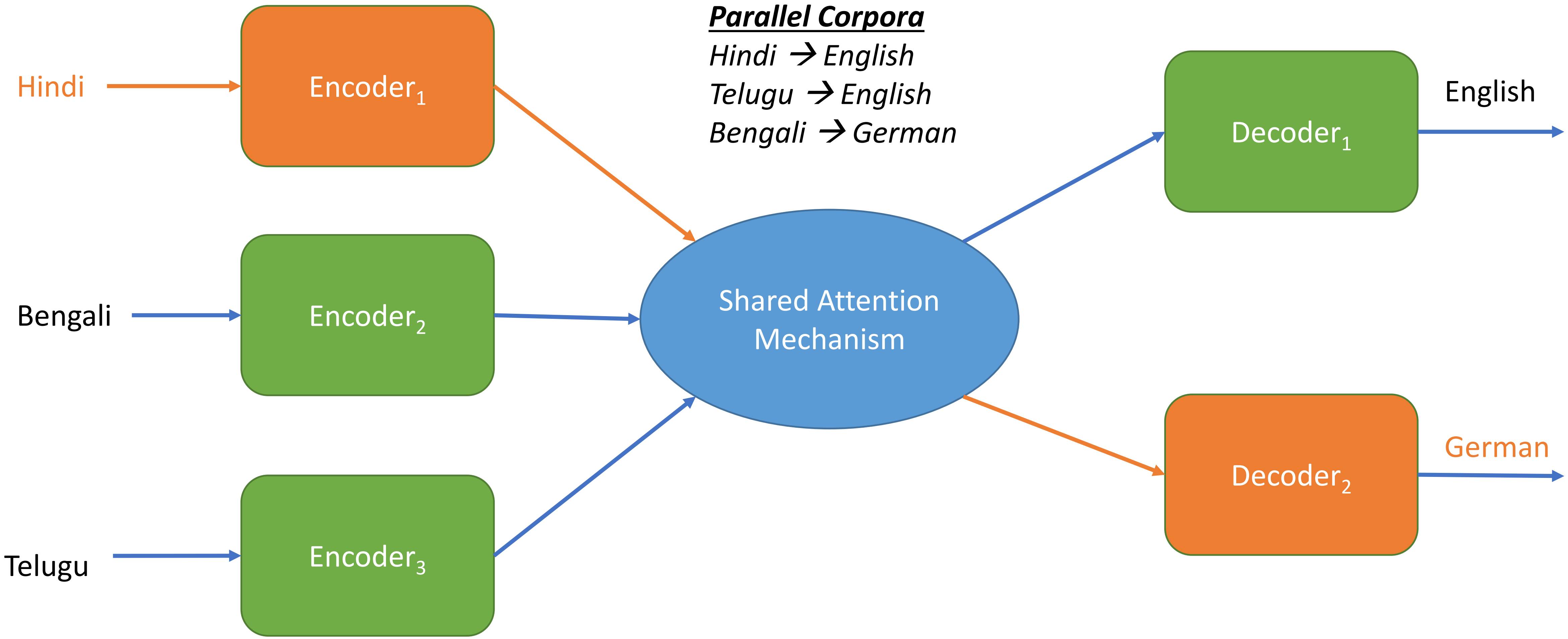
Correlation

Euclidean Distance

Unsupervised Distance Functions

KL/JS Divergence

Wasserstein



Parallel Corpora

Hindi → English

Telugu → English

Bengali → German

Hindi

Bengali

Telugu

Encoder$_1$

Encoder$_2$

Encoder$_3$

Shared Attention Mechanism

Decoder$_1$

Decoder$_2$

English

German

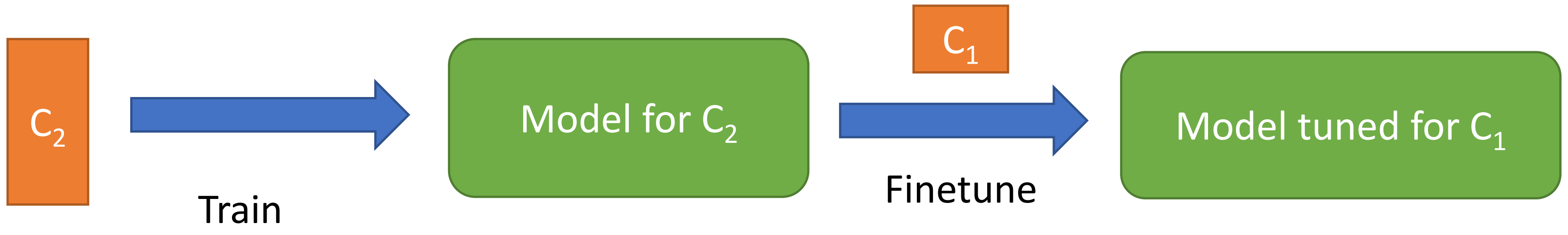
Multilingual NMT makes possible translation between unseen pairs

Zeroshot NMT (Johnson et al., 2017)

Transfer Learning

We want Gujarati → English translation → but little parallel corpus is available

We have lot of Marathi → English parallel corpus



Transfer learning works best for related languages

Thank You!

Write to me at anoop.kunchukuttan@gmail.com in case you have any questions

Thank You!

Write to me at anoop.kunchukuttan@gmail.com in case you have any questions