Introduction to Machine Translation

Anoop Kunchukuttan

Microsoft Translator, Hyderabad

NLP Course, IIT Hyderabad, 16 May 2020

Outline

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

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?

<u>Word order: SOV (Hindi), SVO (English)</u>

E: $\overbrace{\text{Germany}}^{S}\ \overbrace{\text{won}}^{V}\ \text{the last}\ \overbrace{\text{World Cup}}^{O}$

H: $\underbrace{\text{जर्मनी}}_{S}\ \text{ने}\ \text{पिछला}\ \underbrace{\text{विश्व कप}}_{O}\ \underbrace{\text{जीता था}}_{V}$

<u>Free (Hindi) vs rigid (English) word order</u>

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

Why is Machine Translation difficult?

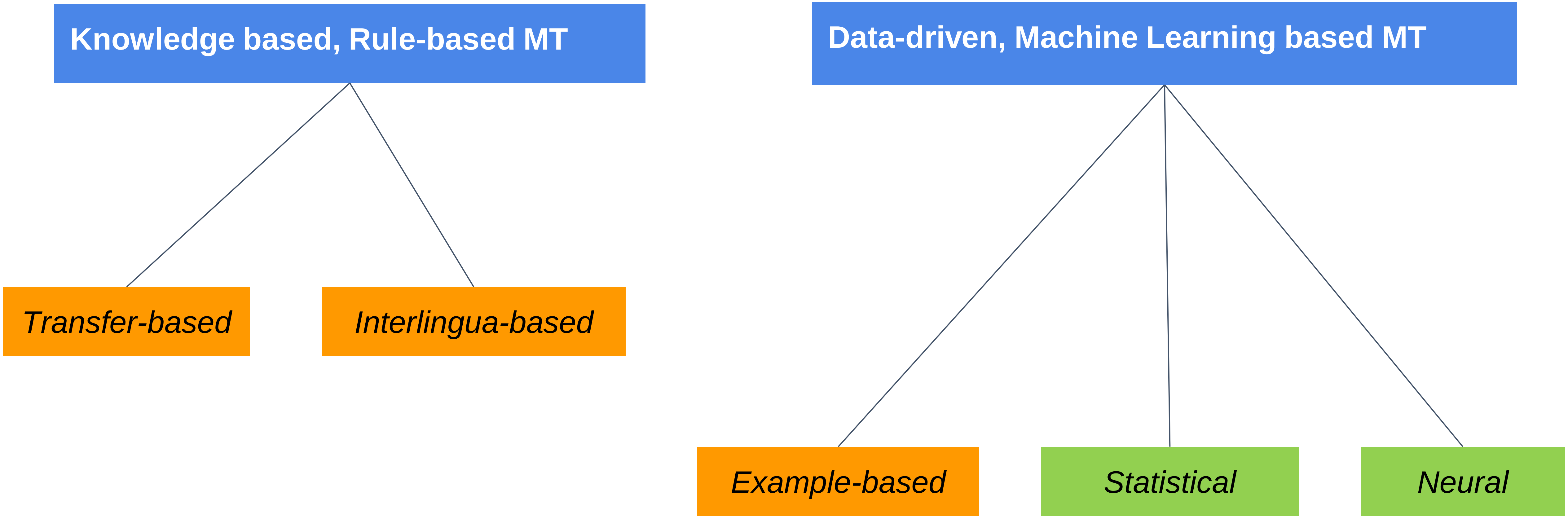
- **Ambiguity**
  - Same word, multiple meanings: मंत्री (minister or chess piece)
  - Same meaning, multiple words: जल, पानी, नीर (water)
- **Word Order**
  - Underlying deeper syntactic structure
  - Phrase structure grammar?
  - Computationally intensive
- **Morphological Richness**
  - Identifying basic units/internal structure of words

घरामागचा: घर ◌ा माग चा: that which is behind 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



Outline

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

Statistical Machine Translation

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

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

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

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

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

Let's assume 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

Parallel Corpus	
A boy is **sitting** in the kitchen	एक लडका रसोई मे **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

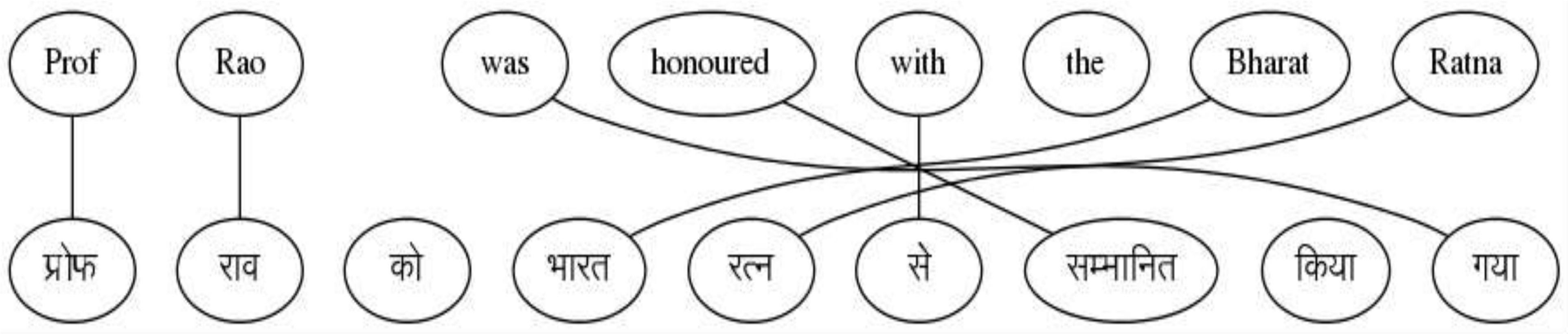
Key Idea 1

Co-occurrence of translated words

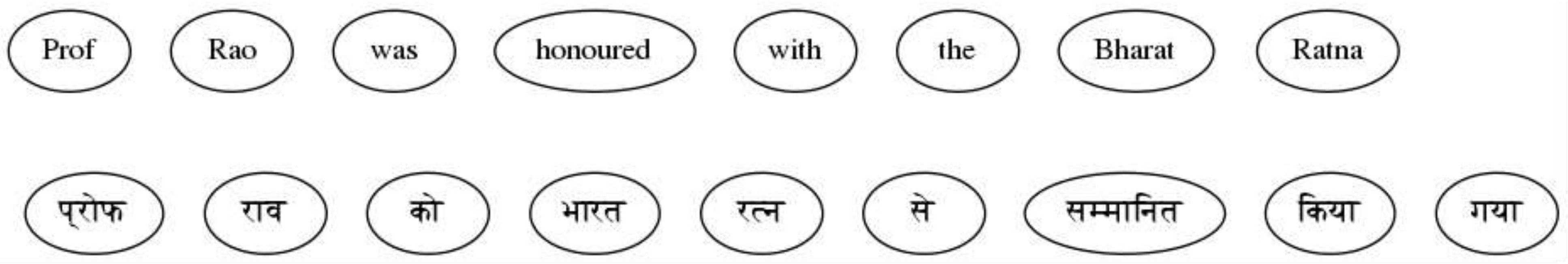
Words which occur together in the parallel sentence are likely to be translations (higher P(f|e))

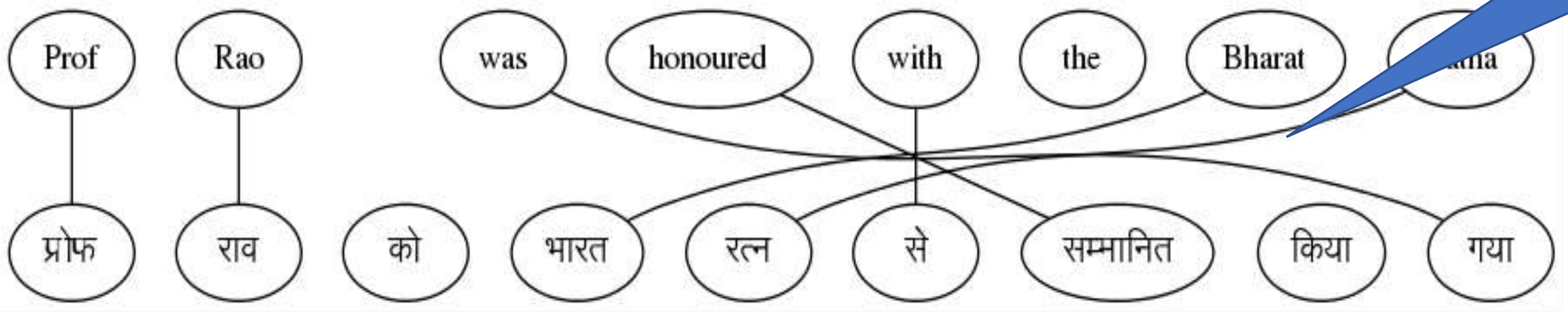
Key Idea 2

Constraints:
A source word can be aligned to a small number target language words in a parallel sentence.



Given a parallel sentence pair, find word level correspondences

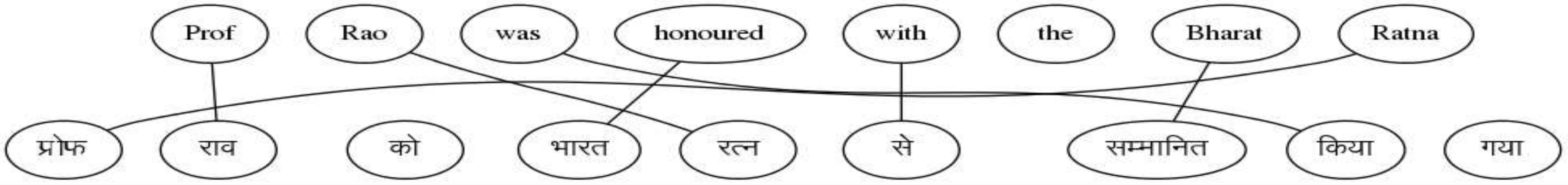


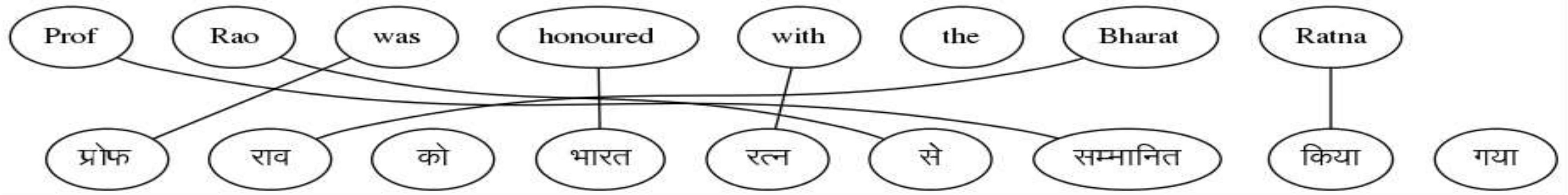


This set of links for a sentence pair is called an 'ALIGNMENT'

But there are multiple possible alignments

Sentence 1

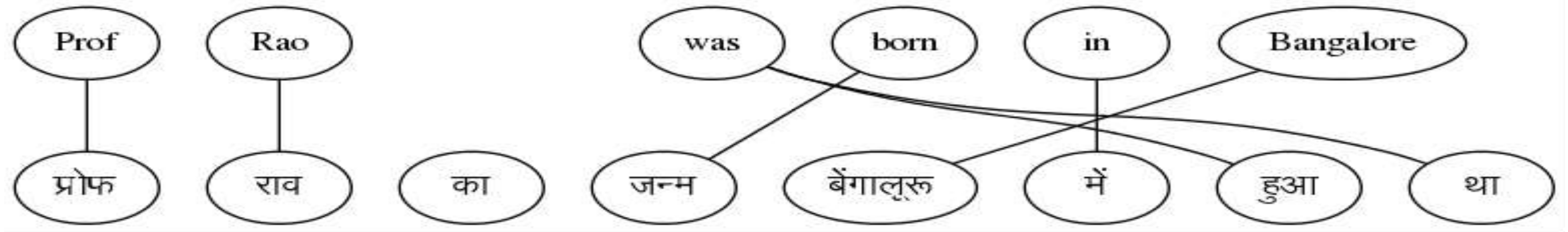


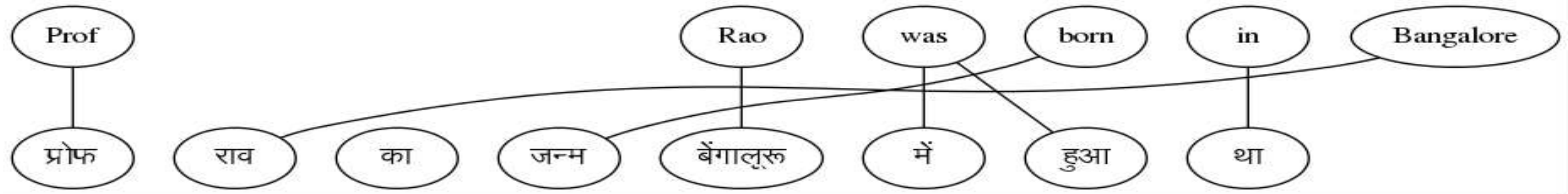


With one sentence pair, we cannot find the correct alignment

Can we find alignments if we have multiple sentence pairs?

Sentence 2

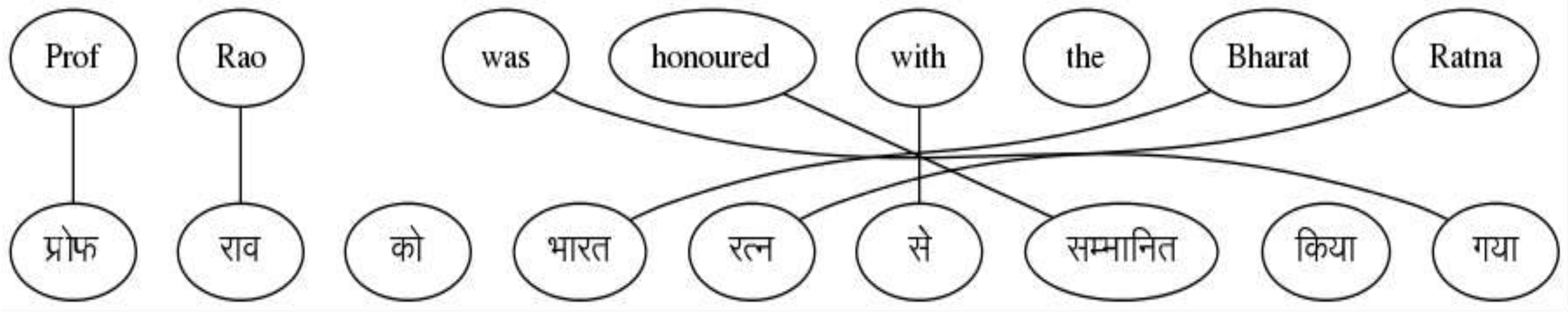




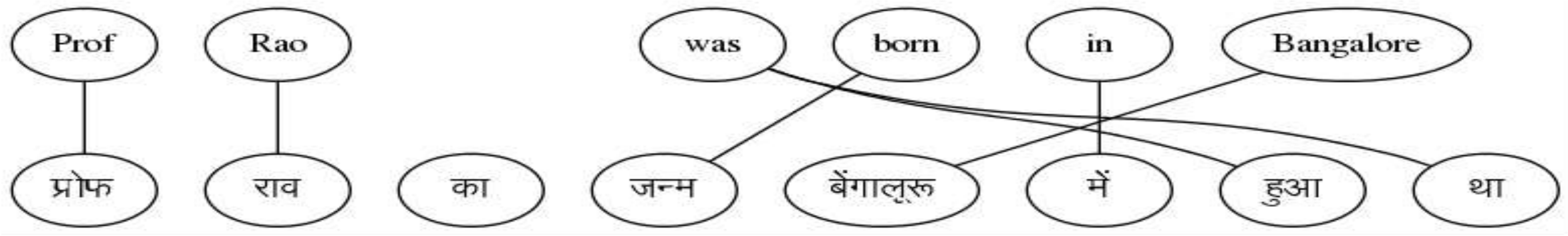
Yes, let's see how to do that …

If we knew the alignments, we could compute P(f|e)

Sentence 1



Sentence 2



$$P(f|e) = \frac{\#(f,e)}{\#(*,e)}$$

$$P(Prof|\text{प्रोफ}) = \frac{2}{2}$$

$\#(a,b)$: number of times word a is aligned to word b

But, we can find the best alignment only if we know the word translation probabilities

The best alignment is the one that maximizes the sentence translation probability

$$P(\boldsymbol{f}, \boldsymbol{a}|\boldsymbol{e}) = P(a) \prod_{i=1}^{i=m} P\left(f_i \middle| e_{a_i}\right) \quad \Longrightarrow \quad \boldsymbol{a}^* = \underset{\boldsymbol{a}}{\operatorname{argmax}} \prod_{i=1}^{i=m} P(f_i|e_{a_i})$$

This is a chicken and egg problem! How do we solve this?

We can solve this problem using a two-step, iterative process

Start with random values for word translation probabilities

Step 1: Estimate alignment probabilities using word translation probabilities

Step 2: Re-estimate word translation probabilities

- We don't know the best alignment

- So, we consider all alignments while estimating word translation probabilities

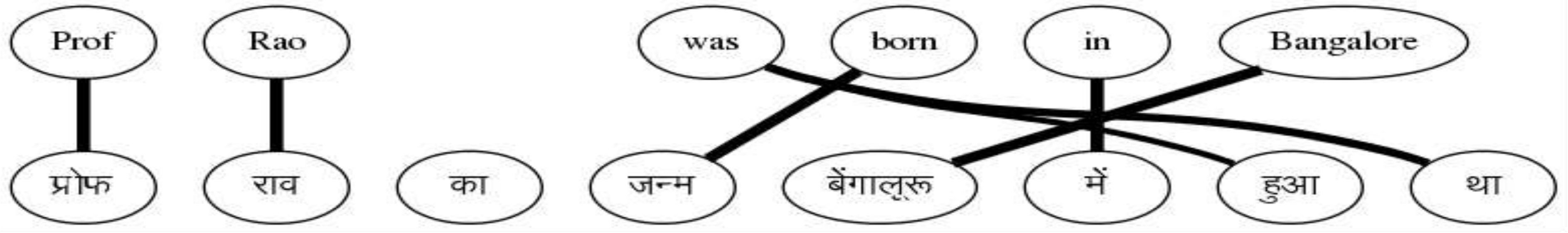
- Instead of taking only the best alignment, we consider all alignments and weigh the word alignments with the alignment probabilities

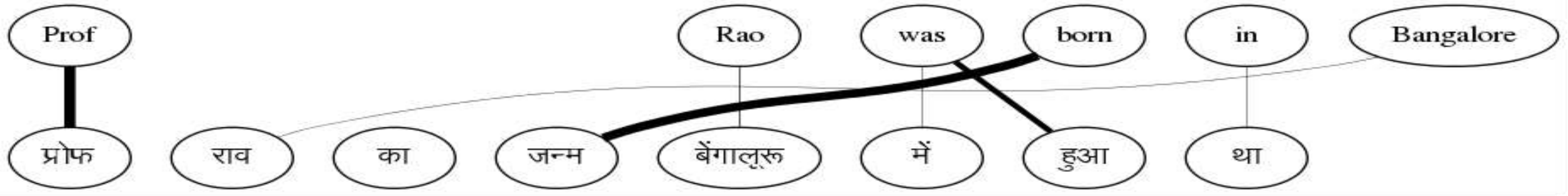
$$P(f|e) = \frac{expected\ \#(f,e)}{expected\ \#(*,e)}$$

Repeat Steps (1) and (2) till the parameters converge

At the end of the process …

Sentence 2





Expectation-Maximization Algorithm: *guaranteed to converge, maybe to local minima*
Hence we need to good initialization and training regimens.

IBM Models

- IBM came up with a series of increasingly complex models
- Called Models 1 to 5
- Differed in assumptions about alignment probability distributions
- Simpler models are used to initialize the more complex models
- This pipelined training helped ensure better solutions

Phrase Based SMT

Parallel Corpus	
A boy is **sitting** in the kitchen	एक लडका रसोई मे **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

Why stop at learning word correspondences?

KEY IDEA

Use “Phrase” as the basic translation unit

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

(Sequence of Words)

Examples of phrase pairs

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

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)

Mathematical Model

Let's revisit the decision rule for SMT model

$$\begin{aligned}\mathbf{e}_{\text{best}} &= \text{argmax}_{\mathbf{e}}\ p(\mathbf{e}|\mathbf{f}) \\ &= \text{argmax}_{\mathbf{e}}\ p(\mathbf{f}|\mathbf{e})\ p_{\text{LM}}(\mathbf{e})\end{aligned}$$

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

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

Distortion probability

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

Phrase Translation Probability

start_i :start position in **f** of i^{th} phrase of **e**
end_i :end position in **f** of i^{th} phrase of **e**

Learning The Phrase Translation Model

Involves Structure + Parameter Learning:

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

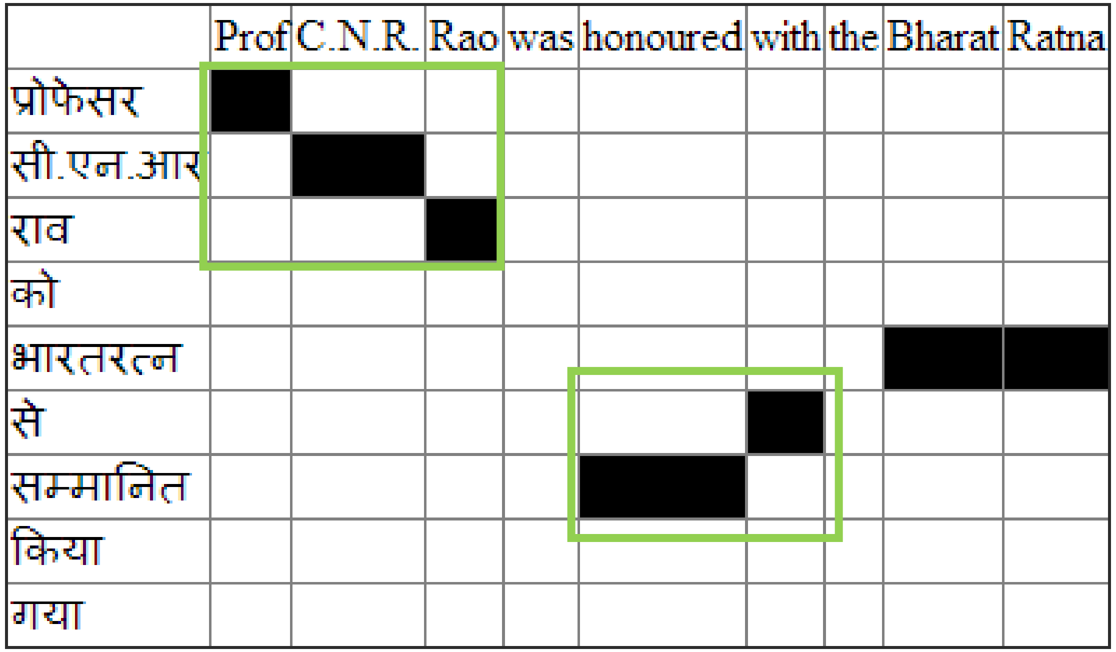
The Prime Minister of India	भारत के प्रधान मंत्री
is running fast	तेज भाग रहा है
the boy with the telescope	दूरबीन से लड़के को
Rahul lost the match	राहुल मुकाबला हार गया

- Learn the **Phrase Translation Probabilities**

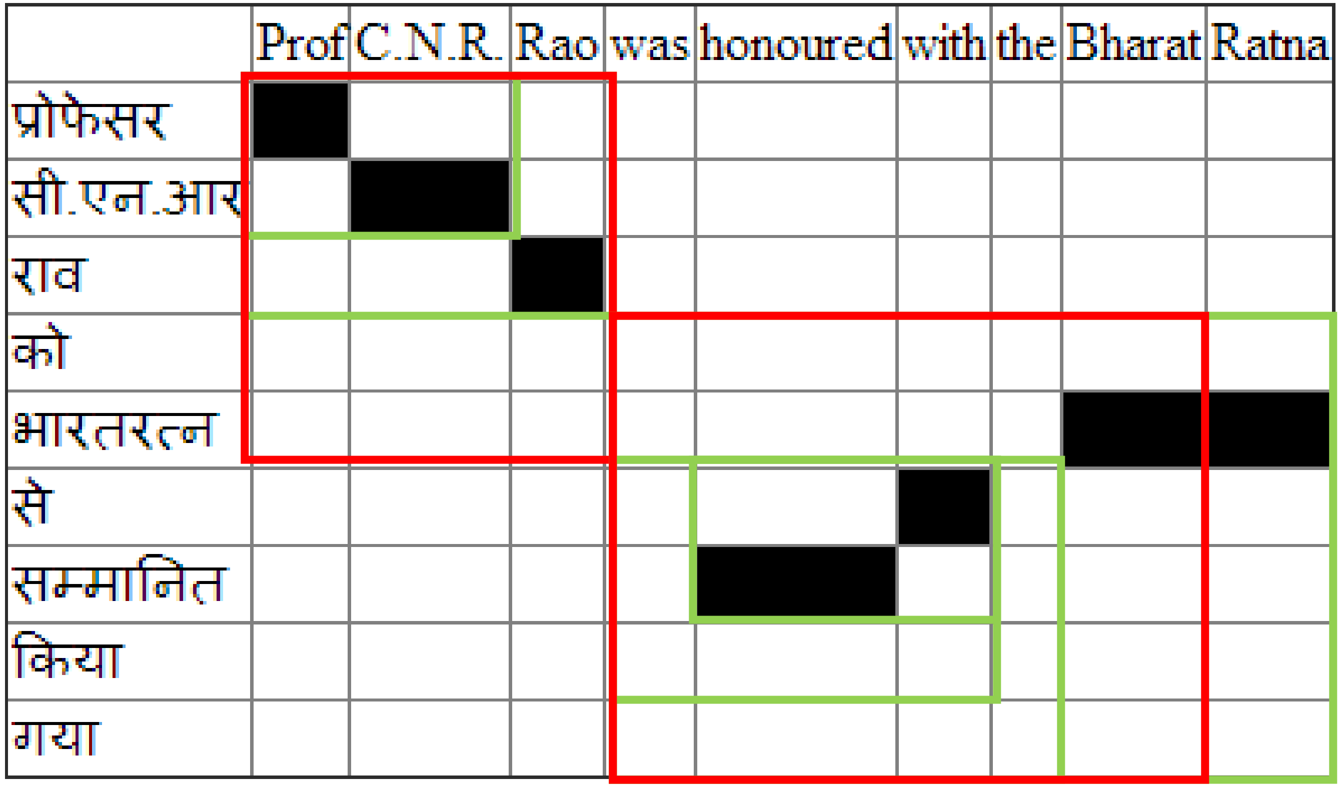
Prime Minister of India	भारत के प्रधान मंत्री India of Prime Minister	0.75
Prime Minister of India	भारत के भूतपूर्व प्रधान मंत्री India of former Prime Minister	0.02
Prime Minister of India	प्रधान मंत्री Prime Minister	0.23

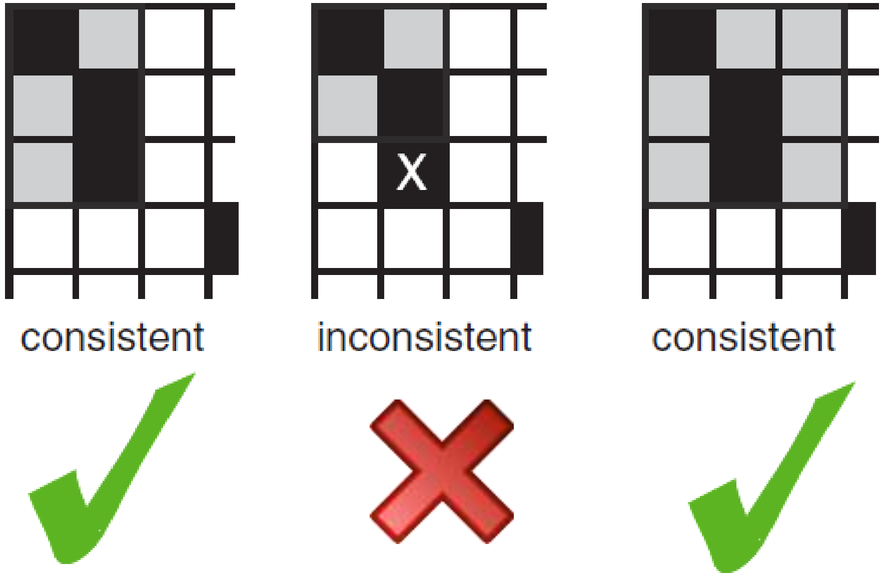
Learning Phrase Tables from Word Alignments

- 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"



Which phrase pairs to include in the phrase table?





Source: SMT, Phillip Koehn

Professor CNR	प्रोफेसर सी.एन.आर
Professor CNR Rao	प्रोफेसर सी.एन.आर राव
Professor CNR Rao was	प्रोफेसर सी.एन.आर राव
Professor CNR Rao was	प्रोफेसर सी.एन.आर राव को
honoured with the Bharat Ratna	भारतरत्न से सम्मानित
honoured with the Bharat Ratna	भारतरत्न से सम्मानित किया
honoured with the Bharat Ratna	भारतरत्न से सम्मानित किया गया
honoured with the Bharat Ratna	को भारतरत्न से सम्मानित किया गया

Discriminative Training of PB-SMT

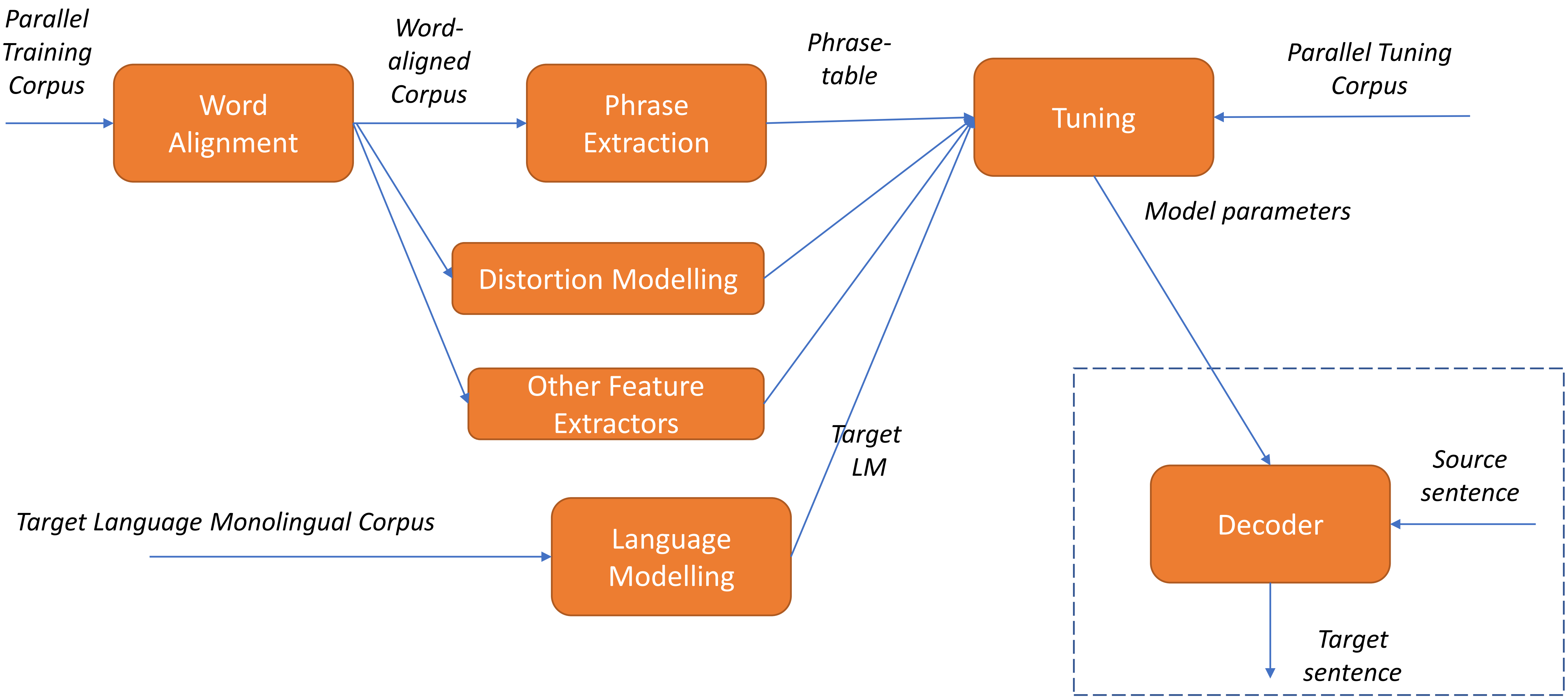
- Directly model the posterior probability p(**e|f**)
- Use the Maximum Entropy framework

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

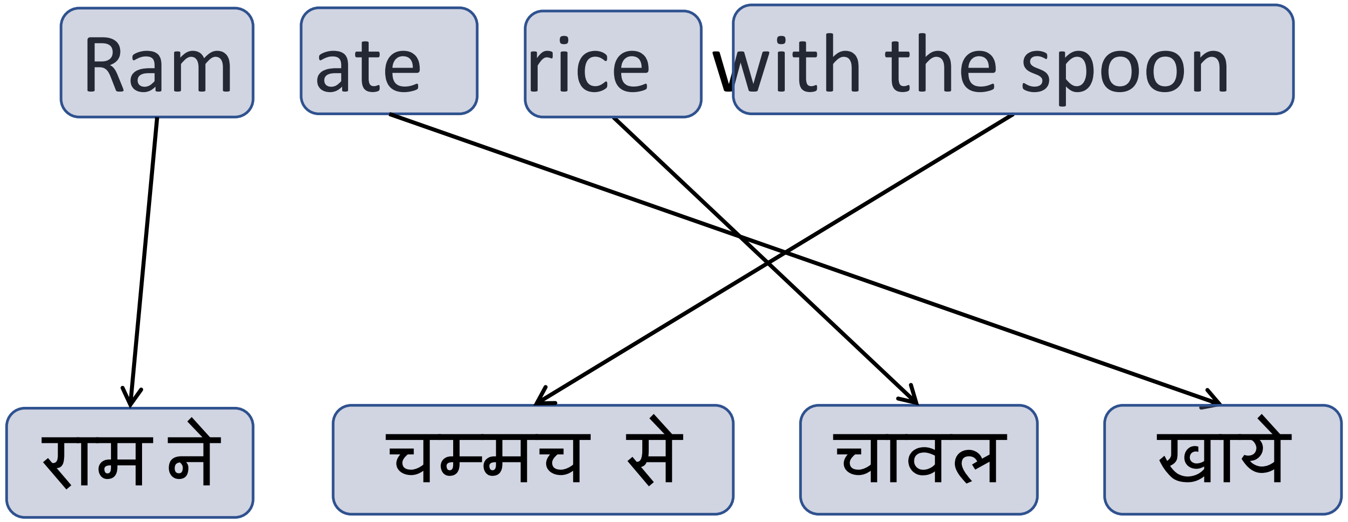
$$e^* = \arg\max_{e_i} \sum_i \lambda_i h_i(f_1^I, e_1^J)$$

  - 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



Decoding

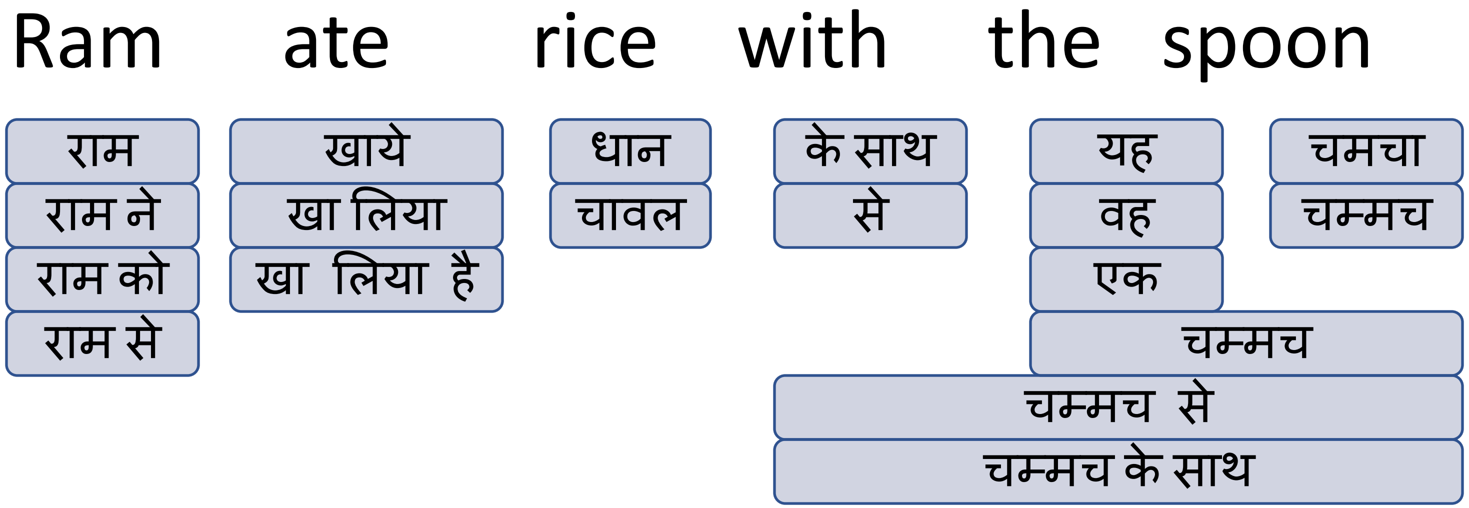


Searching for the best translations in the space of all translations

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

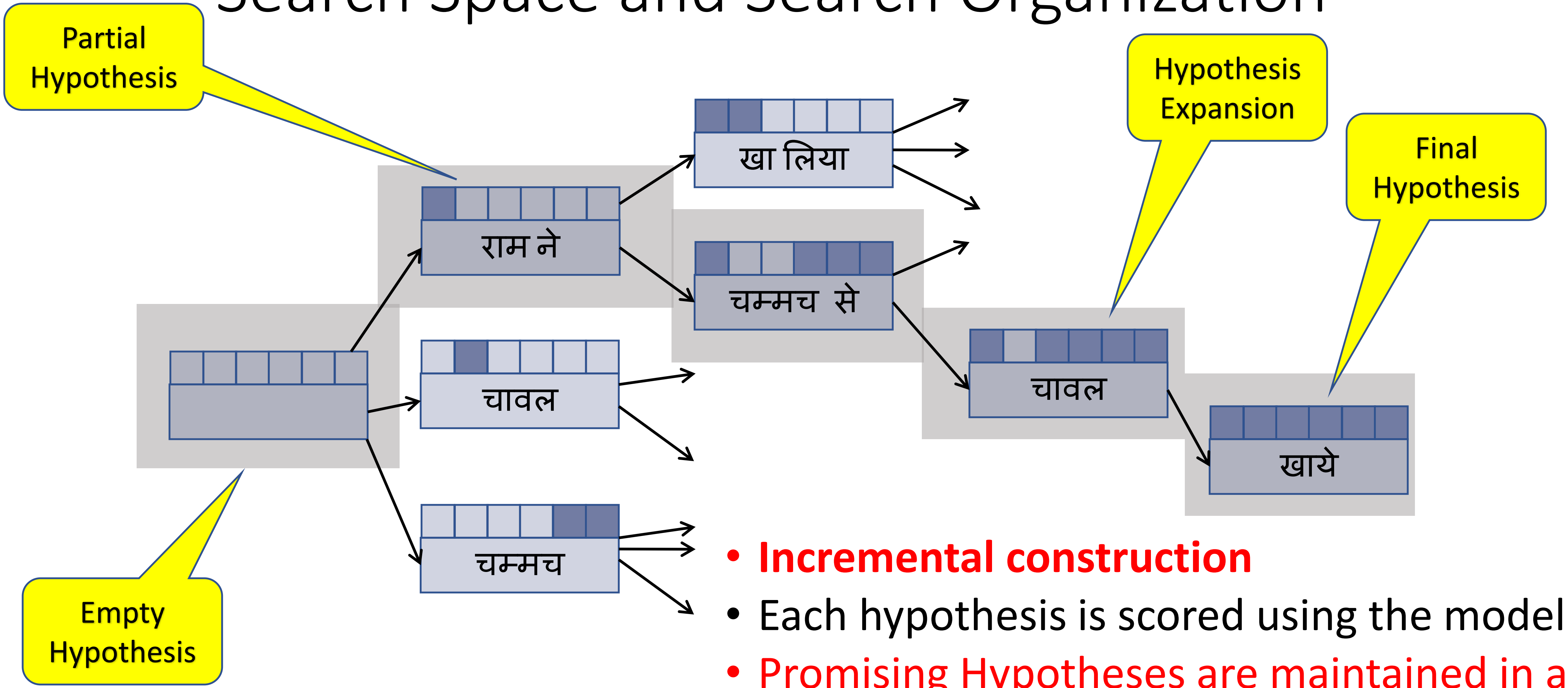
Decoding is challenging

- We picked the phrase translation that made sense to us
- The computer has less intuition
- Phrase table may give many options to translate the input sentence
- Multiple possible word orders



An <u>NP complete</u> search problem ➔ Needs a heuristic search method

Search Space and Search Organization



- **Incremental construction**
- Each hypothesis is scored using the model
- Promising Hypotheses are maintained in a bounded priority queue
- Limit to the reordering window for efficiency

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

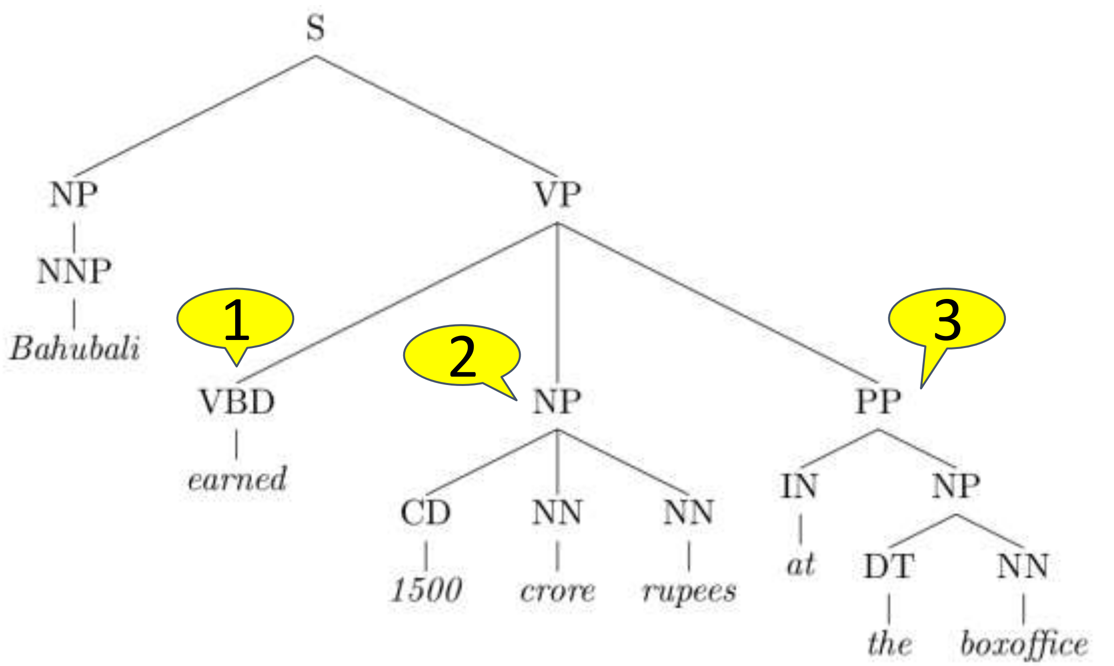
Solution: Let’s help PB-SMT with some preprocessing of the input

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

Bahubali earned more than 1500 crore rupees at the boxoffice

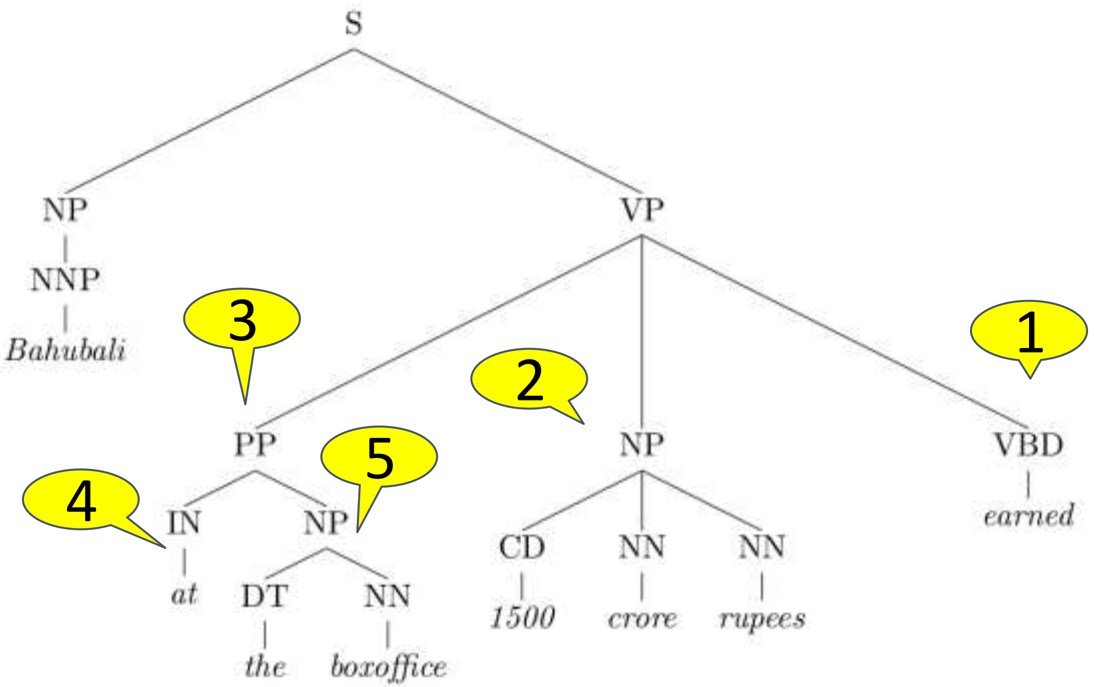
Bahubali the boxoffice at 1500 crore rupees earned

बाहुबली ने बॉक्सओफिस पर 1500 करोड रुपए कमाए



Parse the sentence to understand its syntactic structure

Apply rules to transform the tree

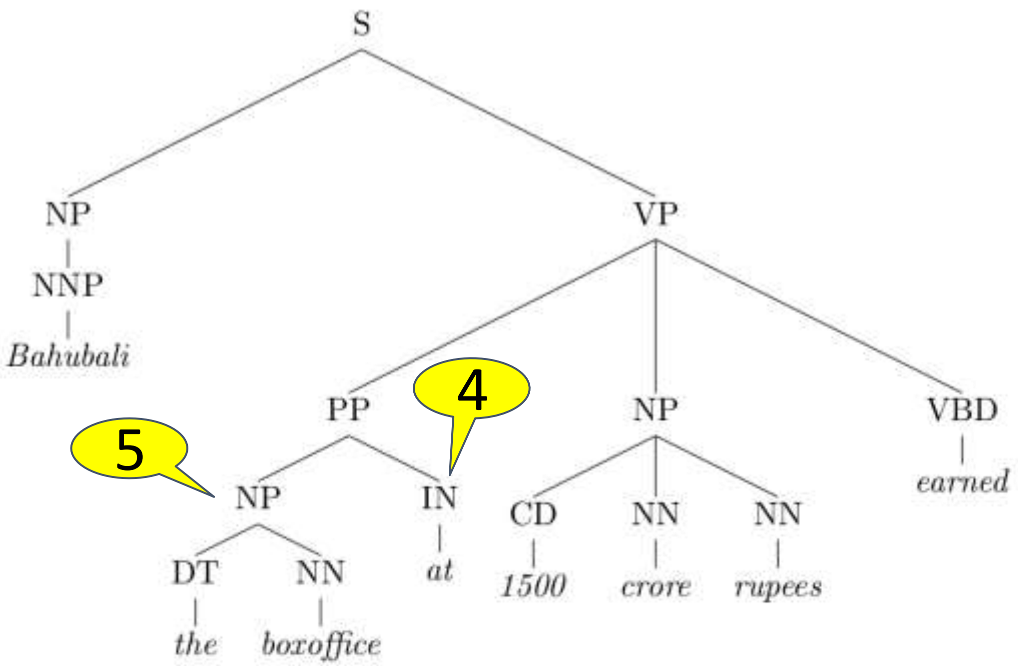


VP → VBD NP PP ⇒ VP → PP NP VBD

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 बॉक्सऑफिस 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

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*

SMT, Rule-based MT and Example based MT **manipulate symbolic representations of knowledge**

Every word has an atomic representation, which can't be further analyzed

home	0
water	1
house	2
tap	3

1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

No notion of similarity or relationship between words

- *Even if we know the translation of* `home`*, we can't translate* `house` *if it an OOV*

Difficult to represent new concepts

- *We cannot say anything about 'mansion' if it comes up at test time*
- *Creates problems for language model as well ⇒ whole are of smoothing exists to overcome this problem*

Symbolic representations are ***discrete representations***

- *Generally computationally expensive to work with discrete representations*
- *e.g. Reordering requires evaluation of an exponential number of candidates*

Neural Network techniques work with distributed representations

<u>*Every word is represented by a vector of numbers*</u>

- *No element of the vector represents a particular word*
- *The word can be understood with all vector elements*
- *Hence distributed representation*
- *But less interpretable*

home	0.5	0.6	0.7
Water	0.2	0.9	0.3
house	0.55	0.58	0.77
tap	0.24	0.6	0.4

Word vectors or embeddings

Can define similarity between words

- *Vector similarity measures like cosine similarity*
- *Since* representations of `home` *and* `house`, *we may be able to translate* `house`

New concepts can be represented using a vector with different values

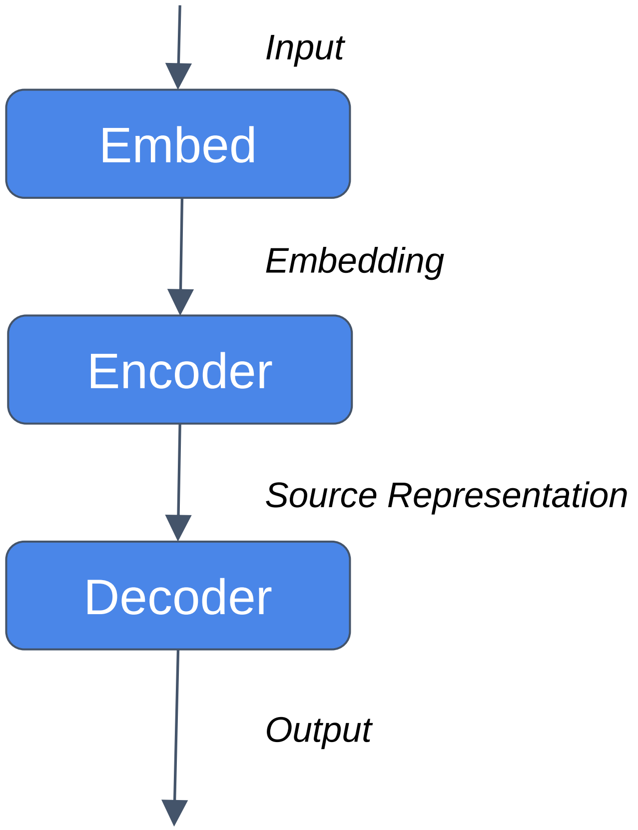
Symbolic representations are ***continuous representations***

- *Generally computationally more efficient to work with continuous values*
- *Especially optimization problems*

Topics

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

Encode - Decode Paradigm



Entire input sequence is processed before generation starts
⇒ In PBSMT, generation was piecewise

The input is a sequence of words, processed one at a time

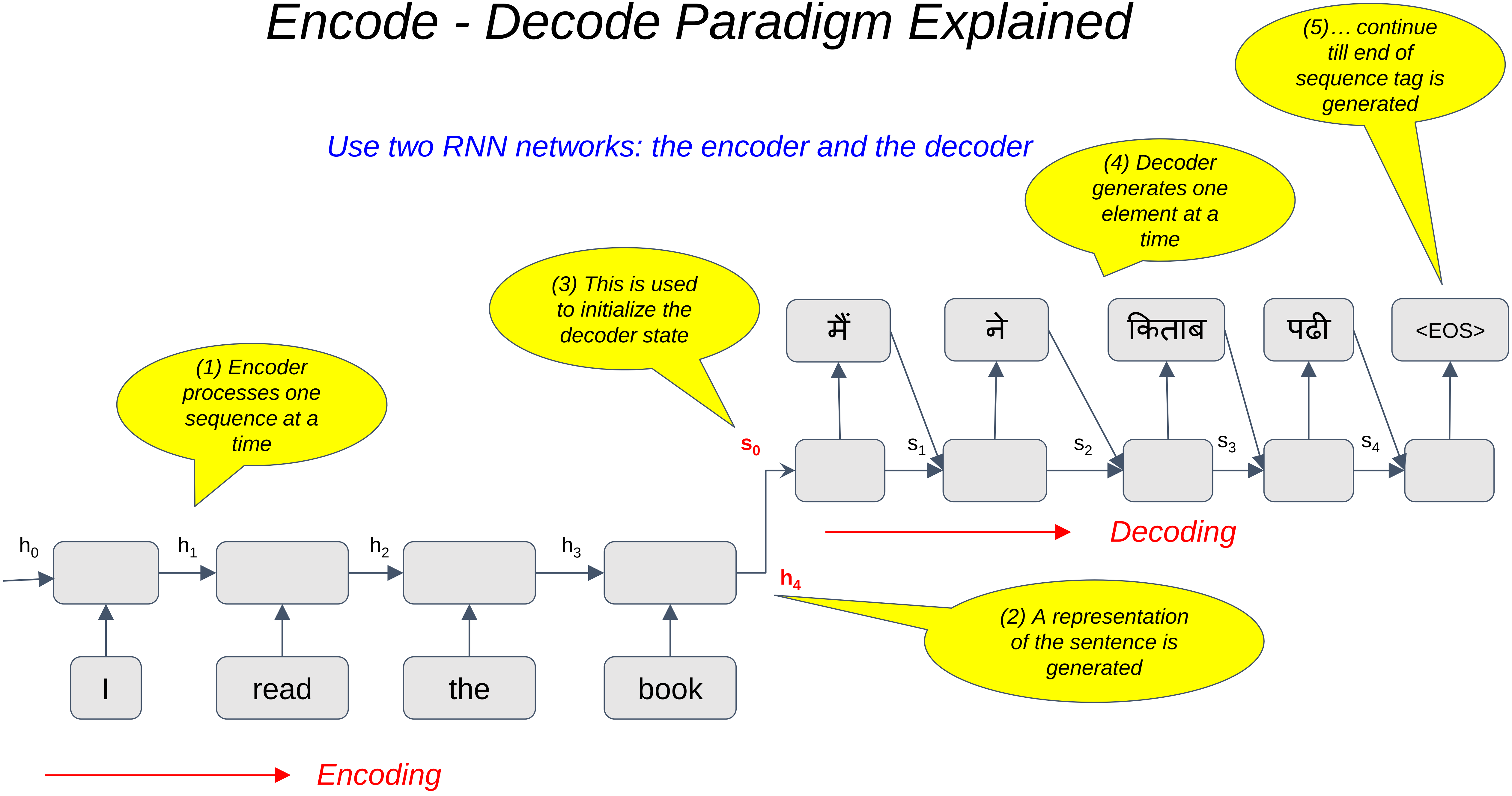
- *While processing a word, the network needs to know what it has seen so far in the sequence*
- *Meaning, know the history of the sequence processing*
- *Needs a special kind of neural network: Recurrent neural network unit which can keep state information*

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

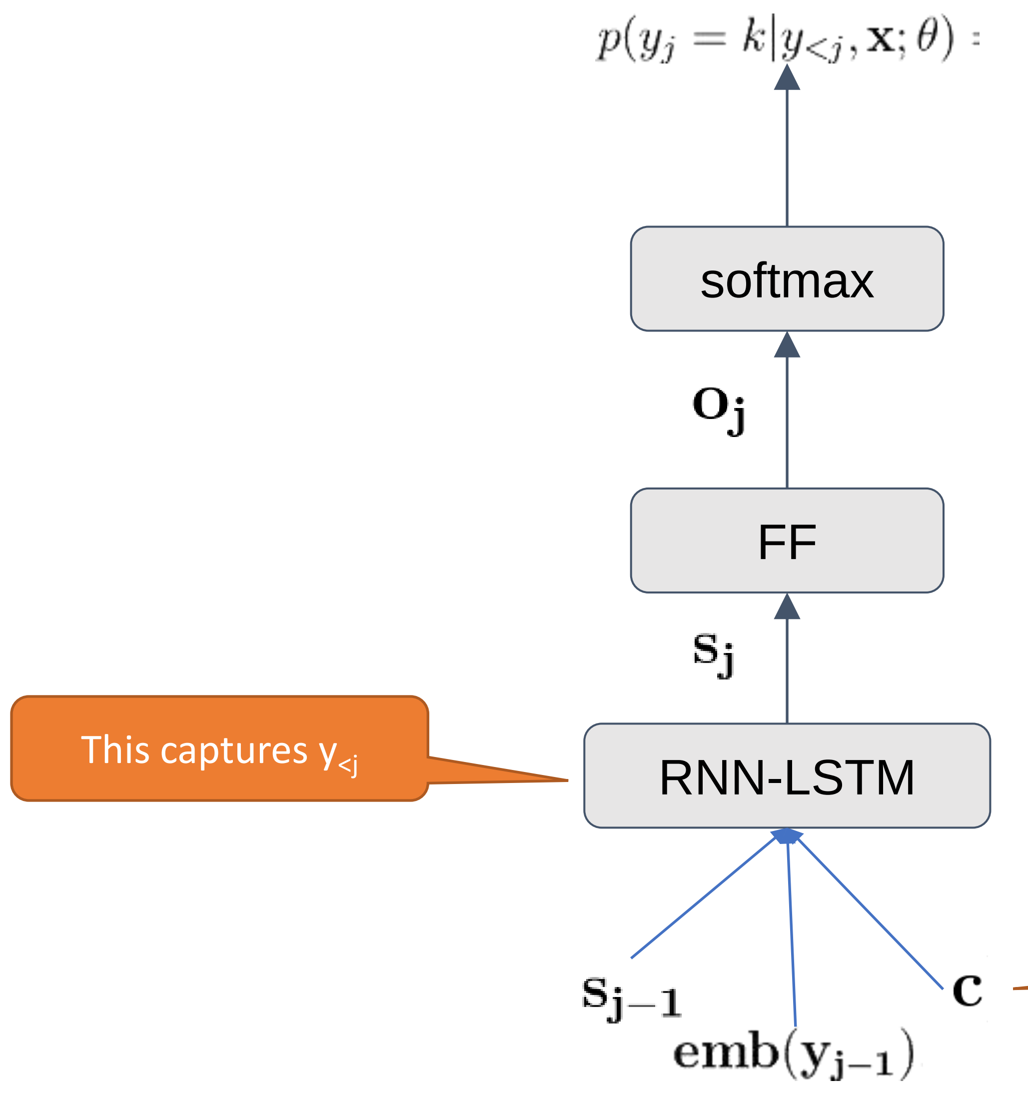
$$p(y_j = k|y_{<j},\mathbf{x};\theta) = softmax(o_{jk})$$

Encode - Decode Paradigm Explained

Use two RNN networks: the encoder and the decoder



What is the decoder doing at each time-step?



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

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

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

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

- Optimized with Stochatic Gradient Descent or variants like ADAM in mini-batches
- End to end training
- **Target Forcing**: Gold-Standard previous word is used, otherwise performance deteriorates
    - Discrepancy in train and test scenarios
    - Solutions: scheduled sampling
- Word-level objective is only an approximation to sentence-level objectives
- Likelihood objective is different from evaluation metrics

Decoding Strategies

- Exhaustive Search: *Score each and every possible translation – Forget it!*

- Sampling

- Greedy

- Beam Search

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

Greedy Search is not optimal

	t_1		t_2
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 sequence w_1w_3 *=0.15*

	t_1		t_2
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

Probability of sequence w_2w_2 *=0.18*

Topics

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

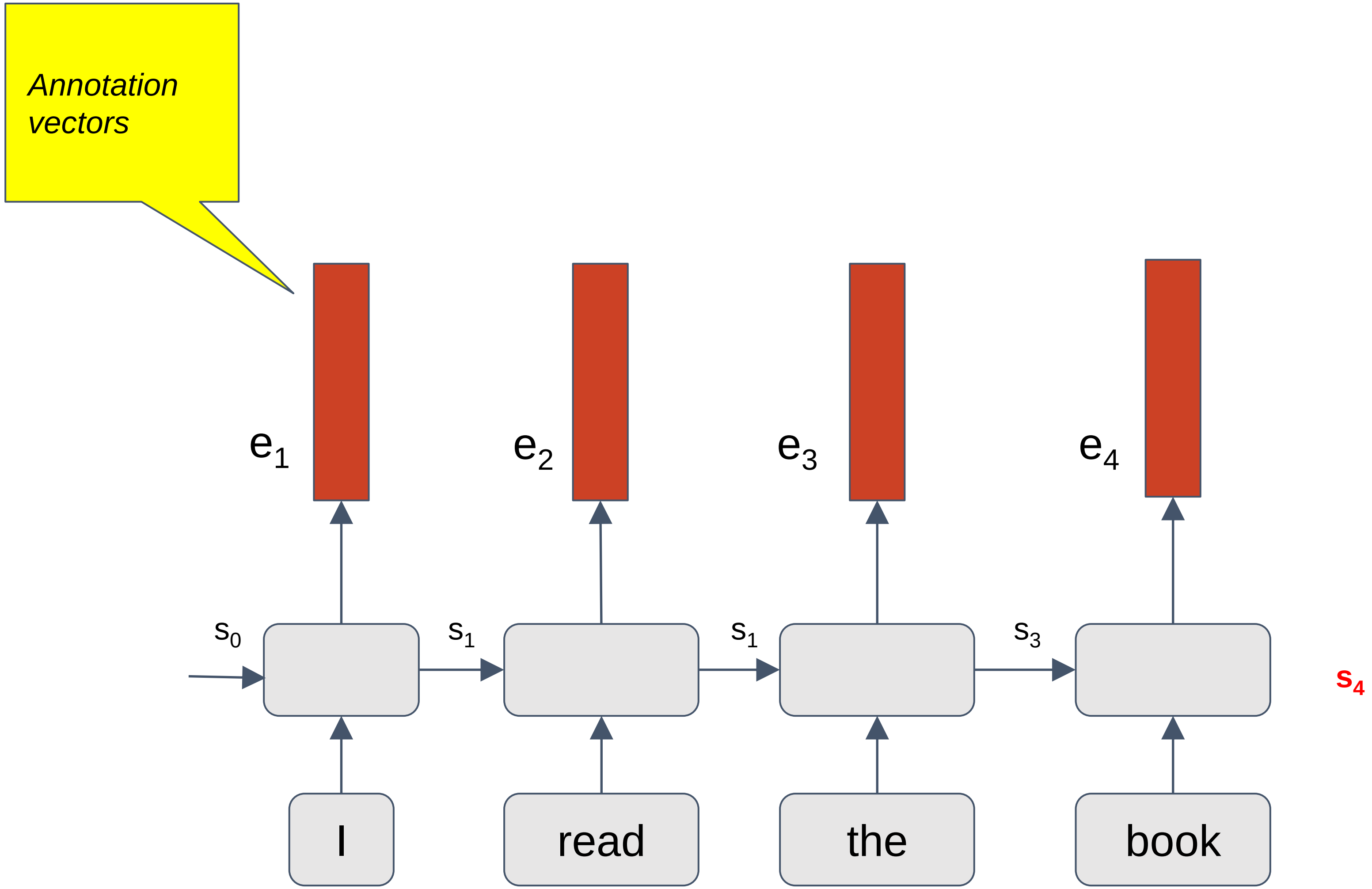
The entire sentence is represented by a single vector

Problems

- A single vector is not sufficient to represent to capture all the syntactic and semantic complexities of a sentence
  - *Solution: Use a richer representation for the sentences*
- Problem of capturing long term dependencies: The decoder RNN will not be able to make use of source sentence representation after a few 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*

Note: in the encoder-decode paradigm, we ignore the encoder outputs

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

How should the decoder use the set of annotation vectors while predicting the next character?

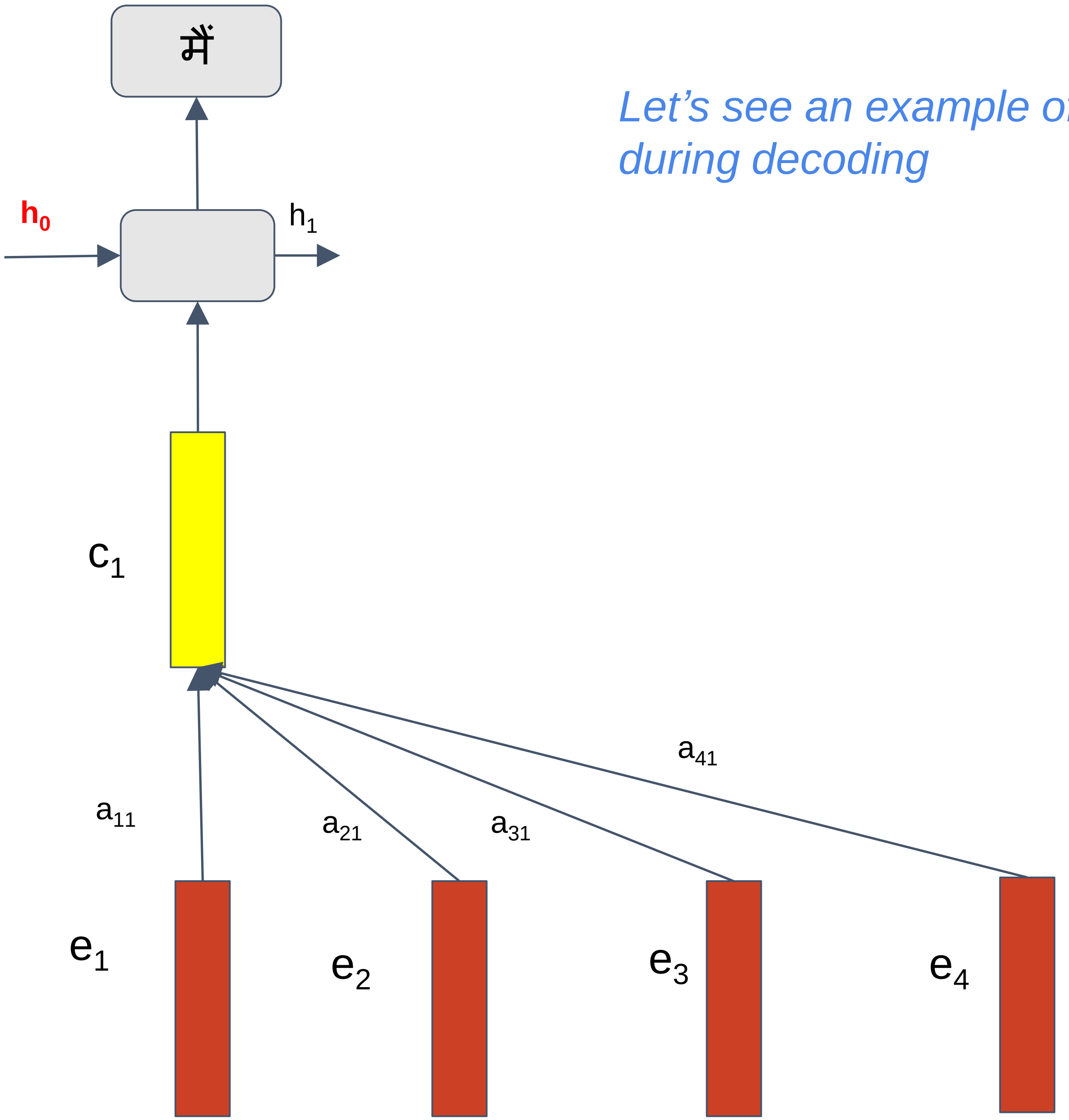
Key Insight:

(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 3^{rd} target word, the 3^{rd} annotation vector (hence 3^{rd} source word) is most important

One way to achieve this:
Take a weighted average of the annotation vectors, with more weight to annotation vectors which need more **focus or attention**

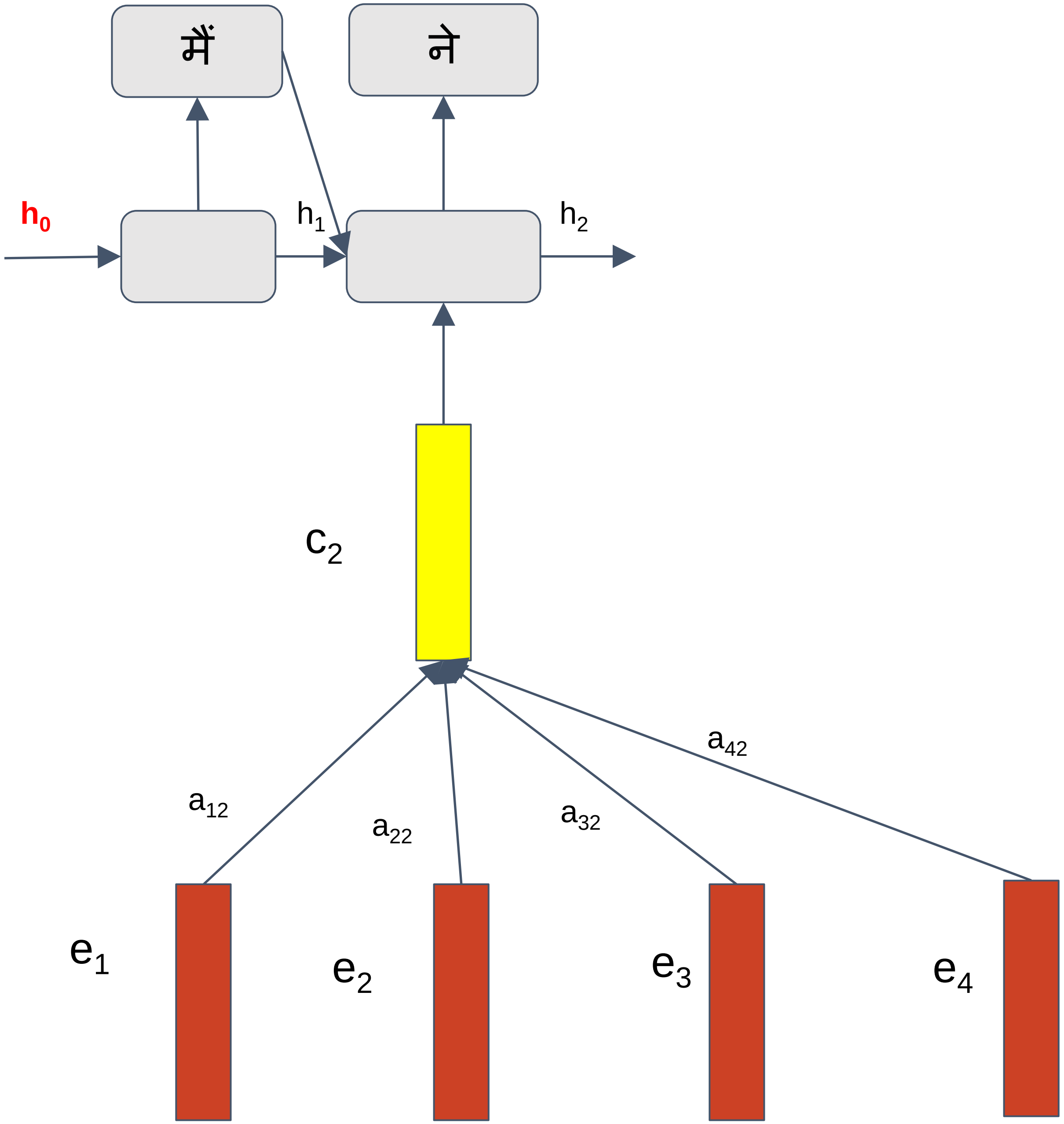
This averaged ***context vector*** is an input to the decoder

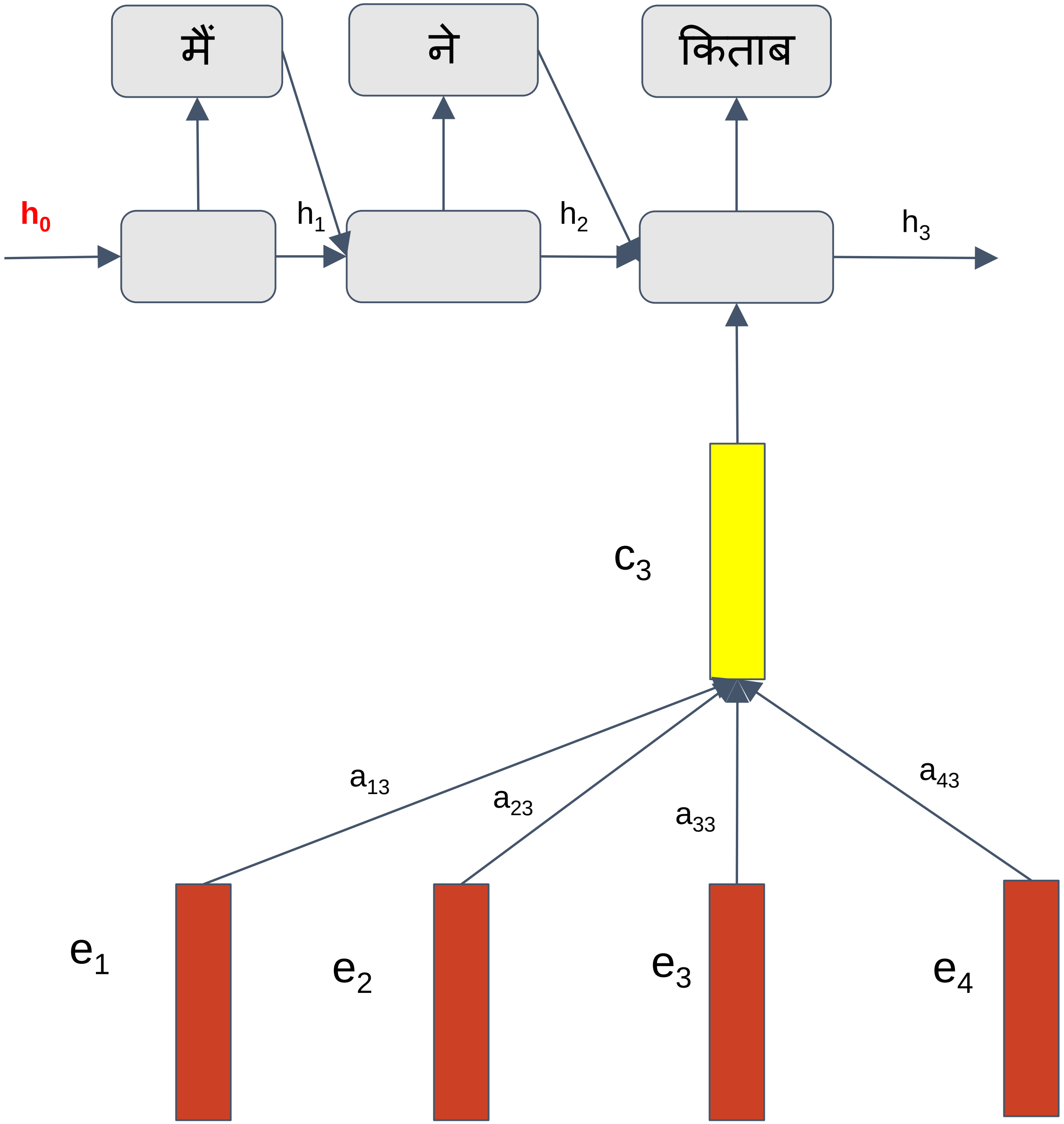


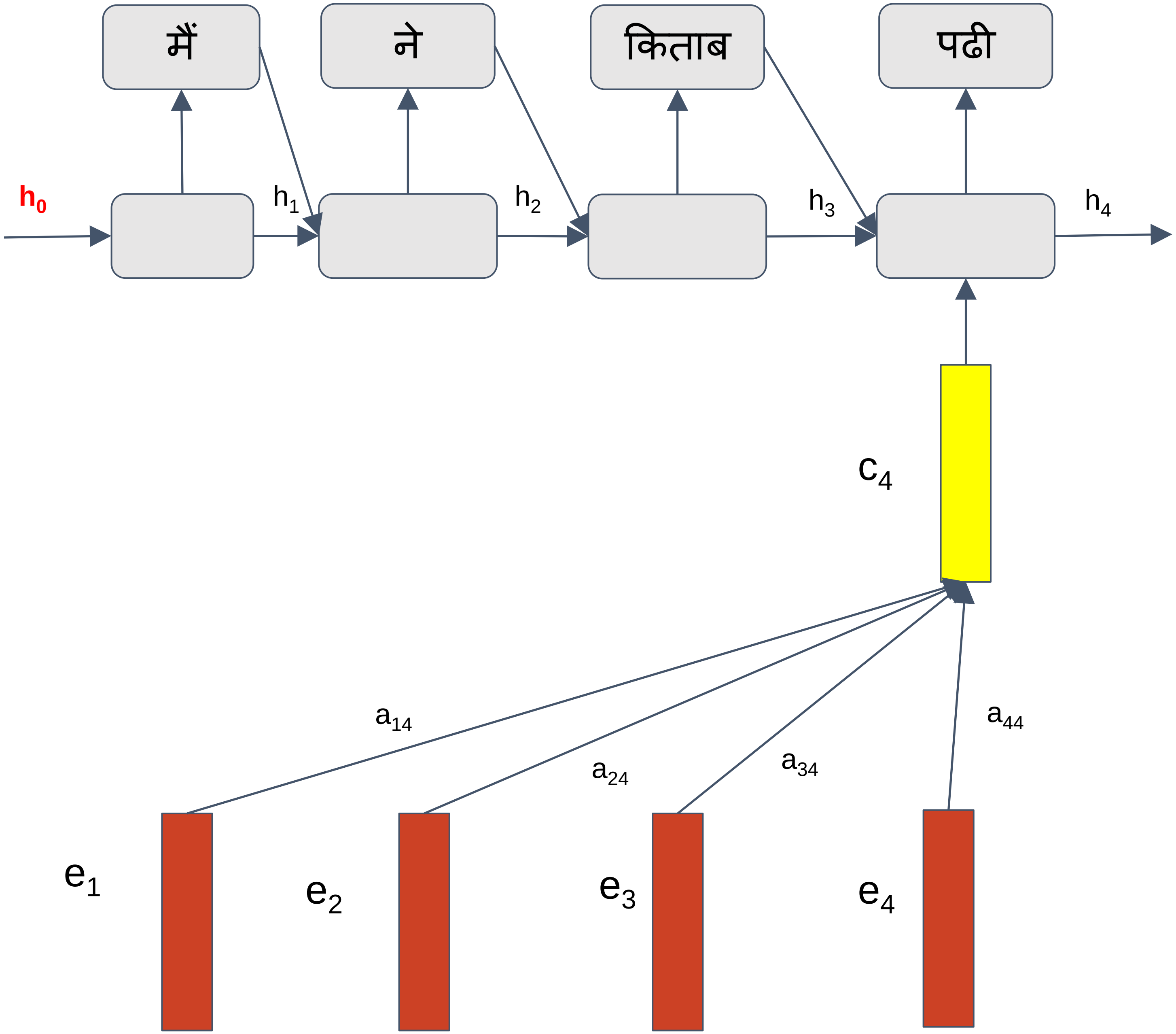
Let's see an example of how the ***attention mechanism*** *works during decoding*

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

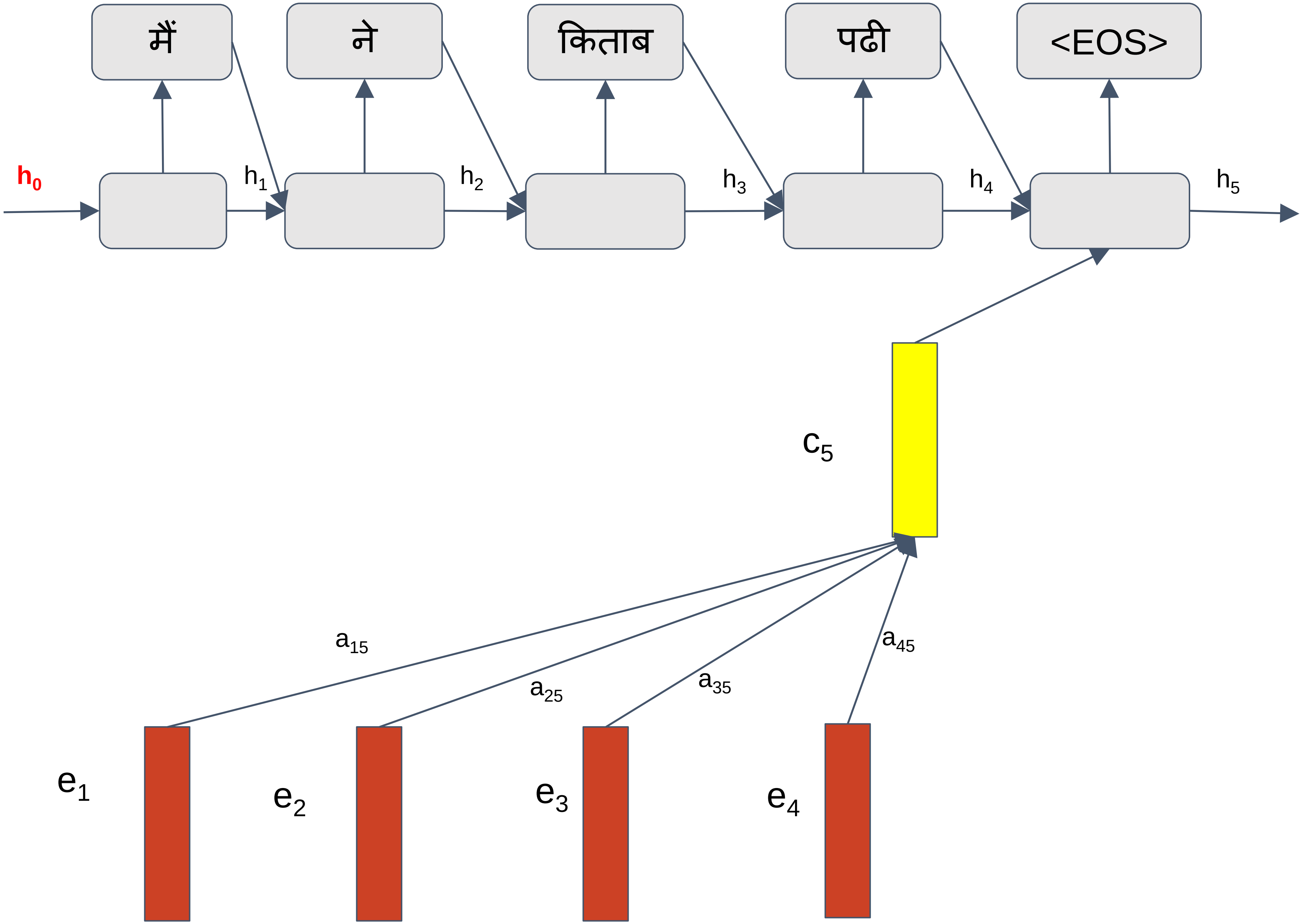
For generation of i^{th} output character:
c_i : context vector
a_{ij} : annotation weight for the j^{th} annotation vector
e_j: j^{th} annotation vector


मैं
ने
h0
h1
h2
c2
a12
a22
a32
a42
e1
e2
e3
e4


मैं
ने
किताब
h0
h1
h2
h3
c3
a13
a23
a33
a43
e1
e2
e3
e4



मैं
ने
किताब
पढी
h_0
h_1
h_2
h_3
h_4
c_4
a_{14}
a_{24}
a_{34}
a_{44}
e_1
e_2
e_3
e_4



मैं
ने
किताब
पढी
<EOS>
h_0
h_1
h_2
h_3
h_4
h_5
c_5
a_{15}
a_{25}
a_{35}
a_{45}
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

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

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

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

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

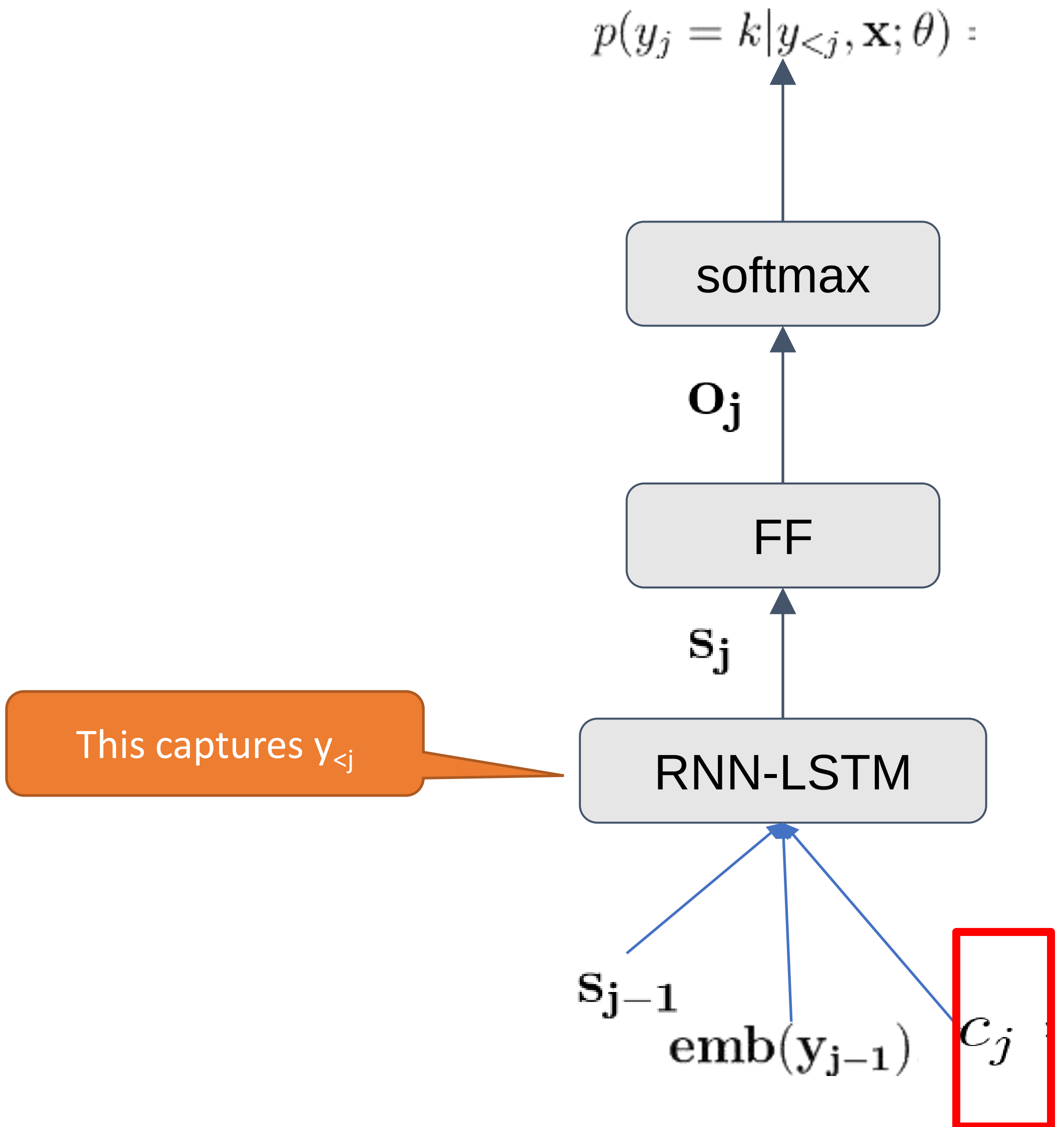
$$\alpha_{ij} = g(s_{j-1}, e_i, \mathbf{emb}(y_{j-1}))$$

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

Final context vector is weighted average of encoder outputs

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

Let us revisit what the decoder does at time step t



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

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

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

This captures source (x)

Topics

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

The models discussed so far do not use monolingual data

Can monolingual data help improve NMT models?

Backtranslation

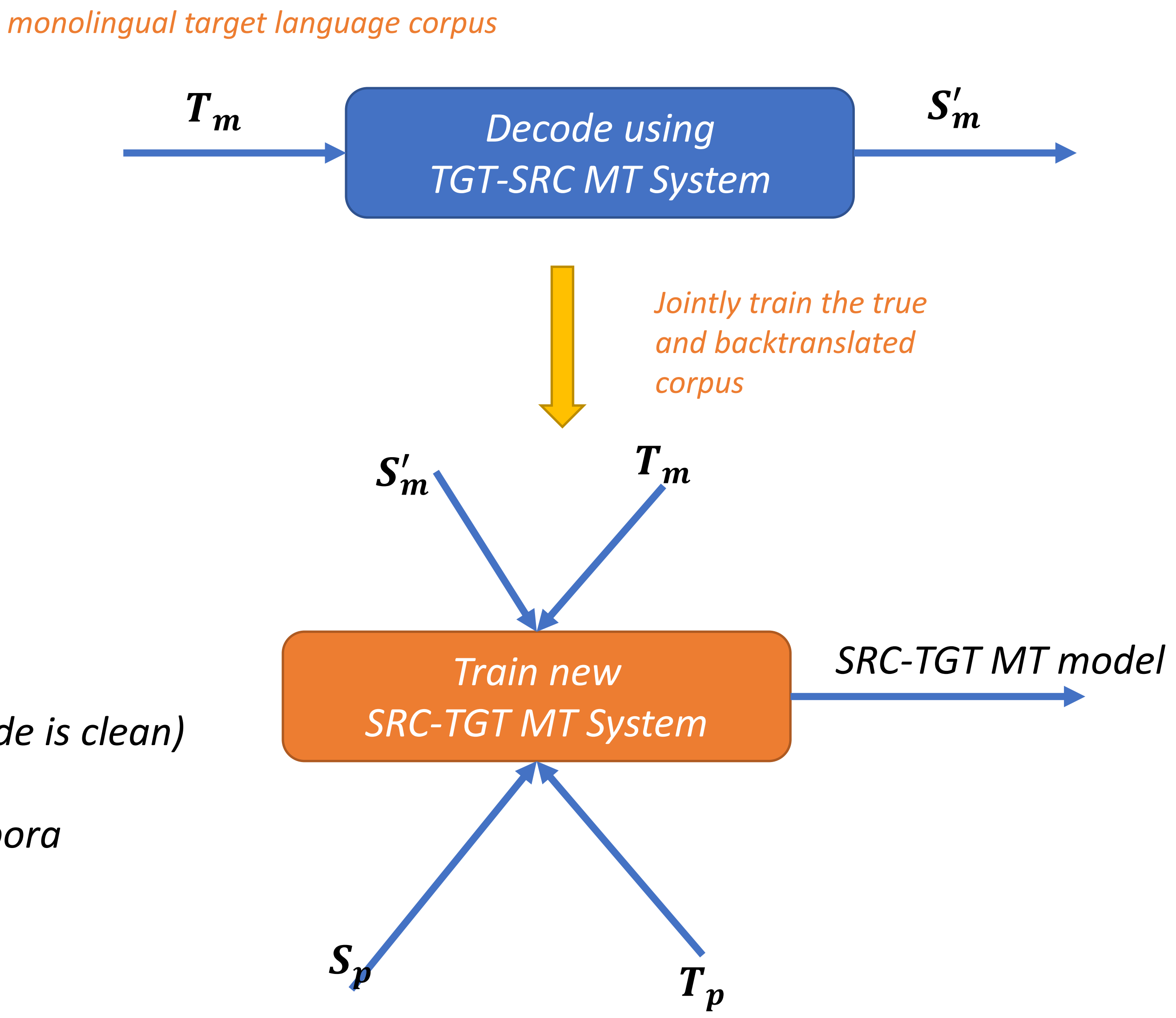
Create pseudo-parallel corpus using Target to source model *(Backtranslated corpus)*

Need to find the right balance between true and backtranslated corpus

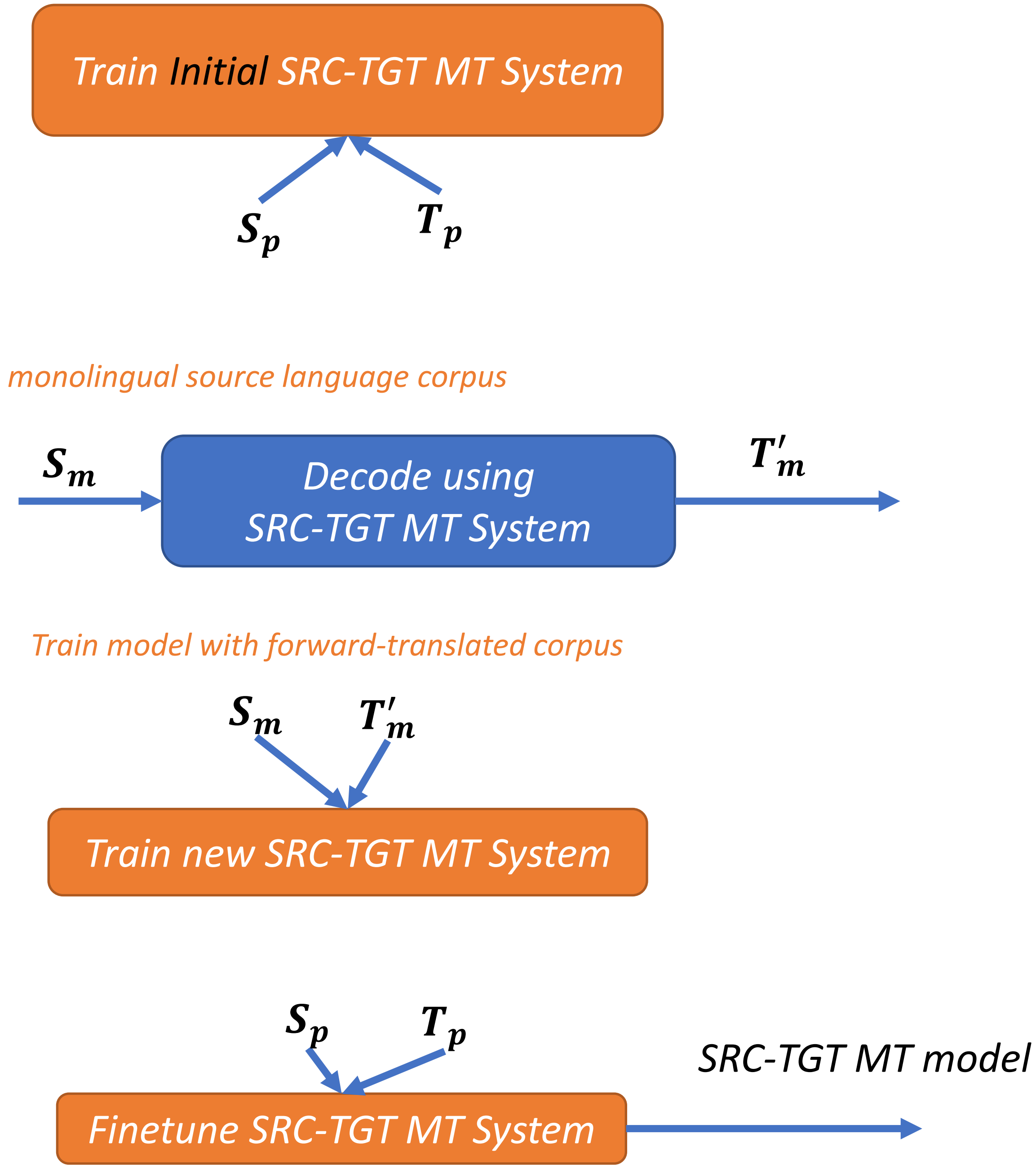
Why is backtranslation useful?

- *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



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*

Why is self-training useful?
- *Adaptation to source language domain*
- *Prevent overfitting by exposure to diverse corpora*

Works well if the initial model is reasonably good

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

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

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, Sentence-piece

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

<u>*Words to encode*</u> <u>*Iterations*</u>

Words to encode	1	2	3	4
BADD	BADD	BP_1D	BP_1D	BP_3
FAD	FAD	FP_1	FP_1	FP_1
FEEDE	FEEDE	FEEDE	FP_2DE	FP_2DE
ADDEEF	ADDEEF	P_1DEEF	P_1DP_2F	P_3P_2F

<u>Data-dependent segmentation</u>

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

Problems with subword level translation

Unwanted splits:

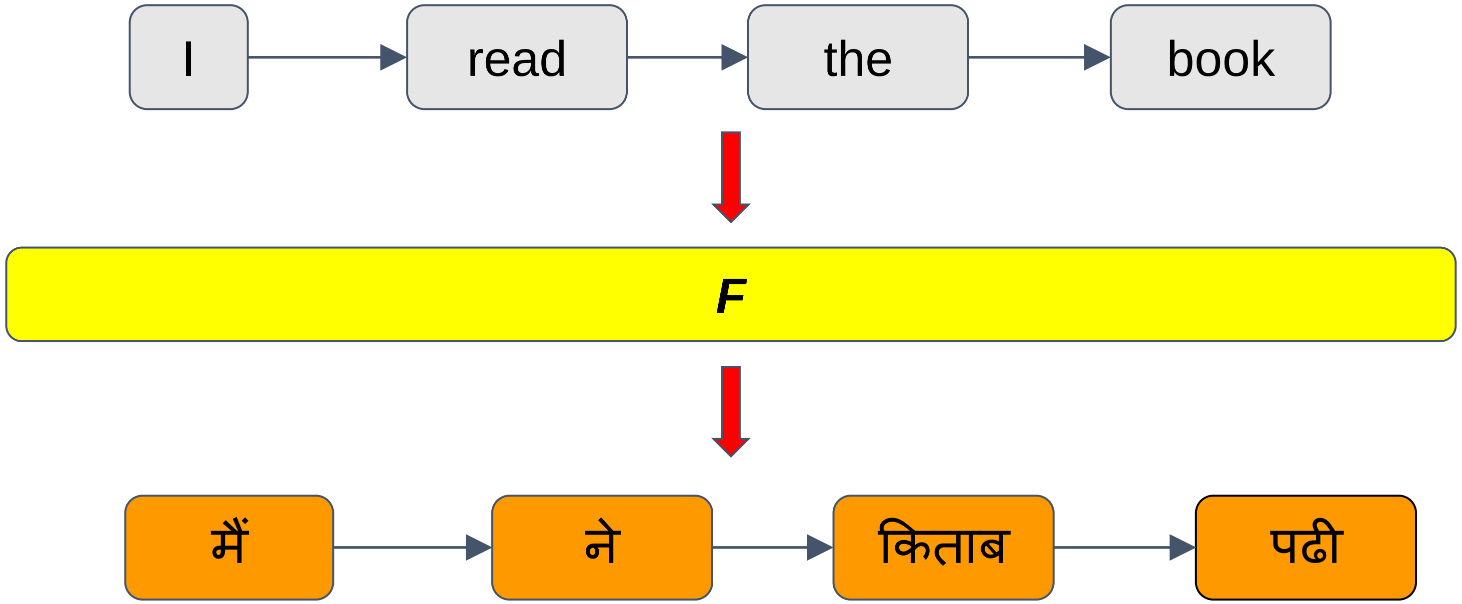
नाराज़ → ना राज़ ➔ no secret

Problem is exacerbated for:

- Named Entities
- Rare Words
- Numbers

We can look at translation as a sequence to sequence transformation problem

*Read the entire sequence and predict the output sequence (using function **F**)*



- Length of output sequence need not be the same as input sequence
- Prediction at any time step *t* has access to the entire input
- A very general framework

Sequence to Sequence transformation is a very general framework

Many other problems can be expressed as sequence to sequence transformation

- *Summarization: Article ⇒ Summary*
- *Question answering: Question ⇒ Answer*
- *Transliteration: character sequence ⇒ character sequence*

- *Image labelling: Image ⇒ Label*
- *Speech Recognition, TTS, etc.*

- *Note $\Rightarrow$ no separate language model*
- *Neural MT generates fluent sentences*
- *Quality of word order is better*
- *No combinatorial search required for evaluating different word orders:*
  - *Decoding is very efficient compared to PBSMT*
- *End-to-end training*
- *Attention as soft associative lookup*

Outline

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

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

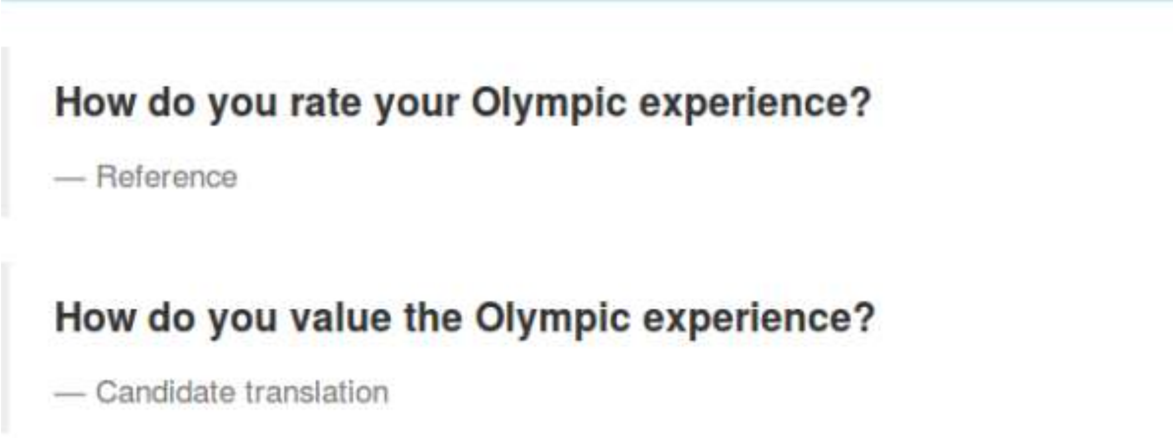
For example, I am attending a lecture

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



Adequacy:

Is the meaning translated correctly?

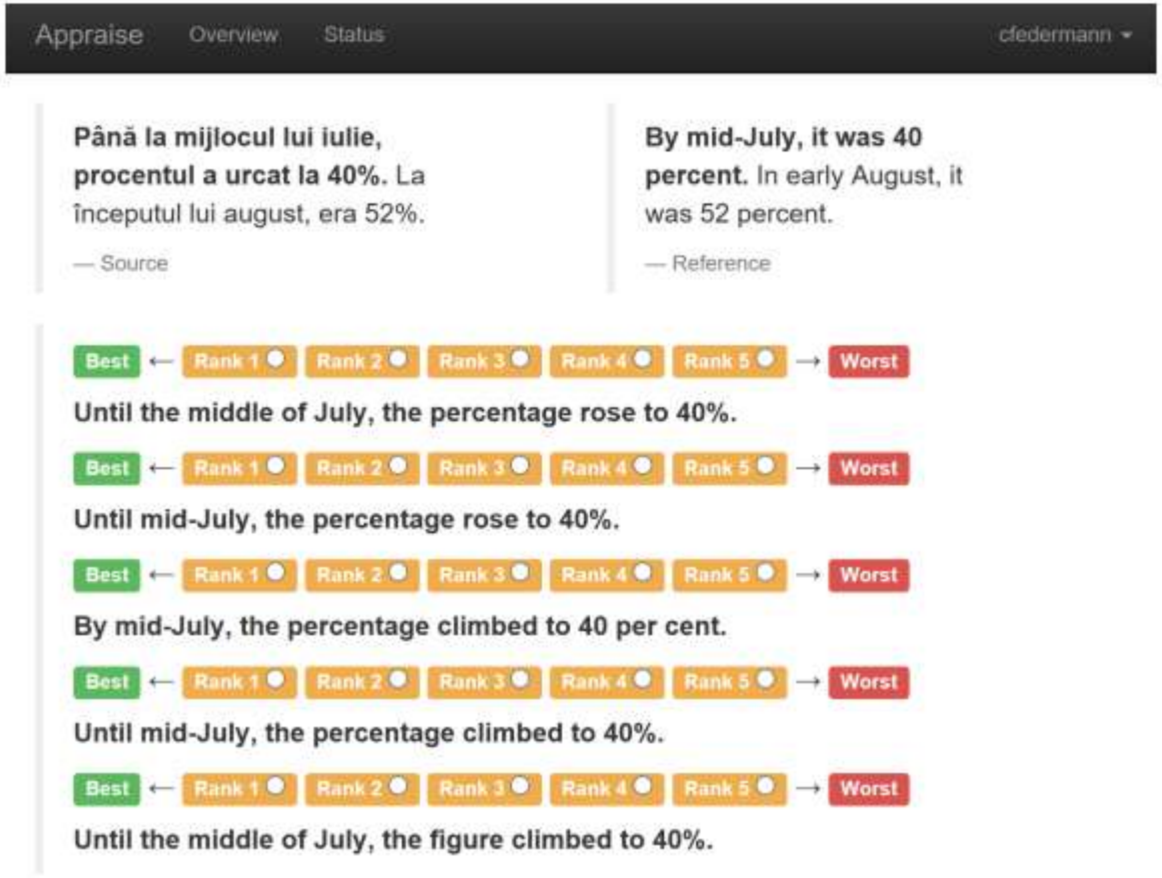
5 = All
4 = Most
3 = Much
2 = Little
1 = None

Fluency:

Is the sentence grammatically valid?

5 = Flawless
4 = Good
3 = Non-native
2 = Disfluent
1 = Incomprehensible

Ranking Translations



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

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

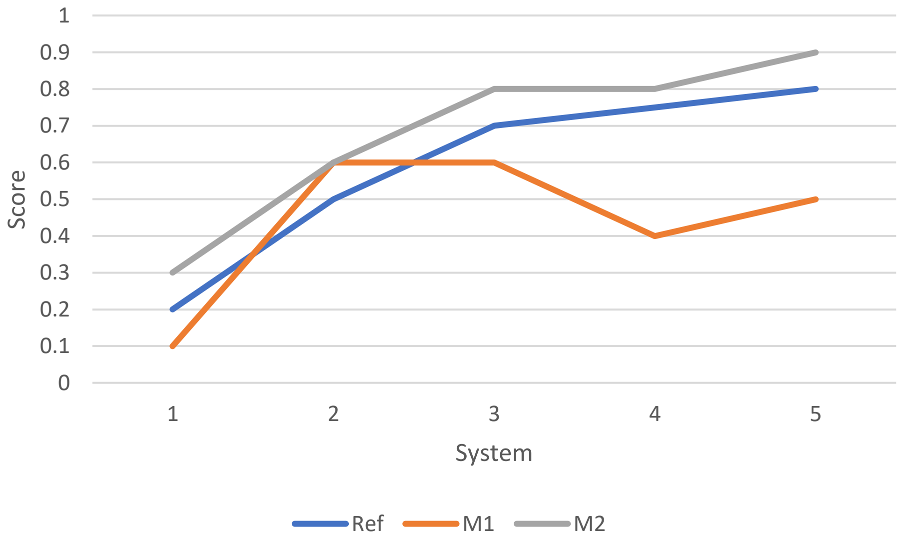
Multiple references are better

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?



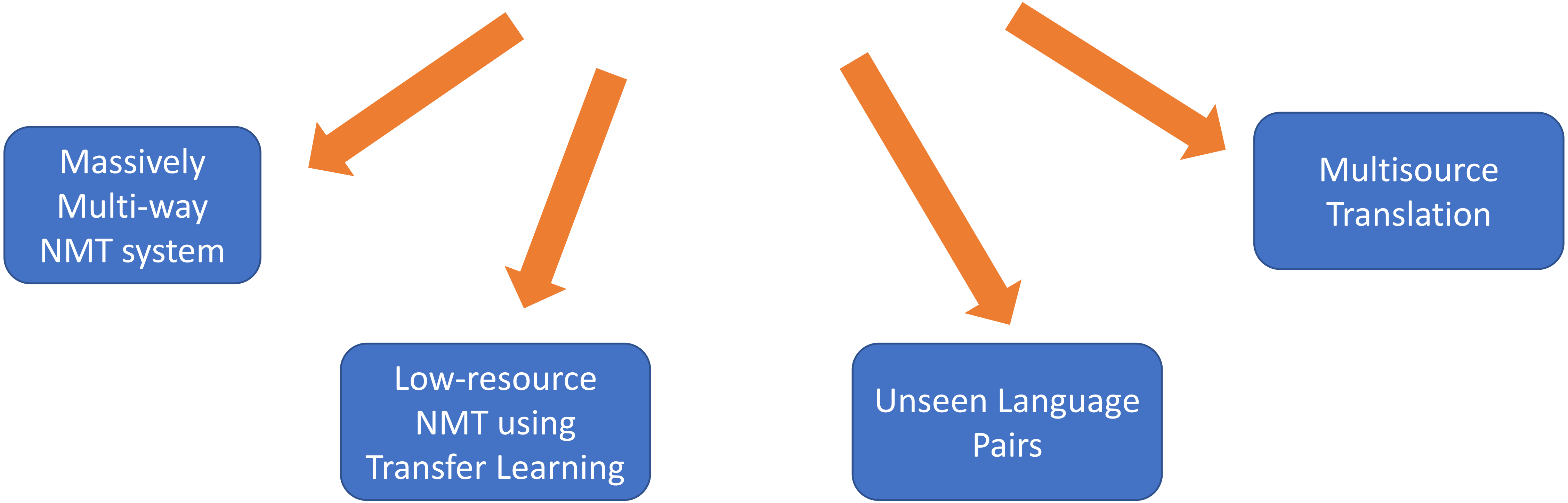
Outline

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

Multilingual Neural Machine Translation

NMT Models involving more than two languages

Use-cases for Multilingual NMT

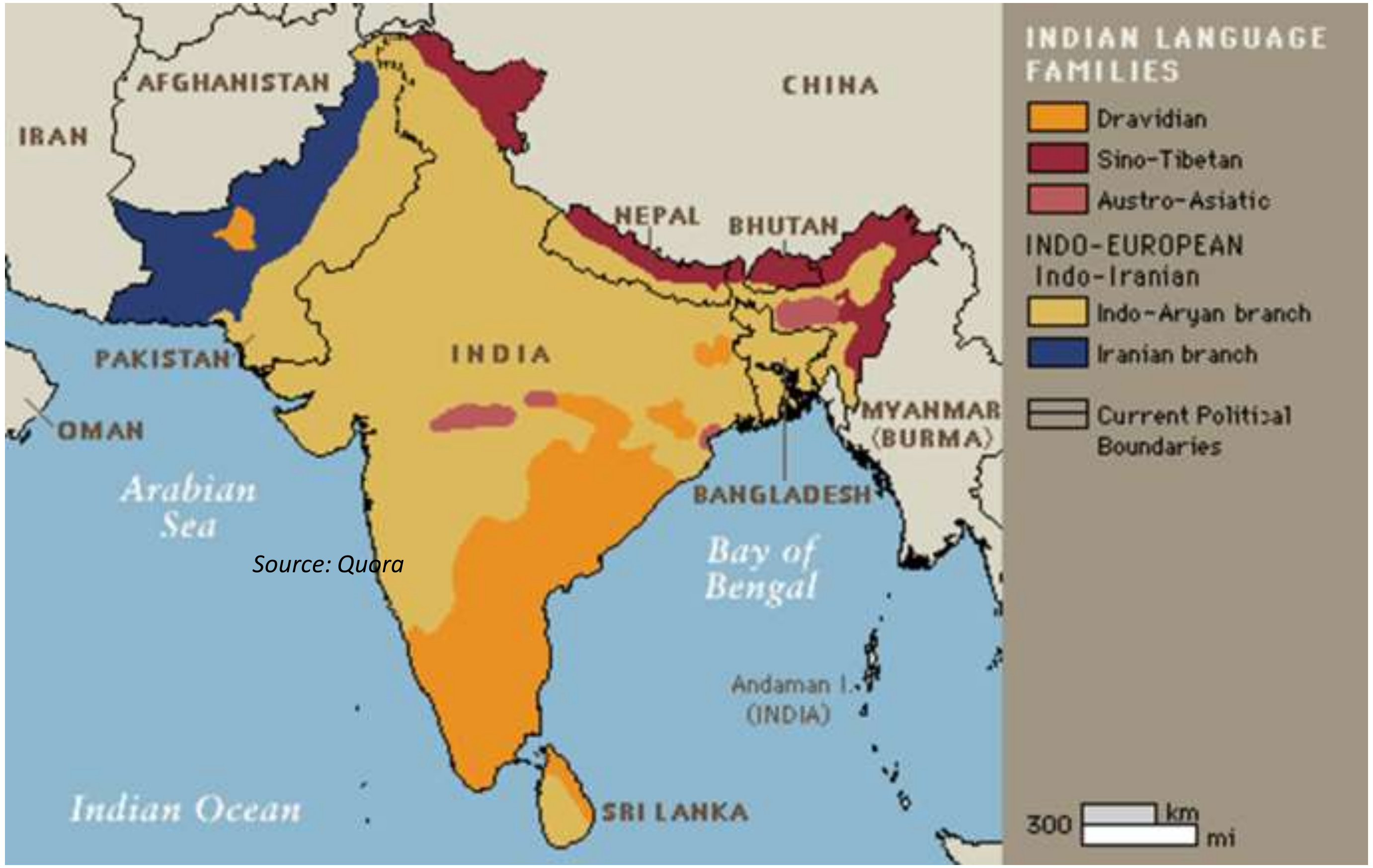


Raj Dabre, Chenhui Chu, Anoop Kunchukuttan. *A Comprehensive Survey of Multilingual Neural Machine Translation*. pre-print arxiv: 2001.01115

Diversity of Indian Languages

Highly multilingual country

Greenberg Diversity Index 0.9



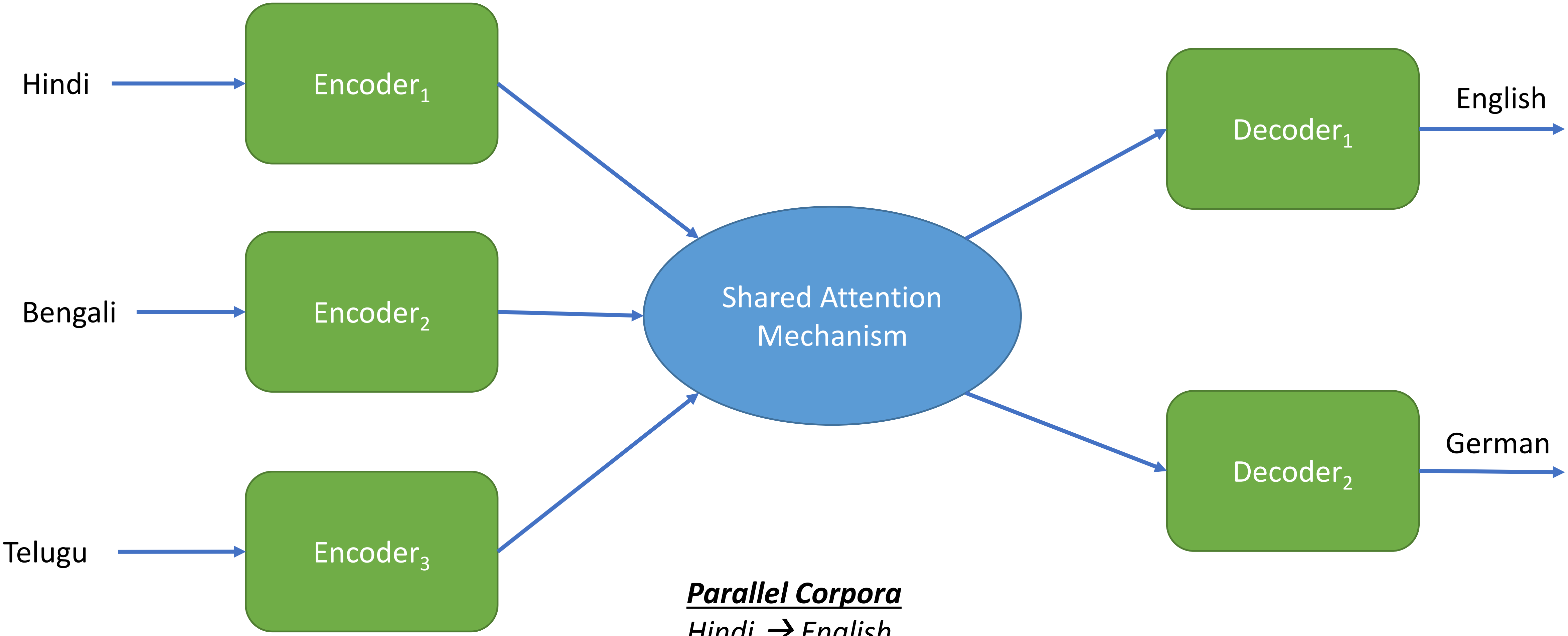
Source: Quora

- *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

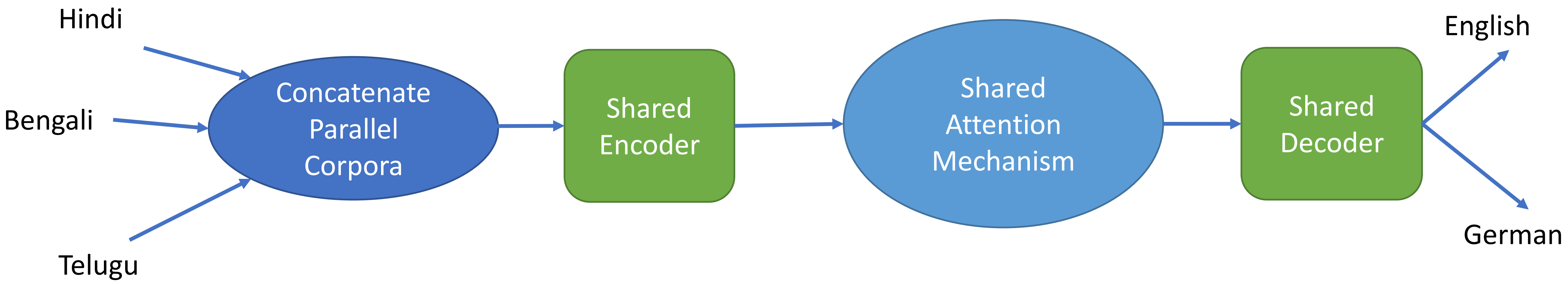
Hindi → English

Telugu → English

Bengali → German

Compact Multilingual NMT

(Johnson et al., 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

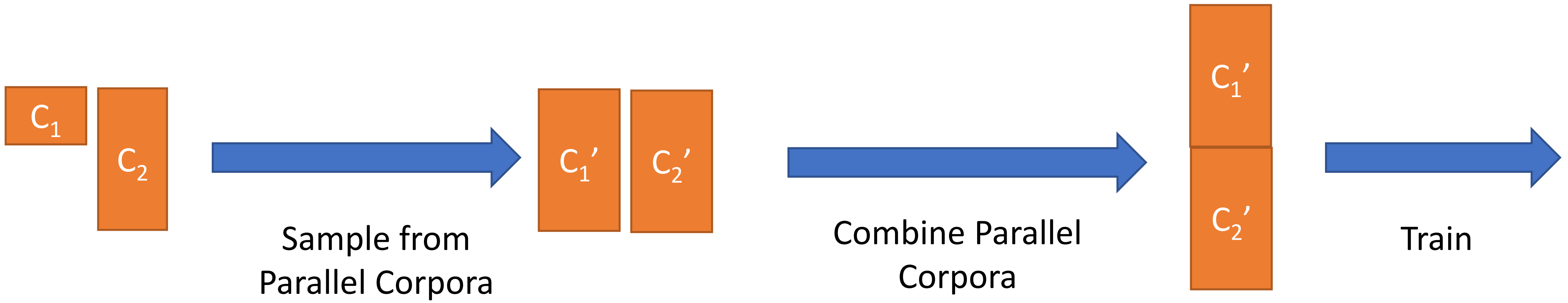
There is only one decoder, how do we generate multiple languages?

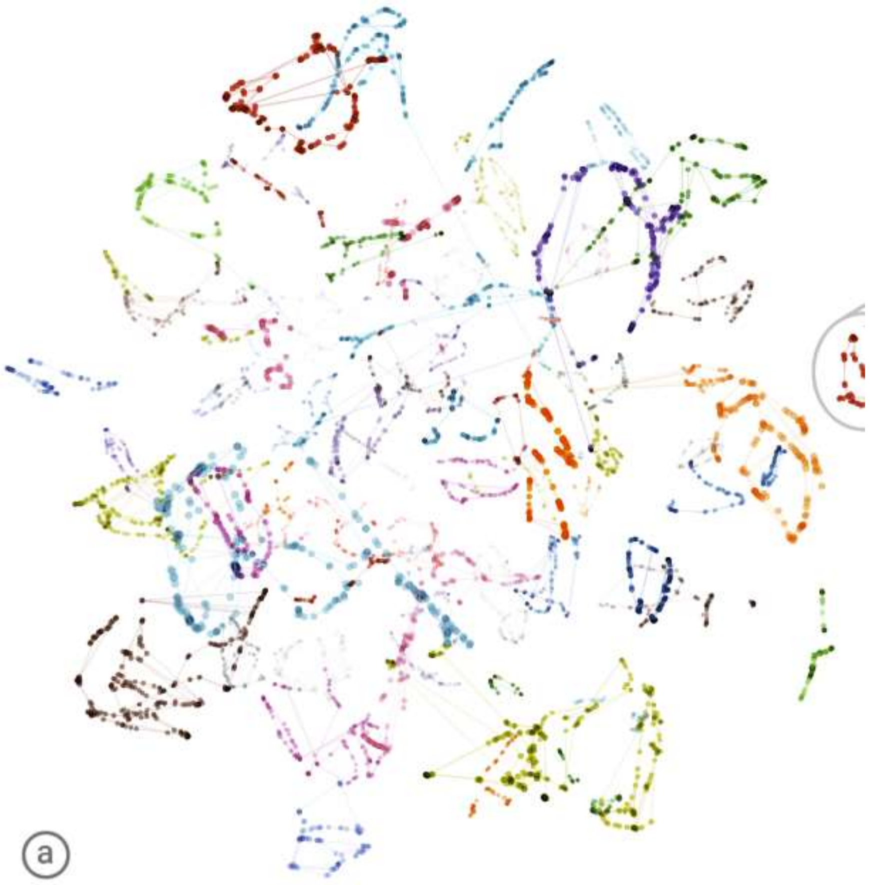
Language Tag Trick → Special token in input to indicate target language

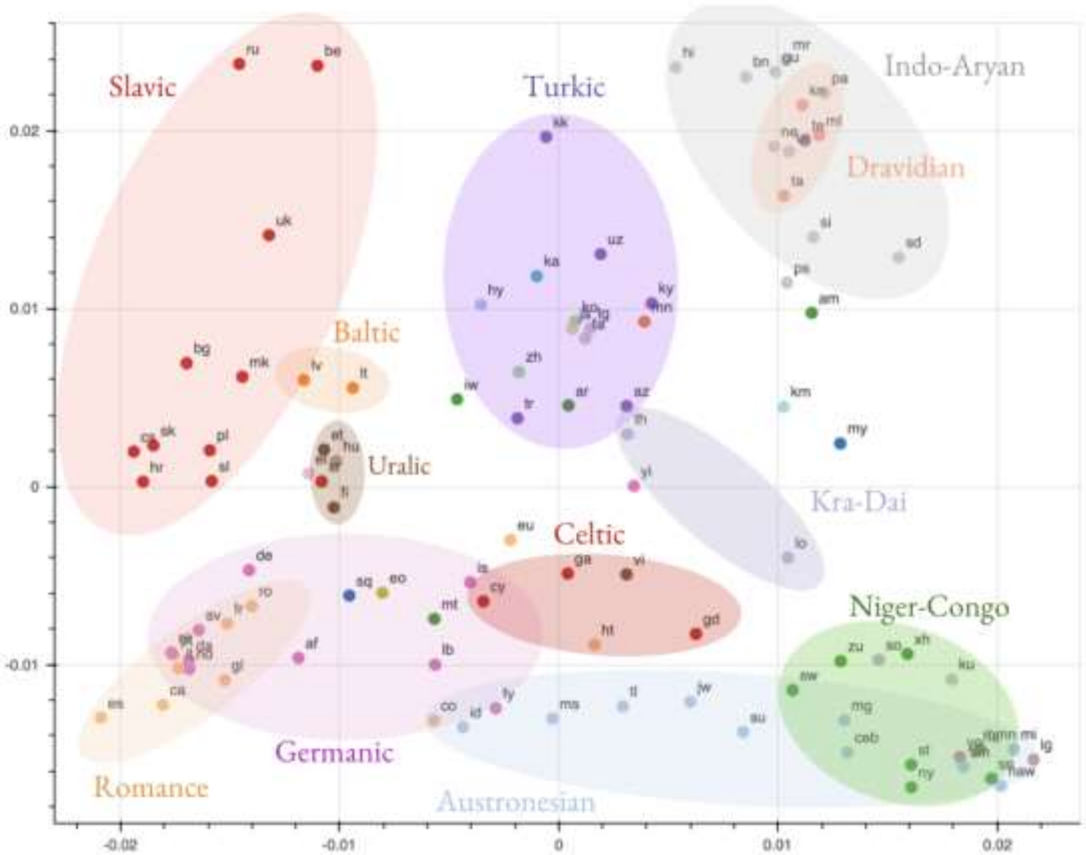
Original Input: *मकर संक्रांति भगवान सूर्य के मकर में आने का पर्व है*

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

Joint Training



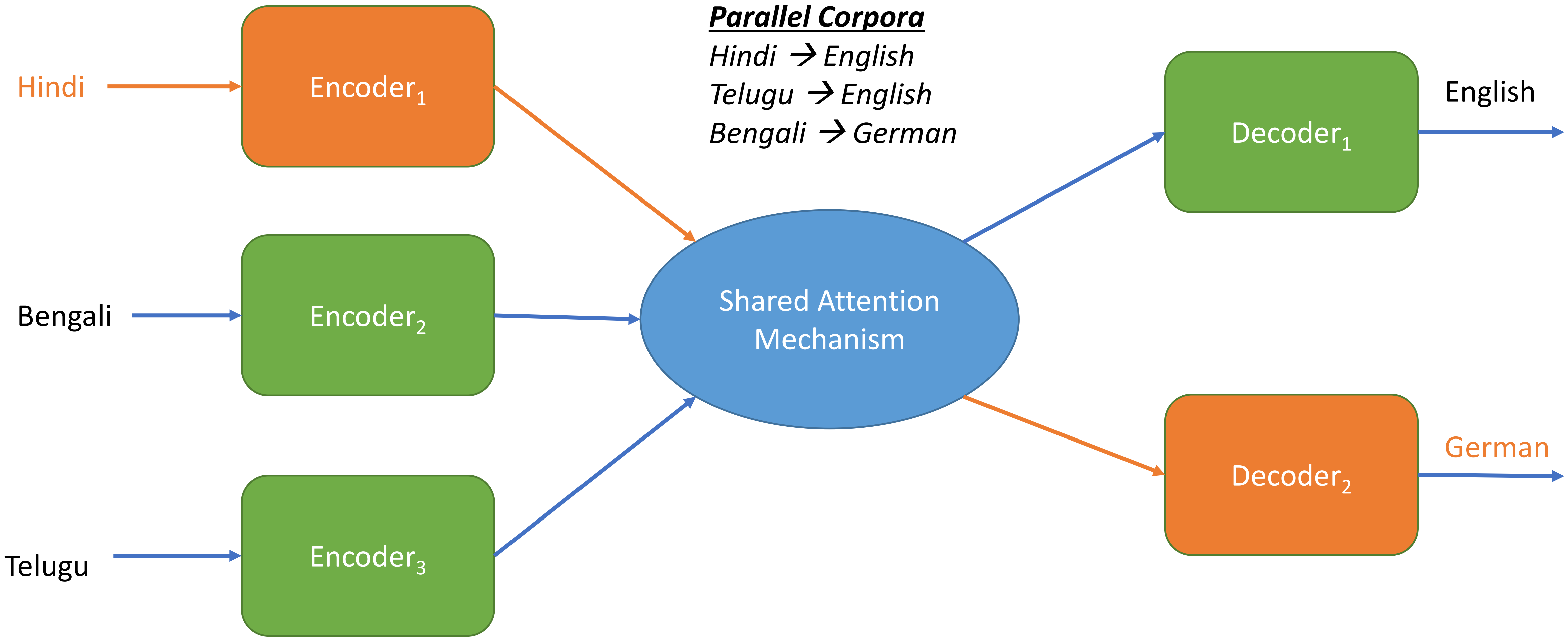




Similar sentences have similar encoder representations

But the multilingual representation is not perfect

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



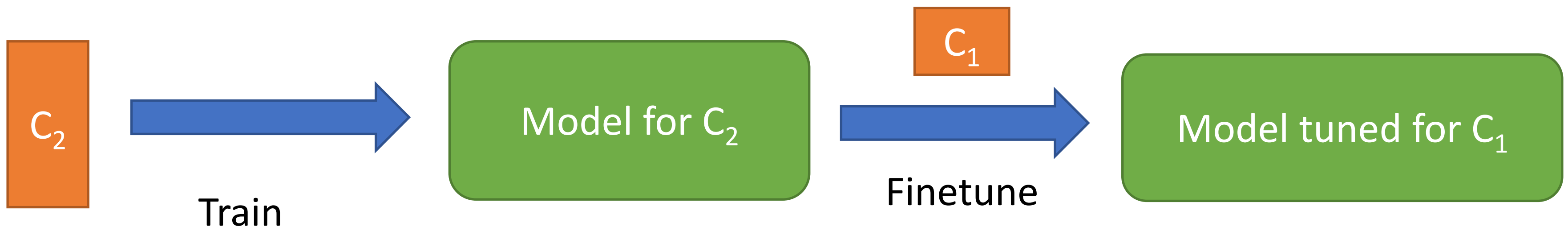
Multilingual NMT makes possible translation between unseen pairs

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

Outline

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

Summary

- Machine Translation is one of the most challenging and exciting NLP problems
  - Watch out for advances in MT!
- Machine Translation is important to build multilingual NLP systems
- NMT has been a great success story for Deep Learning
- NMT has the following benefits
  - Improved Fluency & better Word Order
  - Opens up new avenues: Transfer learning, Unsupervised NMT, Zeroshot NMT

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]

Machine Translation for Related Languages. *Statistical Machine Translation between related languages. Tutorial at NAACL 2016.* [slides]

Multilingual Learning: A related area you should read about. [slides]

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
- IITB English-Hindi Parallel Corpus
- IIIT-Hyderabad PIB and MKB 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