An Introduction to Machine Translation and Transliteration

Anoop Kunchukuttan

Center for Indian Language Technology,
Indian Institute of Technology Bombay

anoopk@cse.iitb.ac.in

https://www.cse.iitb.ac.in/~anoopk

THINK Summer School on Machine Learning.
Vidyalankar Institute of Technology, Mumbai
19th June 2017

Outline

- Introduction
- Machine Translation Paradigms
- Phrase-based SMT
- Extensions to Phrase-based SMT
- Evaluation of Machine Translation
- Neural Machine Translation
- Summary

Outline

- Introduction
- Machine Translation Paradigms
- Phrase-based SMT
- Extensions to Phrase-based SMT
- Evaluation of Machine Translation
- Neural Machine Translation
- Summary

What is Machine Translation?

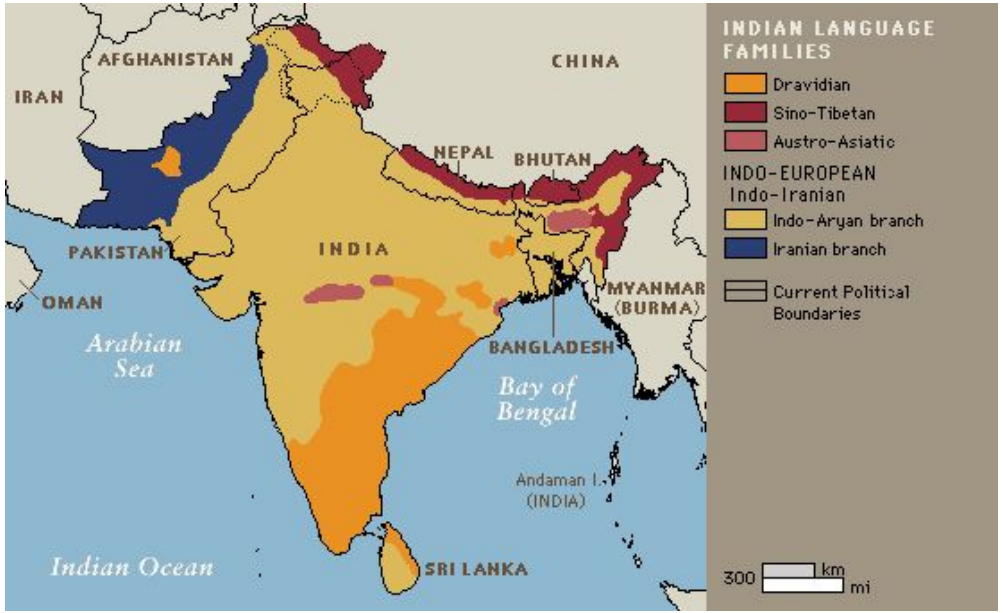
Automatic conversion of text/speech from one natural language to another

Be the change you want to see in the world

वह परिवर्तन बनो जो संसार में देखना चाहते हो

Why do we need machine translation?

- **5+1 language families**
  - Indo-Aryan (74% population)
  - Dravidian (24%)
  - Austro-Asiatic (1.2%)
  - Tibeto-Burman (0.6%)
  - Andaman languages (2 families?)
  - + English (West-Germanic)
- **22 scheduled languages**
- **11 languages with more than 25 million speakers**
  - 30 languages with more than 1 million speakers
  - Only India has 2 languages (+English) in the world's 10 most spoken languages
  - 7-8 Indian languages in the top 20 most spoken languages



Machine Translation Usecases

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

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

Why is Machine Translation difficult?

Language Divergence: the great diversity among languages of the world

- Word order: SOV (Hindi), SVO (English), VSO, OSV,
- Free (Sanskrit) vs rigid (English) word order
- Analytic (Chinese) vs Polysynthetic (Finnish) languages
- Different ways of expressing same concept
- Case marking systems
- Language registers
- Inflectional systems [infixing (Arabic), fusional (Sanskrit), agglutinative (Marathi)]

… and much more

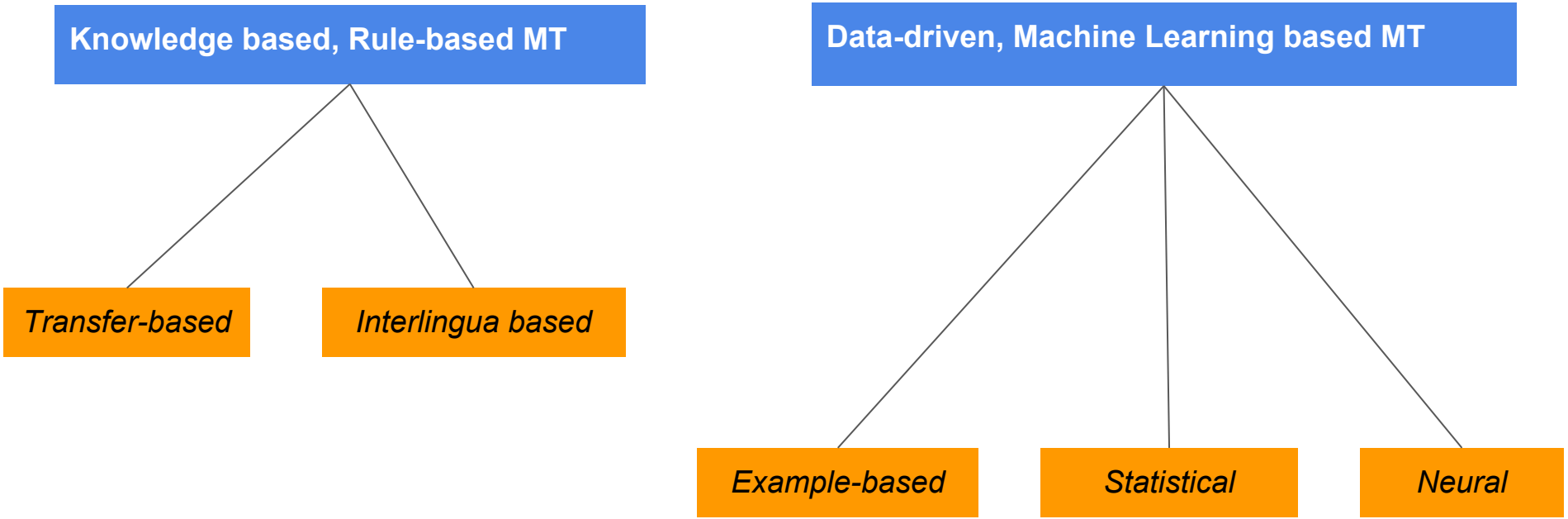
Why is Machine Translation difficult?

- **Ambiguity**
  - Same word, multiple meanings:
  - 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

Outline

- Introduction
- Machine Translation Paradigms
- Phrase-based SMT
- Extensions to Phrase-based SMT
- Evaluation of Machine Translation
- Neural Machine Translation
- Summary

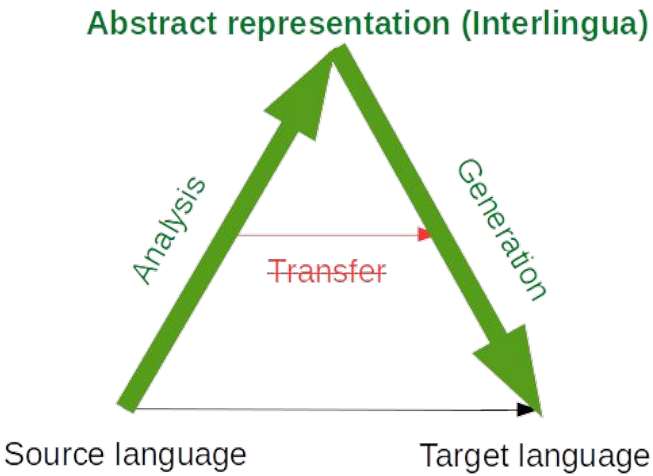
Approaches to build MT systems



Rule-based MT

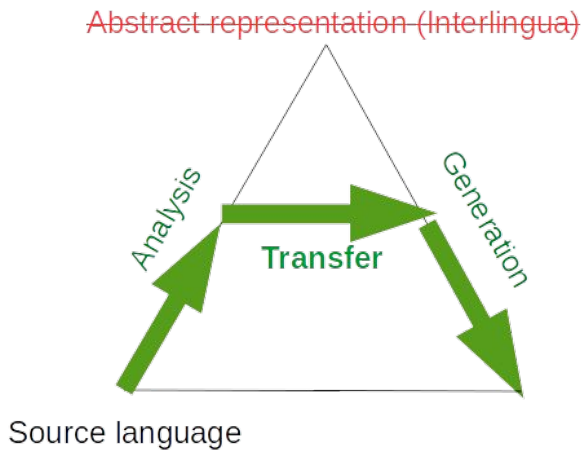
- Rules are written by ***linguistic experts*** to analyze the source, generate an intermediate representation, and generate the target sentence
- Depending on the depth of analysis: interlingua or transfer-based MT

Interlingua based MT



Deep analysis, complete disambiguation and language independent representation

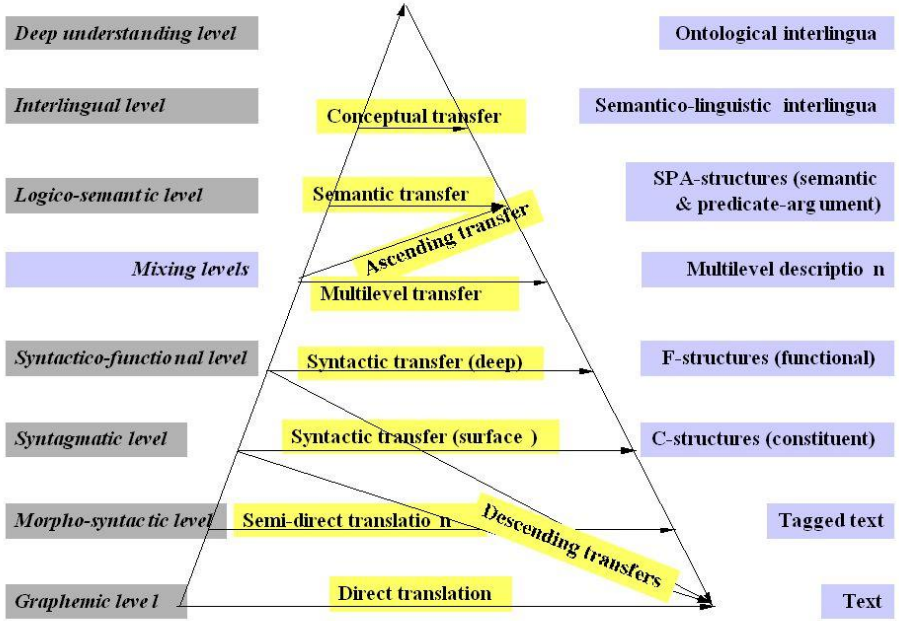
Transfer based MT



Partial analysis, partial disambiguation and a bridge intermediate representation

Vauquois Triangle

Translation approaches can be classified by the depth of linguistic analysis they perform

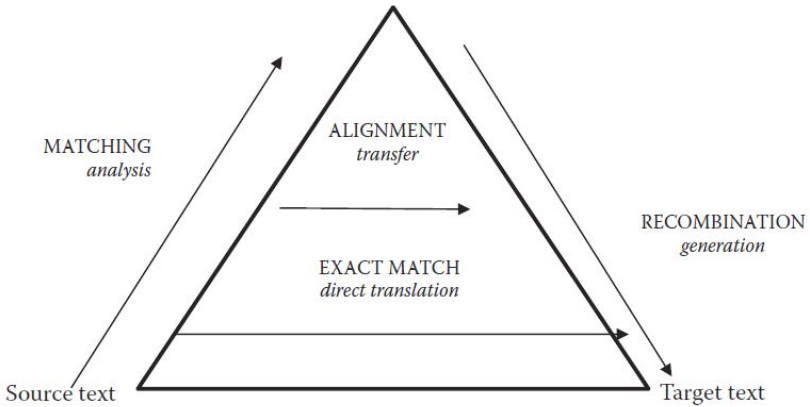


Problems with rule based MT

- Required linguistic expertise to develop systems
- Maintenance of system is difficult
- Difficult to handle ambiguity
- Scaling to a large number of language pairs is not easy

Example-based MT

Translation by analogy ⇒ *match parts of sentences to known translations and then combine*



Input: *He buys a book on international politics*

1. **Phrase fragment matching: (*data-driven)***

   he buys
   a book
   international politics

2. **Translation of segments: (*data-driven)***

   वह खरीदता है
   एक किताब
   अंतर राष्ट्रीय राजनीति

3. **Recombination:** *(human crafted rules/templates)*

   वह अंतर राष्ट्रीय राजनीति पर एक किताब खरीदता है

- *Partly rule-based, partly data-driven.*
- *Good methods for matching and large corpora did not exist when proposed*

Outline

- Introduction
- Machine Translation Paradigms
- Phrase-based SMT
- Extensions to Phrase-based SMT
- Evaluation of Machine Translation
- Neural Machine Translation
- Summary

Statistical Machine Translation

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 बैठी है

Machine Learning

Let's begin with a simplified view of Statistical Machine Translation (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

Machine Learning

* Learn word/phrase alignments

Let's begin with a simplified view of Statistical Machine Translation (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

Machine Learning

* Learn word/phrase alignments
* Learning to reorder

Let's begin with a simplified view of Statistical Machine Translation (SMT)!!

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: source language sentence
f : target 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

We explain translation using the Noisy Channel Model

A very general framework for many NLP problems

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

The SMT Workflow

Training

- Given: Parallel Corpus
- Learn Model: P(e) and P(f|e)
- Offline, one-time process
- Learning Objective: Maximize Likelihood

$$P^*(f|e) = \arg\max \mathbf{Likelihood}(data; P(f|e))$$

Decoding

- Given:
  - Sentence **f** in language **F**
  - Model: P(e) and P(f|e)
- Output: Translation **e** for **f**
- Online process, should be fast
- TM & LM are used for scoring translation candidates

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

Let's see how to learn translation model P(f|e) and language model P(e)

Phrase-based Translation Model

- *Let's see one of the most successful translation models: PBSMT*
- *Widely used in commercial systems like Google Translate (till recently)*
- *Basic unit of translation is a phrase*
- *A phrase is just a sequence of words*

- 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 do the right translation

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

So how the model look now?

- Source sentence can be segmented in $\mathbf{I}$ phrases
- Then, $p(\mathbf{f}|\mathbf{e})$ 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 i^{th} phrase of **e**
end_i :end position in **f** of i^{th} phrase of **e**

Language Model

Measures how likely a sentence is P(e) ⇒ a proxy for grammatical correctness/fluency

$$P(\mathbf{e}) = P(e_1, e_2, \ldots, e_k)$$

$$= \Pi_{i=1..k} P(e_i | e_{i-1..} e_1)$$ ← *Chain Rule*

$$= \Pi_{i=1..k} P(e_i | e_{i-1..} e_{i-n+1})$$ ← *Markov assumption*

How to estimate $P(e_i | e_{i-1..} e_{i-n+1})$?

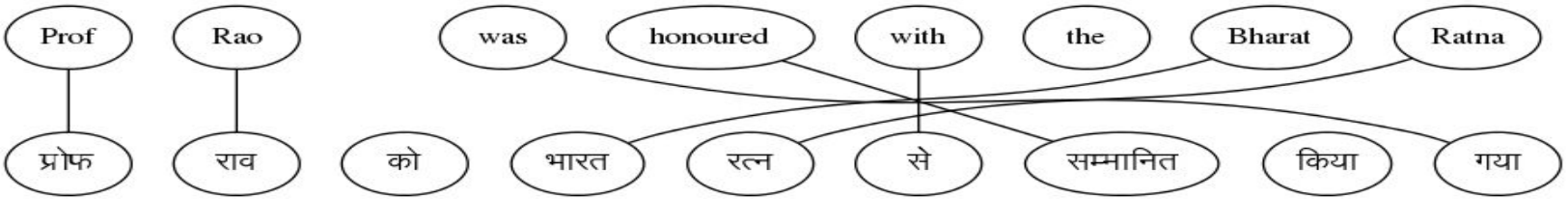
- We can estimate the probabilities by counting from a monolingual corpus
- P(book|the)=#(the,book)/#(the)
- A little complication: what happens if *book* never comes in the training corpus
- That's the complicated part of language modelling, let's skip it for now!

Training a Phrase-based SMT system

- Building the Language Model
- Building the Translation Model
  - Word Alignment (find word-level correspondences)
  - Phrase Extraction (extract phrase pairs)
- Tuning the Model

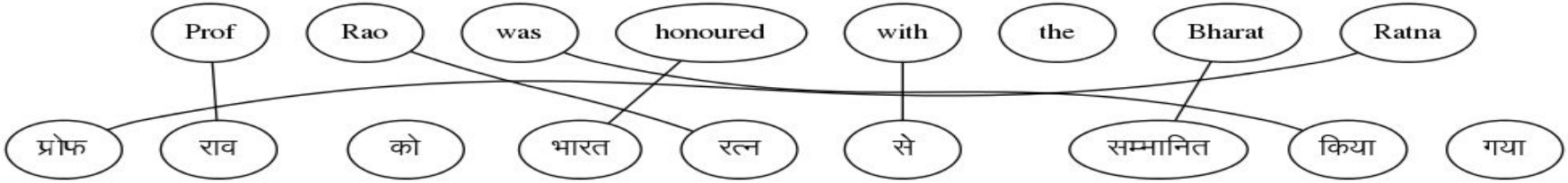
Word Alignment

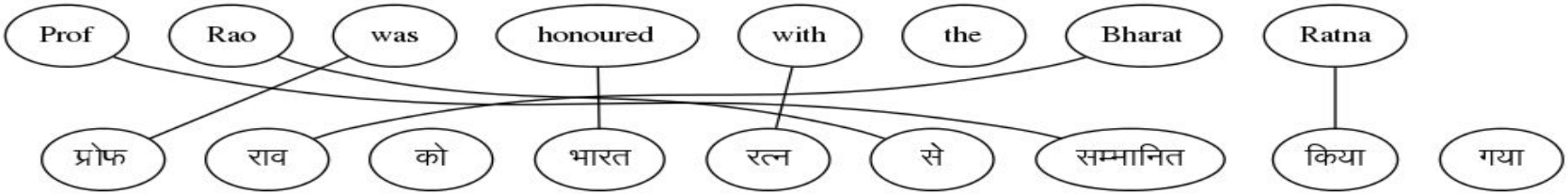
- Central Task in Statistical Machine Translation
- Given a parallel sentence pair, find word level correspondences *(alignment, let's say a)*



But there are multiple possible alignments

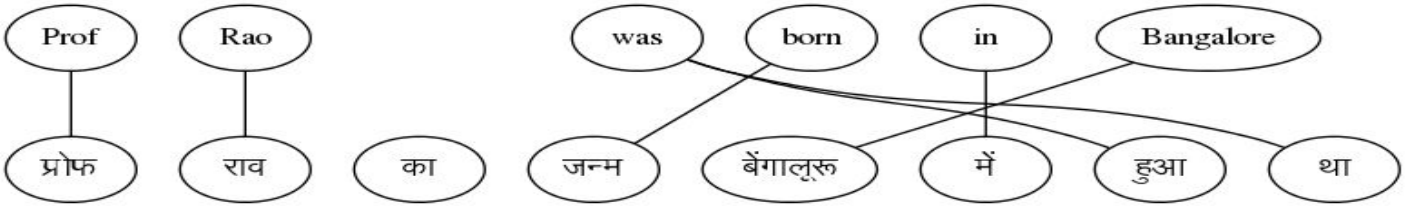
Sentence 1

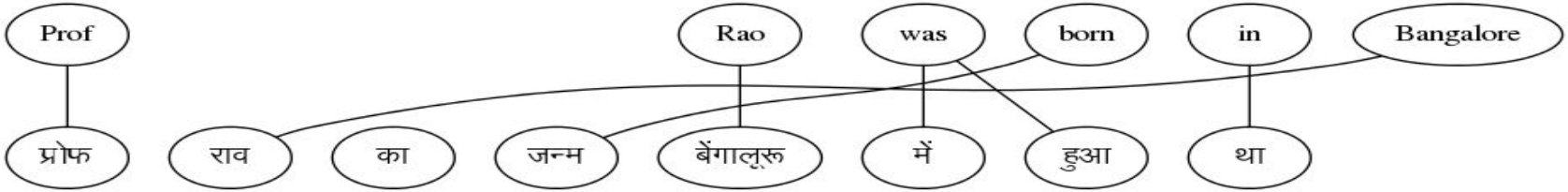




But there are multiple possible alignments

Sentence 2





How do we find the correct alignment?

Key idea

Co-occurrence of words

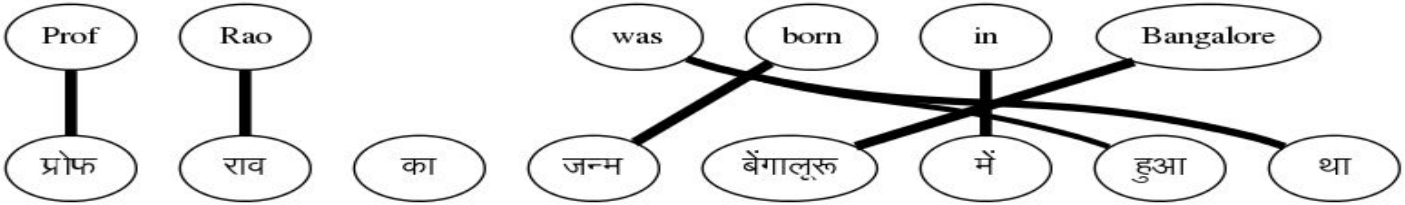
- Words which occur together in the parallel sentence are likely to be translations (*higher P(f|e)*)
- Alignments which have more likely word-translation pairs are more likely (*higher P(a)*)
- It’s a chicken-and-egg problem!
- How to actually find the best alignment?

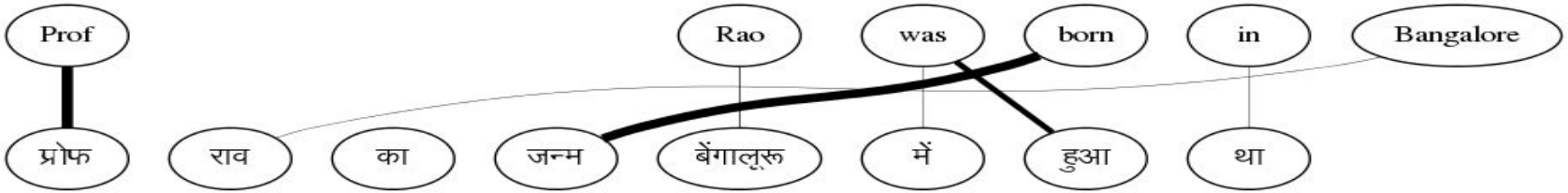
Expectation-Maximization Algorithm

- Find the best ***hidden*** alignment
- A key algorithm for various machine learning problems
  - Start with a random alignment
  - Find P(f|e) given the alignments
  - Now compute alignment probabilities P(a) with these new translation probabilities
  - Do this repeatedly till P(f|e) does not change

At the end of the process

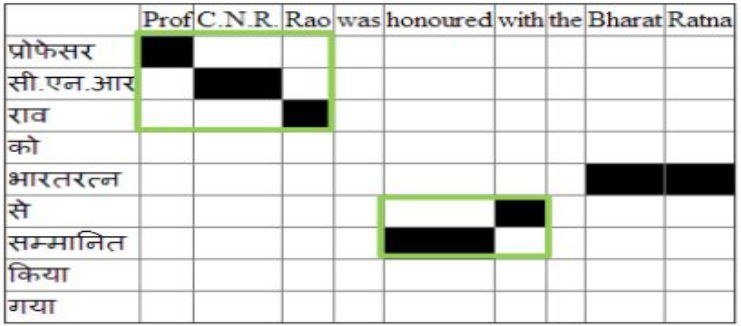
Sentence 2





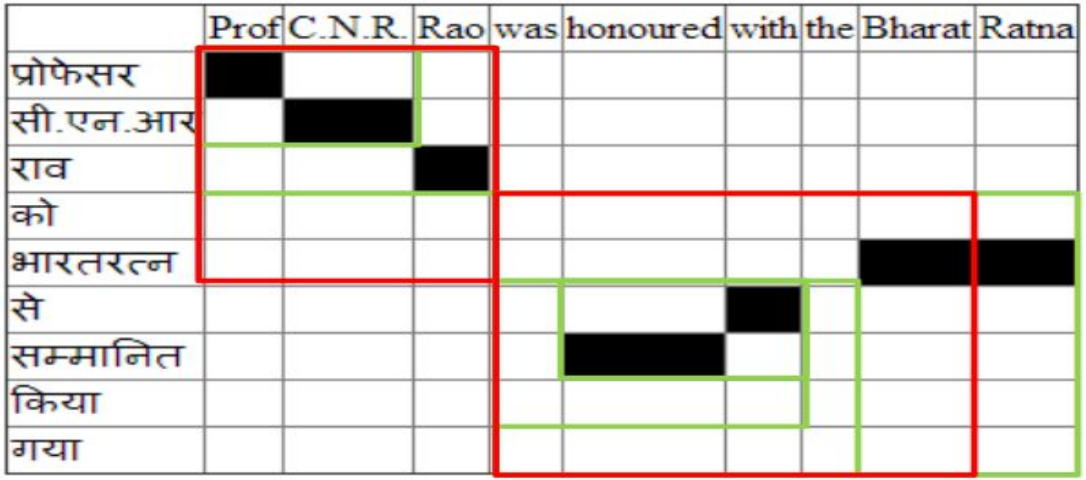
Learning Phrase Tables from Word Alignments

- Leverages word alignments learnt from IBM models
- 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"

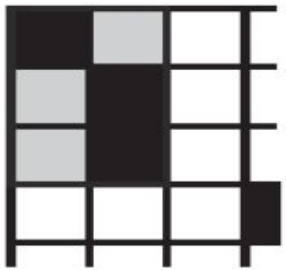


Which phrase pairs to include in the phrase table?

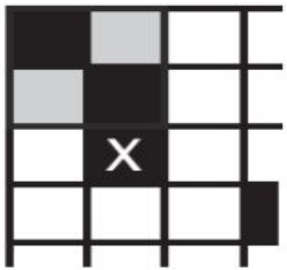
Extracting Phrase Pairs



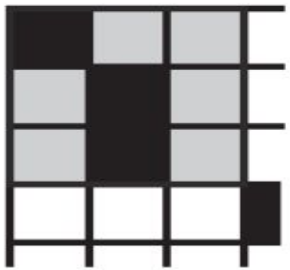
Phrase Pairs “consistent” with word alignment



consistent



inconsistent



consistent

Source: SMT, Phillip Koehn

Examples

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

Computing Phrase Translation Probabilities

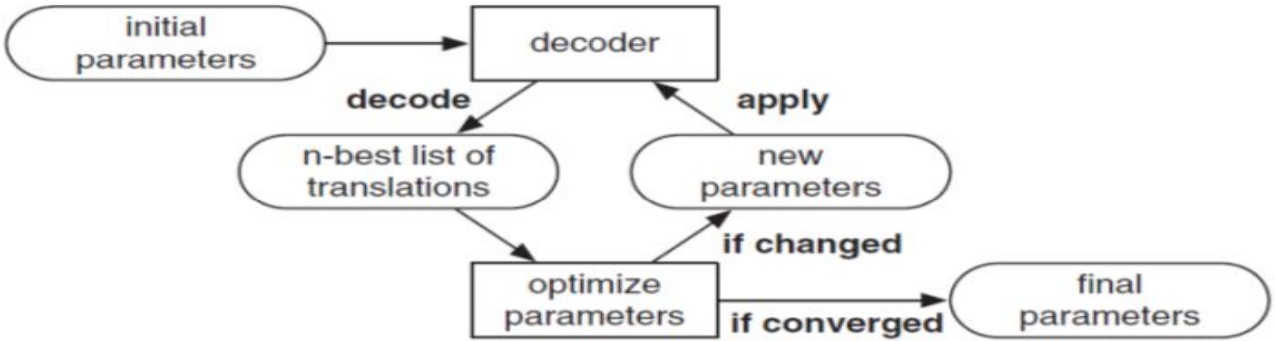
- Estimated from the relative frequency:

$$\phi(\bar{f}|\bar{e}) = \frac{\text{count}(\bar{e},\bar{f})}{\sum_{\bar{f}_i} \text{count}(\bar{e},\bar{f}_i)}$$

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***

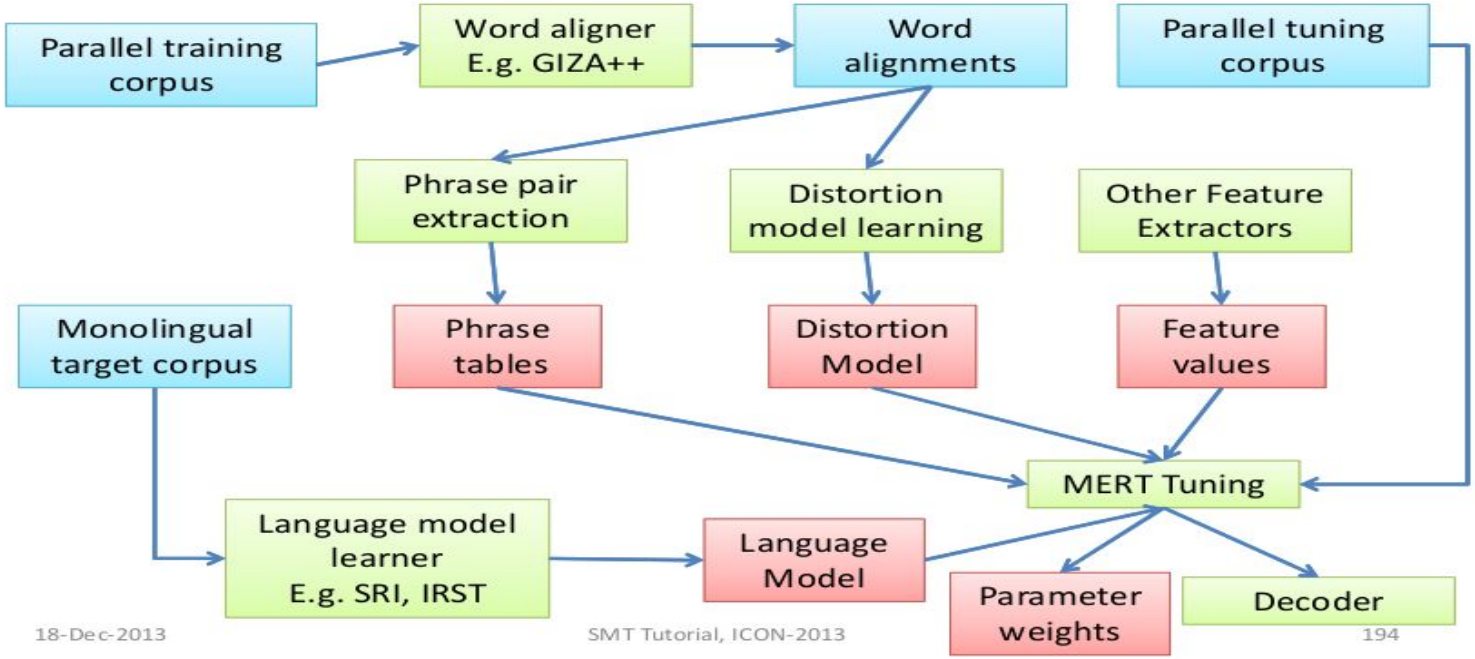
Tuning

- Learning feature weights from data – λ_i
- Minimum Error Rate Training (MERT)
- Search for weights which minimize the translation error on a held-out set (tuning set)
  - Translation error metric : (1 – *BLEU)*



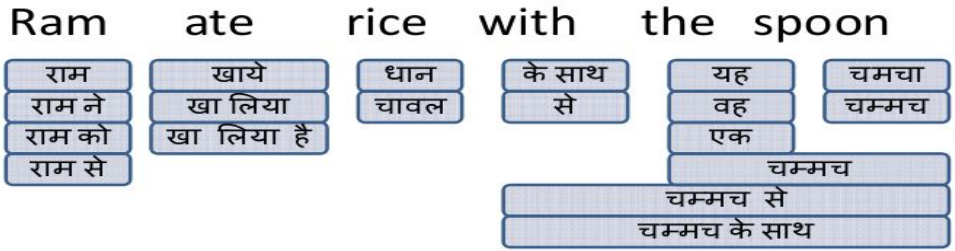
Source: SMT, Phillip Koehn

Overall Training Process for PB-SMT



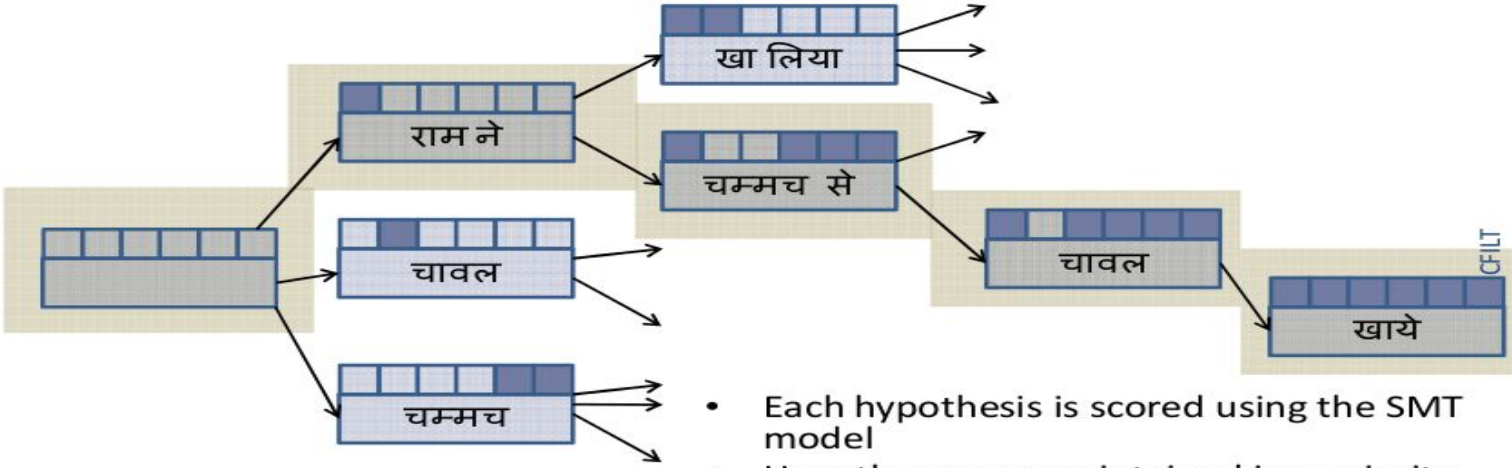
Decoding

We have learnt a translation model, how do we translate a new sentence?



- Isn't it ok to just score all possible translations using the model? $\bar{e} = \arg\max_{e} P(e) \times P(f|e)$
- NP-hard problem: 10-word sentence, 5 translations per word: $10^{5*}10!$ ~ 362 billion possible translations ⇒ Not possible to score each candidate
- Look for approximate solutions
  - Restrict search space: *some word orders are not possible*
  - Incremental construction and scoring
  - Remove candidates that are unlikely to eventually generate good translations

Search Space and Search Organization



- Each hypothesis is scored using the SMT model
- Hypotheses are maintained in a priority queue (called stack decoding historically)
- Limit to the reordering window for efficiency

Outline

- Introduction
- Machine Translation Paradigms
- Phrase-based SMT
- Extensions to Phrase-based SMT
- Evaluation of Machine Translation
- Neural Machine Translation
- Summary

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

- Rich morphology
- Out of Vocabulary words
- Divergent Word Order

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

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!

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 houses
घर ◌ा ◌ं समोर	

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

Now we need to only learn translations for the morphemes

Far more likely to find morphemes in the corpus

Tools for obtaining morphemes: *Morfessor* and *Indic NLP Library*

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?

Note: We want to do a mapping of characters so that they sound the same in Hindi
*⇒ We call this process **'transliteration'**. More on transliteration later ...*

So far we have seen how to learnt how to translate words and phrases

Let’s see how we can generate the correct word order

Getting word order right

Phrase based MT is not good at learning word ordering

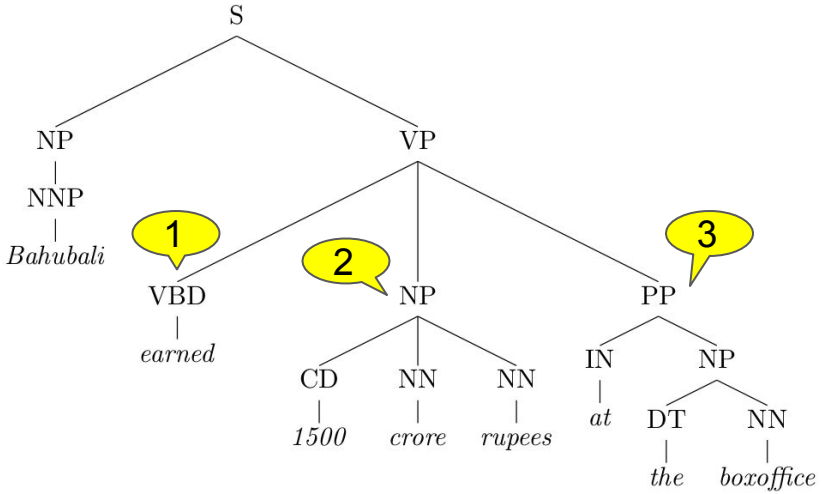
Solution: Let's help PB-SMT with some preprocessing of the input

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 rupee sat the boxoffice

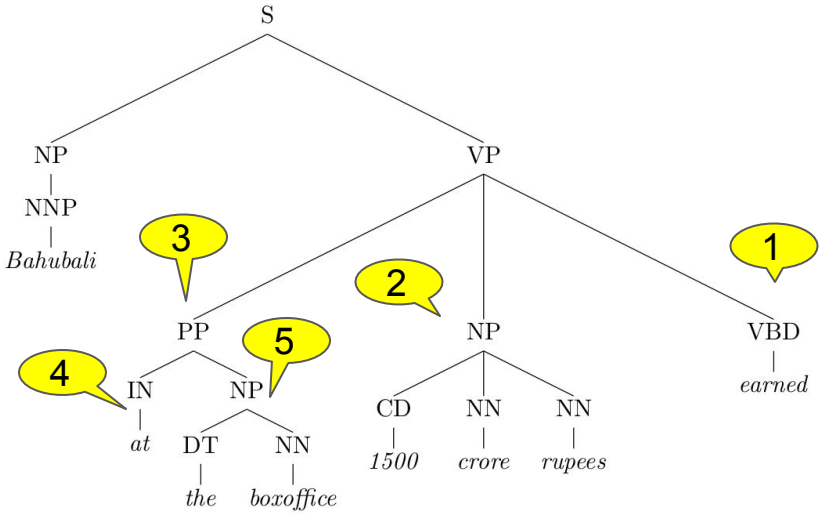
Parse the sentence to understand its syntactic structure



Apply rules to transform the tree

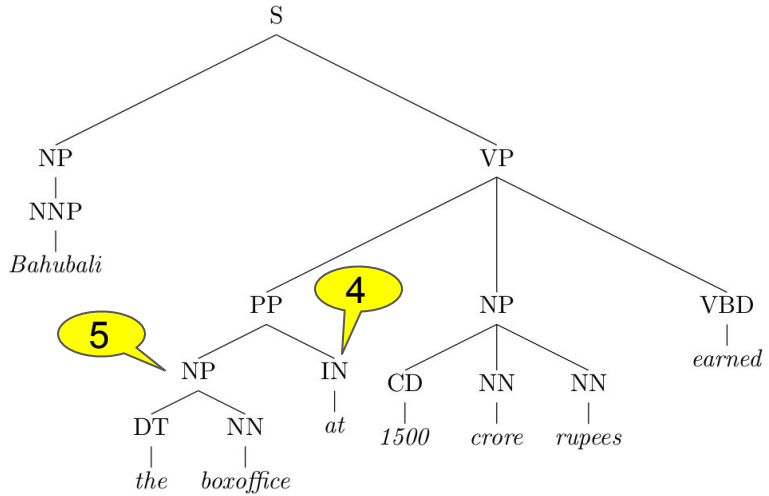
VP → VBD NP PP ⇒ VP → PP NP VBD

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

Better methods exist for generating the correct word order

Incorporate learning of reordering is built into the SMT system

Hierarchical PBSMT ⇒ Provision in the phrase table for limited & simple reordering rules

Syntax-based SMT ⇒ Another SMT paradigm, where the system learns mappings of “treelets” instead of mappings of phrases

Outline

- Introduction
- Machine Translation Paradigms
- Phrase-based SMT
- Extensions to Phrase-based SMT
- Evaluation of Machine Translation
- Neural Machine Translation
- Summary

Evaluation of Machine Translation

How do we judge a good translation?

Can a machine do this?

Why should a machine do this?

Because humans take time!

- Assign scores to specific qualities of output
  - Intelligibility: How good the output is as a well-formed target language entity
  - Accuracy: How good the output is in terms of preserving content of the source text

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 : Accurate but not intelligible

मैं व्याख्यान हूँ

Main vyakhyan hoon

I lecture am

I am lecture: Intelligible but not accurate.

How is translation performance measured?

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

- A corpus of good quality human reference translations

- A numerical “translation closeness” metric

Preliminaries

- **Candidate Translation(s):** Translation returned by an MT system
- **Reference Translation(s):** 'Perfect' translation by humans

Human Evaluation

Adequacy	
5	all meaning
4	most meaning
3	much meaning
2	little meaning
1	none

Fluency	
5	flawless English
4	good English
3	non-native English
2	disfluent English
1	incomprehensible

- Adequacy: Does the output convey the same meaning as the input sentence? Is part of the message lost, added, or distorted?
- Fluency: Is the output fluent? This involves both grammatical correctness and idiomatic word choices.

The most popular metric for MT evaluation: BLEU

Bilingual Language Evaluation Understudy

- Simple metric which computes precision and some notion of recall
- Language Independent

Formulating BLEU (Step 1): Precision

I had lunch now.

Reference 1: मैने अभी खाना खाया
maine abhi khana khaya
I now food ate
I ate food now.

Reference 2 : मैने अभी भोजन किया
maine abhi bhojan kiyaa
I now meal did
I did meal now

Candidate 1: **मैने** अब **खाना खाया**
maine ab khana khaya
I now food ate
I ate food now

matching unigrams: 3,
matching bigrams: 1

Candidate 2: **मैने अभी** लंच एट
maine abhi lunch ate.
I now lunch ate
I ate lunch(OOV) now(OOV)

matching unigrams: 2,
matching bigrams: 1

Unigram precision: Candidate 1: 3/4 = 0.75, Candidate 2: 2/4 = 0.5
Similarly, bigram precision: Candidate 1: 0.33, Candidate 2 = 0.33

Precision: Not good enough

Reference: मुझ पर तेरा सुरूर छाया

mujh-par tera suroor chhaaya

me-on your spell cast

Your spell was cast on me

Candidate 1: मेरे **तेरा सुरूर छाया** matching unigram: 3

mere tera suroor chhaaya

my your spell cast

Your spell cast my

Candidate 2: **तेरा तेरा तेरा सुरूर** matching unigrams: 4

tera tera tera suroor

your your your spell

Unigram precision: Candidate 1: 3/4 = 0.75, Candidate 2: 4/4 = 1

Formulating BLEU (Step 2): Modified Precision

- Clip the total count of each candidate word with its maximum reference count

- $Count_{clip}$(n-gram) = min (count, max_ref_count)

Reference: मुझ पर तेरा सुरूर छाया
mujh-par tera suroor chhaaya
me-on your spell cast
Your spell was cast on me

Candidate 2: **तेरा तेरा तेरा सुरूर**
tera tera tera suroor
your your your spell

- matching unigrams:
  (**तेरा** : min(3, 1) = 1) (सुरूर : min (1, 1) = 1)

Modified unigram precision: 2/4 = 0.5

Modified n-gram precision

For entire test corpus, for a given n,

n-gram: Matching n-grams in C

$$p_n = \frac{\sum_{C \in \{Candidates\}} \sum_{n\text{-}gram \in C} Count_{clip}(n\text{-}gram)}{\sum_{C' \in \{Candidates\}} \sum_{n\text{-}gram' \in C'} Count(n\text{-}gram')}$$

Modified precision for n-grams

Overall candidates of test corpus

n-gram': All n-grams in C

This metric prefers shorter candidates translations, hence a brevity penalty is added

BLEU score

Recall -> Brevity Penalty

$$\mathrm{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \le r \end{cases}$$

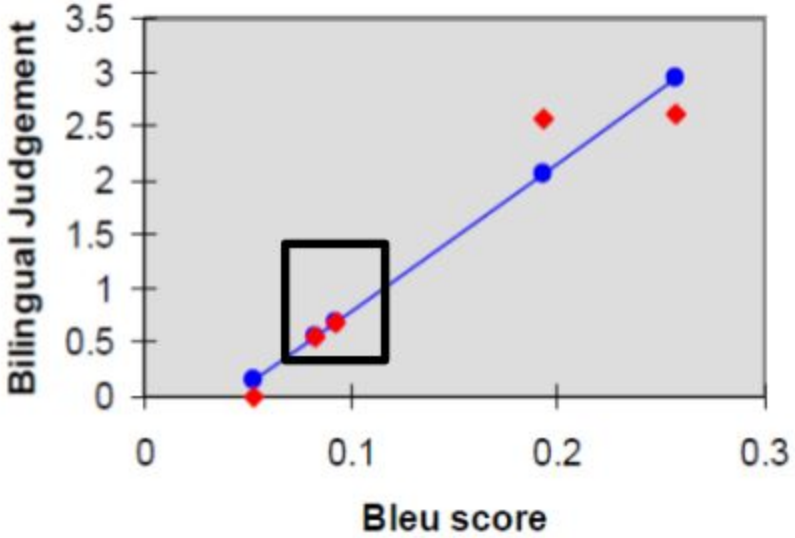
Precision -> Modified n-gram precision

$$p_n = \frac{\sum_{C \in \{Candidates\}} \sum_{n\text{-}gram \in C} Count_{clip}(n\text{-}gram)}{\sum_{C' \in \{Candidates\}} \sum_{n\text{-}gram' \in C'} Count(n\text{-}gram')}$$

$$\mathrm{BLEU} = \mathrm{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right)$$

Why does BLEU score work at all?

Well co-related with human judgments
This is the principle for evaluation of evaluation metrics!



Outline

- Introduction
- Machine Translation Paradigms
- Phrase-based SMT
- Extensions to Phrase-based SMT
- Evaluation of Machine Translation
- Neural Machine Translation
- Summary

Neural Machine Translation

***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 nothing about 'mansion' if it comes up at test time*
- *Creates problems 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

Every word is represented by a vector of numbers

- *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

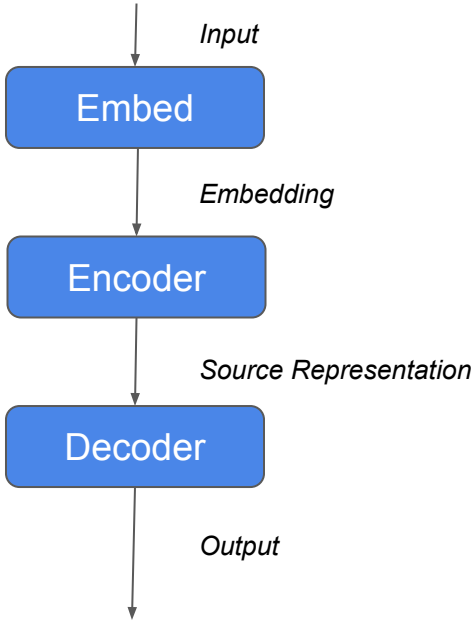
- *Vector similarity measures like cosine similarity*
- *Since* representations of `home` *and* `house`*, we may be able to translate* `house`

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*

Encode - Decode Paradigm

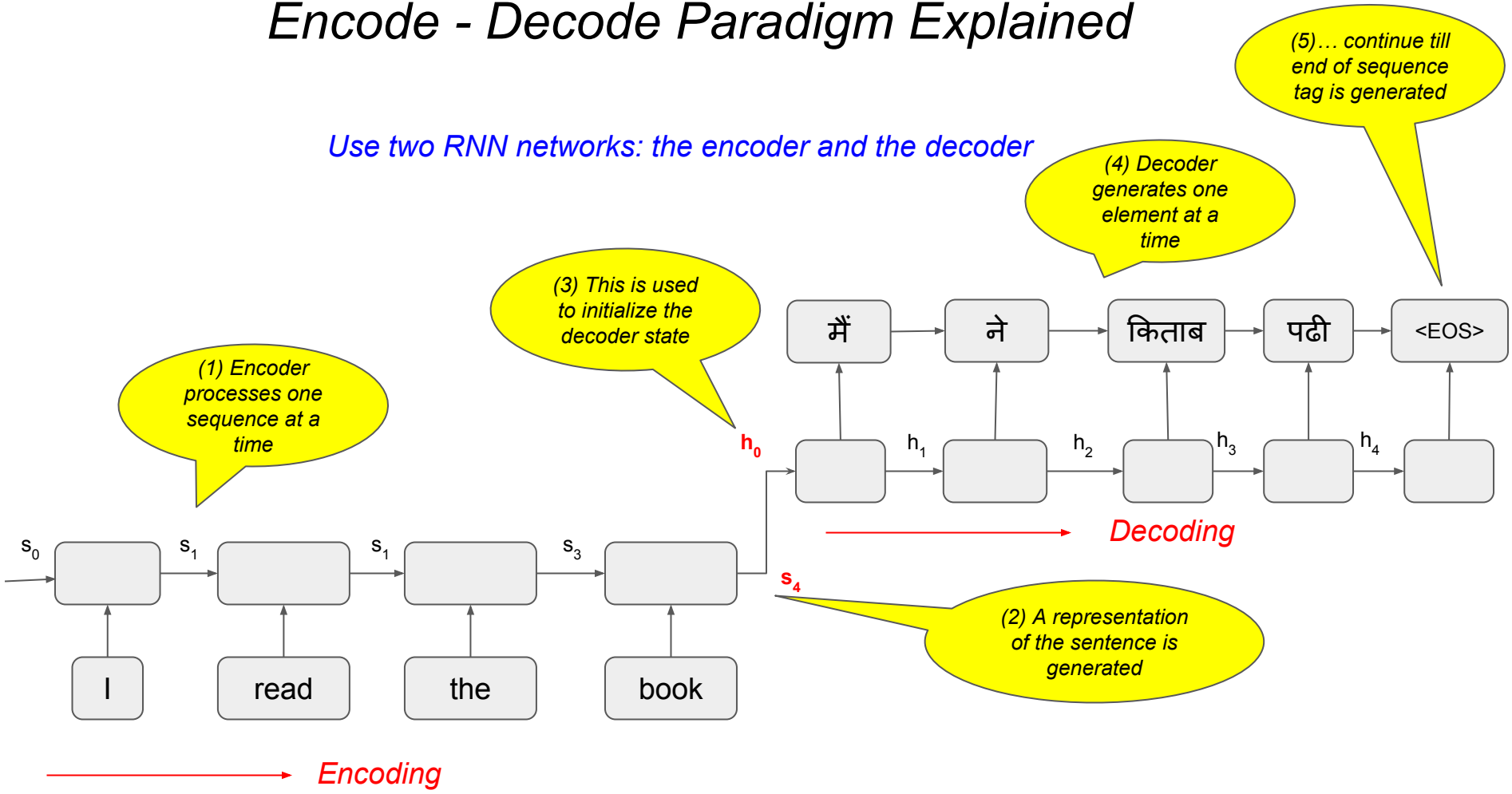


Entire input sequence is processed before generation starts
⇒ In PBSMT, generation was piecewise

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: Recurrent neural network unit which can keep state information*

Encode - Decode Paradigm Explained



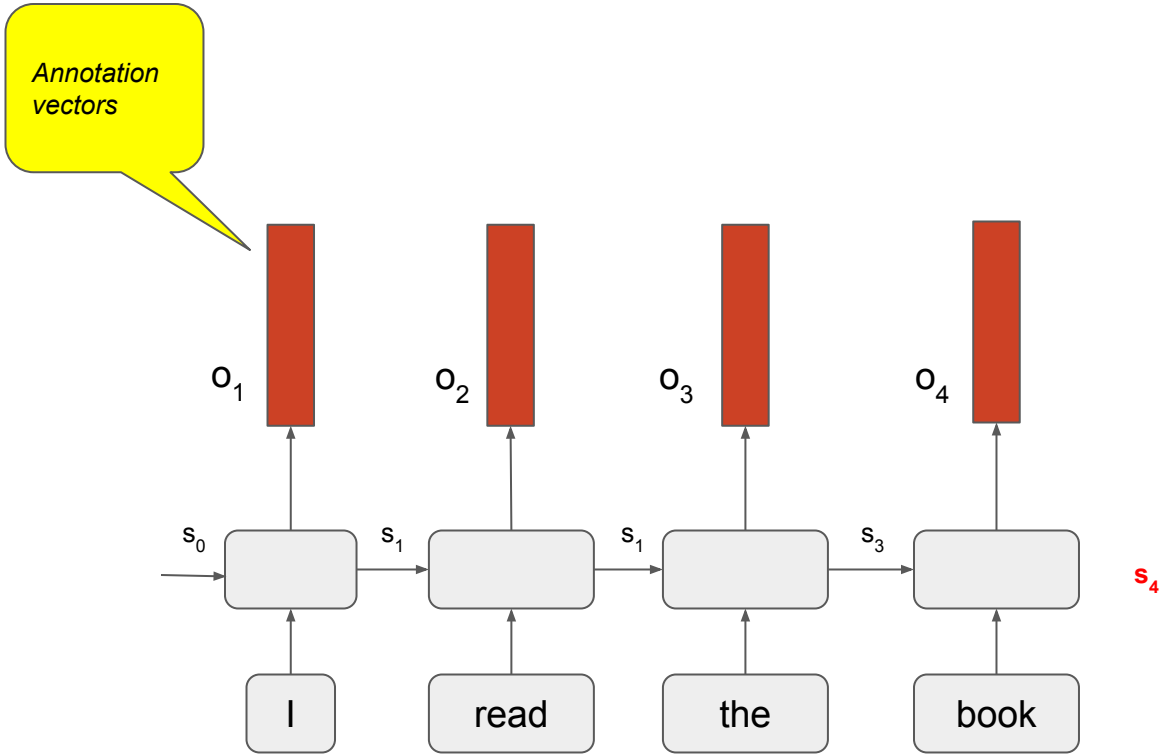
This approach reduces the entire sentence representation to a single vector

Two problems with this design choice:

- 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***

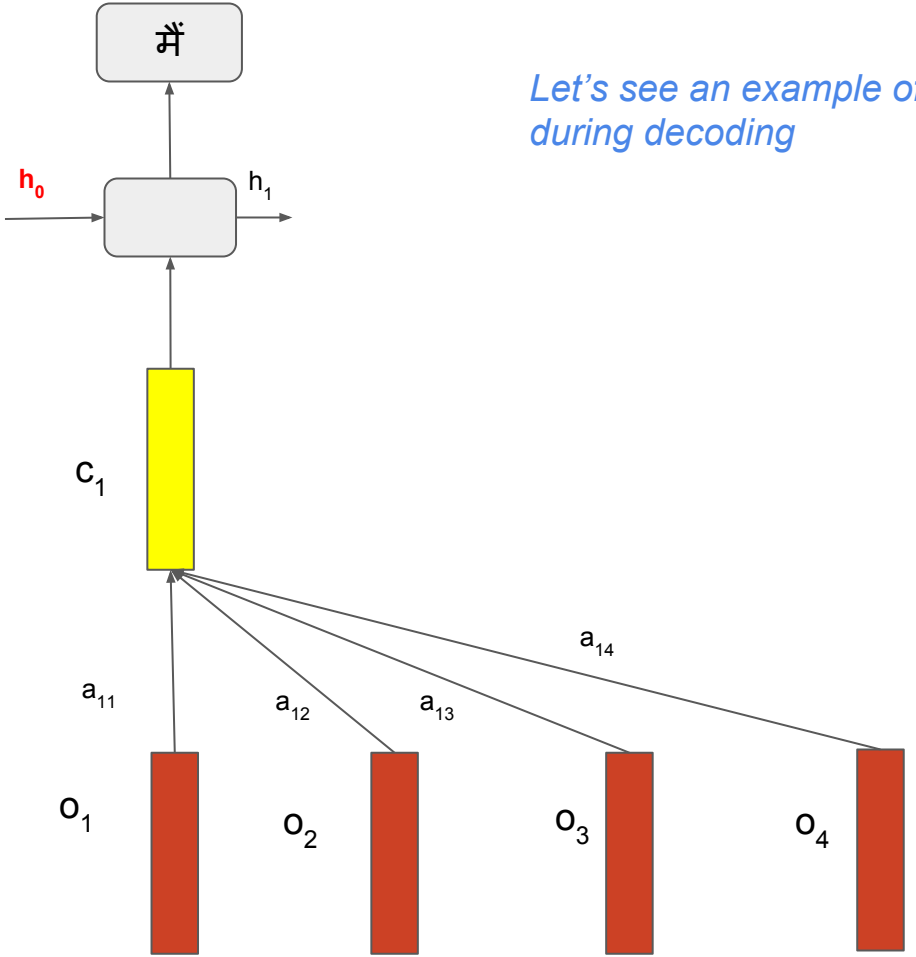
How should the decoder use the set of annotation vectors while predicting the next character?

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 3rd target word, the 3rd annotation vector (hence 3rd 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**

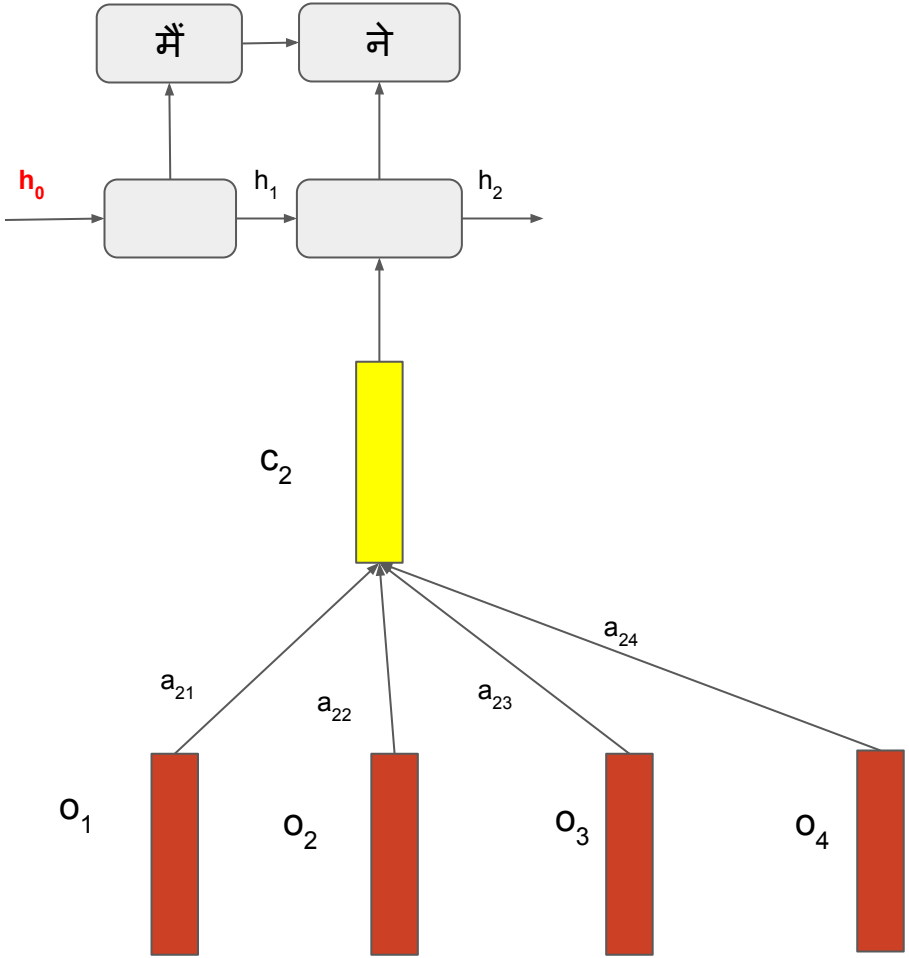
This averaged ***context vector*** is an input to the decoder

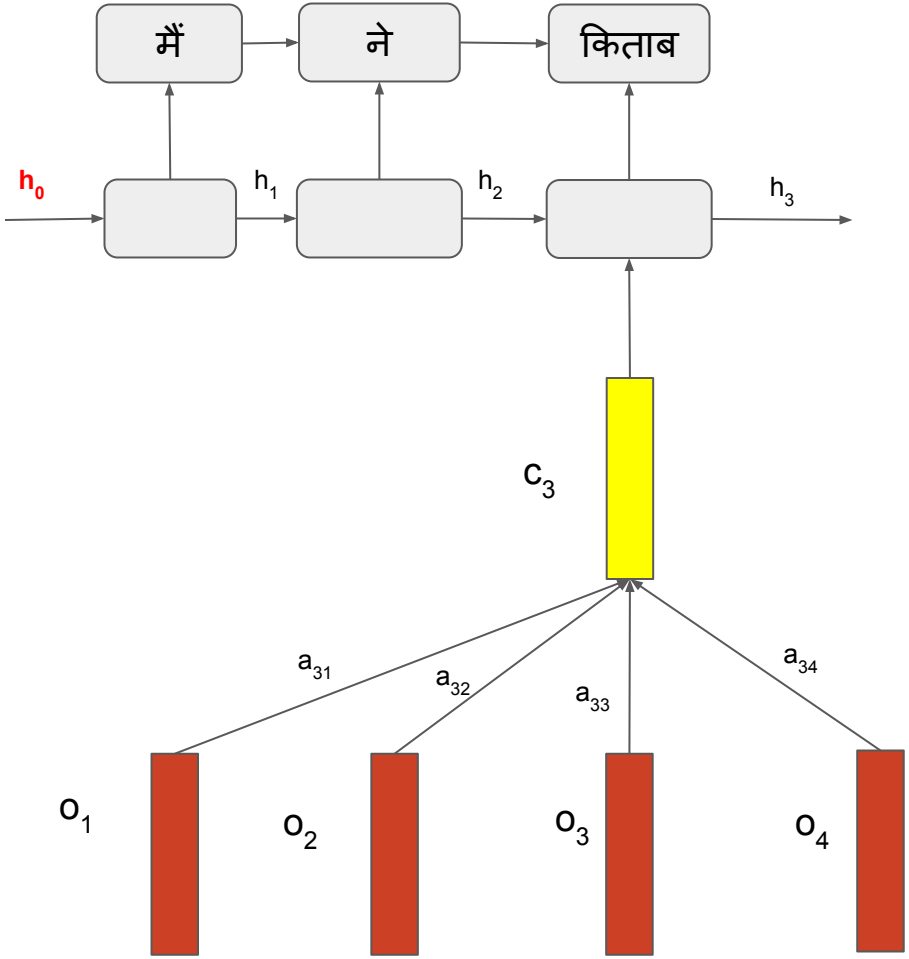


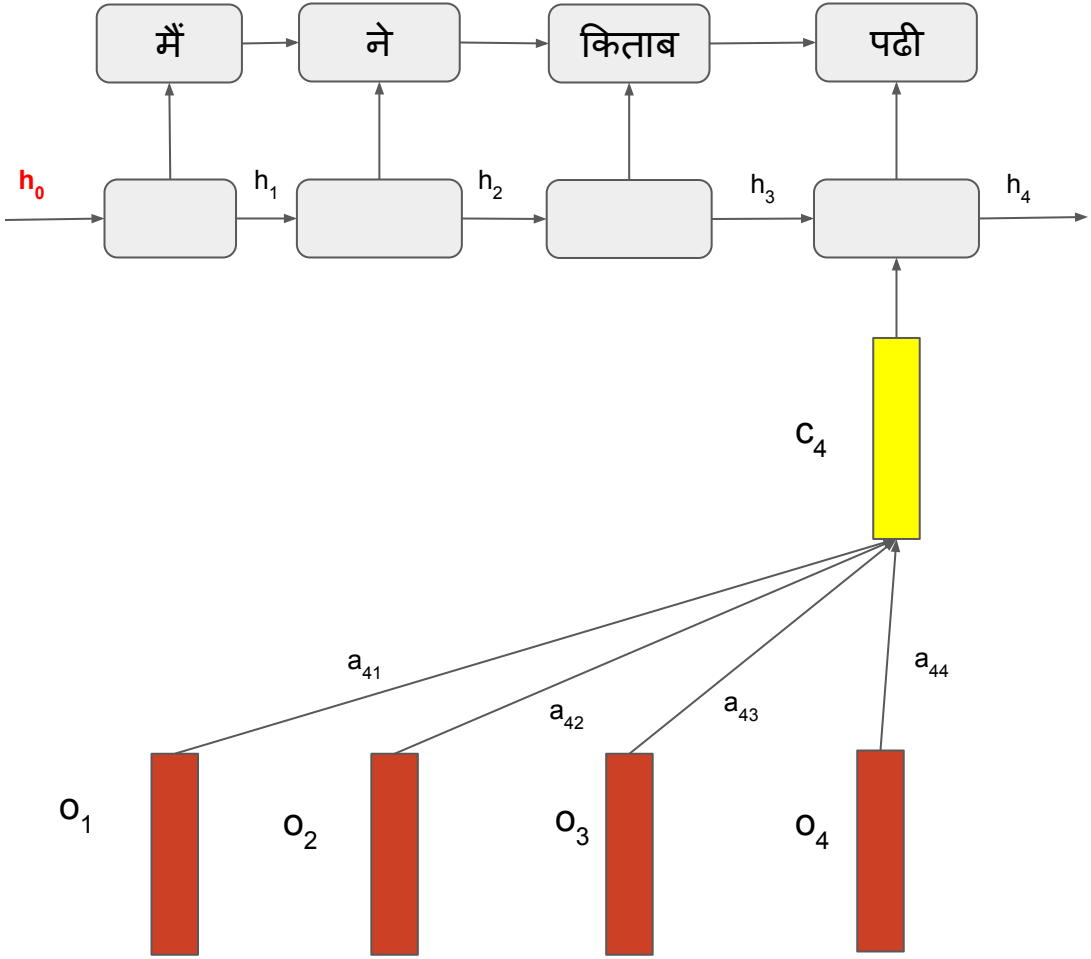
*Let's see an example of how the **attention mechanism** works during decoding*

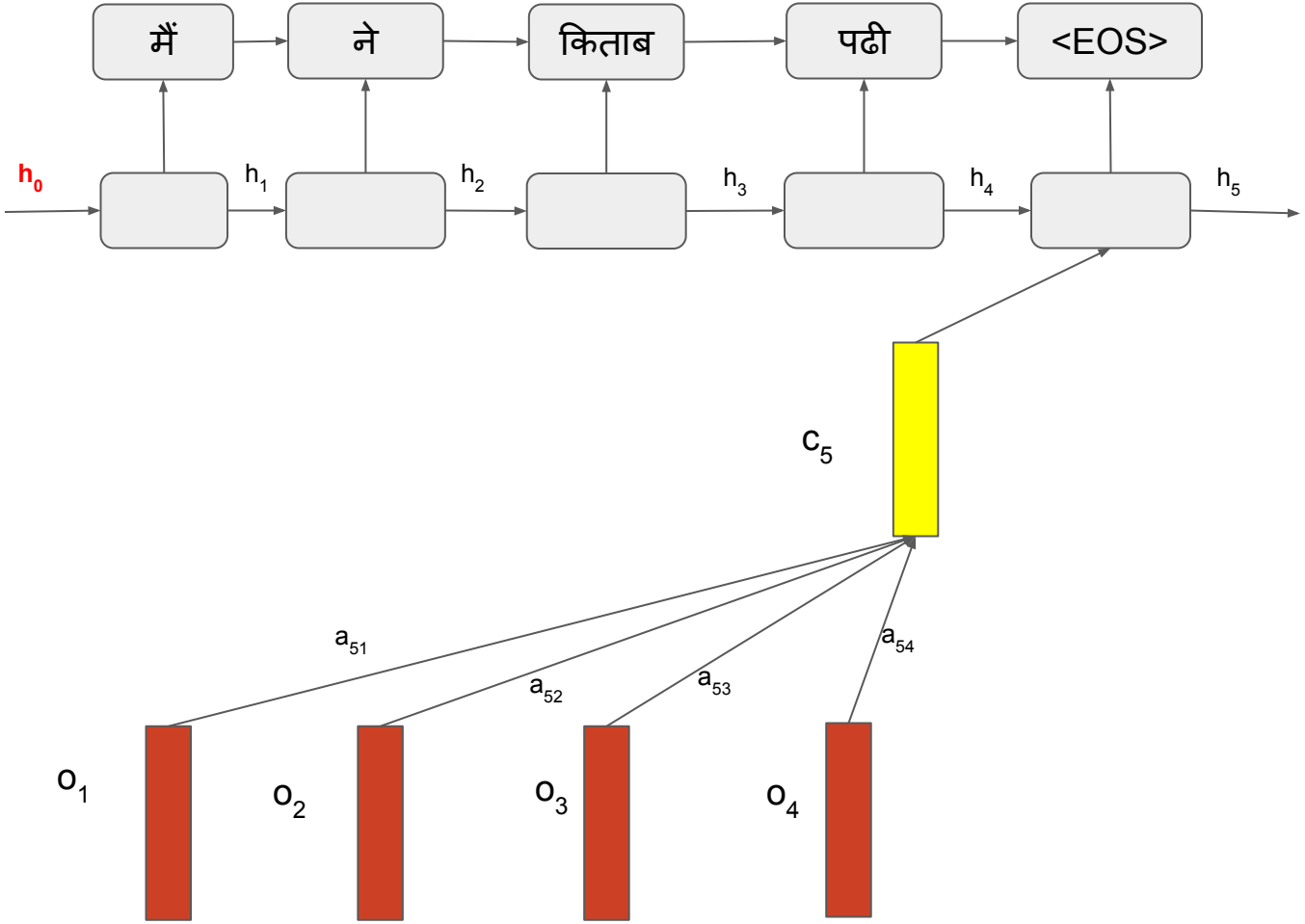
$$c_i = \sum_{j=1}^{n} a_{ij} o_j$$

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


मैं
ने
h_0
h_1
h_2
c_2
a_{21}
a_{22}
a_{23}
a_{24}
o_1
o_2
o_3
o_4


मैं
ने
किताब
h_0
h_1
h_2
h_3
c_3
a_{31}
a_{32}
a_{33}
a_{34}
o_1
o_2
o_3
o_4


मैं
ने
किताब
पढी
h0
h1
h2
h3
h4
c4
a41
a42
a43
a44
o1
o2
o3
o4


मैं
ने
किताब
पढी
<EOS>
h0
h1
h2
h3
h4
h5
c5
a51
a52
a53
a54
o1
o2
o3
o4

But we do not know the attention weights?
How do we find them?

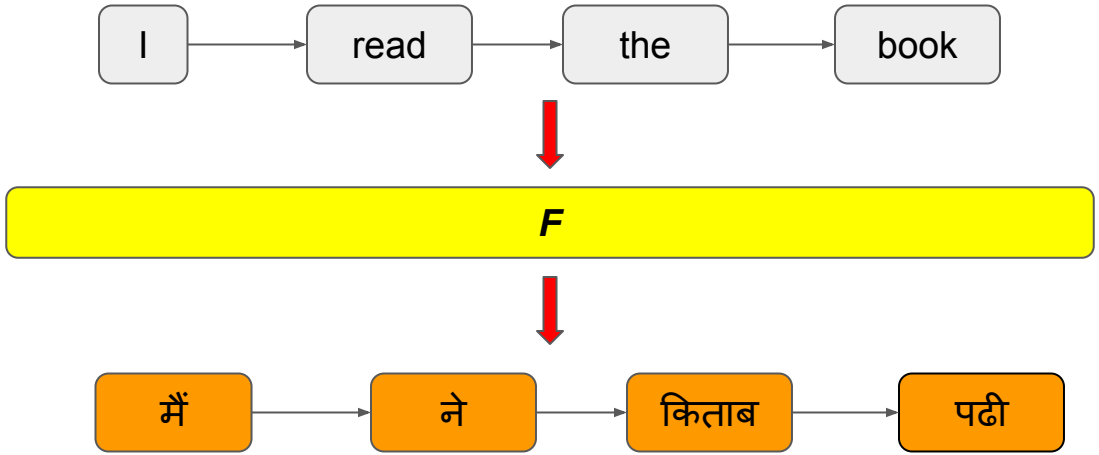
Let the training data help you decide!!

Idea: Pick the attention weights that maximize the translation accuracy
(more precisely, decrease training data loss)

- *Note ⇒ 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*
- *Exciting times ahead!*

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*
- *Image labelling: Image ⇒ Label*
- *Transliteration: character sequence ⇒ character sequence*

Summary

- Introduction
- Machine Translation Paradigms
- Phrase-based SMT
- Extensions to Phrase-based SMT
- Evaluation of Machine Translation
- Neural Machine Translation

Getting Started with Machine Translation

Software

- Statistical Machine Translation: Moses
- Neural Machine Translation: Nematus

Corpora

- Technology Development in Indian Language (TDIL) website
- Europarl Corpus
- IIT Bombay English-Hindi Parallel Corpus

Resources for Reading

Books & Articles

- Statistical MT Tutorial Workbook, Kevin Knight (online)
- Statistical Machine Translation, Phillip Koehn (book)
- Machine Translation. Pushpak Bhattacharyya (book)
- Neural Machine Translation. Kyunghyun Cho (online)
    - https://devblogs.nvidia.com/parallelforall/introduction-neural-machine-translation-with-gpus/

Presentations

- *Machine Learning for Machine Translation (An Introduction to Statistical Machine Translation)*. **Tutorial at ICON 2013**
    - https://www.cse.iitb.ac.in/~anoopk/publications/presentations/icon_2013_smt_tutorial_slides.pdf

Transliteration

You are in Kerala … waiting to travel by bus

Not a hypothetical situation …. Read this:
http://www.thehindu.com/todays-paper/tp-national/tp-kerala/call-to-bring-on-board-bus-signage-in-three-languages/article5224039.ece

How do you translate Xi Xinping?

Xi Jinping is the President of China
शी चिनफिंग चीन के राष्ट्रपति है

.

Ok, we got lucky here … but there are so many names you will not find in any corpus

Transliteration can simplify Translation

Hindi

यदि श्वास प्रणालिका में सूजन आ जाये तब भी रक्त मुँह के रास्ते बाहर आने लगता है ।

Punjabi Translation

ਜੇਕਰ ਸਾਹ ਪ੍ਰਣਾਲੀ ਵਿਚ ਸੋਜ ਆ ਜਾਵੇ ਤਦ ਵੀ ਖੂਨ ਮੂੰਹ ਦੇ ਰਾਸਤੇ ਬਾਹਰ ਆਉਣ ਲਗਦਾ ਹੈ ।

जेकर साह प्रणाली विच सोज आ जावे तद वी खून मू□ह दे रासते बाहर आउण लगदा है ।

Hindi-Punjabi Transliteration

ਆਦਿ ਸਾਹ ਪ੍ਰਣਾਲੀ ਮੈਂ ਸੂਜਨ ਆ ਜਾਵੇ ਤਦ ਵੀ ਰਕਤ ਮੂੰਹ ਦੇ ਰਾਸਤੇ ਬਾਹਰ ਆਉਣ ਲਗਦਾ ਹੈ ।

आदि साह प्रणाली मैं सूजन आ जावे तद वी रकत मू□ह दे रासते बाहर आउण लगदा है ।

Some Concepts

- Natural Language: A system of communication among humans with sound
- Script: A system of symbols for representing language in writing
  - *A language can have multiple scripts:*
  - *Sanskrit is written in many scripts (Devanagari, Malayalam, Tamil, Telugu, Roman, etc.)*
  - *A script can be used for multiple languages*
  - *Devanagari is used to write Sanskrit, Hindi, Marathi, Konkani, Nepali*
- Phoneme: basic unit of sound in a language that is meaningful
- Grapheme: basic distinct unit of a script
  - *A phoneme can be represented by multiple graphemes*
  - *cut, dirt*
  - *A grapheme can be used to represent multiple sounds*
  - *cut, put*

What is transliteration?

- *Transliteration is the conversion of a given name in the source language (from source script) to a name in the target language (target script), such that the target language name is:*
- phonemically equivalent to the source name
  - *मुम्बई → Mumbai*
- conforms to the phonology of the target language
  - नरेन्द्र → ನರೆಂದರ (ನರೇಂದರ)
- matches the user intuition of the equivalent of the source language name in the target language, considering the culture and orthographic character usage in the target language
  - ആലപ്പുഴ (aalappuzha)→ Alappuzha

Isn't it easy to just map characters from one script to another?

- Local spelling conventions
  - लता in Roman: Latha (South India) vs Lata (North India)
  - Laxmi → लक्षमी
- Missing sounds
  - കോഴിക്കോട് (kozhikkoT)→ कोषिक्कोड (koShikkod)
- Transliterate or translate
  - കോഴിക്കോട് (kozhikkoT) → Calicut
- Transliteration variants
  - मुंबई, मुम्बई

Why English spellings caused trouble in school ...

Ambiguity in character to sound mapping

- ionize vs nation

- *fish* can be pronounced as *ghoti*
  - *gh* as in *tough*
  - *o as in women*
  - *ti as in nation*

… and Hindi spellings didn't

Unambiguous mapping from character to sound

*Remember the **varnamala**? – organized according to scientific principles*

	***sparśa* (Plosive)**								***anunāsika* (Nasal)**		***antastha* (Approximant)**		***ūṣma/saṃghaṣhrī* (Fricative)**			
Voicing →	*aghoṣa*				*ghoṣa*								*aghoṣa*		*ghoṣa*	
Aspiration →	*alpaprāṇa*		*mahāprāṇa*		*alpaprāṇa*		*mahāprāṇa*		*alpaprāṇa*				*mahāprāṇa*			
***kaṇṭhya* (Guttural)**	क	ka /k/	ख	kha /kʰ/	ग	ga /g/	घ	gha /gʰ/	ङ	ṅa /ŋ/					ह	ha /ɦ/
***tālavya* (Palatal)**	च	ca /c, t͡ʃ/	छ	cha /cʰ, t͡ʃʰ/	ज	ja /ɟ, d͡ʒ/	झ	jha /ɟʰ, d͡ʒʰ/	ञ	ña /ɲ/	य	ya /j/	श	śa /ɕ, ʃ/		
***mūrdhanya* (Retroflex)**	ट	ṭa /ʈ/	ठ	ṭha /ʈʰ/	ड	ḍa /ɖ/	ढ	ḍha /ɖʰ/	ण	ṇa /ɳ/	र	ra /r/	ष	ṣa /ʂ/		
***dantya* (Dental)**	त	ta /t̪/	थ	tha /t̪ʰ/	द	da /d̪/	ध	dha /d̪ʰ/	न	na /n/	ल	la /l/	स	sa /s/		
***oṣṭhya* (Labial)**	प	pa /p/	फ	pha /pʰ/	ब	ba /b/	भ	bha /bʰ/	म	ma /m/	व	va /w, ʋ/				

The extent of Devanagari-like scripts

How do we solve the transliteration problem?

- Transliteration is very similar to translation
- Instead of words, we have characters
- However, it is much simpler
  - No reordering
  - Small vocabulary (except Chinese and Japanese Kanji)
  - Regular grammar
- Similar to Vouquois triangle, you can transliterate at different levels:
  - Phoneme (like transfer based MT)
  - Grapheme (like direct MT)

References

- About Scripts
  - Omniglot: http://www.omniglot.com/
  - Wikipedia pages on Devanagari & Brahmi script
    -
- About Transliteration
  - Karimi, Sarvnaz, Falk Scholer, and Andrew Turpin. "Machine transliteration survey." ACM Computing Surveys. 2011.
- Hands on
  - Google Transliterate
    - http://www.google.com/inputtools/
  - Brahmi-Net: IITB's transliteration system
    - http://www.cfilt.iitb.ac.in/brahminet/

Thank You!

https://www.cse.iitb.ac.in/~anoopk