Natural Language Processing

A Distributional Approach

Anoop Kunchukuttan
Microsoft AI & Research

ankunchu@microsoft.com

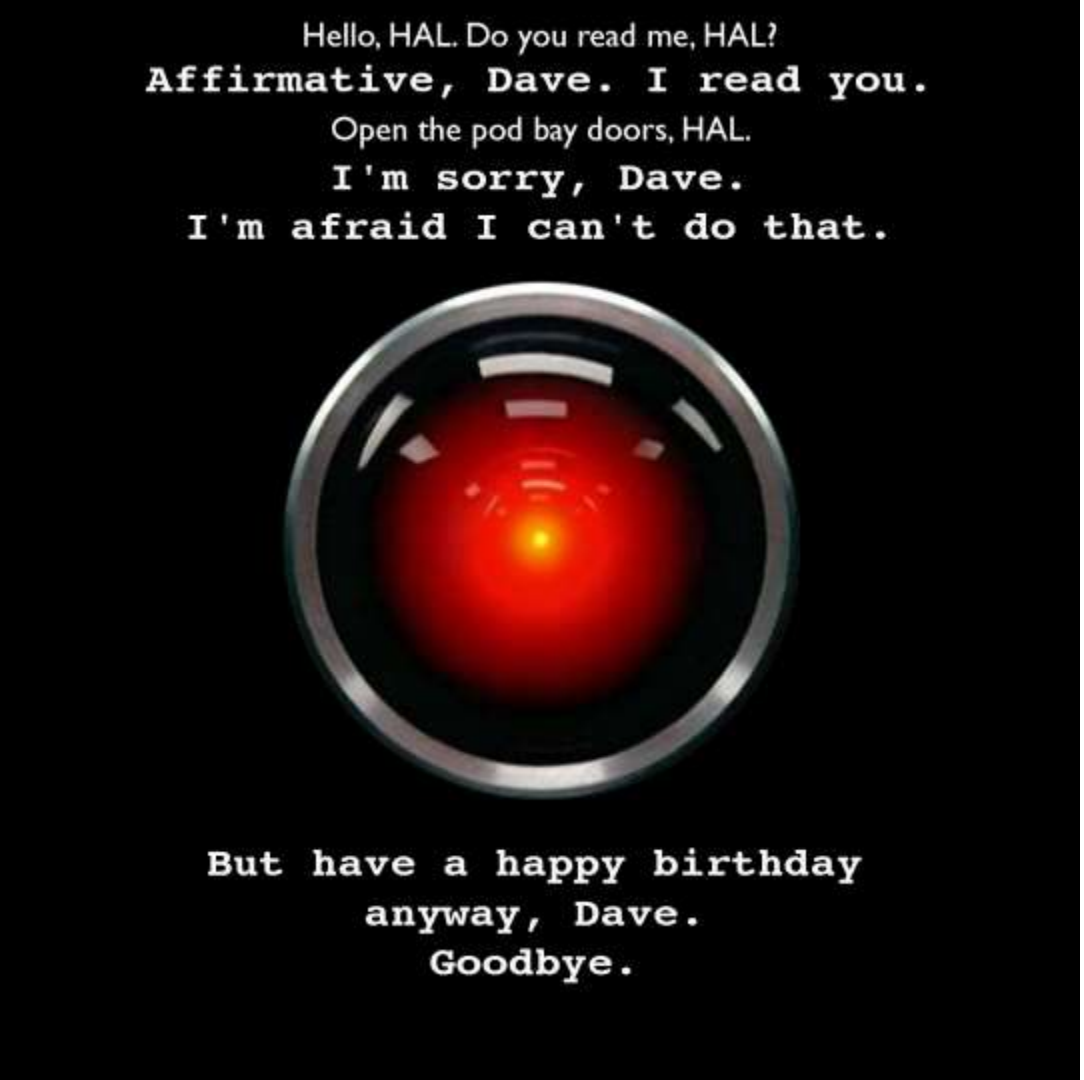
AI Deep Dive Workshop at IIT Alumni Center Bengaluru, 27th July 2019

Outline

- *What is Natural Language Processing?*
- *A Linguistics Primer*
- *Symbolic vs. Connectionist Approaches*
- *Distributional Semantics*
- *Word Embeddings*
- *Sentence Embeddings*
- *Building simple NLP applications*
- *Summary*

Outline

- ***What is Natural Language Processing?***
- *A Linguistics Primer*
- *Symbolic vs. Connectionist Approaches*
- *Distributional Semantics*
- *Word Embeddings*
- *Sentence Embeddings*
- *Building simple NLP applications*
- *Summary*



Natural Language Processing deals with the interaction between computers and humans using natural language.

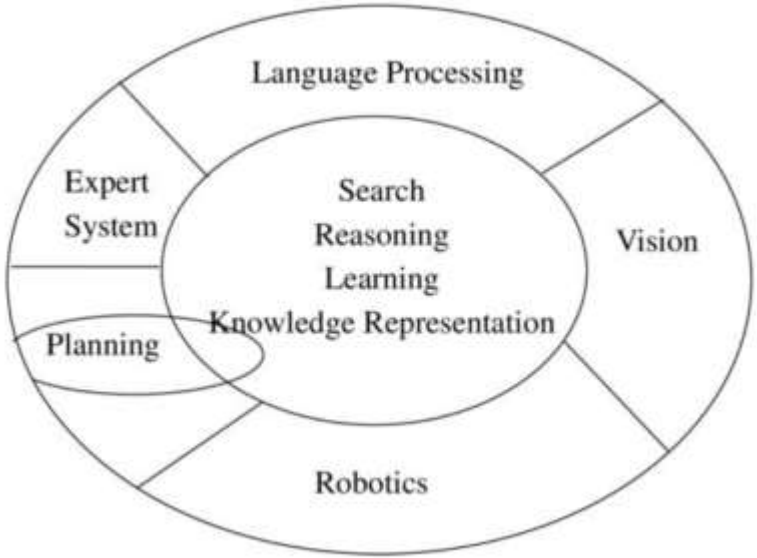
An intelligent agent like HAL can do:

- *Natural Language Understanding*
- *Natural Language Generation*

Many other useful applications

- *Text Classification*
- *Spelling Correction*
- *Grammar Checking*
- *Essay Scoring*
- *Machine Translation*

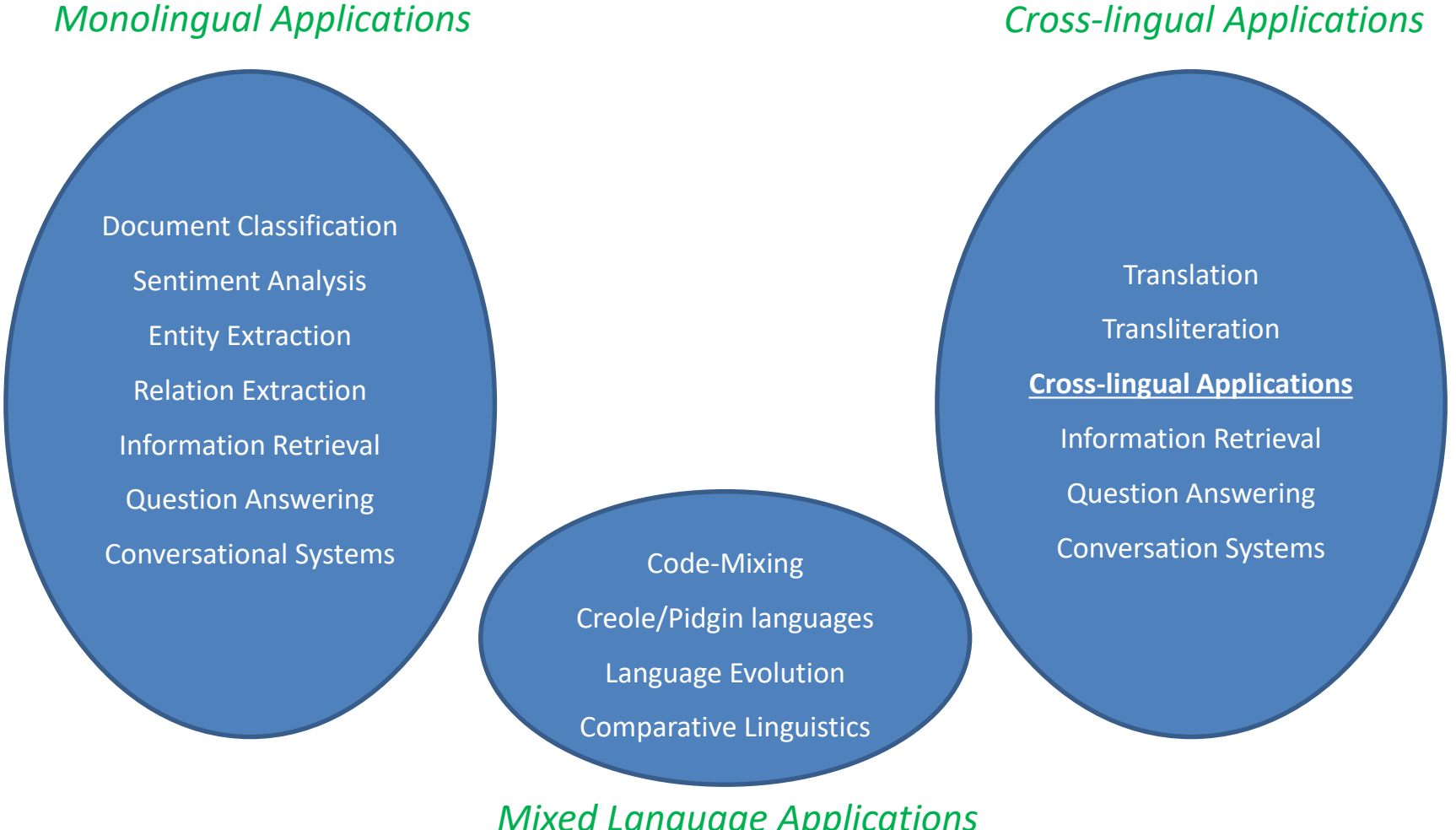
NLP and Artificial Intelligence



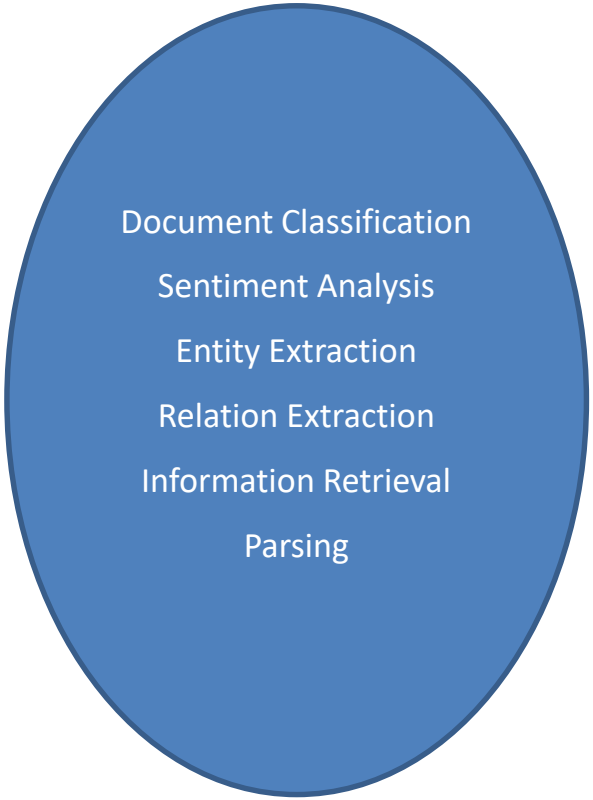
- Branch of AI
- Interface with humans
- Deal with a complex artifact like language
- Diagram
- Deep and Shallow NLP
- Super-applications of NLP

Difference from other AI tasks

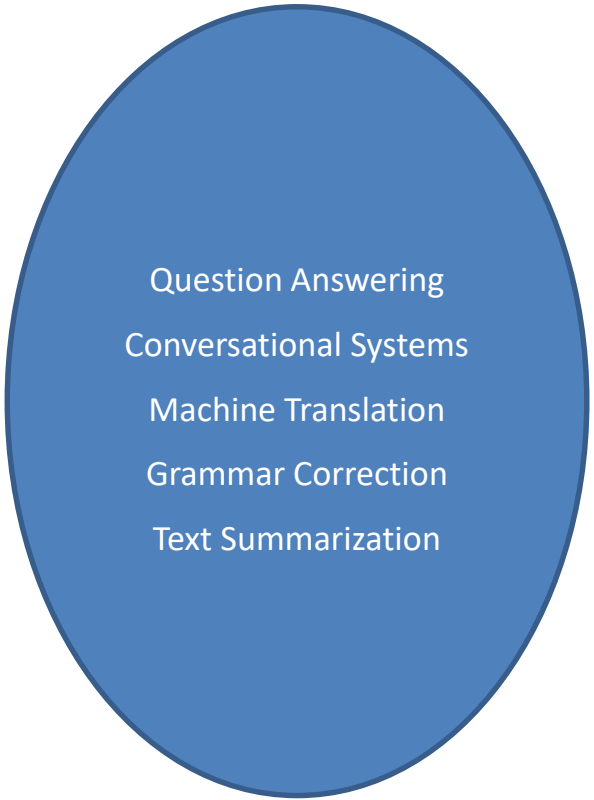
- Higher-order cognitive skills
- Inherently discrete
- Diversity of languages


Monolingual Applications
Document Classification
Sentiment Analysis
Entity Extraction
Relation Extraction
Information Retrieval
Question Answering
Conversational Systems
Cross-lingual Applications
Translation
Transliteration
Cross-lingual Applications
Information Retrieval
Question Answering
Conversation Systems
Code-Mixing
Creole/Pidgin languages
Language Evolution
Comparative Linguistics
Mixed Language Applications

Analysis

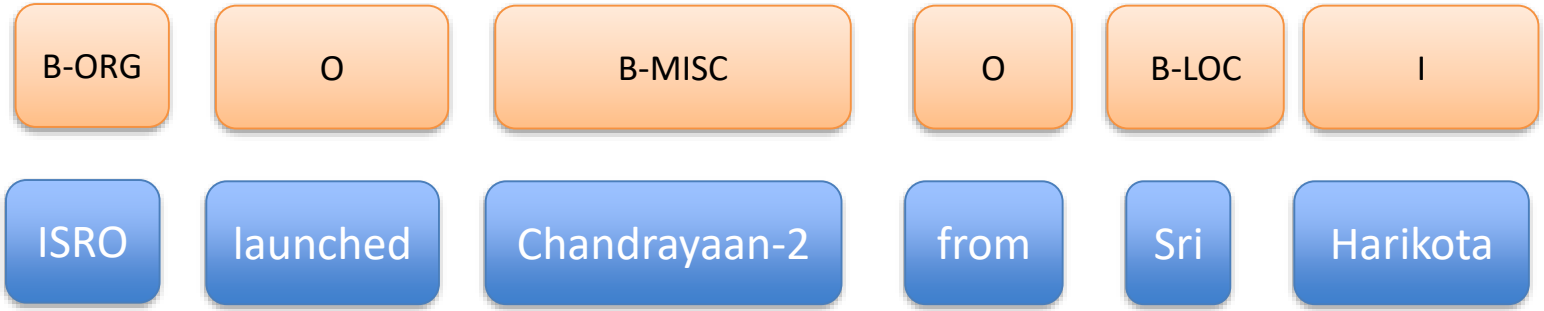


Synthesis



Classification Tasks

Sequence Labelling Tasks



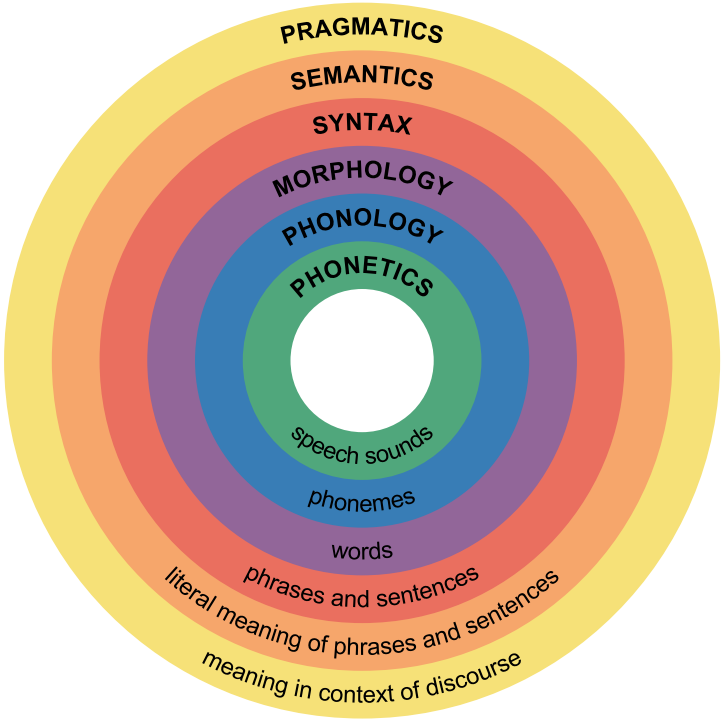
Sequence to Sequence Tasks

Outline

- *What is Natural Language Processing?*
- ***A Linguistics Primer***
- *Symbolic vs. Connectionist Approaches*
- *Distributional Semantics*
- *Word Embeddings*
- *Sentence Embeddings*
- *Building simple NLP applications*
- *Summary*

A LINGUISTICS PRIMER

Natural language is the object to study of NLP
Linguistics is the study of natural language



Source: Wikipedia

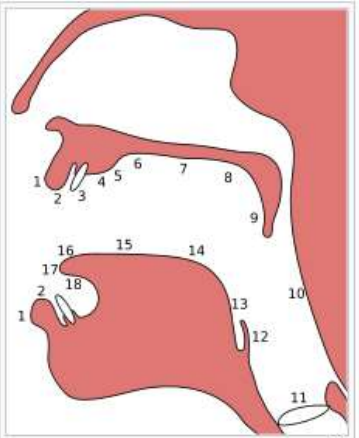
Just as you need to know the laws of physics to build mechanical devices, you need to know the nature of language to build tools to understand/generate language

<u>Some interesting reading material</u>

1) *Linguistics: Adrian Akmajian et al.*
2) *The Language Instinct: Steven Pinker – for a general audience – highly recommended*
3) *Other popular linguistic books by Steven Pinker*

Phonetics & Phonology

Vocal Tract



Passive and active places of articulation: (1) *Exo-labial*; (2) *Endo-labial*; (3) *Dental*; (4) *Alveolar*; (5) *Post-alveolar*; (6) *Pre-palatal*; (7) *Palatal*; (8) *Velar*; (9) *Uvular*; (10) *Pharyngeal*; (11) *Glottal*; (12) *Epiglottal*; (13) *Radical*; (14) *Postero-dorsal*; (15) *Antero-dorsal*; (16) *Laminal*; (17) *Apical*; (18) *Sub-apical* or *sub-laminal*.

	BILABIAL		LABIODENTAL		DENTAL		ALVEOLAR		POST ALVEOLAR		RETROFLEX		PALATAL		VELAR		UVULAR	
PLOSIVE	p	b					t	d			ʈ	ɖ	c	ɟ	k	g	q	ɢ
NASAL		m		ɱ				n				ɳ		ɲ		ŋ		ɴ
TRILL		ʙ						r										ʀ
TAP OR FLAP								ɾ				ɽ						
FRICATIVE	ɸ	β	f	v	θ	ð	s	z	ʃ	ʒ	ʂ	ʐ	ç	ʝ	x	ɣ	χ	ʁ
LATERAL FRICATIVE							ɬ	ɮ										
APPROXIMANT				ʋ				ɹ				ɻ		j		ɰ		
LATERAL APPROXIMANT								l				ɭ		ʎ		ʟ		

International Phonetic Alphabet (IPA) chart

- *Phonemes are the basic distinguishable sounds of a language*
- *Every language has a sound inventory*

Morphology

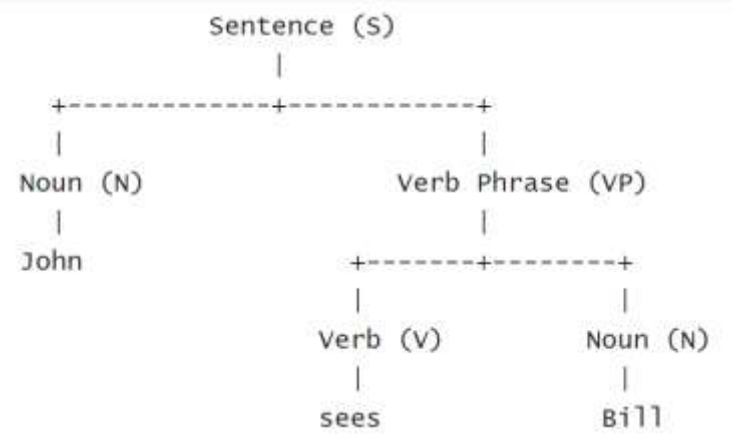
Inflectional Morphology

घरासमोरचा ➔ घर समोर चा

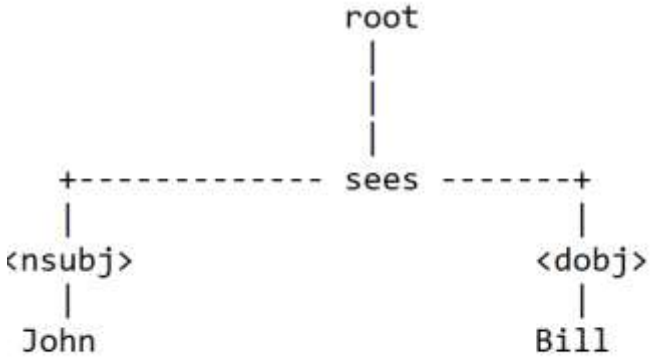
Derivational Morphology

नीलांबर ➔ नील अंबर

Syntax



Constituency Parse



Dependency Parse

Language Diversity

Phonology/Phonetics:

- *Retroflex sounds most found in Indian languages*
- *Tonal languages (Chinese, Thai)*

Morphology:

Chinese → isolating language

Malayalam → agglutinative language

Syntax:

SOV language (Hindi): मैं बाज़ार जा रहा हूँ

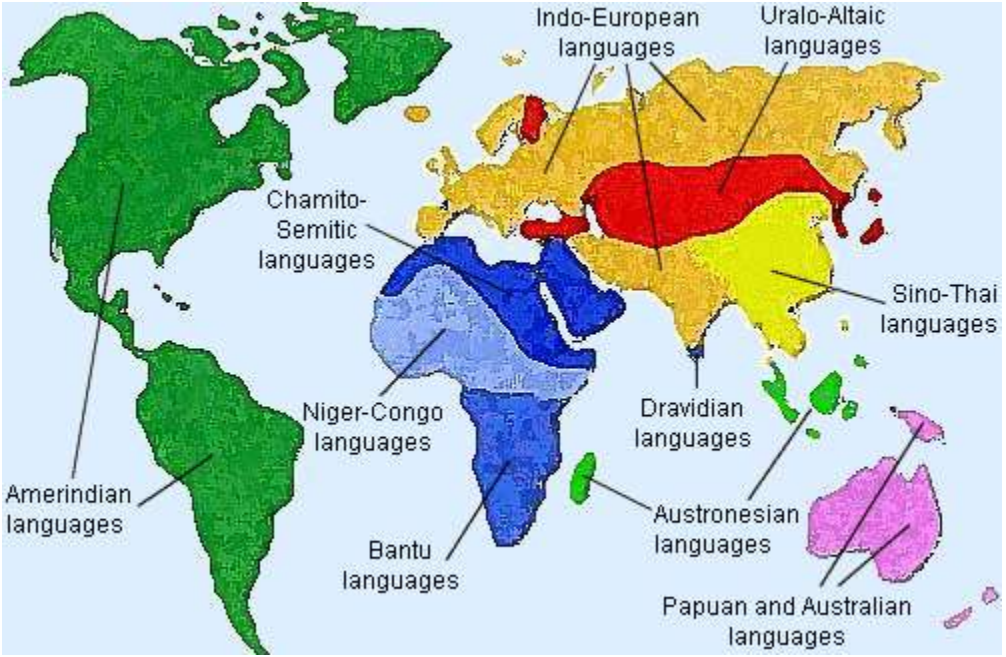
SVO language (English): I am going to the market

Subject (S) Verb (V) Object (O)

Free-order vs. Fixed-order languages

https://www.ethnologue.com/statistics/family

Language Families



Source: https://www.freelang.net/families/

Writing Systems

https://www.omniglot.com/
https://home.unicode.org/

Logographic: characters stand for concepts e.g. Chinese

Abjad: characters stand for consonants; vowels not represented. e.g. Arabic, Hebrew

Abugida: both vowels and consonants represented; vowels indicated by diacritics e.g. most Indic scripts like Devanagari

Alphabet: both vowels and consonants have independent symbols e.g. Latin, Cyrillic

The above three systems approximate phonemes as basic units

Syllabic: each character stands for a syllable e.g. Korean Hangul, Japanese Katakana

Outline

- *What is Natural Language Processing?*
- *A Linguistics Primer*
- ***Symbolic vs. Connectionist Approaches***
- *Distributional Semantics*
- *Word Embeddings*
- *Sentence Embeddings*
- *Building simple NLP applications*
- *Summary*

Let us look at a simple NLP application – Sentiment Analysis

Kabir Singh Movie Review

Review by **Bollywood Hungama News Network**
20 June 2019 22:39 pm IST

4.0
Kabir Singh Movie Rating

Listen to this article in audio 00:00/00:00

One of the most loved love stories of Bollywood is DEVDAS. It has been remade several times and ten years ago, Anurag Kashyap gave a different touch to the tale through DEV D [2009]. All the interpretations have been liked as there's a charm in the story of a man who goes on a self-destructive path when he fails to get the girl he loves. Two years ago, Sandeep Reddy Vanga made a Telugu film named ARJUN REDDY, which had a kind of a deja vu of DEVDAS. Yet, it stood out due to the treatment, execution and performances. ARJUN REDDY became a cult success and now its Hindi remake KABIR SINGH is all set to hit theatres. So does KABIR SINGH turn out to be as good as or better than ARJUN REDDY? Or does it fail to stir the emotions of the viewers? Let's analyse.

?

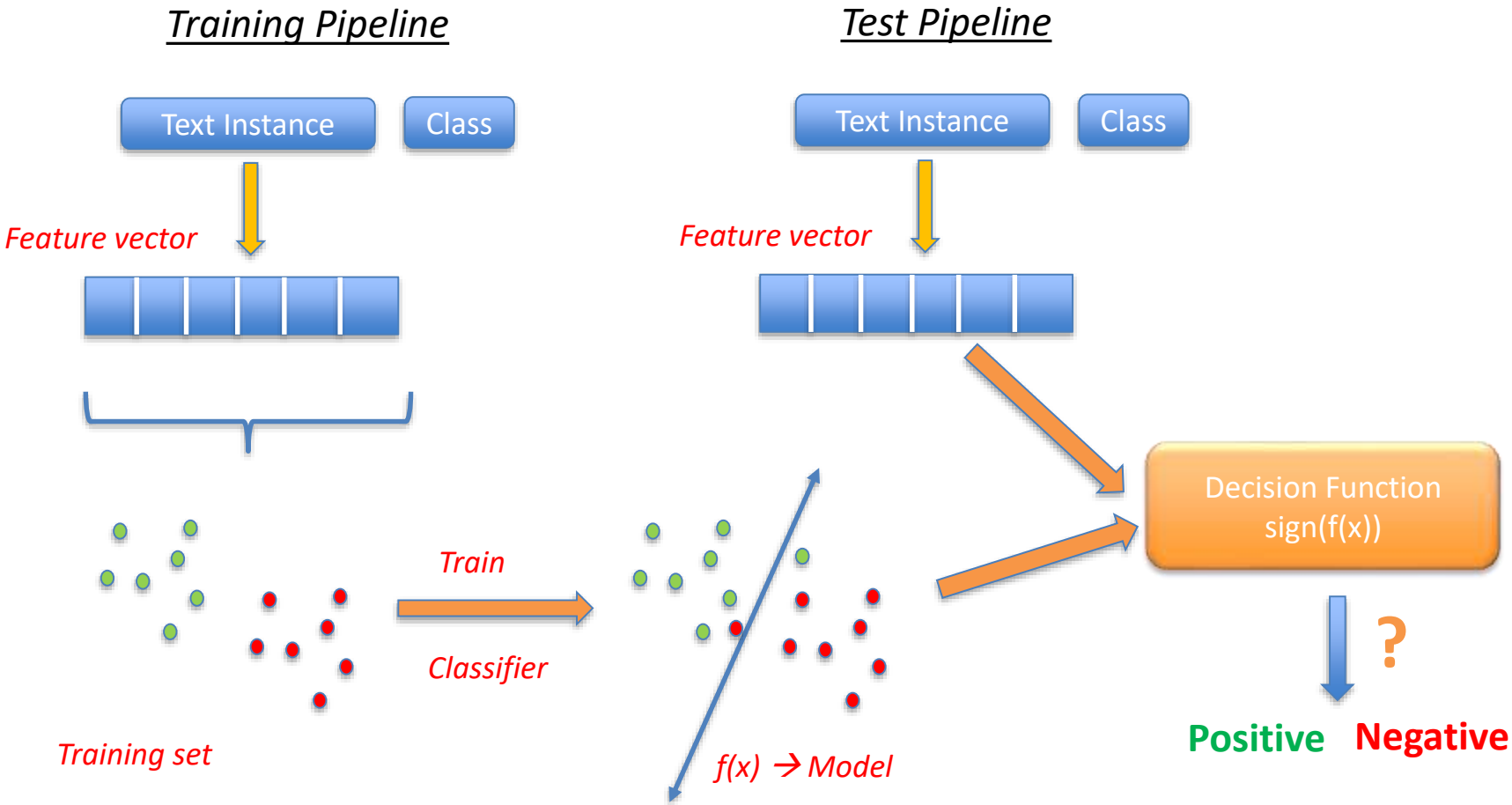
Positive

Negative

Neutral

An example of a text classification problem

A Machine Learning Pipeline for Text Classification



How do we design features?

Hints for positive review:

- *"well-made love saga"*
- *"deadly cocktail of hit music, taut script and bravura performances"*
- *"The funny and medical-inspired one liners are quite witty"*

Hints for negative review:

- *"It has been remade several times"*
- *"Kiara Advani doesn't have much dialogues and her screen time is limited in the second half."*

Confusing signals:

- *"Or does it fail to stir the emotions of the viewers?"*
- *"Yet another Tere Naam"*
- *Sarcasm*
- *Thwarted expressions*

A feature vector characterizes the text → its signature
Similar texts should have similar feature vectors

Simple Features

Bag-of-words (presence/absence)

Well-made	hit	script	lovely	boring	music
1	1	1	1	0	1

Term-frequency (tf) → word frequency is an indicator of importance of the word

Well-made	hit	script	lovely	boring	music
1	3	5	2	0	1

Tf-idf → discount common words which occur in all examples

$$idf(w) = \frac{d_w}{D}$$

Well-made	hit	script	lovely	boring	music
0.3	0.5	0.7	2	0.1	1

d_w: number of documents containing word *w*

D: total number of documents

idf: inverse document frequency

Large and sparse feature vector: size of vocabulary

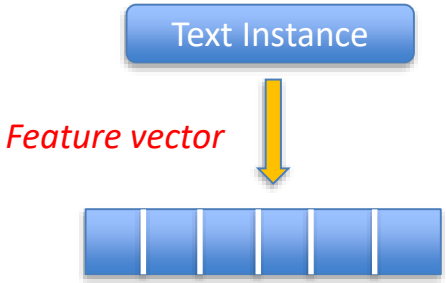
Each feature is atomic → similarity between features, synonyms not captured

More features

- Bigrams: e.g. *lovely_script*
- Part-of-speech tags
- Presence in [positive/negative] sentiment word list
- Negation words
- Is the sentence sarcastic (output from saracasm classifier?)

- *These features have to be **hand-crafted manually** – repeat for domains and tasks*
- ***Need linguistic resources** like POS, lexicons, parsers for building features*
- *Can some of these features be discovered from the text in an unsupervised manner using raw corpora?*

Where do we want to go?

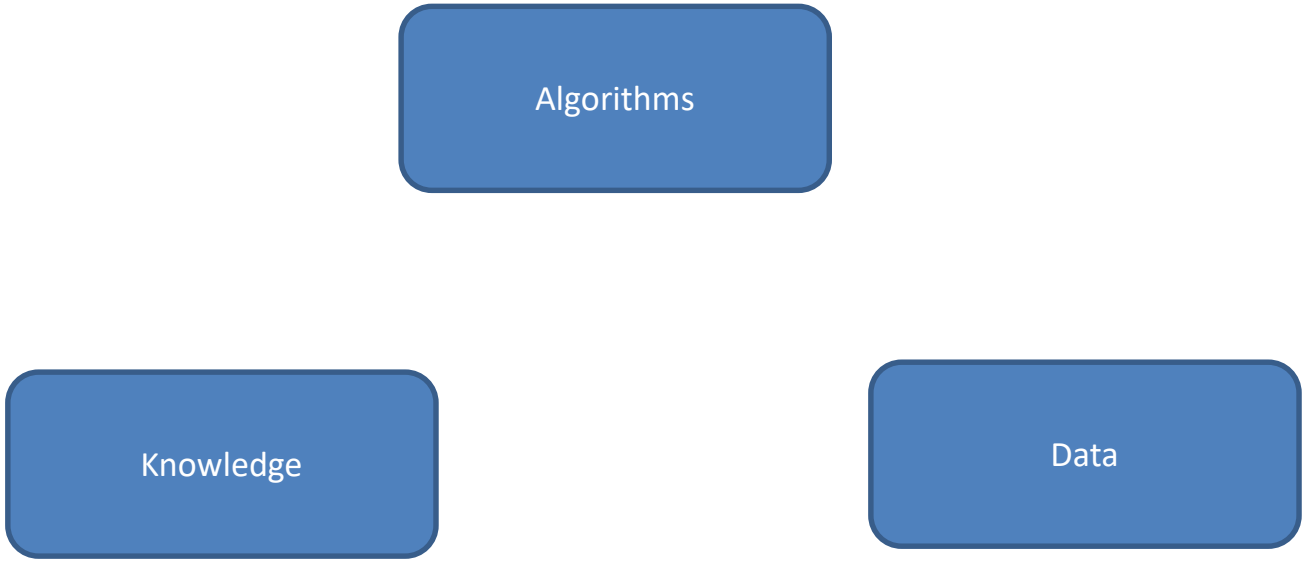


Can we replace the
high-dimensional, resource-heavy document feature vector

with

- *low-dimensional vector*
- *learnt in an unsupervised manner*
- *subsumes many linguistic features*

Facets of an NLP Application



Facets of an NLP Application

RULE-BASED SYSTEMS

Expert Systems
Theorem Provers
Parsers
Finite State Transducers

Largely language independent

Rules for morphological analyzers, Production rules, etc.

Lot of linguistic knowledge encoded

Paradigm Tables, dictionaries, etc.

Lot of linguistic knowledge encoded

Some degree of language independence through good software engineering and knowledge of linguistic regularities

Facets of an NLP Application

STATISTICAL ML SYSTEMS (Pre-Deep Learning)

Largely language independent, could solve non-trivial problems efficiently
Supervised Classifiers
Sequence Learning Algorithms
Probabilistic Parsers
Weighted Finite State Transducers

Feature Engineering

Lot of linguistic knowledge encoded
Feature engineering is easier than maintain rules and knowledge-bases

***Annotated Data**, Paradigm Tables, dictionaries, etc.*

Lot of linguistic knowledge encoded

General language-independent ML algorithms and easy feature learning

Facets of an NLP Application

DEEP LEARNING SYSTEMS

Algorithms

Largely language independent

Fully Connected Networks
Recurrent Networks
Convolutional Neural Networks
Sequence-to-Sequence Learning

Knowledge

Representation Learning, *Architecture Engineering, AutoML*

Feature engineering is unsupervised, largely language independent

Data

Annotated Data*, ~~Paradigm Tables, dictionaries, etc.~~*

Very little knowledge; annotated data is still required

Neural Networks provide a convenient language for expressing problems, representation learning automated feature engineering

Facets of an NLP Application

DEEP LEARNING SYSTEMS

Algorithms

Largely language independent

Fully Connected Networks
Recurrent Networks
Convolutional Neural Networks
Sequence-to-Sequence Learning

Knowledge

Representation Learning, *Architecture Engineering, AutoML*

Data

Annotated Data*, ~~Paradigm Tables, dictionaries, etc.~~*

Very little knowledge; annotated data is still required

Feature engineering is unsupervised, largely language independent

Neural Networks provide a convenient language for expressing problems, representation learning automated feature engineering

The core of a Deep Learning NLP system:

Ability to represent linguistic artifacts (words, sentences, paragraphs, etc.) with low-dimensional vectors that capture relatedness

How do we learn such representations?

Outline

- *What is Natural Language Processing?*
- *A Linguistics Primer*
- *Symbolic vs. Connectionist Approaches*
- ***Distributional Semantics***
- *Word Embeddings*
- *Sentence Embeddings*
- *Building simple NLP applications*
- *Summary*

DISTRIBUTIONAL SEMANTICS

Distributional Hypothesis

“A word us known by the company it keeps” *- Firth (1957)*

“Words that occur in similar contexts tend to have similar meanings”
- Turney and Pantel (2010)

He is unhappy about the failure of the project

The failure of the team to successfully finish the task made him sad

Distributed Representations

Sad: (the, failure, of, team, to, successfully, finish, task, made, him)
Unhappy: (he, is, about, the, failure, of, project)

- A word is represented by its context
- Context:
  - Fixed-window
  - Sentence
  - Document
- The distribution of the context defines the word
- The distributed representation has intrinsic structure
- Can define notion of similarity based on contextual distributions

What similarities do distributed models capture?

Words similar to 'unhappy'

displeased
dissatisfied
annoyed
frustrated
miffed
angry
incensed
livid
peeved
irked
unsatisfied
disillusioned
disappointed
disgusted
happy
unimpressed
disenchanted
fuming
angered
irritated
infuriated
dismayed
unhappiness
satisfied
ambivalent
upset
disheartened
concerned
uneasy

Paradigmatic Relationshop

Words which can occur in similar contexts are related

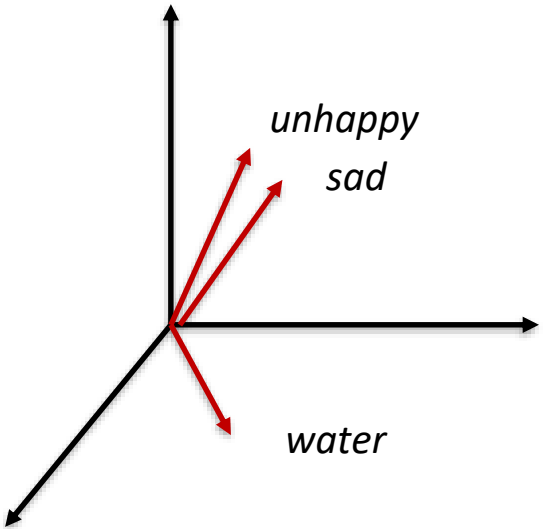
Attributional Similarity

- *degree of correspondence between the properties of words*
- *Loosely means the same as semantic similarity, semantic relatedness*
- *Could capture synonyms, antonyms, thesaurus words*

Relational Similarity

- between two pairs of words a : b and c : d
- depends on the degree of correspondence between the relations of a: b and c:d
- Captures analogical relations
- *air: bird, water: fish*

Vector Space Models



Each word is represented by a vector encoding of its context – How?

Similarity of words can be defined in terms of vector similarity: Cosine similarity, Euclidean distance, Mahalanobis distance

Efficient computation of many similarities: Sparse Matrix Multiplication, Locality Sensitive Hashing, Random Indexing

Cosine similarity equation

Long history of Vector Space Models used to capture distributional properties
- IR (Salton, 1975), LSI (Deerwater, 1990)

What embeddings are we interested in?

- Distributed Representations for words (Word embeddings)
- Word embeddings for morphologically rich languages
- Contextual Word Embeddings
- Sentence embeddings

Peter Turney, Patrick Pantel. *From Frequency to Meaning: Vector Space Models of Semantics.* JAIR. 2010.
Jeff Mitchell, Mirella Lapata. *Vector-based models of semantic composition.* ACL. 2008.

Outline

- *What is Natural Language Processing?*
- *A Linguistics Primer*
- *Symbolic vs. Connectionist Approaches*
- *Distributional Semantics*
- ***Word Embeddings***
- *Sentence Embeddings*
- *Building simple NLP applications*
- *Summary*

WORD EMBEDDINGS

What properties should word embeddings have?

- Should capture similarity between words
- Learn word embeddings from raw corpus based on distributional/context information
- Pre-trained embeddings
- Represent words in a low-dimensional vector space

Co-occurrence Matrix

Context

Word

	sad	unhappy	the	of	project
Sad					
Unhappy					
failure					

Word-context co-occurrence matrix filled across corpus

How do we fill this?

One-hot representations

Context

Word

	sad	unhappy	the	of	project
Sad	1	0	1	0	0
Unhappy	1	0	0	0	1
failure	0	1	0	1	1

Cannot capture the quantum of similarity

With frequency information

Context

Word

	sad	unhappy	the	of	project
Sad	5	0	10	0	0
Unhappy	3	0	0	0	2
failure	0	7	0	3	10

- *It is a good idea to length-normalize the vectors*
- *Raw frequencies are problematic*
- *Very high-dimensional representation*

Problem with raw frequencies

- *Some frequent words will dominate*
- *Similarity measurements will be biased*
- *Solutions*
  - *Ignore frequent words like ‘of’, ‘the’*
  - *Use a threshold on maximum frequency*
  - *Pointwise Mutual Information*

Pointwise Mutual Information (PMI)

- *Measure if (word,context) pair occur together by chance*
- *Is the context informative about the word?*
- *Uniformly frequent context words will have low PMI*

$$PMI(w,c) = \log \frac{p(c|w)}{p(c)}$$
$$= \log \frac{count(w,c) * N}{count(c) * count(w)}$$

Positive PMI: negative values are problematic, not reliable with small corpora

$$PPMI(w,c) = PMI(w,c) \quad if\ PMI(w,c) > 0$$
$$= 0 \quad otherwise$$

Singular Value Decomposition

$$\underset{|V|\times|V|}{\begin{bmatrix} X \end{bmatrix}} = \underset{|V|\times|V|}{\begin{bmatrix} W \end{bmatrix}} \underset{|V|\times|V|}{\begin{bmatrix} \sigma_1 & 0 & 0 & \dots & 0 \\ 0 & \sigma_2 & 0 & \dots & 0 \\ 0 & 0 & \sigma_3 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \sigma_V \end{bmatrix}} \underset{|V|\times|V|}{\begin{bmatrix} C \end{bmatrix}}$$

SVD provides a way to factorize a co-occurrence matrix into

- Word embedding Matrix (W)
- Context embedding Matrix (C)
- Singular values which capture variance captured by each dimension (σ_i)

Deerwester, Scott, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. *Indexing by latent semantic analysis.* Journal of the American society for information science. 1990.

Low Rank Approximation

$$\underset{|V|\times|V|}{\begin{bmatrix} X \end{bmatrix}} = \underset{|V|\times k}{\begin{bmatrix} W \end{bmatrix}} \underset{k\times k}{\begin{bmatrix} \sigma_1 & 0 & 0 & \dots & 0 \\ 0 & \sigma_2 & 0 & \dots & 0 \\ 0 & 0 & \sigma_3 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \sigma_k \end{bmatrix}} \underset{k\times|V|}{\begin{bmatrix} C \end{bmatrix}}$$

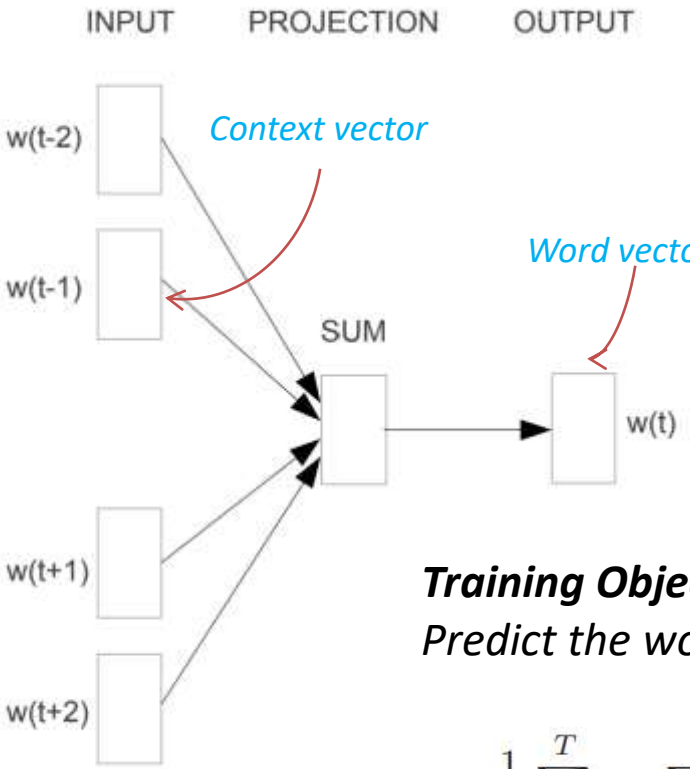
- Singular values are sorted in decreasing order
- Consider k dimensions in W corresponding to first k singular values
- Retains important information to reconstruct the matrix with high level of accuracy (defined by k and singular values)

Word2Vec

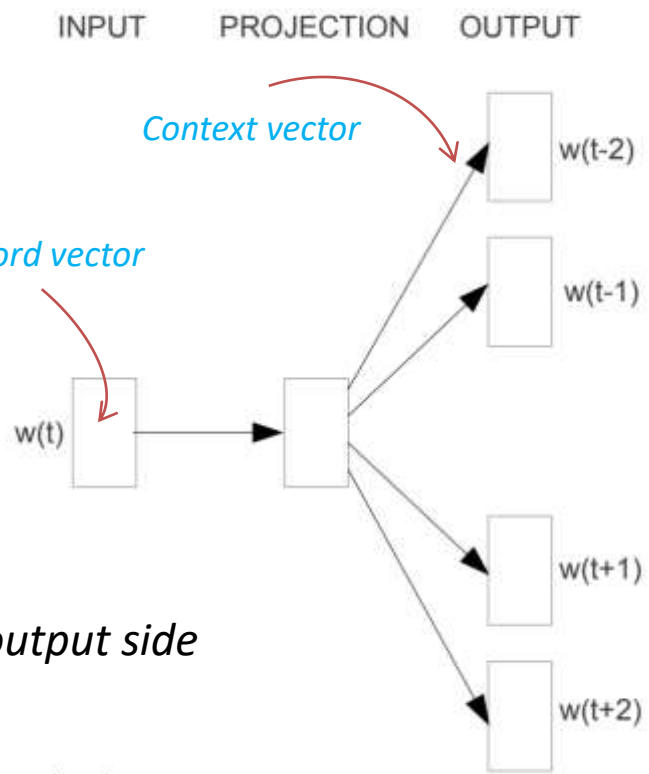
- Seminal work from Mikolov *et al.* 2012/2013
- **Prediction-based:** representation learning as classification problem
- Linear Model
- Very efficient and scalable training
- Can be used to train on large datasets
- Linearity of models enables simple, but interesting manipulations in the vector space
- Two models:
  - Continuous bag-of-words (CBOW)
  - Skip-gram

Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean. *Efficient Estimation of Word Representations in Vector Space*. Arxiv report. 2012.

CBOW



Skip-gram



Training Objective

Predict the words on the output side

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_t)$$

Training Large Vocabularies

- Computing softmax over entire vocab is expensive
- Reduce the training to a binary classification problem
  given (w, w_c): does w_c occur in the context of w
- Add *k* negative samples for every positive sample
- Speeds up training

$$\log \sigma({v'_{w_O}}^\top v_{w_I}) + \sum_{i=1}^{k} \mathbb{E}_{w_i \sim P_n(w)} \left[\log \sigma(-{v'_{w_i}}^\top v_{w_I}) \right]$$

Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J. *Distributed representations of words and phrases and their compositionality*. NIPS. 2013.

Count vs prediction-based methods (Levy et al.)

Are prediction-based methods better?

- Prediction-based methods are also matrix factorizations
  - They are not inherently better than count-based methods
- Various design decisions and hyper-parameters choices can explain success of prediction-based models:
  - Different importance to different context words
  - Frequency subsampling
  - Negative sampling and sample size
- Incorporating similar ideas into count-based models
  - Count-based better at **similarity** tasks
  - Prediction-based better at **analogy** tasks

Omer Levy, Yoav Goldberg and Ido Dagan. *Improving Distributional Similarity with Lessons Learned from Word Embeddings*. TACL. 2015.

GloVe (Global Vectors)

Co-occurrence-based algorithms use global context information

- *Effective use of co-occurrence statistics*
- *Difficult to scale to large datasets*

Prediction based models use local context information

- *Do not effectively use co-occurrence statistics*
- *Long training time*
- *Can be trained on large datasets*

Can we combine the benefits of the two approaches?

Jeffrey Pennington, Richard Socher, Christopher D. Manning. *Glove: Global Vectors for Word Representation*. EMNLP. 2014.

GloVe (Global Vectors)

Question: *How is meaning captured in word vectors?*

Key Insight: *Meaning difference is captured by ratio of conditional probabilities*

	x = solid	x = gas	x = water	x = fashion
$P(x\|\text{ice})$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(x\|\text{steam})$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$\frac{P(x\|\text{ice})}{P(x\|\text{steam})}$	8.9	8.5×10^{-2}	1.36	0.96

*GloVe **explicitly** models this intuition*

$$w_i \cdot w_j = \log P(i|j)$$

$$w_x \cdot (w_a - w_b) = \log \frac{P(x|a)}{P(x|b)}$$

Morphology

Inflectional Morphology

play
plays
played
playing

घर
घरात
घरासमोर
घरी
घराचा
घरासमोरचा
घरासमोरच्या

Languages like Marathi have large number of inflectional variations

Capture grammatical properties

Derivational Morphology

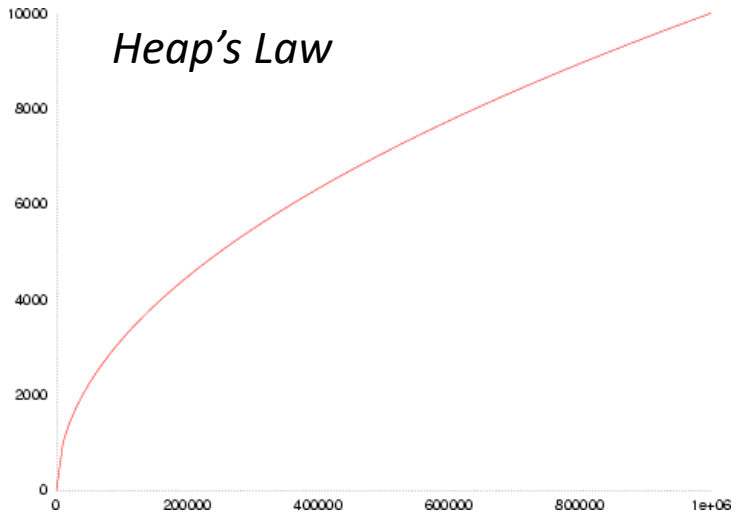
capitalism
communism
socialism
fascism

disregard
disrespect
disjoint
dislike

New words by composing existing words

Morphologically related words should have similar embeddings

The Morphological Challenge



Vocabulary increases with corpus size

For morphologically rich languages, potential vocabulary is large (theoretically infinite)

It is not possible to learn embeddings for all possible words

Large vocabulary
→ *too may words with small counts*
→ *cannot estimate embeddings effectively*

How to estimate embeddings for morphological variants not seen in training corpus?
How to ensure that data sparsity does not adversely affect learning word embeddings?

How to incorporate morphological information into word embeddings?

Define word as a composition of subword elements

Unit	Example
Character	घ र ◌ा स म ◌ो र च ◌ा
Character 3-gram	घरा रास ◌ासम समो मोर ◌ोरच रचा
Character overlap 3-gram	घरा समो रचा
Syllable	घ रा स मो र चा
Morpheme	घर ◌ा समोर चा

Morphology aware-embeddings

Define word embeddings as a functions of subword embeddings

$$emb_{final}(w) = F(S, w)$$

Where, S is the set of subwords of w

$$emb_{final}(w) = emb(w) + \sum_{s \in w} emb(s)$$

$$emb_{final}(w) = emb(w) + emb(\text{घर}) + emb(\text{ा}) + emb(\text{समोर}) + emb(\text{चा})$$

With the redefined word embedding, train the embeddings on the data

FastText

- A variant of the word2vec algorithm that can handle morphology
- Simple model: word is a bag overlapping n-grams
- Final word embedding is sum of n-gram embedding + intrinsic word embedding
- Can generate embeddings for OOVs
- Highly scalable implementation which can train large datasets very efficiently

Piotr Bojanowski, Edouard Grave, Armand Joulin, Tomas Mikolov. *Enriching Word Vectors with Subword Information*. TACL. 2017.

Evaluating Quality of Word embeddings

Extrinsic Evaluation

- How well do word embeddings perform for some NLP task?
  - Text classification, sentiment analysis, question answering
- ***Cons***:
  - task specific – does not give general insight
  - some tasks may be time-consuming to evaluate
- ***Pros:*** Sometimes data may just be available

Intrinsic Evaluation

- Specifically designed to understand word embedding quality
  - Semantic relatedness, semantic analogy, syntactic analogy
  - synonym detection, hypernym detection
- ***Cons:***
  - *Careful design of testsets and evaluation tasks*
  - *Cost and expertise required to create testsets*
- ***Pros:*** *typically quick to run to speed up development cycle*

(See SemEval tasks to discover tasks and datasets)

Semantic Relatedness

- Humans judge relatedness:
  $sim_{human}(bird, sparrow) = 0.8$
- Cosine similarity using word embeddings:
  $sim_{human}(bird, sparrow) = cosine\_sim(v_{bird}, v_{sparrow})$
- Embeddings quality: Correlation (sim_{human}, sim_{model}) over test dataset.
- **Popular datasets:**
  - RG-65, MC30, WordSim-353, SimLex-999, SimLex-3500
  - 7 Indian languages from IIIT-Hyderabad (Link)
    - Translations of RG-65 and WordSim-353

- Tests attributional similarity
- Design issues:
  - How are the test pairs decided?
  - Inter-annotator agreement

Word Analogy

a:b :: c: d
Japan: Tokyo :: France: ?
Japan: Tokyo :: France: Paris

Find the nearest word which satisfies

$$d = \underset{d' \in V}{\operatorname{argmin}}\ distance(d', c + b - a)$$

Tests relational similarity

Semantic Analogies: Japan: Tokyo :: France: Paris
Syntactic Analogies: play: playing :: think: thinking

Embedding quality: Accuracy of prediction over testset

Popular datasets:

- Google, MSR, BATS, SemEval 2012
- Hindi analogy dataset from FastText project

Practical tips for building word embeddings

- The larger corpora the better
  - More than 500 million words is a good thumb rule
  - Look at linear models with efficient implementations
- 300-500 dimensional embeddings work well
- Morphologically rich languages
  - Use a model which uses subword units e.g. FastText
- No single good algorithm: try different approaches
- Hyper-parameter tuning gives decent gains
- Normalize vectors to unit length

Resources

Software

- Word2Vec implementation in GenSim
- FastText
- GloVe

Reading

- Sebastin Ruder's lucid articles: Part 1 here .. follow the rest
- Prof. Mitesh Khapra's slides: [link]
- *word2vec Parameter Learning Explained* by Xin Rong
- *word2vec Explained: deriving Mikolov et al.'s negative-sampling wordembedding method* by Yoav Goldberg and Omer Levy

Outline

- *What is Natural Language Processing?*
- *A Linguistics Primer*
- *Symbolic vs. Connectionist Approaches*
- *Distributional Semantics*
- *Word Embeddings*
- ***Sentence Embeddings***
- *Building simple NLP applications*
- *Summary*

SENTENCE EMBEDDINGS

A nice summary of many sentence embeddings:
https://medium.com/huggingface/universal-word-sentence-embeddings-ce48ddc8fc3a

Semantically similar sentences should have similar embeddings

Can we have a distributed representation of larger linguistic units like phrases and sentences?

Can phrase/sentence representations be composed from word representations? (Compositional Distributional Semantics)

How do we evaluate the quality of sentence embeddings?

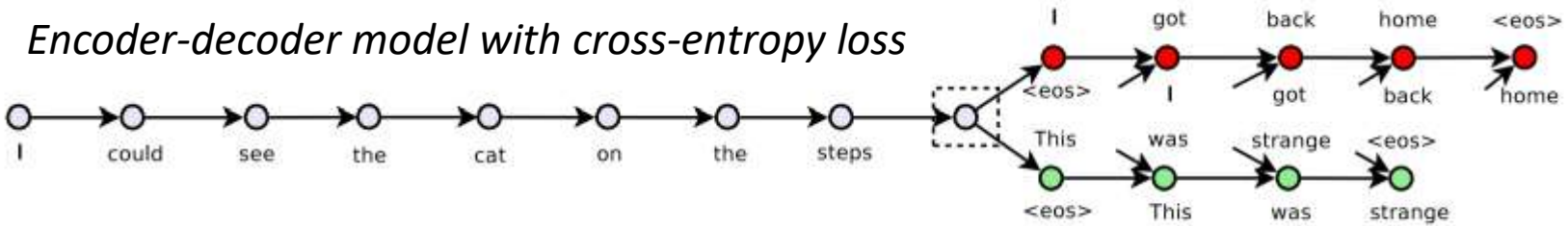
Bag-of-Word approaches

Method	Key idea	Reference	Example
Average of word embeddings	Strong baseline		$z = 0.5(x + y)$
+ concatenation of diverse embeddings	Increase model capacity	https://arxiv.org/abs/1803.01400	$x = x_{\{glove\}} \odot x_{\{w2v\}}$
Weighted Average	Frequent words not important	https://openreview.net/pdf?id=SyK00v5xx	$z = \alpha_x x + \alpha_y y$
Elementwise product		https://www.aclweb.org/anthology/P08-1028	$z_j = x_j y_j$
Power Means + Concatenation	Different means capture different informatio	https://arxiv.org/abs/1803.01400	$z = \sqrt[p]{\frac{1}{2}(x^p + y^p)}$

Skip-Thought Vectors

https://arxiv.org/abs/1506.06726

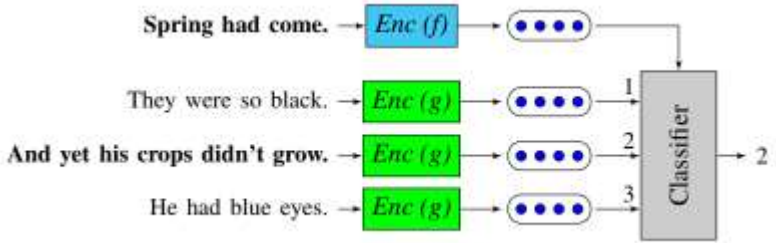
Encoder-decoder model with cross-entropy loss



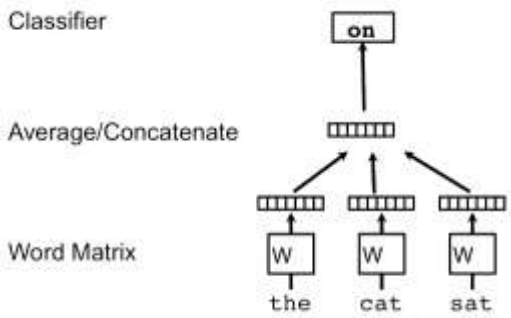
- Distributional hypothesis applied to sentences
- Sentence-level analog of skip-gram model
- Given a sentence, predict previous and next sentence in a discourse

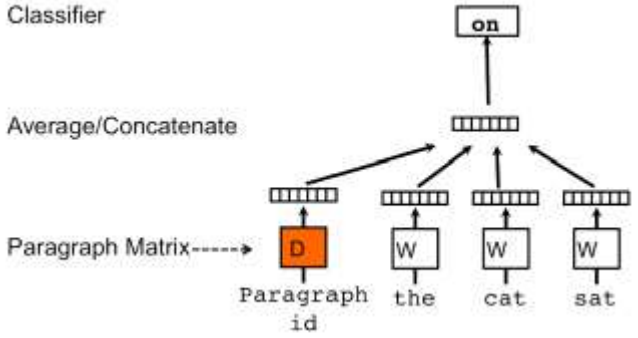
Quick-thought Vectors https://arxiv.org/abs/1803.02893

- Pose as classification problem
- Predict if a sentence belongs in context
- Add negative examples



Paragraph Vector





At inference time, paragraph vector needs to be computed for new para with a backpropagation update

Directly Learning Sentence Embeddings

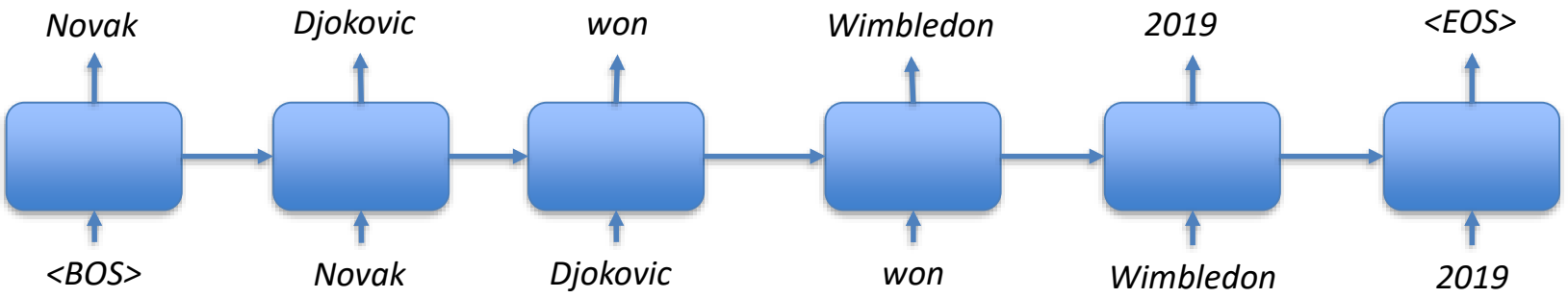
Previous approaches composed word vectors

Can we directly train sentence embeddings

What would be a good unsupervised objective to train sentence embeddings?

A Language Model!

Language Model



$$P(w_1, ..., w_m) = \prod_{i=1}^{i=m} P(w_i | w_1, ..., w_{i-1})$$

$$h_t = \sigma(W^{(hh)} h_{t-1} + W^{(hx)} x_{[t]})$$

Recurrent Neural Network

- A Neural Network cell with state
- Useful for modelling sequences
- Output is a function of previous state and current input

$$\hat{y}_t = softmax(W^{(S)} h_t)$$

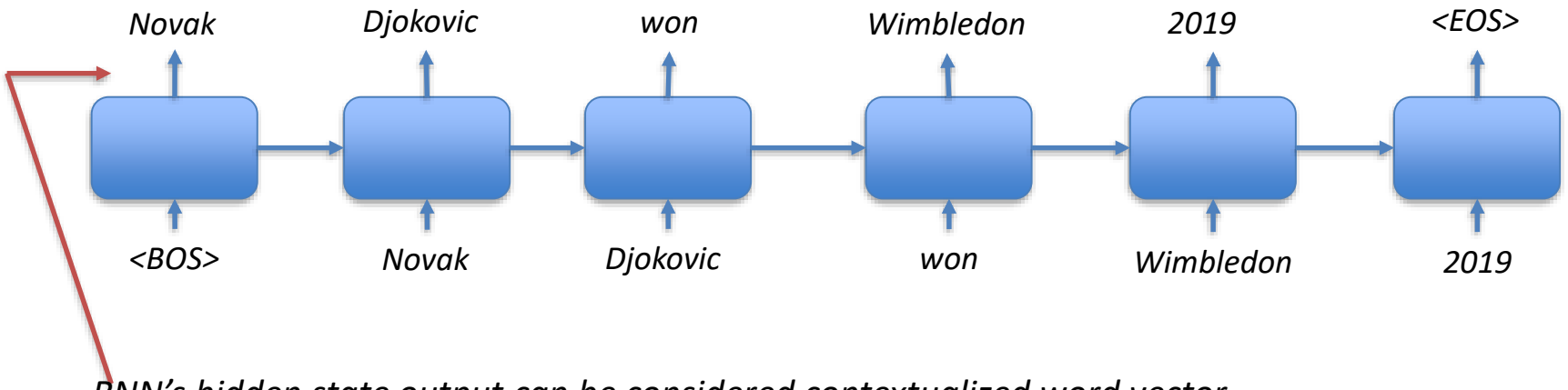
$$J = \frac{1}{T} \sum_{t=1}^{T} J^{(t)}(\theta) = -\frac{1}{T} \sum_{t=1}^{T} \sum_{j=1}^{|V|} y_{t,j} \times log(\hat{y}_{t,j})$$

Recurrent NN Approaches

- Train a Language Model on monolingual corpus
- The encoder states represent contextualized word vectors
  - Sense disambiguation
  - Some applications need these contextualized embeddings
- Sentence embedding can be a composition of contextualized word embeddings
  - See composition methods discussed previously
- Use LSTM or GRU units instead of RNN cell units
  - To solve exploding/vanishing gradient issues
- Use bi-LSTM instead of LSTM
  - Use information from both directions

Contextualized Word Vectors (*ELMO, COVE*)

ELMO: https://arxiv.org/abs/1802.05365, COVE: https://arxiv.org/abs/1708.00107



RNN's hidden state output can be considered contextualized word vector

- *Context considered in RNN hidden state ➔ some sort of disambiguation*
- *Deep Representations: take contextualized representations from multiple layers*

$$\mathbf{ELMo}_k^{task} = E(R_k; \Theta^{task}) = \gamma^{task} \sum_{j=0}^{L} s_j^{task} \mathbf{h}_{k,j}^{LM}.$$

- *Use Bi-LSTM instead of LSTM to capture bi-directional context*

How to use the pre-trained LM?

Pre-trained LM can be used as lower layer of neural network

Feature-based approach *(CoVE, ELMO): Application can directly use contextualized word vector*

Discriminative fine-tuning *(ULMFit, BERT, GPT):*

- *LM layers can be fine-tuned for downstream application*
- *Fine-tuning can include LM as an auxiliary objective*

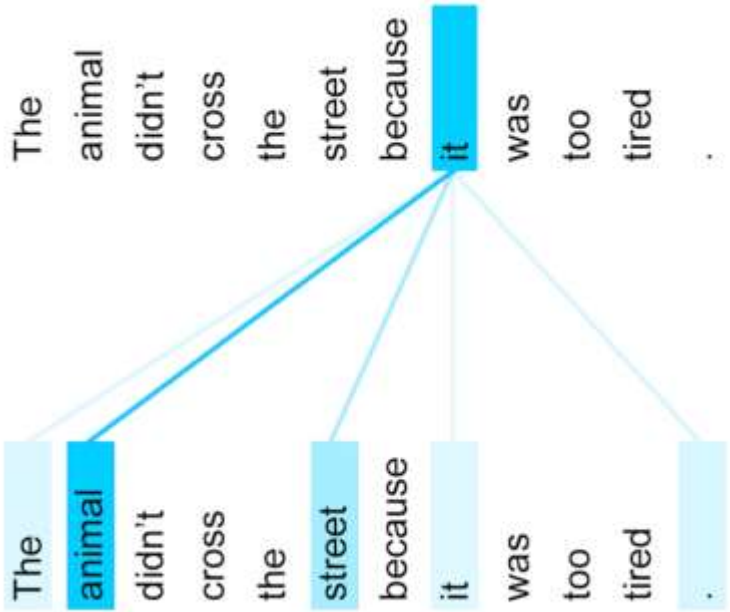
$$L(\theta) = L_{task}(\theta) + L_{LM}(\theta)$$

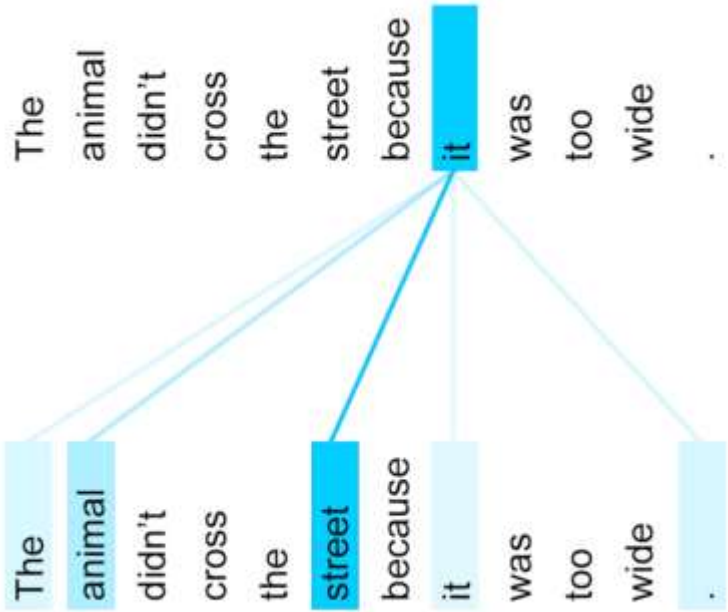
Sentence embeddings *(Infersent): Composition of contextualized word embeddings*

Transformer-based Approaches

- Weakness of RNN approaches: sequential processing
- Can CNN overcome this limitation?
  - Deep networks needed to handle long-range dependencies
- Transformer network relies on **self-attention** instead of recurrent connections
  - Self-attention relies on pairwise word similarity
- Advantages:
  - Parallelizes training
  - Train deeper networks
  - Handle larger datasets
  - Handle long range dependencies better

Self-attention

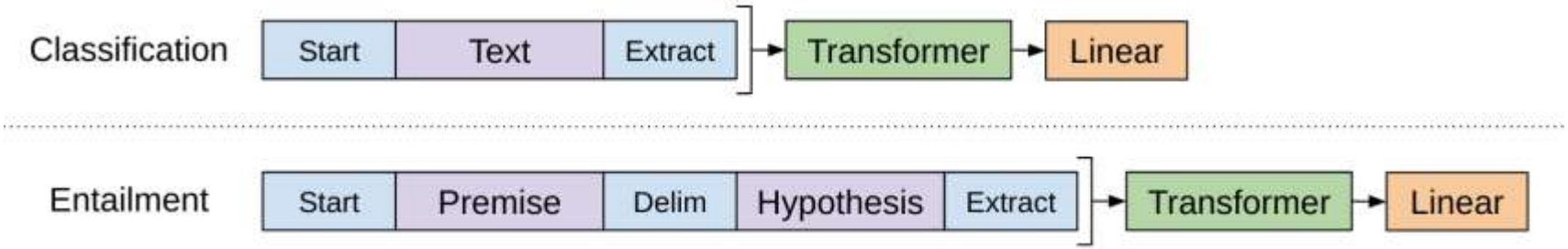




Open AI's GPT

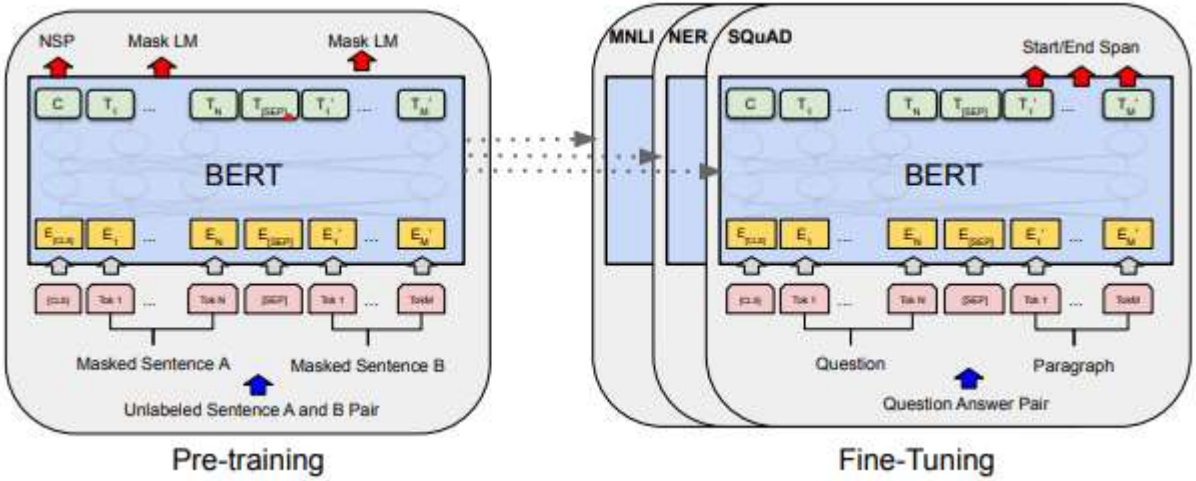
- Train a standard LM using *transformer decoder*
- Fine-tune the network on supervised tasks
- An interesting idea: task-specific input transformations

Reduce task-specific finetuning parameters



Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever. *Improving Language Understanding by Generative Pre-Training.* 2018.

Bidirectional Encoder Representation Transformer (BERT)



- **Jointly train on left and right context**
- Achieved via Masked LM objective → randomly delete a few words
- Achieved state-of-art results on most benchmarks by a big margin!

Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.* NAACL. 2019.

Supervised Approaches

- Language Modelling is an unsupervised objective that is representative of the language
- Can we do better with supervised tasks that capture the complexities of language?

What are such possible tasks?

- Natural Language Inference / Textual Entailment (InferSent)
  https://arxiv.org/abs/1705.02364
- Machine Translation (CoVE)
  https://arxiv.org/abs/1708.00107

Premise	Label	Hypothesis
Fiction		
The Old One always comforted Ca'daan, except today.	*neutral*	Ca'daan knew the Old One very well.
Letters		
Your gift is appreciated by each and every student who will benefit from your generosity.	*neutral*	Hundreds of students will benefit from your generosity.
Telephone Speech		
yes now you know if if everybody like in August when everybody's on vacation or something we can dress a little more casual or	*contradiction*	August is a black out month for vacations in the company.
9/11 Report		
At the other end of Pennsylvania Avenue, people began to line up for a White House tour.	*entailment*	People formed a line at the end of Pennsylvania Avenue.

Multi-task Approaches

- Why just train on one task?
- **MSR/MILA**
  - NMT, NLI, Constituency Parsing, Skip-thought vectors
- **Google Universal Sentence Encoder**
  - Language Model, NLI
- **MSR MT-DNN**
  - Masked LM, Next Sentence Prediction, Single-sentence classification, Pairwise Text Similarity, Pairwise Text Classification, Pairwise Ranking

Prevents overfitting, better generalization

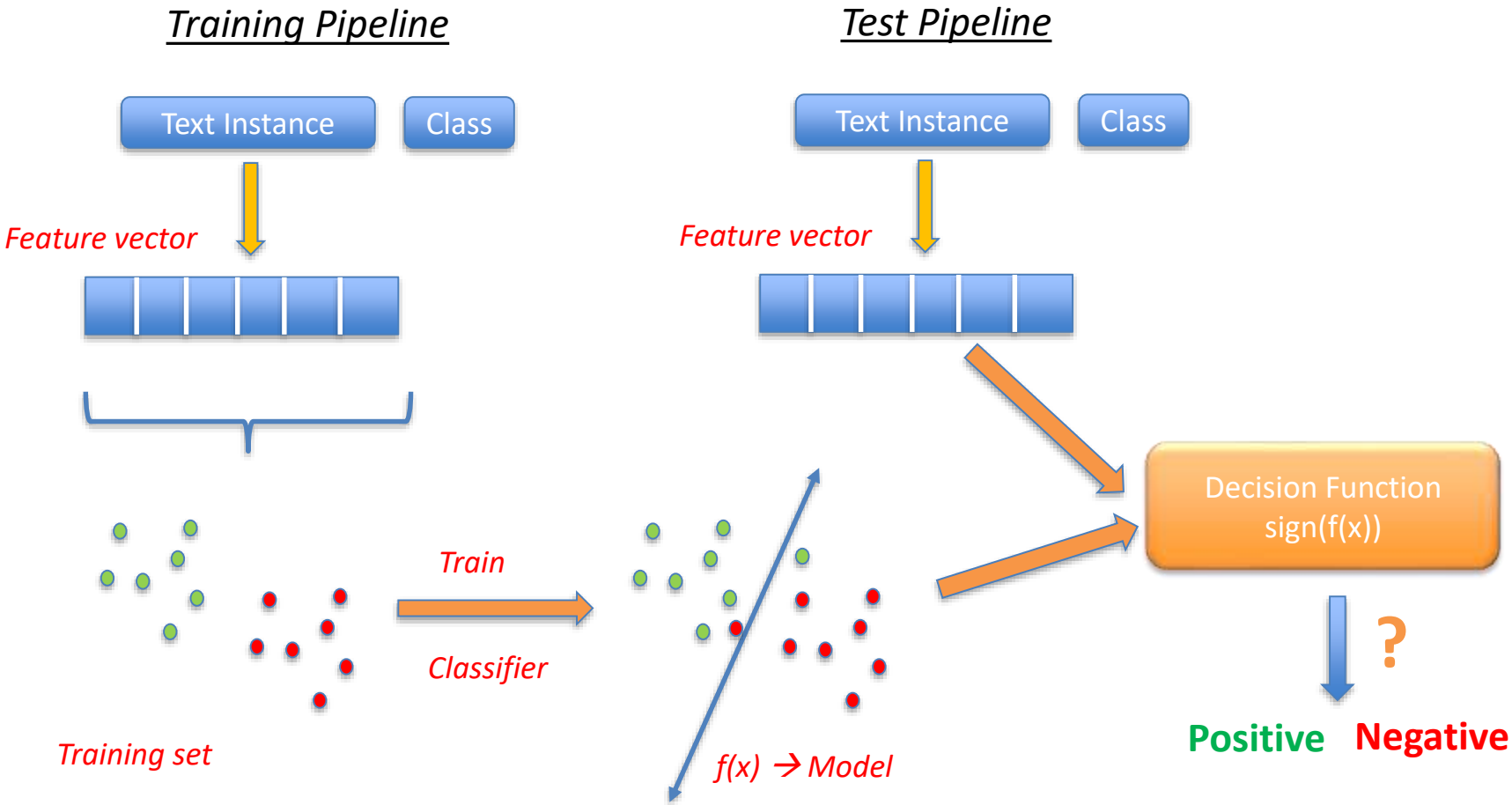
Evaluation Tasks

- SentEval downstream tasks
  - Movie review, product review, semantic textual similarity, image-caption retrieval, NLI, *etc.*
- SentEval probing tasks
  - evaluate what linguistic properties are encoded in your sentence embeddings
- GLUE dataset
  - Linguistic acceptability, sentiment analysis, paraphrase tasks, NLI

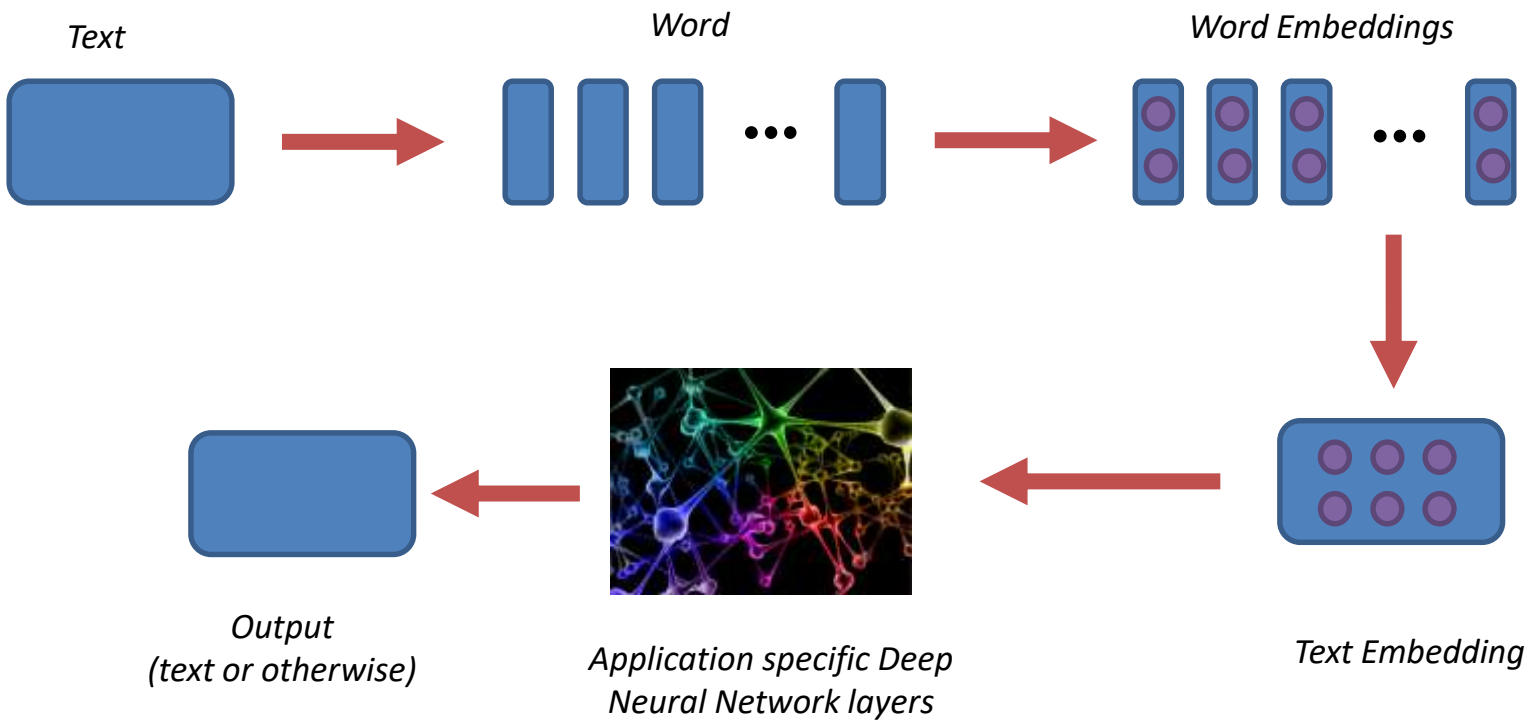
Outline

- *What is Natural Language Processing?*
- *A Linguistics Primer*
- *Symbolic vs. Connectionist Approaches*
- *Distributional Semantics*
- *Word Embeddings*
- *Sentence Embeddings*
- ***Building simple NLP applications***
- *Summary*

A Machine Learning Pipeline for Text Classification



A Typical Deep Learning NLP Pipeline



Training for a classification problem

Application layer outputs values for K classes: f_k k=1 to K

Softmax: Convert to probabilities $p_k = \frac{e^{f_k}}{\sum_j e^{f_j}}$

Objective: Minimize Negative Log-likelihood/Cross Entropy

$$NLL(D) = -\sum_{n=1}^{N} \log p_{y_n}$$

y_n is the label of the n^{th} training example between 1 and K

Optimizer: Stochastic Gradient Descent or its variants *(AdaGrad, ADAM, RMSProp)*

Decision Rule $y_x^* = \underset{k=1\ to\ K}{\operatorname{argmax}} \log p_k\ (NN(x))$

Training for a sequence labelling problem

Objective: Minimize Negative Log-likelihood/Cross Entropy of entire sequence

$$NLL(D) = -\sum_{n=1}^{N}\sum_{t=1}^{T} \log p_{y_{nt}}$$

y_n is the label of the n^{th} training example between 1 and K

Optimizer: Stochastic Gradient Descent or its variants *(AdaGrad, ADAM, RMSProp)*

Decision Rule

Find the sequence which maximizes the probability of the entire sequence

- Greedy Decoding
- Beam Search

Outline

- *What is Natural Language Processing?*
- *A Linguistics Primer*
- *Symbolic vs. Connectionist Approaches*
- *Distributional Semantics*
- *Word Embeddings*
- *Sentence Embeddings*
- *Building simple NLP applications*
- ***Summary***

Summary

- Shift in NLP solutions from classical ML to neural network approaches
- Less feature engineering
- Use of pre-trained embeddings
- End-to-end training

Natural Language Processing

NLP Super Applications

Anoop Kunchukuttan
Microsoft AI & Research

ankunchu@microsoft.com

The “big” super applications for NLP

- Machine Translation
- Question Answering
- Conversational Systems

- *Complex applications which need processing at every NLP layer*
- *Advances in each of these problems represent advances in NLP*
- *Captures imagination of users*

Another big question

Can we build language independent NLP systems?

Outline

- Machine Translation
- Question Answering
- Multilingual NLP

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

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)

S V O

E: Germany won the last World Cup

H: जर्मनी ने पिछला विश्व कप जीता था

S O V

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

Why is Machine Translation difficult?

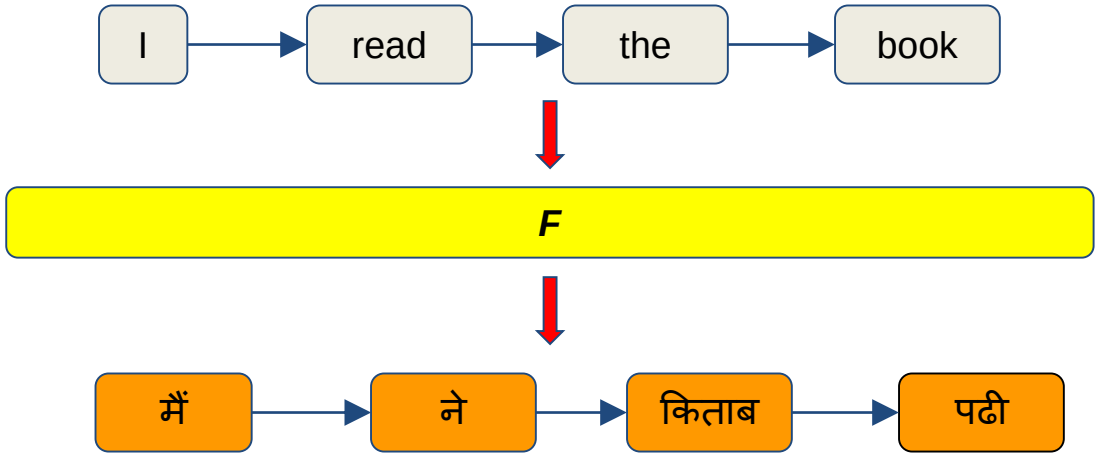
- **Ambiguity**
  - Same word, multiple meanings: मंत्री (minister or chess piece)
  - Same meaning, multiple words: जल, पानी, नीर (water)
- **Word Order**
  - Underlying deeper syntactic structure
  - Phrase structure grammar?
  - Computationally intensive
- **Morphological Richness**
  - Identifying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

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.*

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

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



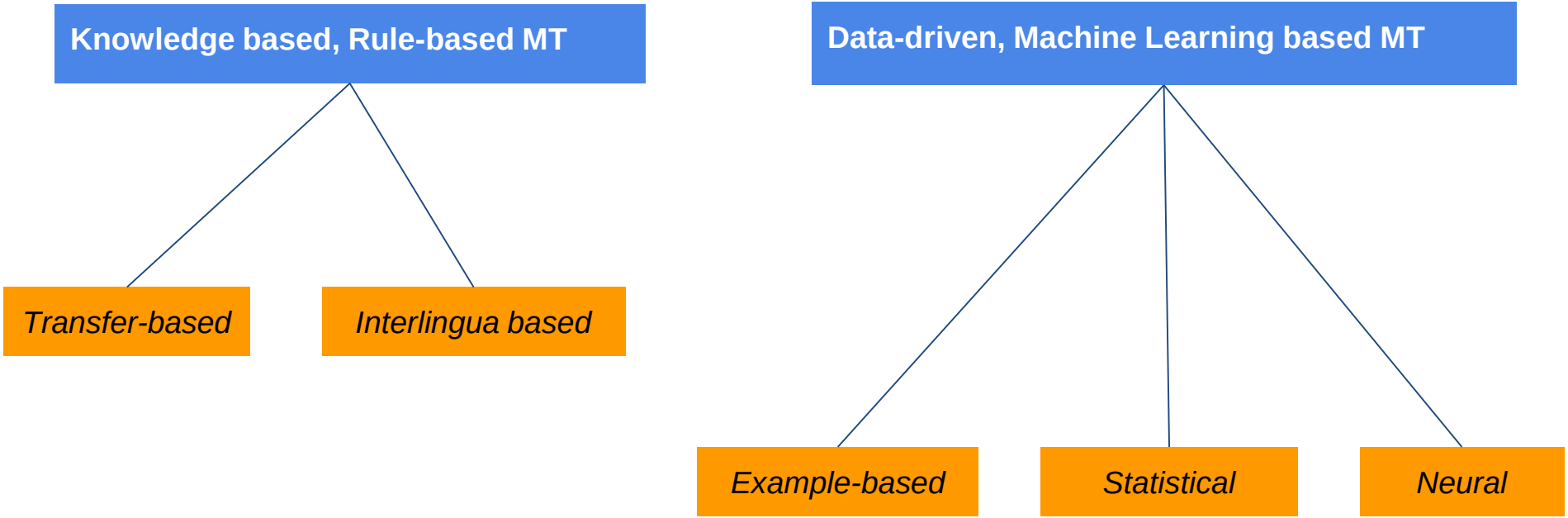
- Length of output sequence need not be the same as input sequence
- Prediction at any time step *t* has access to the entire input
- A very general framework

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*

Approaches to build MT systems



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

E: target language
F: source language

e: source language sentence
f : target language sentence

Best translation

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

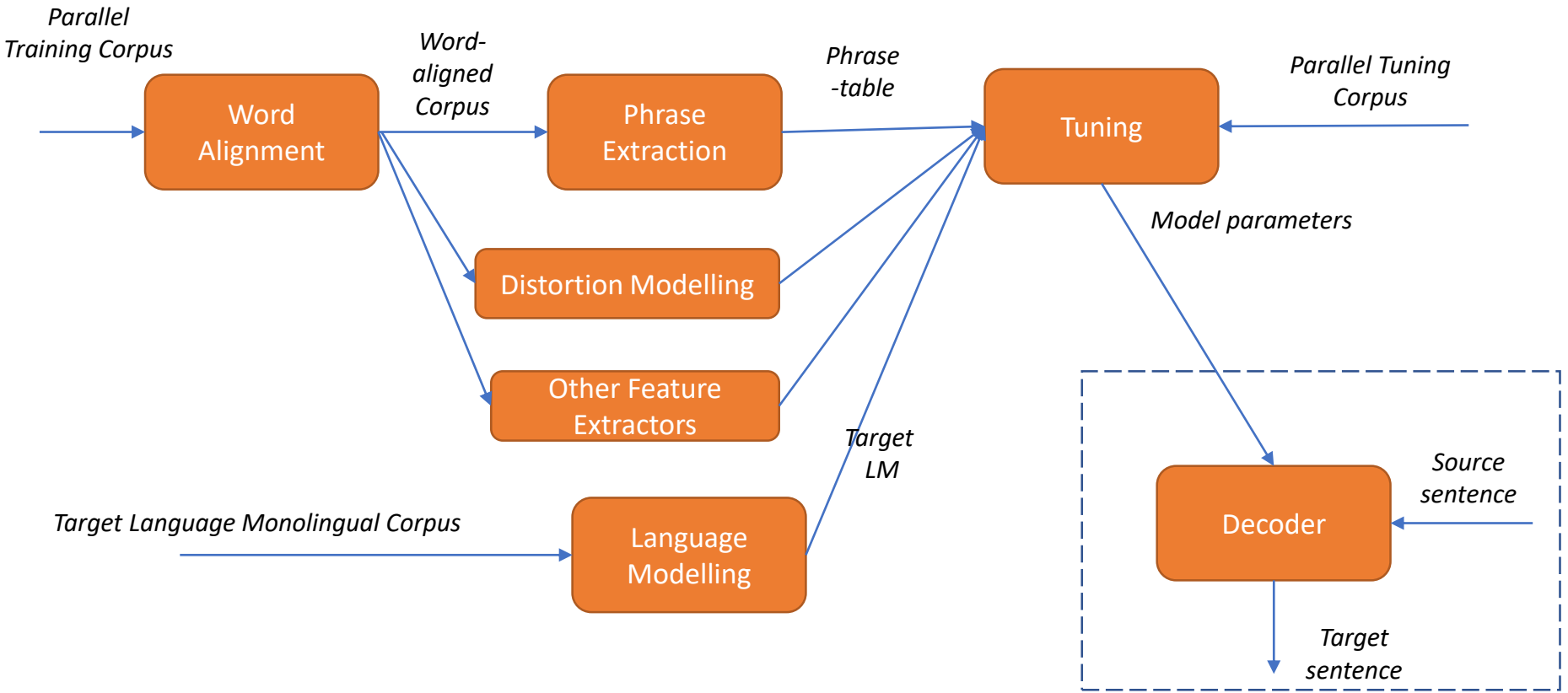
How do we **model** this quantity?

Typical SMT Pipeline

Language Model

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

Translation Model



***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

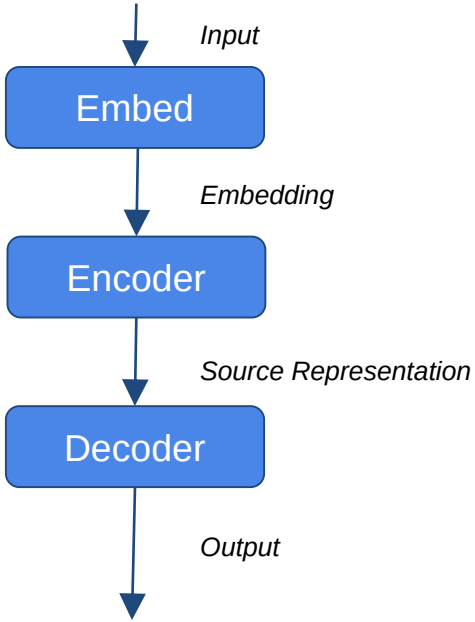
- *We cannot say anything about 'mansion' if it comes up at test time*
- *Creates problems for language model as well ⇒ whole are of smoothing exists to overcome this problem*

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 MACHINE TRANSLATION

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*

$$P(f|e) = softmax(decoder(encoder(x))$$

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
Water
house
tap

0.5	0.6	0.7
0.2	0.9	0.3
0.55	0.58	0.77
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 Explained

Use two RNN networks: the encoder and the decoder

(1) Encoder processes one sequence at a time

(2) A representation of the sentence is generated

(3) This is used to initialize the decoder state

(4) Decoder generates one element at a time

(5)… continue till end of sequence tag is generated

$A = \pi r^2$

s_0 s_1 s_1 s_3 s_4

I read the book

Encoding

h_0 h_1 h_2 h_3 h_4

y_1 y_2

मैं ने किताब पढी <EOS>

Decoding

$$P(y_i|y_{i-1} \dots y_1) = LSTM(h_{i-1}, y_{i-1})$$

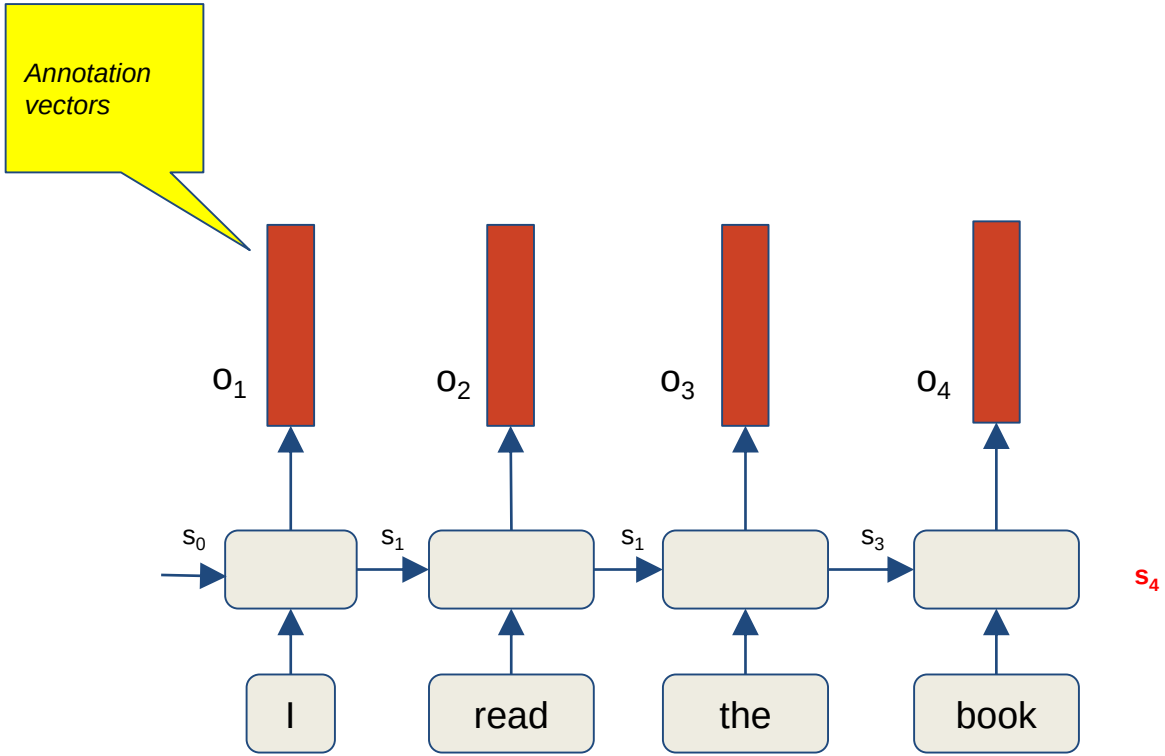
This approach reduces the entire sentence representation to a single vector

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:

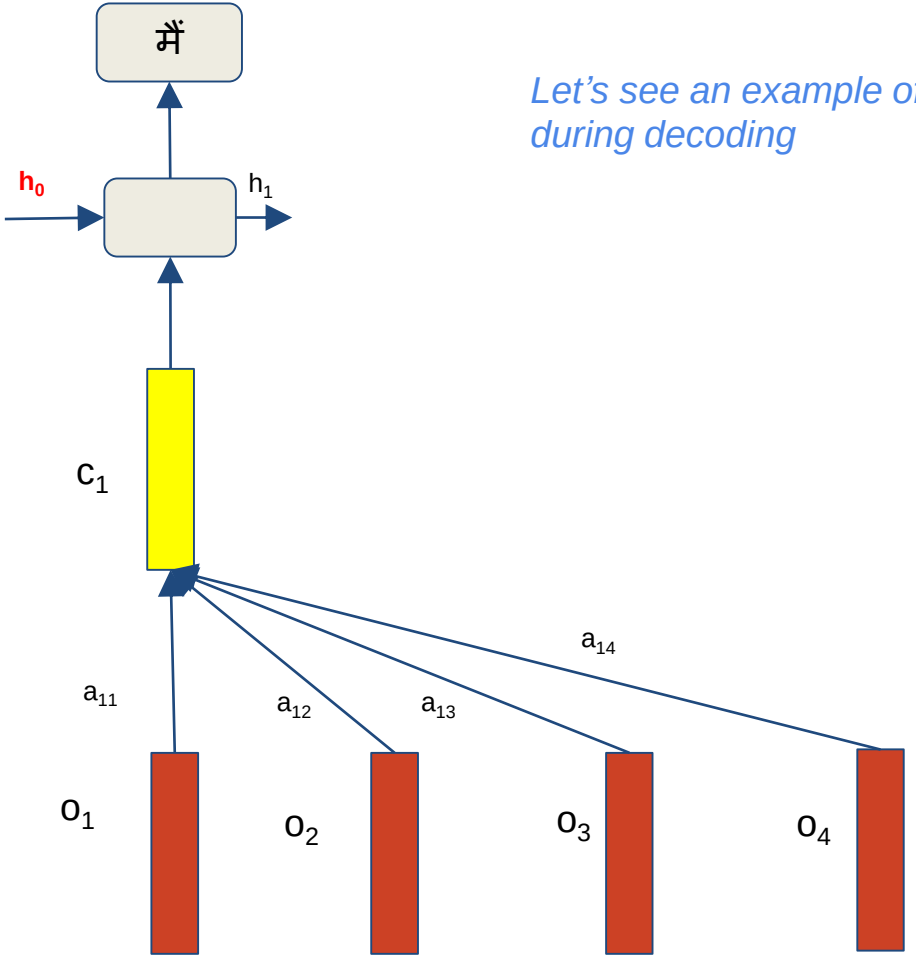
(1) Not all annotation vectors are equally important for prediction of the next element

(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**

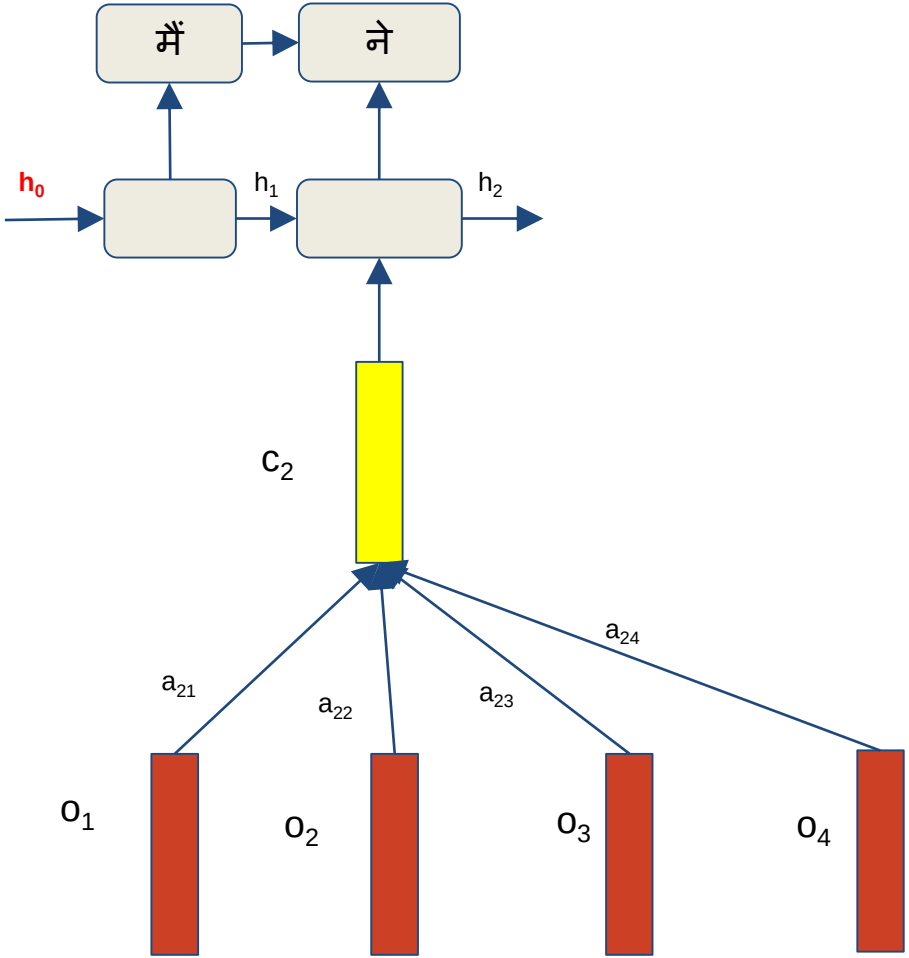
This averaged ***context vector*** is an input to the decoder

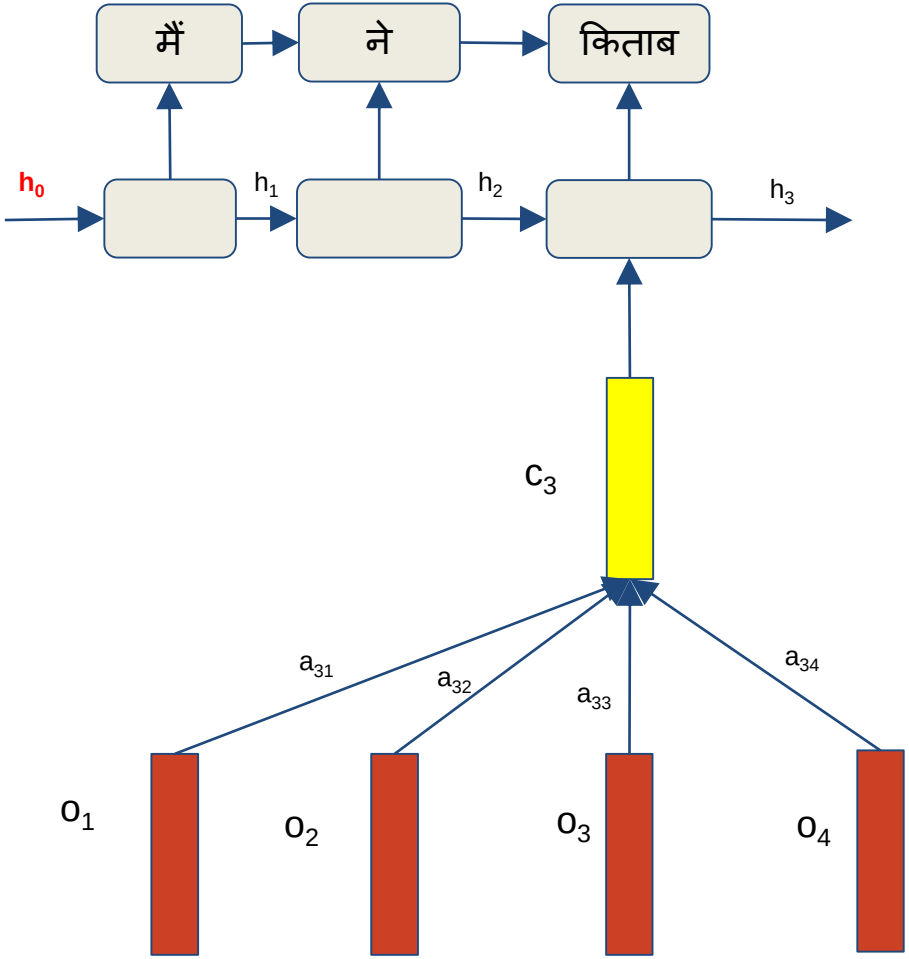


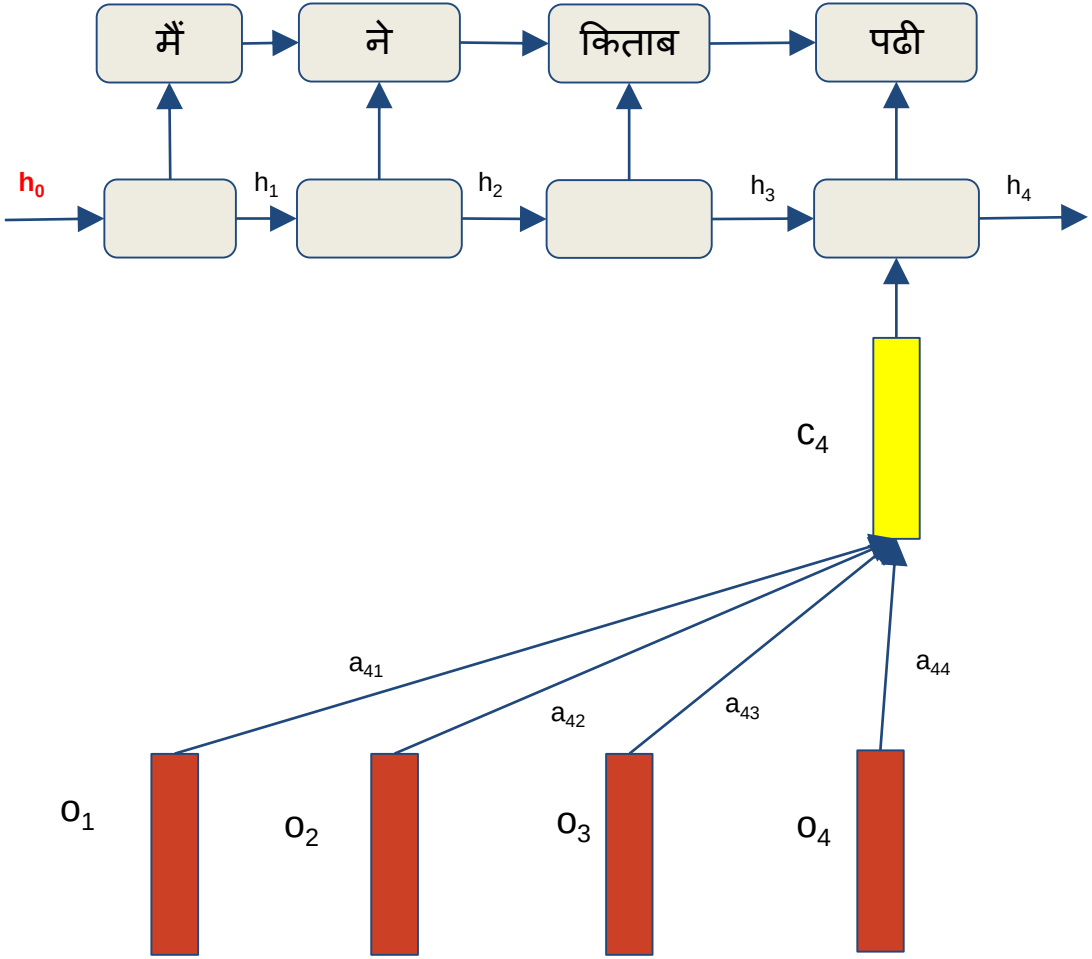
*Let's see an example of how the **attention mechanism** works during decoding*

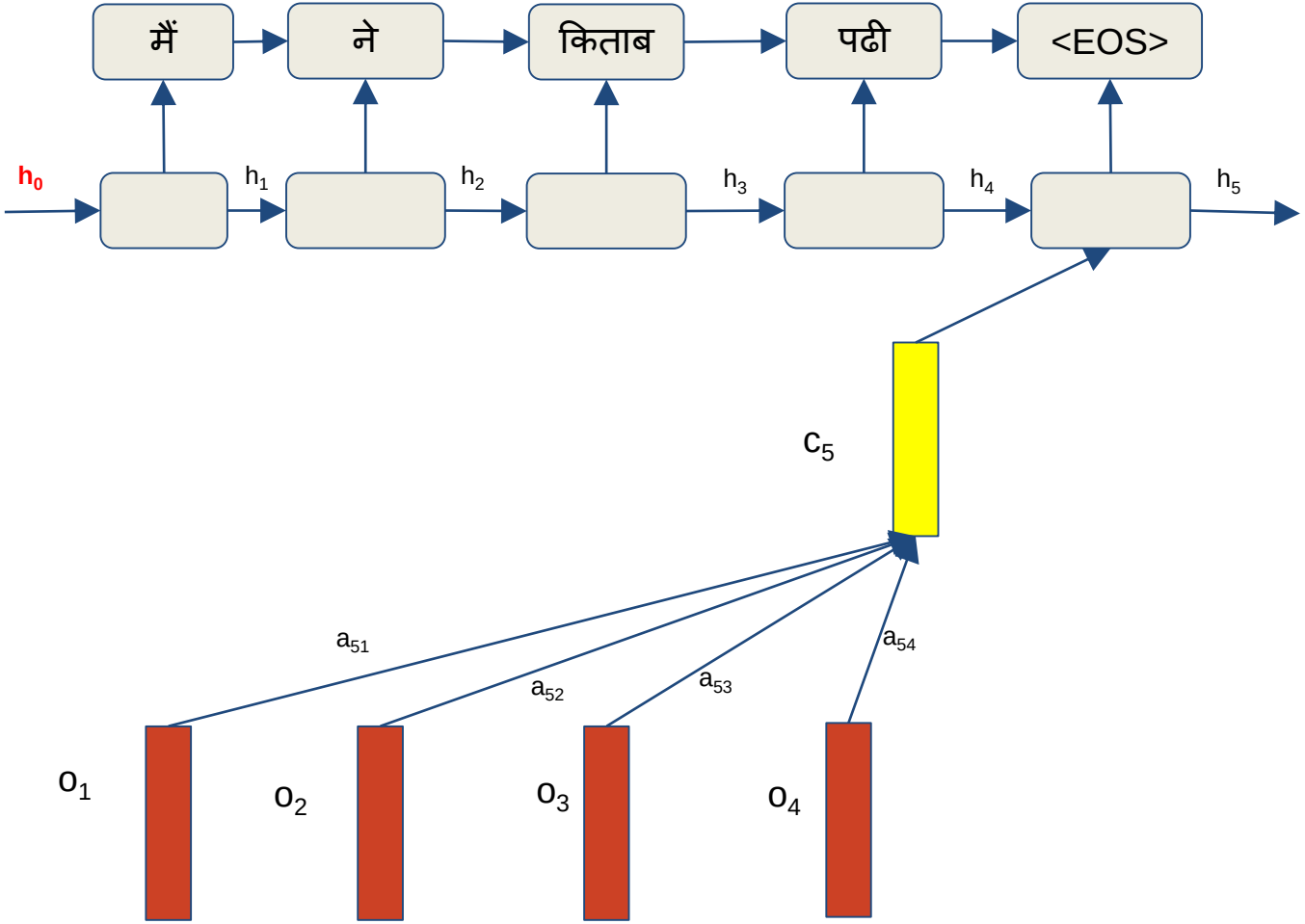
$$c_i = \sum_{j=1}^{n} a_{ij} o_j$$

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
h0
h1
h2
h3
c3
a31
a32
a33
a34
o1
o2
o3
o4

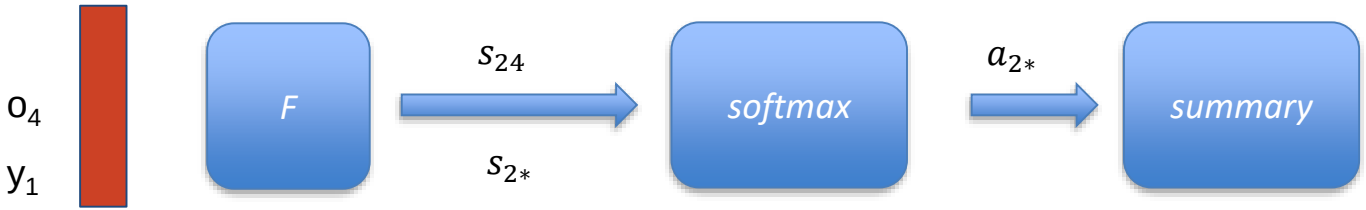

मैं
ने
किताब
पढी
h0
h1
h2
h3
h4
c4
a41
a42
a43
a44
o1
o2
o3
o4


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)



$$s_{ij} = F(h_{i-1}, O_j, y_{i-1})$$

$$a_{ij} = \frac{e^{s_{ij}}}{\sum_{k=1}^{n} e^{s_{ik}}}$$

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

$$P(y_i | y_{i-1} \dots y_1) = LSTM(h_{i-1}, c_i)$$

Loss: average NLL over sequence

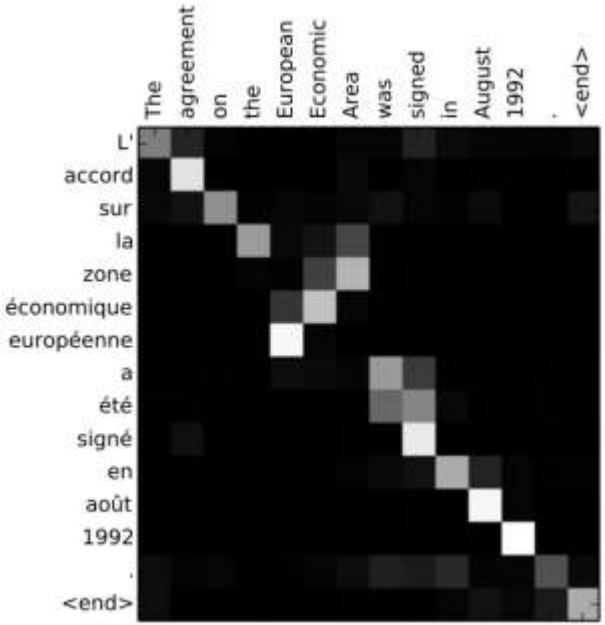
Exposure bias: training on ***true*** history, decoding on ***generated*** history

Attention is a powerful idea!

Attention has been the single biggest advance in helping NMT systems surpass SMT

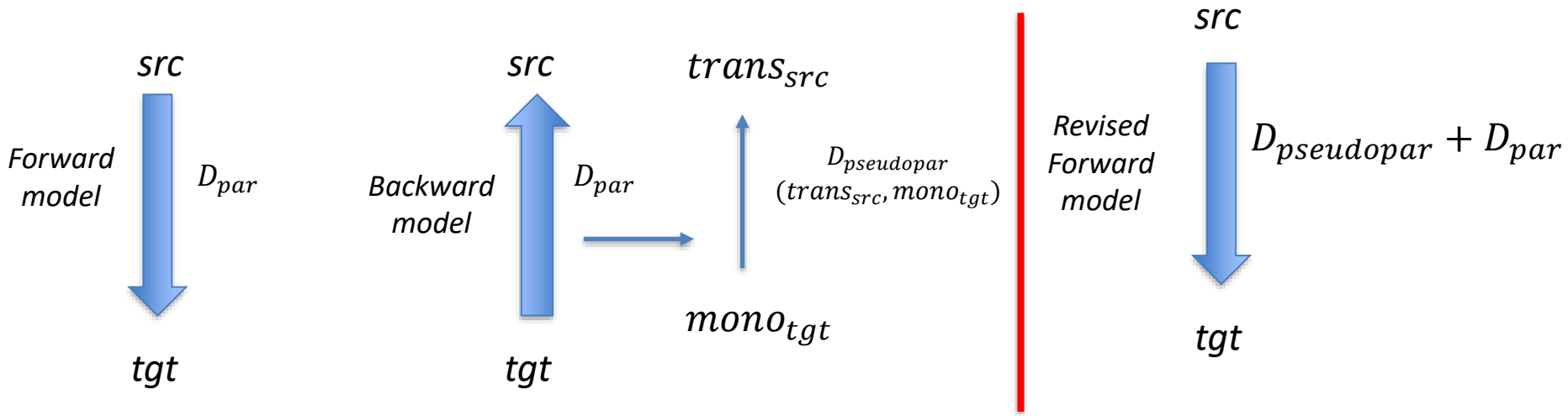
Attention can capture some sort of alignment

Attention is a ***general Deep Learning Technique***



Backtranslation

- NMT does not use monolingual data → decoder is a source-conditioned LM
- Utilizing monolingual data could improve target side fluency
- How to incorporate monolingual data? ➔ Backtranslation



Acts as a regularizer, very useful for low-resource language pairs

Benefits of NMT

- *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*
- *End-to-end training*

Evaluation of MT output

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

Evaluation is a problem for most natural language generation issue

MT can provide some solutions

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?

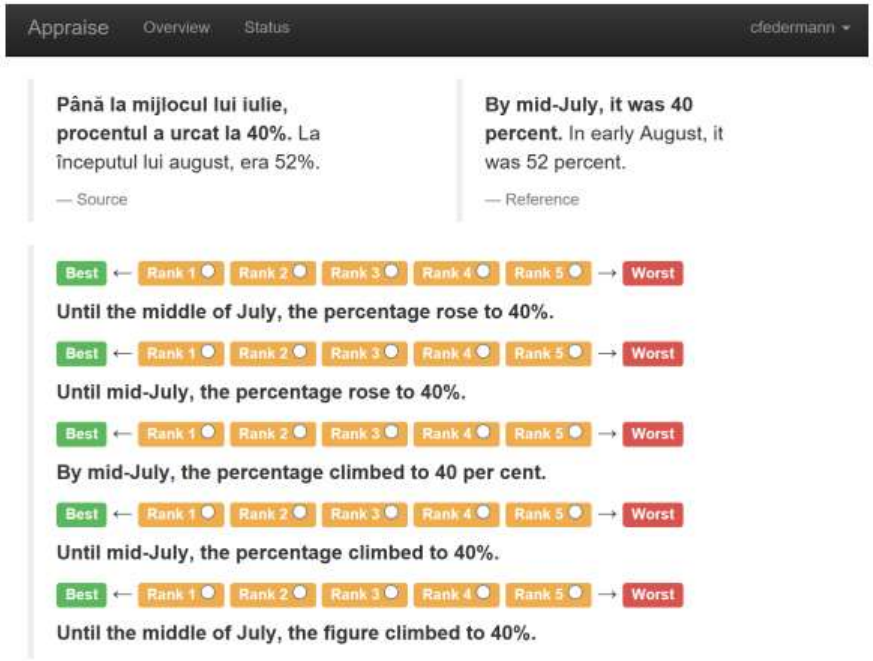
5 = All
4 = Most
3 = Much
2 = Little
1 = None

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

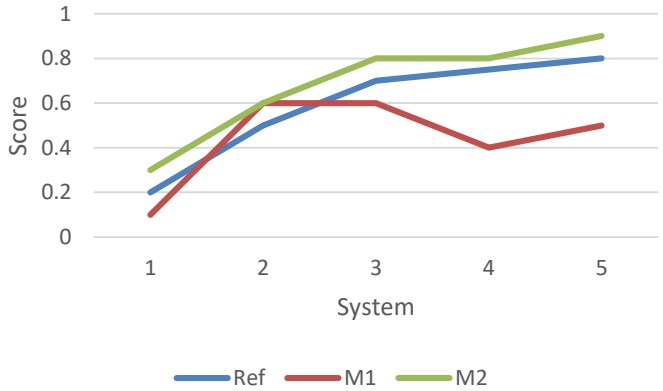
Multiple references are better

Some popular automatic evaluation metrics

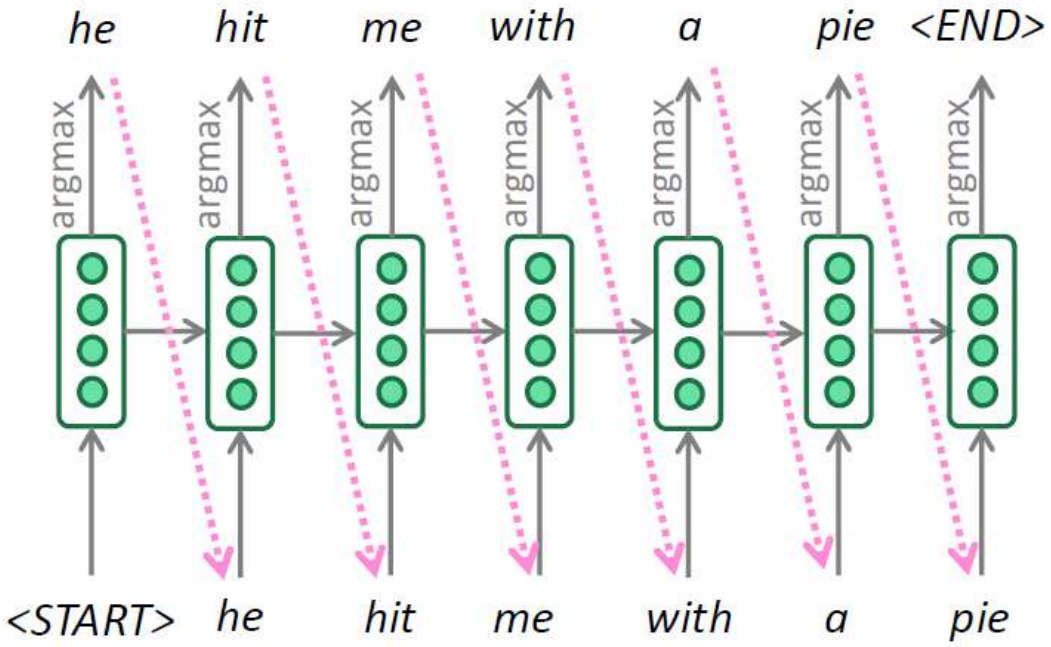
- **BLEU (Bilingual Evaluation Understudy)**
- TER (Translation Edit Rate)
- METEOR (Metric for Evaluation of Translation with Explicit Ordering)

How good is an automatic metric?

How well does it correlate with human judgment?

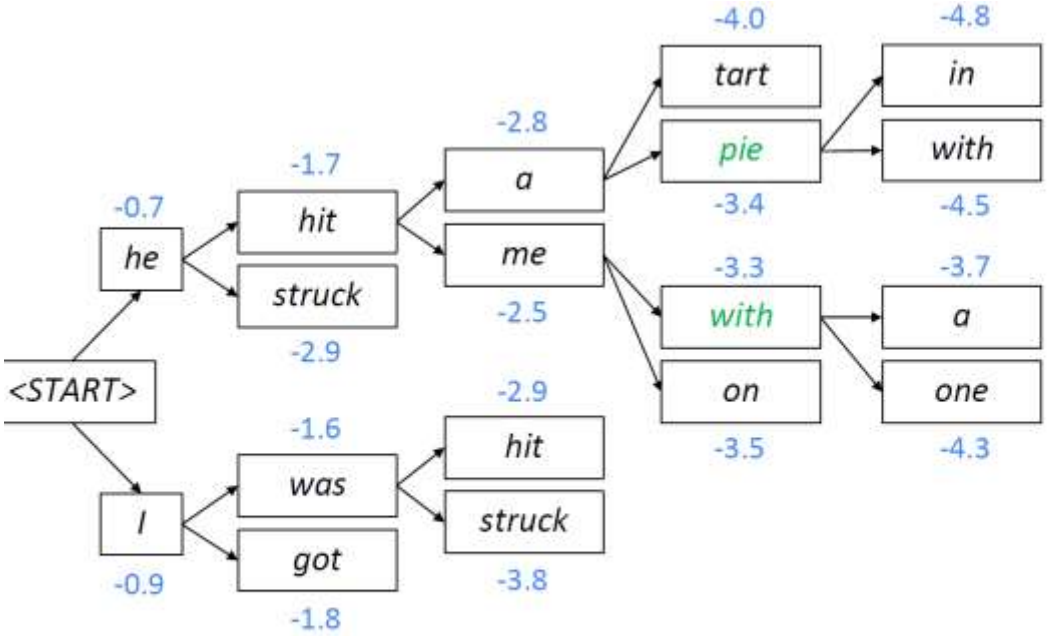


Greedy Decoding



- Like a multi-class decision rule at every time step
- Simple
- May not result in optimal output over entire sequence

Beam Search



For each of the *k* hypotheses, find top *k* next words and calculate scores

Software

- *Moses*: default toolkit for SMT + many utilities
- *FairSeq*: Wide variety of models, based on PyTorch, from FB
- *OpenNMT*: Open-source PyTorch, TF and Torch modular architecture
- *tensor2tensor*: tensorflow-based implementation from Google
- *Marian*: fast C++ implementation used by Microsoft

Datasets and Shared Tasks

- EuroParl
- UN Corpora
- TED talks
- OpenSubtitles

Look at the Opus Repository for many translation datasets

<u>Indian languages</u>

- Indian Language Corpora Initiative
- IIT Bombay English-Hindi Parallel corpus
- Charles Univesity English-Hindi Parallel corpus

Reading Material

SMT Tutorials

- *Machine Learning for Machine Translation (An Introduction to Statistical Machine Translation)*. **Tutorial at ICON 2013** with Prof. Pushpak Bhattacharyya, Piyush Dungarwal and Shubham Gautam. [slides] [handouts]
- *Machine Translation: Basics and Phrase-based SMT*. **Talk at the Ninth IIIT-H Advanced Summer School on NLP (IASNLP 2018), IIIT Hyderabad** . [pdf] [pptx]
- Text Book: Machine Translation. Philipp Koehn

NMT Tutorial

- Graham Neubig: https://arxiv.org/abs/1703.01619

QUESTION ANSWERING

We used to get 10 blue links to questions

who is the prime minister of india

List of Prime Ministers of India - Wikipedia

Prime Minister of India - Wikipedia

Prime Minister of India

List of Prime Ministers of India from 1947-2019 All PM Details, ...

List of prime ministers of India | Britannica.com

Now we are moving towards getting exact answers

who is the prime minister of india

India · Prime Minister

Narendra Modi

who is the finance minister of india

List of Finance ministers of India

- R.K.Shanmukham Chetty.
- John Mathai.
- Chintamanrao Deshmukh.
- Jawaharlal Nehru.
- T.T.Krishnamachari.
- ... *(more items)*

List of Finance ministers of India - Banking Awareness

Minister of Finance (India) - Wikipedia

List of Finance Ministers of India From 1947 Till Date

All Finance Ministers of India 2019 - getelectionresult.com

Question Answering

Query

who is the prime minister of india

Context

Answer

India · Prime Minister

Narendra Modi

Question Answering as a test of general Natural Language Understanding
Almost any problem can be cast as a question answering problem

Open Context/Domain

Query

who is the prime minister of india

A large collection of documents, databases, etc

Context

Open Domain QA

Answer

India · Prime Minister

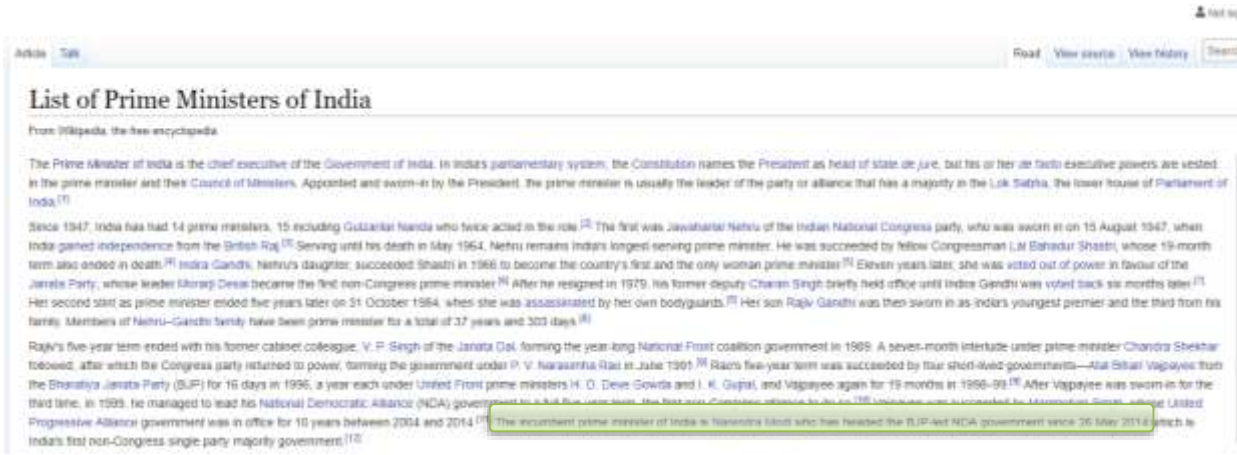
Narendra Modi

Data from various sources have to be aggregated
A lot of world knowledge may be required

Closed Context

Query

who is the prime minister of india



Machine Reading/Comprehension

Answer

India · Prime Minister

Narendra Modi

- *Question can be answered only from the small context (document/paragraph) provided*
- *A truer test of Natural Language Understanding*

Many language skills required for reading comprehension

Task 1: Single Supporting Fact
Mary went to the bathroom.
John moved to the hallway.
Mary travelled to the office.
Where is Mary? A:office

Task 3: Three Supporting Facts
John picked up the apple.
John went to the office.
John went to the kitchen.
John dropped the apple.
Where was the apple before the kitchen? A:office

Task 5: Three Argument Relations
Mary gave the cake to Fred.
Fred gave the cake to Bill.
Jeff was given the milk by Bill.
Who gave the cake to Fred? A: Mary
Who did Fred give the cake to? A: Bill

Task 7: Counting
Daniel picked up the football.
Daniel dropped the football.
Daniel got the milk.
Daniel took the apple.
How many objects is Daniel holding? A: two

Task 9: Simple Negation
Sandra travelled to the office.
Fred is no longer in the office.
Is Fred in the office? A:no
Is Sandra in the office? A:yes

Task 11: Basic Coreference
Daniel was in the kitchen.
Then he went to the studio.
Sandra was in the office.
Where is Daniel? A:studio

Task 13: Compound Coreference
Daniel and Sandra journeyed to the office.
Then they went to the garden.
Sandra and John travelled to the kitchen.
After that they moved to the hallway.
Where is Daniel? A: garden

Task 15: Basic Deduction
Sheep are afraid of wolves.
Cats are afraid of dogs.
Mice are afraid of cats.
Gertrude is a sheep.
What is Gertrude afraid of? A:wolves

Task 17: Positional Reasoning
The triangle is to the right of the blue square.
The red square is on top of the blue square.
The red sphere is to the right of the blue square.
Is the red sphere to the right of the blue square? A:yes
Is the red square to the left of the triangle? A:yes

Task 19: Path Finding
The kitchen is north of the hallway.
The bathroom is west of the bedroom.
The den is east of the hallway.
The office is south of the bedroom.
How do you go from den to kitchen? A: west, north
How do you go from office to bathroom? A: north, west

Machine Comprehension is a building block for Open-domain QA

A large collection of documents, databases, etc

Information Retrieval

Query

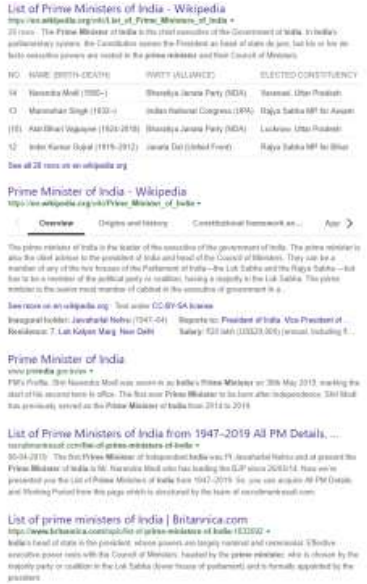
who is the prime minister of india

Machine Reading at scale

India · Prime Minister

Narendra Modi

Machine Reading



Answer

We will focus on machine comprehension

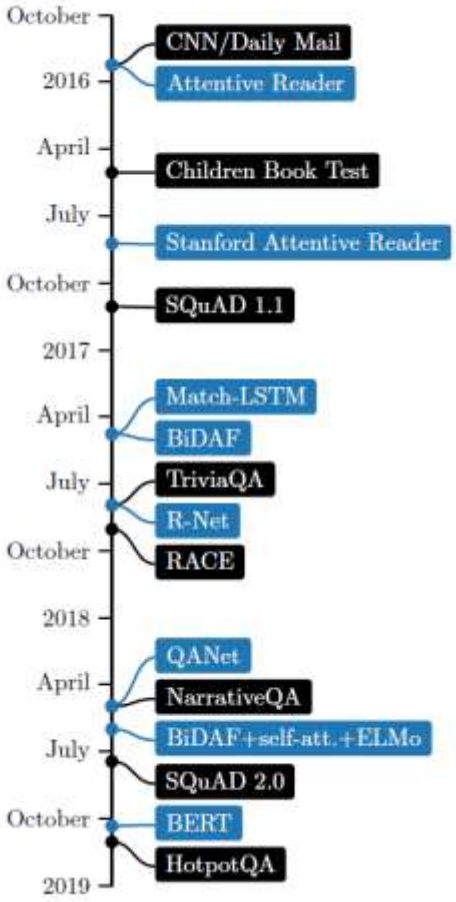
Major Trends in Machine Comprehension

Increase in size and diversity of training data

Supervised learning

+

Sophisticated language representation



Different Kinds of Machine Comprehension tasks

CNN/Daily Mail (cloze style)

passage: (@entity4) if you feel a ripple in the force today , it may be the news that the official @entity6 is getting its first gay character . according to the sci-fi website @entity9 , the upcoming novel " @entity11 " will feature a capable but flawed @entity13 official named @entity14 who " also happens to be a lesbian . " the character is the first gay figure in the official @entity6 – the movies , television shows , comics and books approved by @entity6 franchise owner @entity22 – according to @entity24 , editor of " @entity6 " books at @entity28 imprint @entity26 .

question: characters in " _____ " movies have gradually become more diverse

answer: @entity6

SQuAD (span prediction)

passage: Super Bowl 50 was an American football game to determine the champion of the National Football League (NFL) for the 2015 season. The American Football Conference (AFC) champion Denver Broncos defeated the National Football Conference (NFC) champion Carolina Panthers 24–10 to earn their third Super Bowl title. The game was played on February 7, 2016, at Levi's Stadium in the San Francisco Bay Area at Santa Clara, California. As this was the 50th Super Bowl, the league emphasized the "golden anniversary" with various gold-themed initiatives, as well as temporarily suspending the tradition of naming each Super Bowl game with Roman numerals (under which the game would have been known as "Super Bowl L"), so that the logo could prominently feature the Arabic numerals 50.

question: Which NFL team won Super Bowl 50?

answer: Denver Broncos

MCTest (multiple choice)

passage: Once upon a time, there was a cowgirl named Clementine. Orange was her favorite color. Her favorite food was the strawberry. She really liked her Blackberry phone, which allowed her to call her friends and family when out on the range. One day Clementine thought she needed a new pair of boots, so she went to the mall. Before Clementine went inside the mall, she smoked a cigarette. Then she got a new pair of boots. She couldn't choose between brown and red. Finally she chose red, which the seller really liked. Once she got home, she found that her red boots didn't match her blue cowgirl clothes, so she knew she needed to return them. She traded them for a brown pair. While she was there, she also bought a pretzel from Auntie Anne's.

question: What did the cowgirl do before buying new boots?

hypothesized answers: A. She ate an orange B. She ate a strawberry C. She called her friend D. She smoked a cigarette

answer: D. She smoked a cigarette

NarrativeQA (free-form text)

passage: ... In the eyes of the city, they are now considered frauds. Five years later, Ray owns an occult bookstore and works as an unpopular children s entertainer with Winston; Egon has returned to Columbia University to conduct experiments into human emotion; and Peter hosts a pseudo-psychic television show. Peter's former girlfriend Dana Barrett has had a son, Oscar, with a violinist whom she married then divorced when he received an offer to join the London Symphony Orchestra. . .

question: How is Oscar related to Dana?

answer: He is her son

The SQuAD 1.X dataset

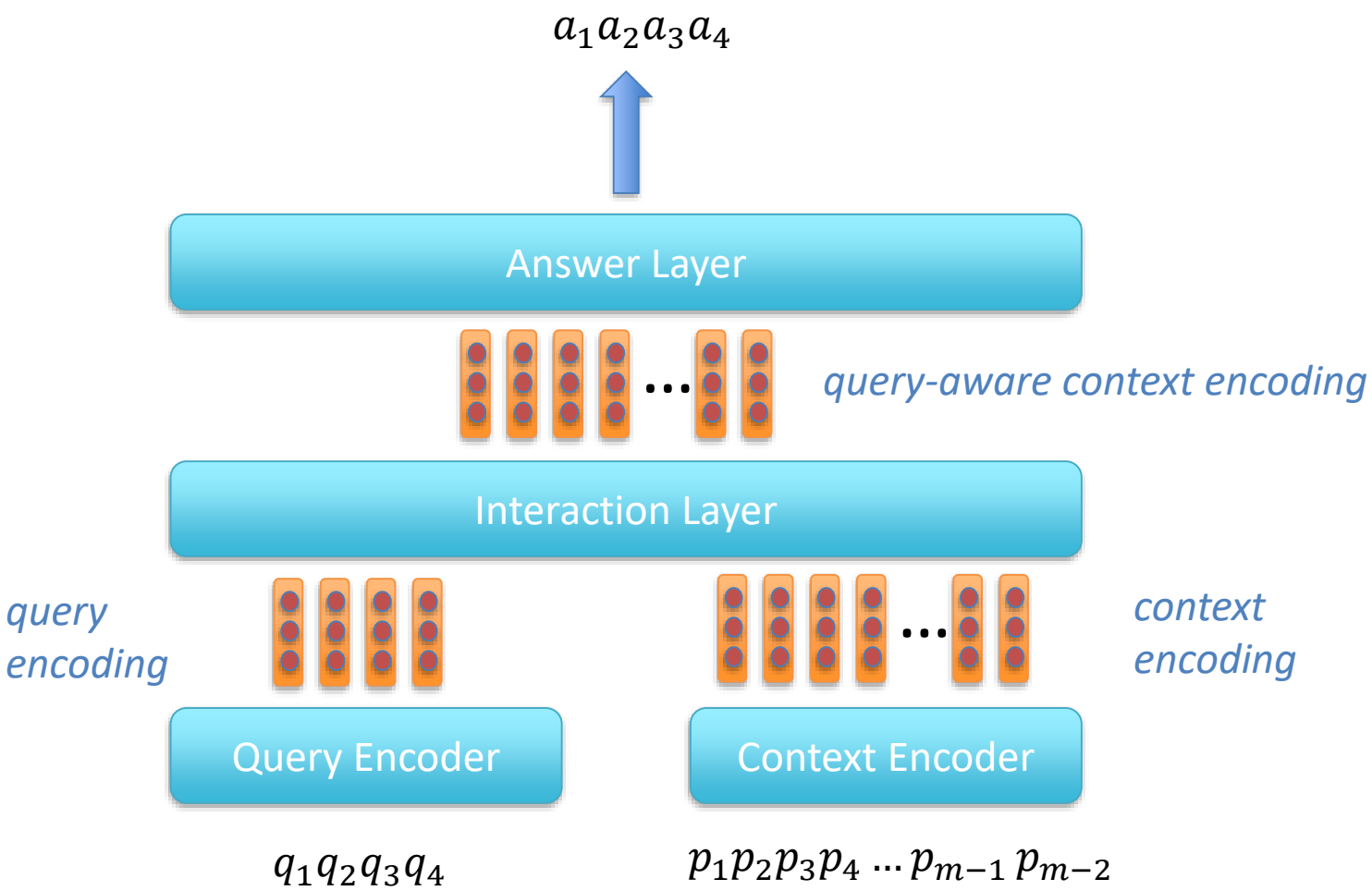
*The **S**tanford **QU**estion **A**nswering **D**ataset*

- *Questions created from Wikipedia articles by crowd-workers*
- *Diverse answer types*
- *Diversity in syntactic divergence*
- *Span based answers makes evaluation easier, yet flexible like free-form answers*
- *Provides human performance for comparison*

Cons

- *Questions not natural and independent of the paragraph → Natural Questions, TriviaQA*
- *Answers in a single span mostly → HotpotQA, Qangaroo, ComplexWebQuestions*
- *Can be solved well by context and type-matching heuristics*

One of the most popular and wildly reported MR datasets


$a_1 a_2 a_3 a_4$
Answer Layer
query-aware context encoding
Interaction Layer
query encoding
context encoding
Query Encoder
Context Encoder
$q_1 q_2 q_3 q_4$
$p_1 p_2 p_3 p_4 \cdots p_{m-1} p_{m-2}$

Question and Context Encoder

Standard embedding methods

Word (and possibly char-word embeddings) followed by

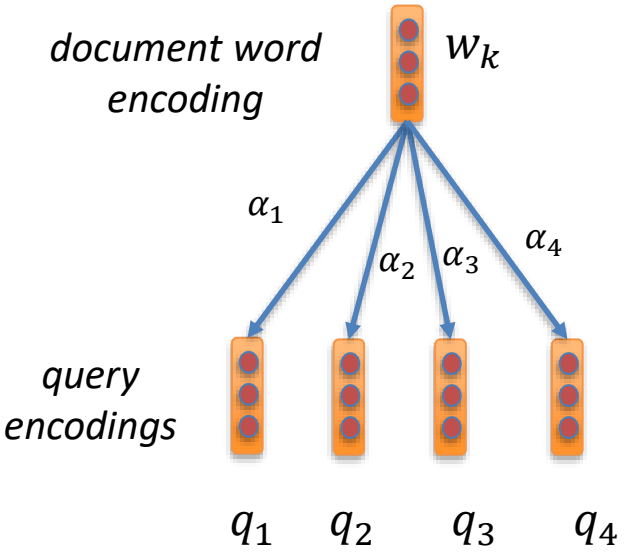
LSTM/bi-LSTM embeddings

Simplest interaction network

Just concat the last states of query and context encodings

- *Does not capture similarities between the query and context words*
- *Long range dependencies cannot be captured*

Attention-based Reader



Query summary corresponding to document word at position k

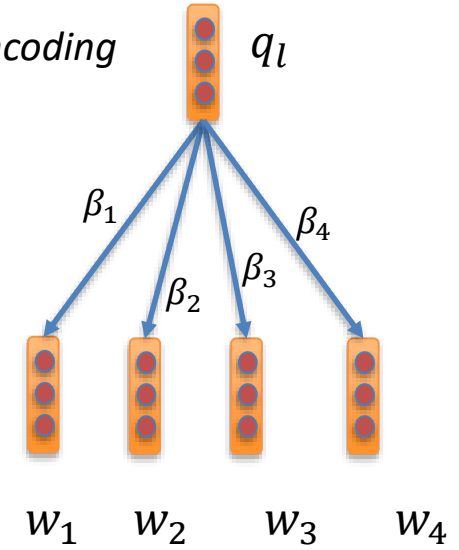
$$q^{(k)} = \sum_{i=1}^{i=4} \alpha_i q_i$$

Build a query-aware document representation

$$r^{(k)} = [w_k; q^{(k)}]$$

Co-attention based Reader

Also attend to the document for each query word



$$d^{(l)} = \sum_{i=1}^{i=4} \beta_i w_i$$

Document summary corresponding to query word at position k

Build a co-attention based document representation

$$r^{(k)} = [w_k; q^{(k)}; G(d^{(1)} \dots d^{(L)})]$$

Some of the best MC models use some kind of co-attention/bidirectional attention flow

Memory Networks

Methods so far: Look at the query and context once

We may want to refine our query and context representations

Different parts of the context may be attended to in later iterations

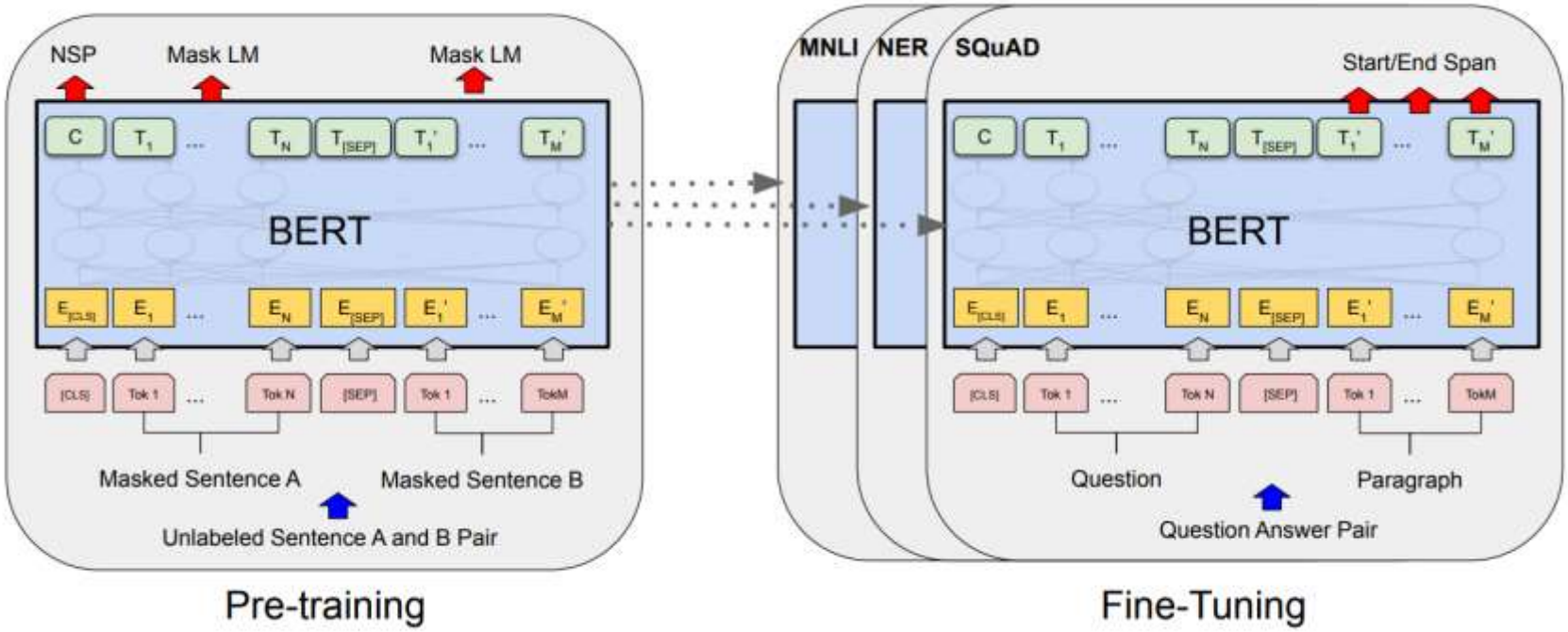
Memory networks: generation of attention networks (with multiple hops)

A more general idea:

- can also write to memory networks – useful in some problems

The BERT Revolution

Self-Attention + Co-attention

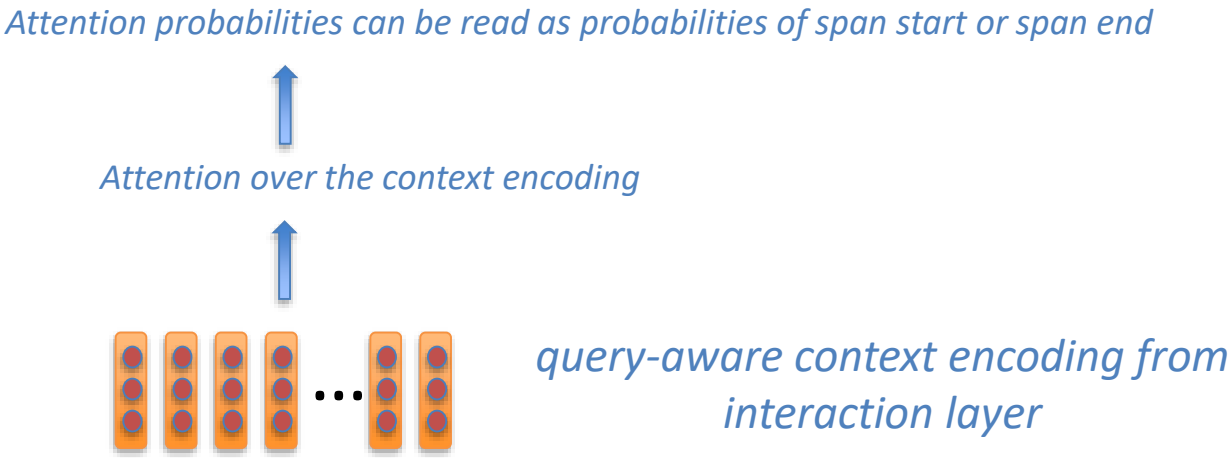


Pointer Networks for Span Identification

For datasets like SQuAD, the answer is a span

The span is defined by a (start_position, end_position) tuple

Answer Layer predicts this tuple using Pointer networks



Does the question have an answer?

For open-domain QA – important to identify that document does not have answer

MC systems trained on datasets always having answers will provide junk answers

SQuAD 2.0: Incorporates no answer questions and plausible answers

Article: Endangered Species Act
Paragraph: *" . . . Other legislation followed, including the Migratory Bird Conservation Act of 1929, a 1937 treaty prohibiting the hunting of right and gray whales, and the Bald Eagle Protection Act of 1940. These later laws had a low cost to society—the species were relatively rare—and little opposition was raised."*

Question 1: *"Which laws faced significant opposition?"*
Plausible Answer: *later laws*

Question 2: *"What was the name of the 1937 treaty?"*
Plausible Answer: *Bald Eagle Protection Act*

How to detect no-answers?

- Span size=0
- Special *no-option* token in input
- Special *no-option* output

Reading Comprehension Datasets

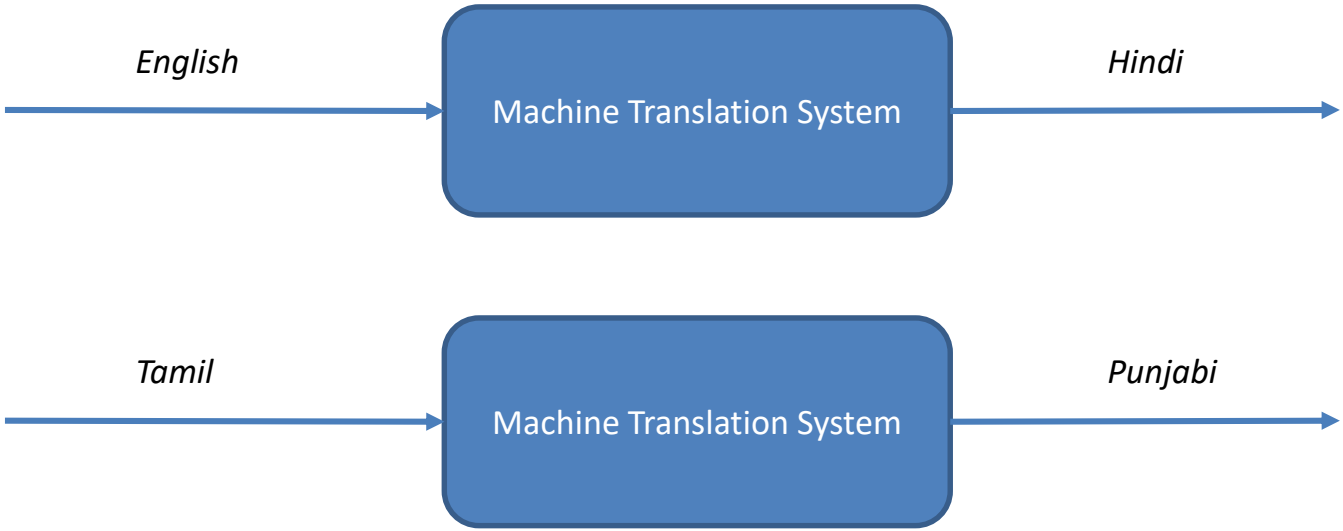
- Deep Read (Hirschmann 1999 et al.)
- MCTest
- CNN/Daily Mail
- SQuAD 1.x
- SQuAD 2.0
- WikiQA
- TriviaQA
- HotPotQA
- Natural Questions

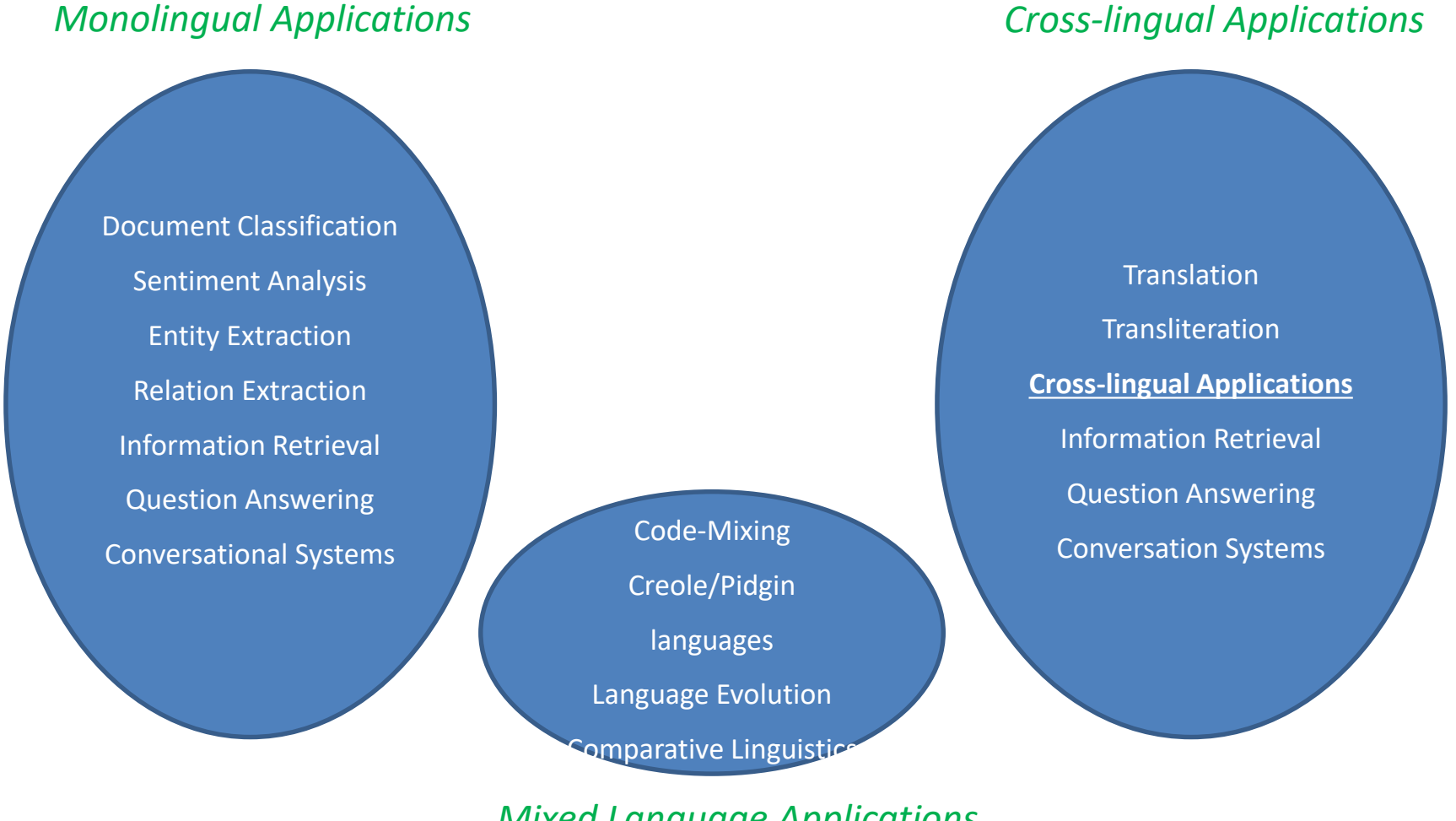
Reading Comprehension Software

- AllenNLP’s BiDAF
- BERT

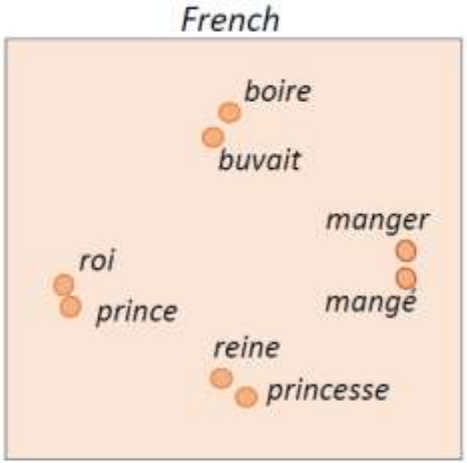
MULTILINGUAL NLP

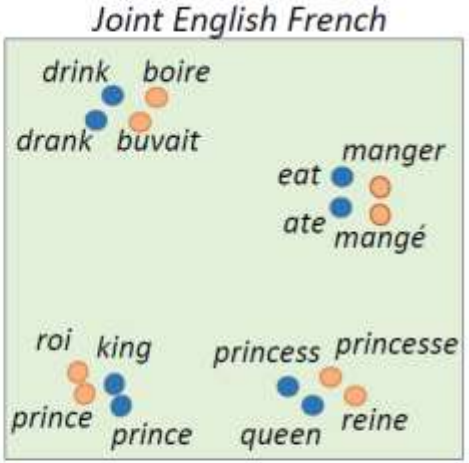
Broad Goal: Build NLP Applications that can work on different languages




Monolingual Applications
Document Classification
Sentiment Analysis
Entity Extraction
Relation Extraction
Information Retrieval
Question Answering
Conversational Systems
Cross-lingual Applications
Translation
Transliteration
Cross-lingual Applications
Information Retrieval
Question Answering
Conversation Systems
Code-Mixing
Creole/Pidgin
languages
Language Evolution
Comparative Linguistics
Mixed Language Applications

Cross Lingual Embeddings





Monolingual Word Representations
(capture syntactic and semantic similarities between words)

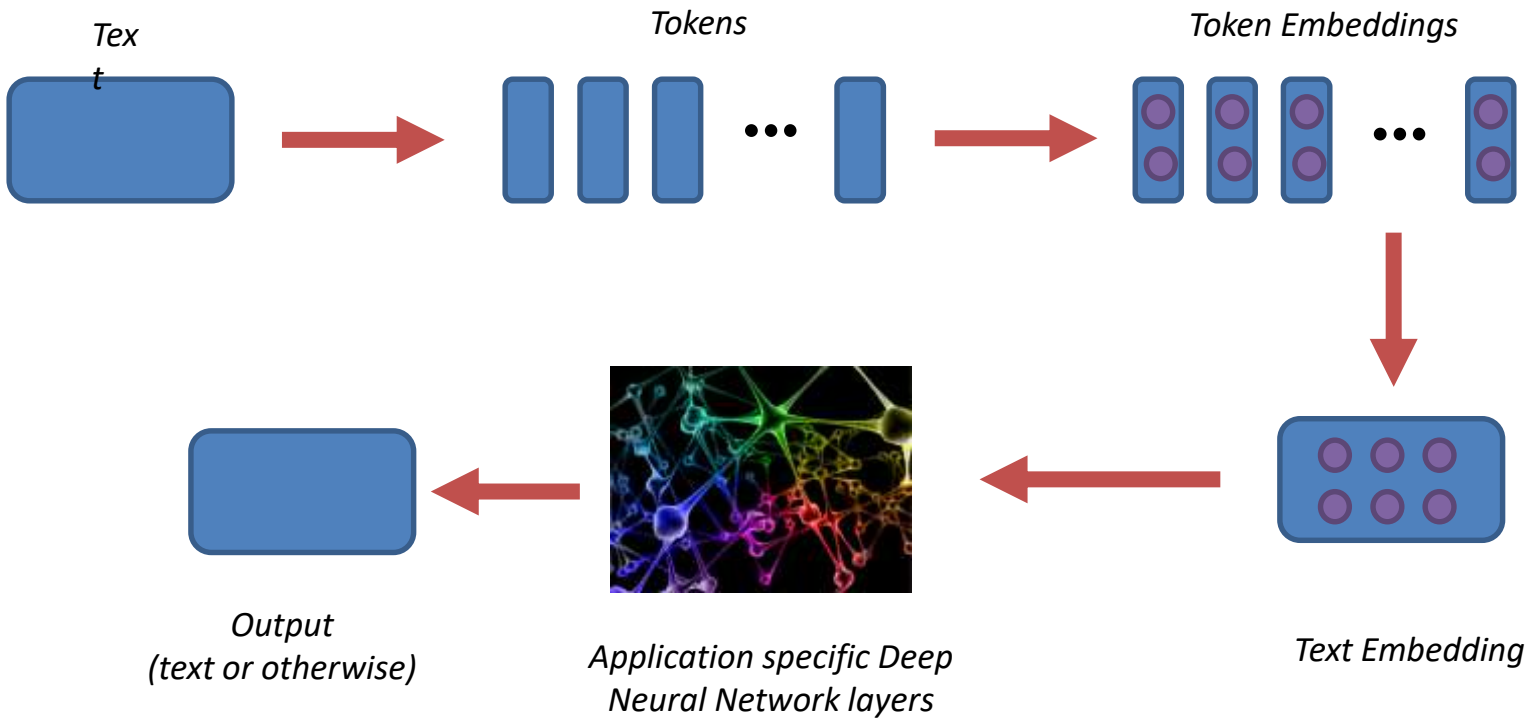
Multilingual Word Representations
(capture syntactic and semantic similarities between words both within and across languages)

$$embed(y) = f(embed(x))$$

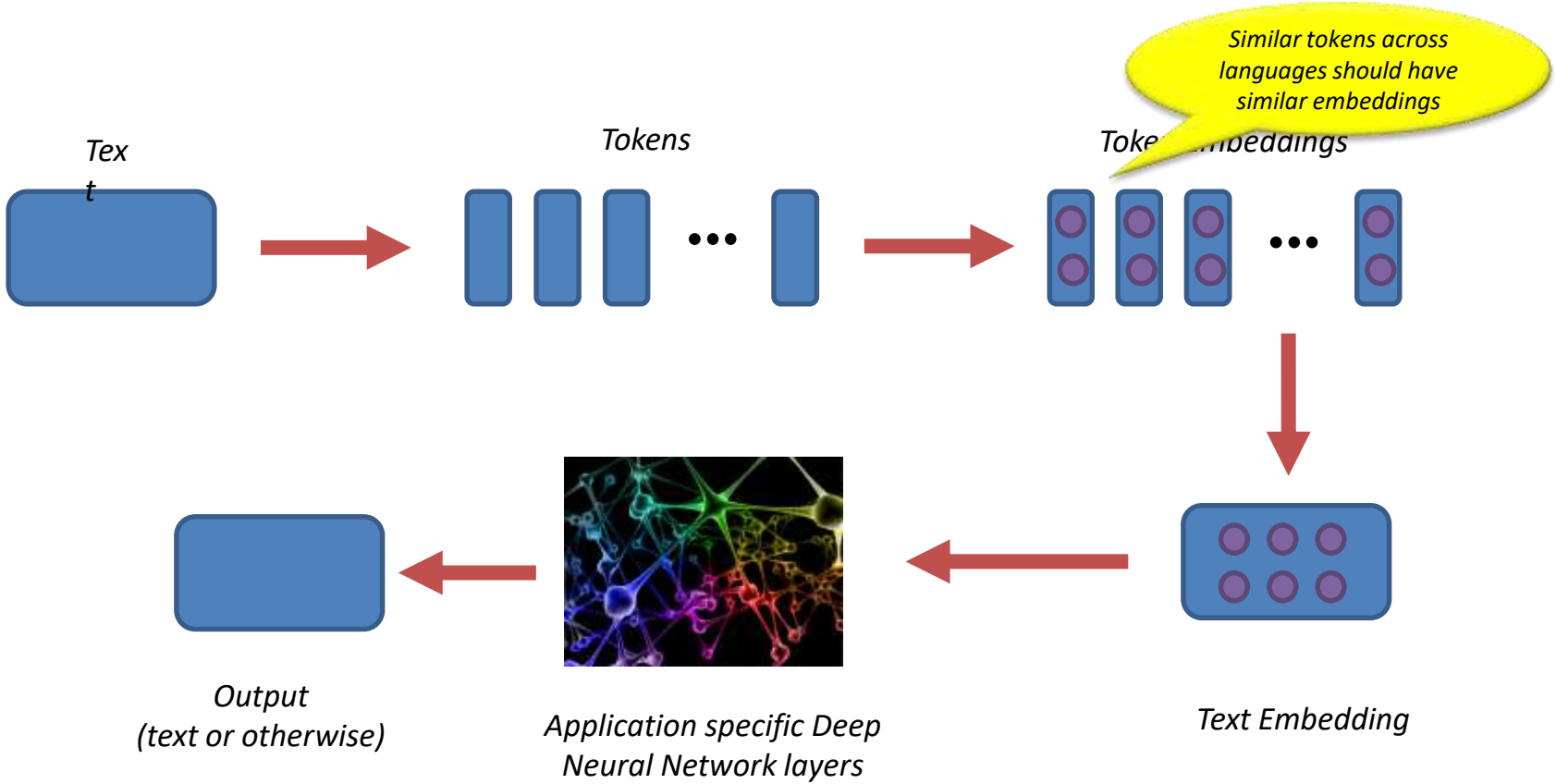
(Source: Khapra and Chandar, 2016)

x, y are source and target words
$embed(w)$: embedding for word w

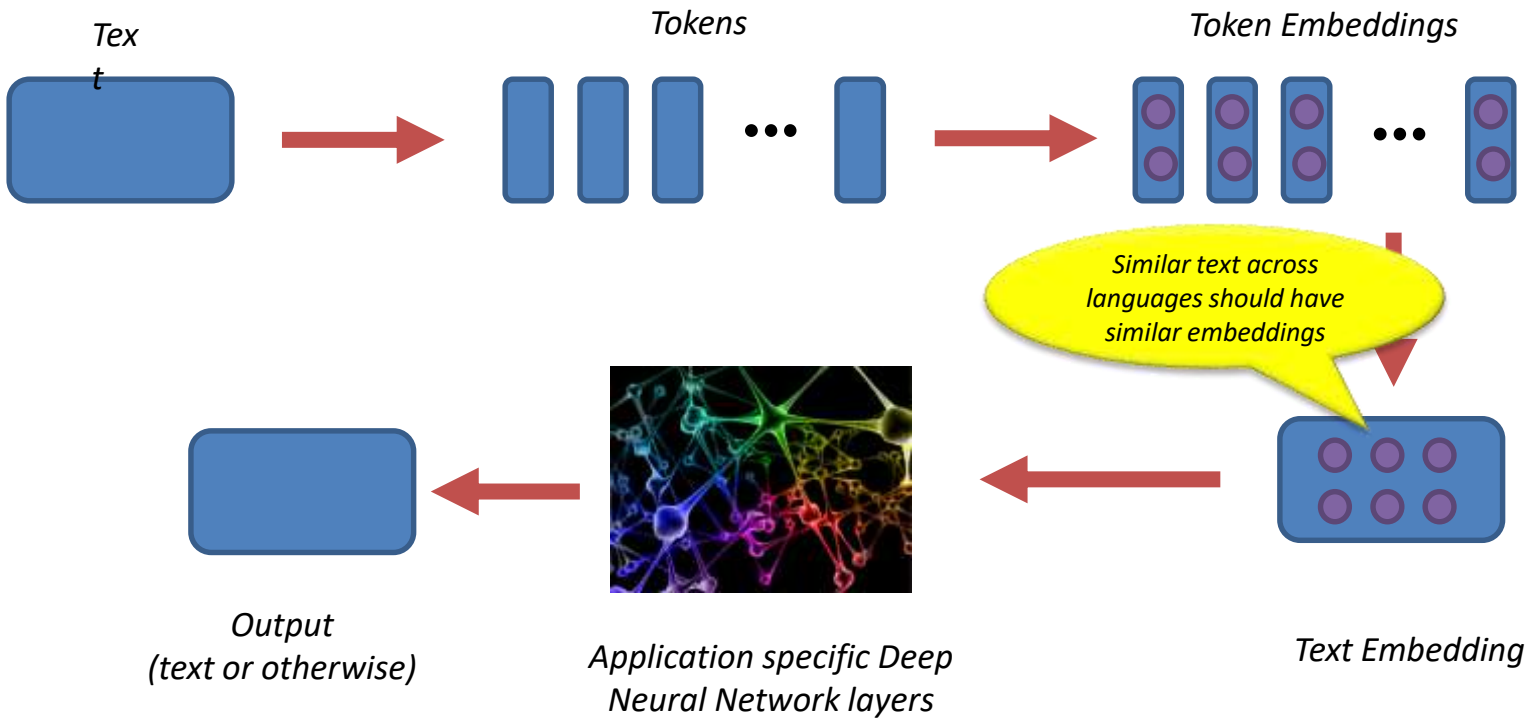
A Typical Multilingual NLP Pipeline



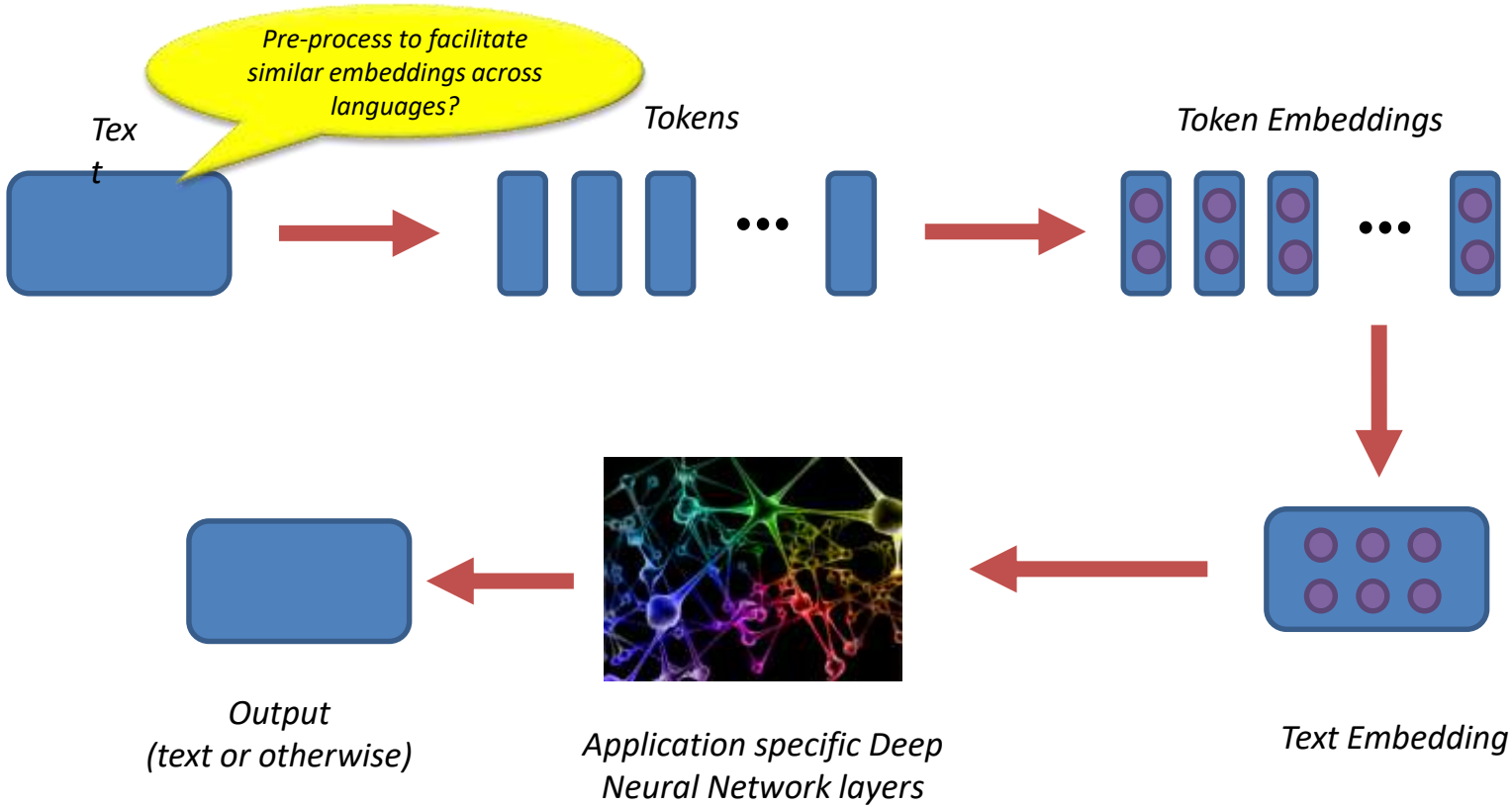
A Typical Multilingual NLP Pipeline



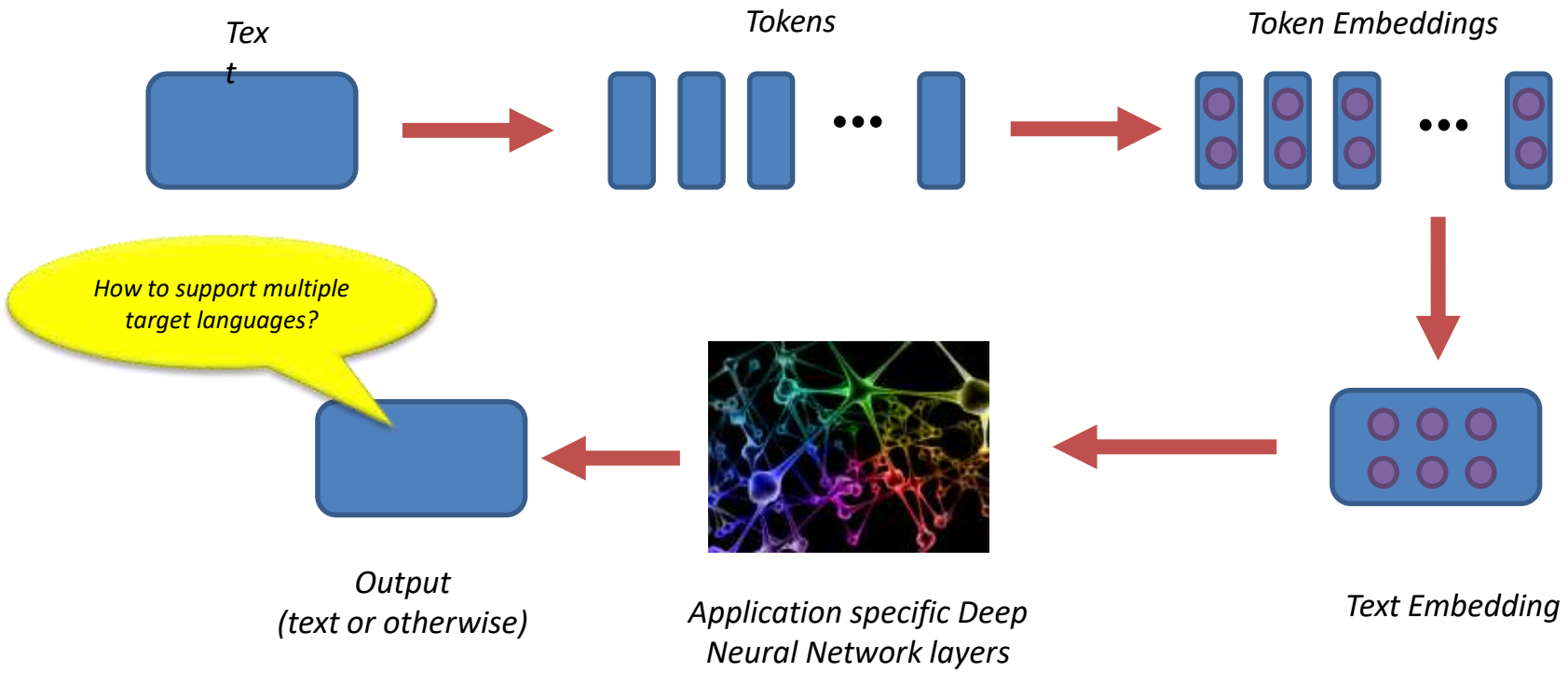
A Typical Multilingual NLP Pipeline



A Typical Multilingual NLP Pipeline



A Typical Multilingual NLP Pipeline



More Reading Material

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

Machine Translation for Related Languages

- *Statistical Machine Translation between related languages. Tutorial at NAACL 2016 with Prof. Pushpak Bhattacharyya and Mitesh Khapra.* [abstract] [slides]
- *Machine Translation for related languages. Tech Talk at AXLE 2018 (Microsoft Academic Accelerator).* [pdf] [pptx]
- *Translation and Transliteration between related languages. Tutorial at ICON 2015 with Mitesh Khapra.* [abstract] [slides] [handouts]

Multilingual Training

- *Cross lingual embeddings survey paper:* https://arxiv.org/abs/1706.04902
- *Multilingual Learning. Invited Talk at IIIT Hyderabad Machine Learning Summer School (Advances in Modern AI) 2018.* [slides]

READING MATERIAL AND RESOURCES

Reading Material

Text Books

- Speech and Language Processing. Dan Jurafsky and James Martin.
- The Language Instinct. Stenven Pinker.
- Linguistics. *Adrian Akmajian, et al.*

Online Courses

- CS224n by Chris Manning. http://web.stanford.edu/class/cs224n/

Software

- LingVo
- AllenNLP
- PyText

References

- Stephen Clark. Vector Space Models of Lexical Meaning. In The Handbook of Contemporary Semantic Theory (eds S. Lappin and C. Fox). 2015.
- Peter Turney, Patrick Pantel. From Frequency to Meaning: Vector Space Models of Semantics. JAIR. 2010.