AI Module: NLP

Deepak Subramani
Assistant Professor
Dept. of Computational and Data Science
Indian Institute of Science Bengaluru

Tabular
Time series
Image
Video
Text
Audio

→ modal

Input → $\boxed{}$ → Output

model

Predictive AI

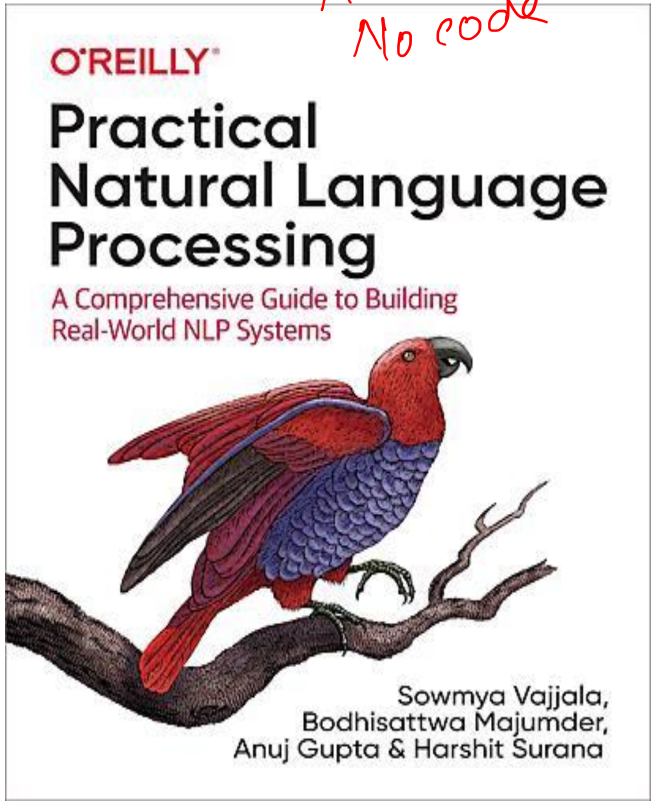
Generative AI

Department of Computational and Data Sciences

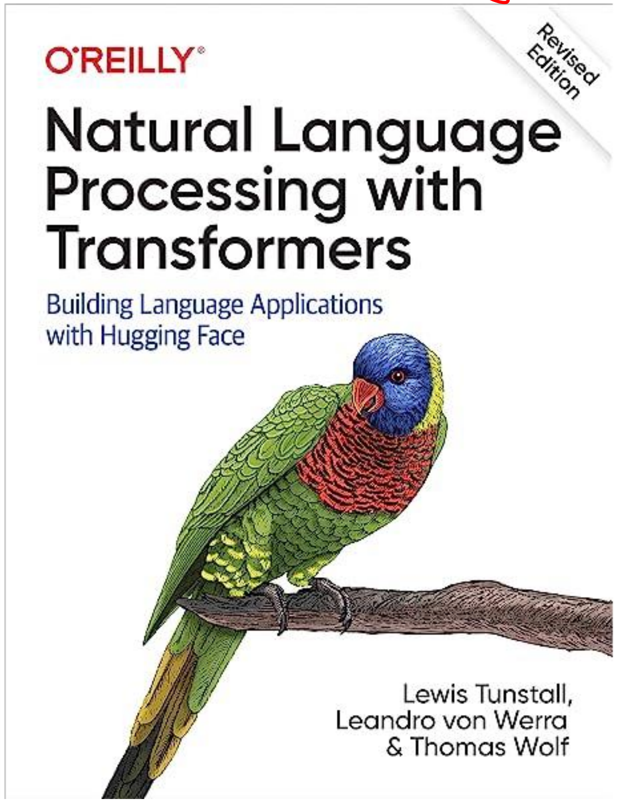
Reference Books

Simple

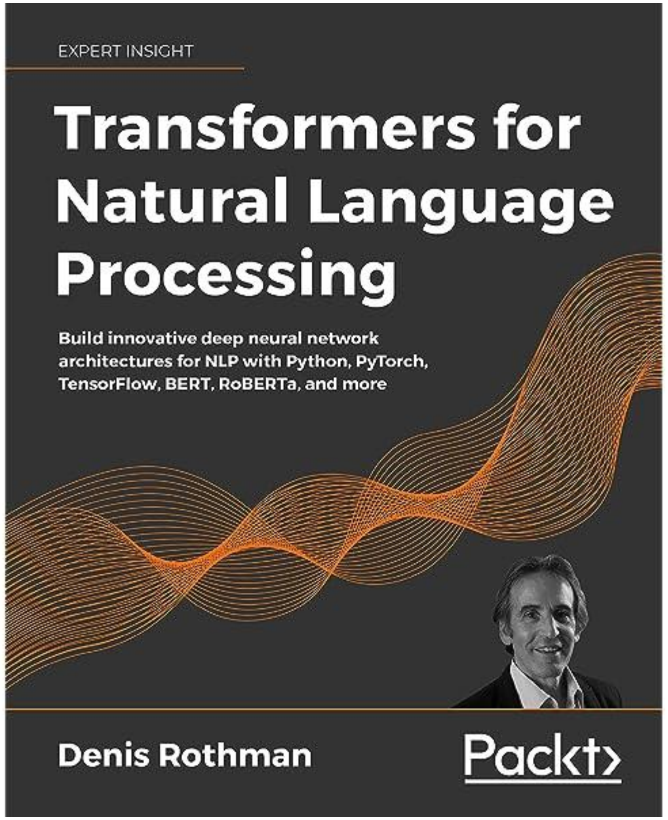
No math
No code



HF Library



Builds from scratch



Outline for Week01

- Part 01
  - Intro to NLP
  - Tasks in NLP
  - NLP Pipeline
  - Data Pre-processing, NLTK and Spacy
  - Two Representation Approaches
- Part 02
  - Text Vectorization
  - Word Embedding
  - Byte Pair Encoding Tokenization
  - Training Word Embedding

Outline for Week 02

- Part 01: Sequence Modeling
  - Essential tasks in sequence modeling
  - When to use a seq model?
  - Names of recurrent architectures and introduction (details in additional slide deck on RNN – not included in the main content)
  - Other Seq Tasks (Time Series Modeling; Not NLP)
  - Additional notebooks on RNN and LSTM (not compulsory)
- Part 02: Attention Mechanism and Transformer Encoders
  - Key, Query, Value
  - Multi Head Attention
  - MHA as text representation learners for use in discriminative tasks
  - Positional Embedding
  - Trick of the trade: Layer Normalization
  - Assignment on Attention Mechanism

Outline for Week 03

- Part 01: BERT Models for Discriminative Tasks (NLU)
  - BERT and friends – the essential foundational model for text tasks
  - How is BERT trained?
  - What can BERT be used for with examples
  - Assignment on BERT
- Part 02: Transformer Decoder
  - Intro to Neural Machine Translation
  - Masked attention
  - Understanding Decoder through animations
  - Assignment on decoder and NMT

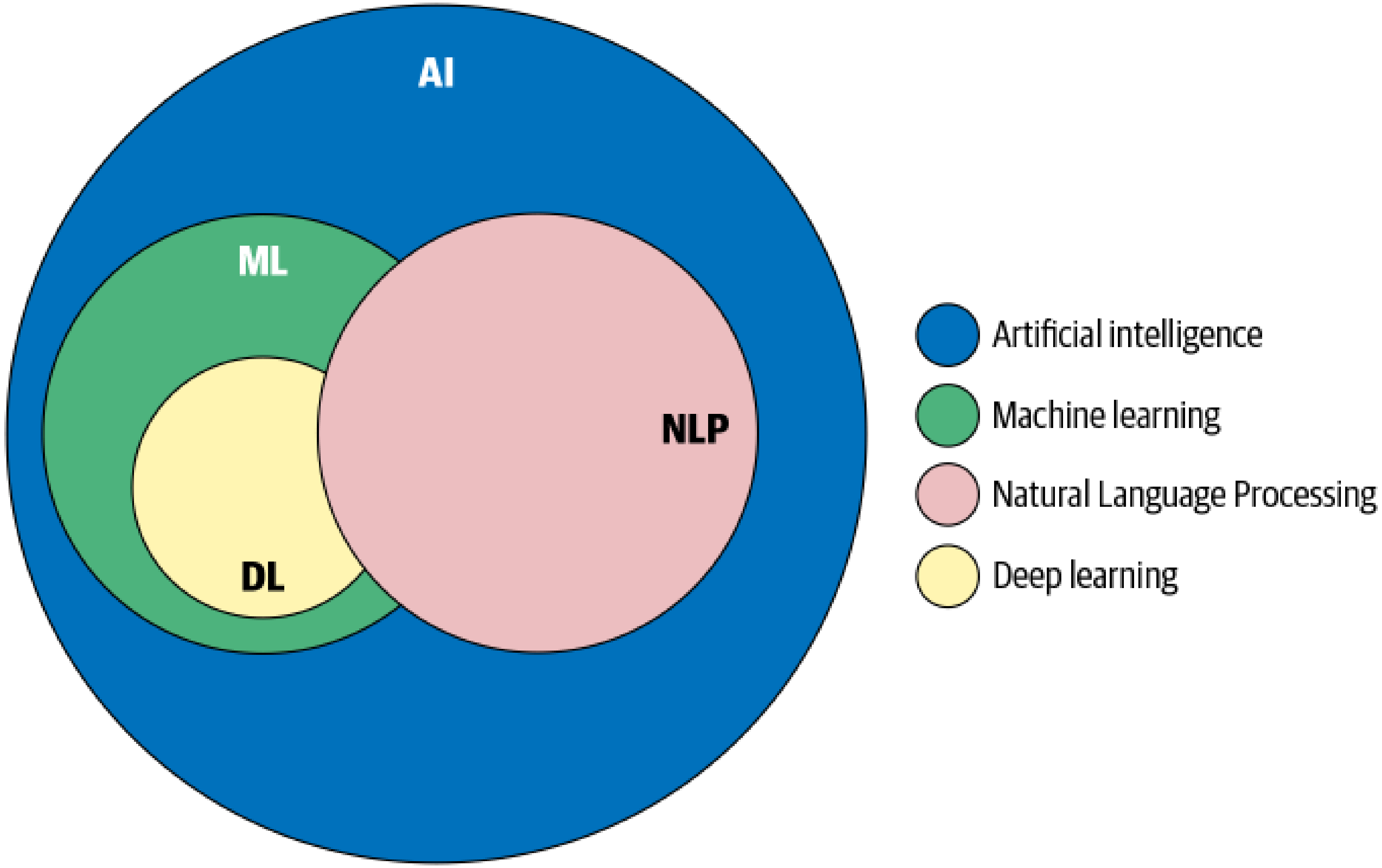
Let's meet a Virtual Assistant

- "Hey Siri. Show me a good breakfast restaurant near me."
- How does Siri process this query?
  1. Convert speech to text
  2. Understand the semantics of the question (e.g., understand keywords like breakfast, restaurant) and formulate a structured query (place type = "restaurant", good for="breakfast", rating 3-5, distance < 3 km)
     - It also needs your current location!
  3. Search for restaurants, filter by the structure above and rank based on ratings (or other metrics)
     - For rating, the system could use the star rating and the sentiments/points in the written review – another NLP task
  4. At the restaurant, it might translate the menu card from Kannada to English

Natural Language Processing

- C, C++ - Machine Languages
  - Starting point was a human writing rules
  - Usage is based on the rules
- Tamil, English - Natural Languages
  - Usage comes first
  - Rules came later
- Early NLP focused on formalizing rules and building systems based on it from an applied linguistics perspective
- Complex hand crafted rules sustained machine translation and chatbots
- From the late 1980s, the thinking of Machine Learning for NLP emerged
  - Can I use a text corpora and discover the rules of language?
- Through 1990s, formal linguist+engineer rule system was popular
- Growth of ML in NLP started with decision trees, logistic regression and SVM

Context


AI
ML
DL
NLP
Artificial intelligence
Machine learning
Natural Language Processing
Deep learning

NLP: Major Tasks

- Modern NLP – Goal is not to understand language, but to ingest a piece of language as input and return useful quantities
- A collection of fundamental tasks repeatedly come in NLP = NLU + NLG
- Natural Language Understanding NLU
  - "What is the topic of this text?" – Topic Modelling
  - "Is this text inappropriate?" – Content Filtering
  - "Is this text, positive, neutral or negative?" – Sentiment Analysis
  - Named Entity Recognition, Part of Speech Tagging
  - Information retrieval (Keyword based)

Naive Bayes Model

① Fill in the blank
② Predict next word → auto regressive

- Natural Language Generation
  - "What is the next word or character?" – Language Modeling, Sentence Completion
  - "What is "AI" in tamil?" – Machine Translation
  - "What is the crux of this paragraph?" – Text Summarization
  - Answer to "Where is the nearest hair salon?" – Question Answering

Seq to $\boxed{\text{Seq}}$
Text to Text

NLP: Major tasks

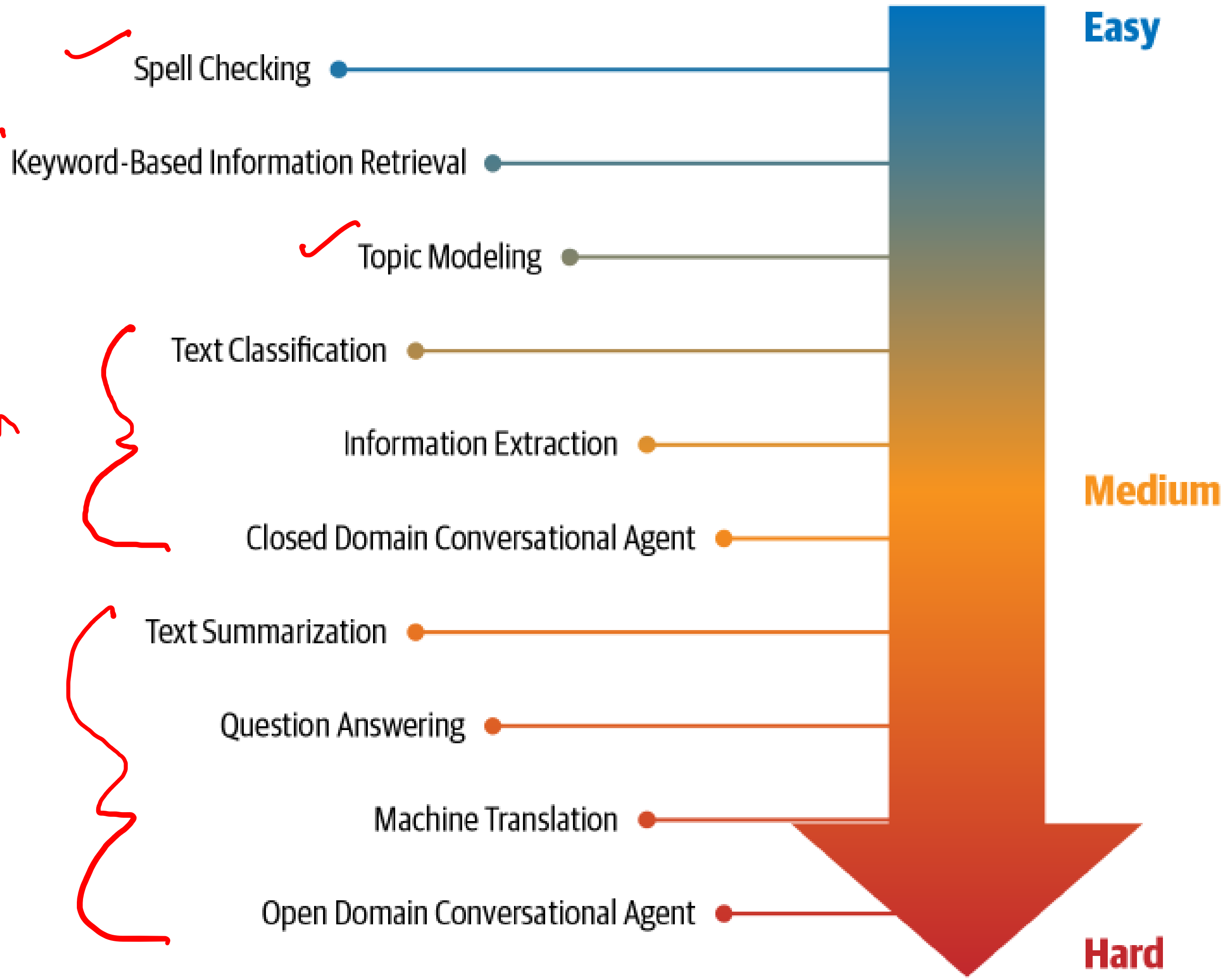
Easy

Medium

Langchain

GPT

Hard



Seq-to-Seq

C++ Python

Fig 1-2, Vajjala et al

Deepak Subramani, deepakns@iisc.ac.in

Top NLP Libraries

- NLTK - tokenization, lemmatization, stemming, parsing, POS tagging, etc. This library has tools for almost all NLP tasks. Supports large number of languages
  - Research standard
- Spacy – The main competitor for NLTK. These two libraries can be used for the same tasks. Limited language support NER
  - Industry standard
- Gensim – Topic and vector space modelling, document similarity
- Polyglot – similar to NLTK and has support for a large number of languages. But slow and not enough support.

Challenges in NLP

- Relationship between words and semantics is highly nonlinear
- Words need to be converted to a numerical representation that can be handled by computer algorithms
- Series of data processing must be done to even start the core NLP task
  - Text parsing, morphological analysis, word sense disambiguation, understanding grammatical structure
- Human language understanding relies on "common knowledge"
  - Example - A: "man bit dog" vs B:"dog bit man" – We all know that B is more likely than A

Example of Challenges

1

The man couldn't lift his son because he was so **weak**. → Who was weak?

The man couldn't lift his son because he was so **heavy**. → Who was heavy?

2

Mary and Sue are **sisters**.

Mary and Sue are **mothers**.

→ How are Mary and Sue related?

Joan made sure to thank Susan for all the help she had **received**. → Who had received help?

Joan made sure to thank Susan for all the help she had **given**. → Who had given help?

John **promised** Bill to leave, so an hour later he left.

John **ordered** Bill to leave, so an hour later he left.

→ Who left an hour later?

Issues with DL for NLP

- Overfitting on small datasets
  - Reliable few shot learning – Becoming better with LLMs
- Domain Adaptation
- Interpretability
- Common sense and world knowledge
- Cost
- Edge deployment

Word of Caution

- Good performance on NLP tasks should not be confused for human-level intelligence
- Text-processing models do not possess human-level understanding of language
- They are simply data processing pipelines that look for patterns in text and perform simple tasks (as in the previous slide) at scale
- Computer Vision – Pattern Recognition at pixel level
- Natural Language Processing – Pattern Recognition at word, sentence level (GPT – token level)

Poll – T or F

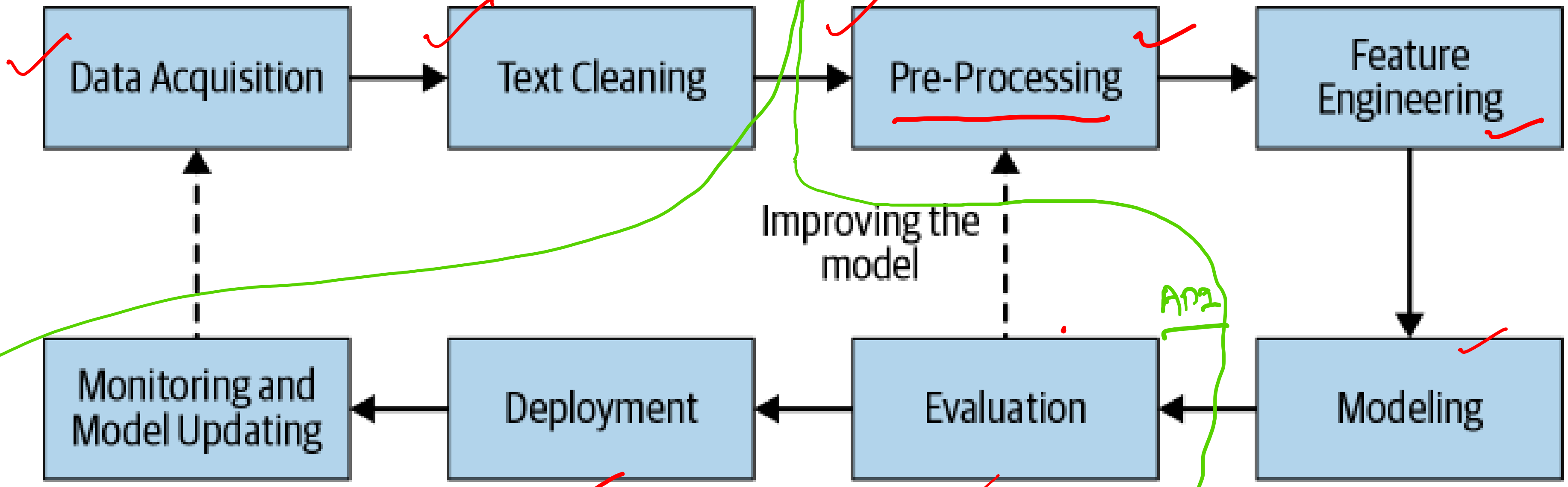
1. In Natural language, rules came first and usage next
    - True, False
2. Text Summarization is a Discriminative Task
    - True, False
3. All NLP tasks can be viewed as a generative or a discriminative task
    - True, False

Generic NLP Pipeline

OCR
SuryaOCR

NLTK

Sequence modelling approach
Bag of words modelling



LLMOps

Data Related Tasks

- Data Acquisition
- Text Extraction and Cleaning

11 am

Data Pre-Processing

- Preliminaries (in all projects) – sentence segmentation and word tokenization
- Frequent steps (in almost all projects) - Stop word removal, stemming and lemmatization, removing digits/punctuation, lowercasing, etc.
- Other steps (in some projects) - Normalization, language detection, code-mixing (mixing of 2 or more languages), transliteration, etc.
- Advanced processing (in specific applications) – POS tagging, parsing, coreference resolution

Building Blocks of Language: NLTK

NLTK – Natural Language Toolkit

Other options:
Spacy is the main competitor for NLTK

Gensim is another python library

Uses rules from English language to perform NLP tasks

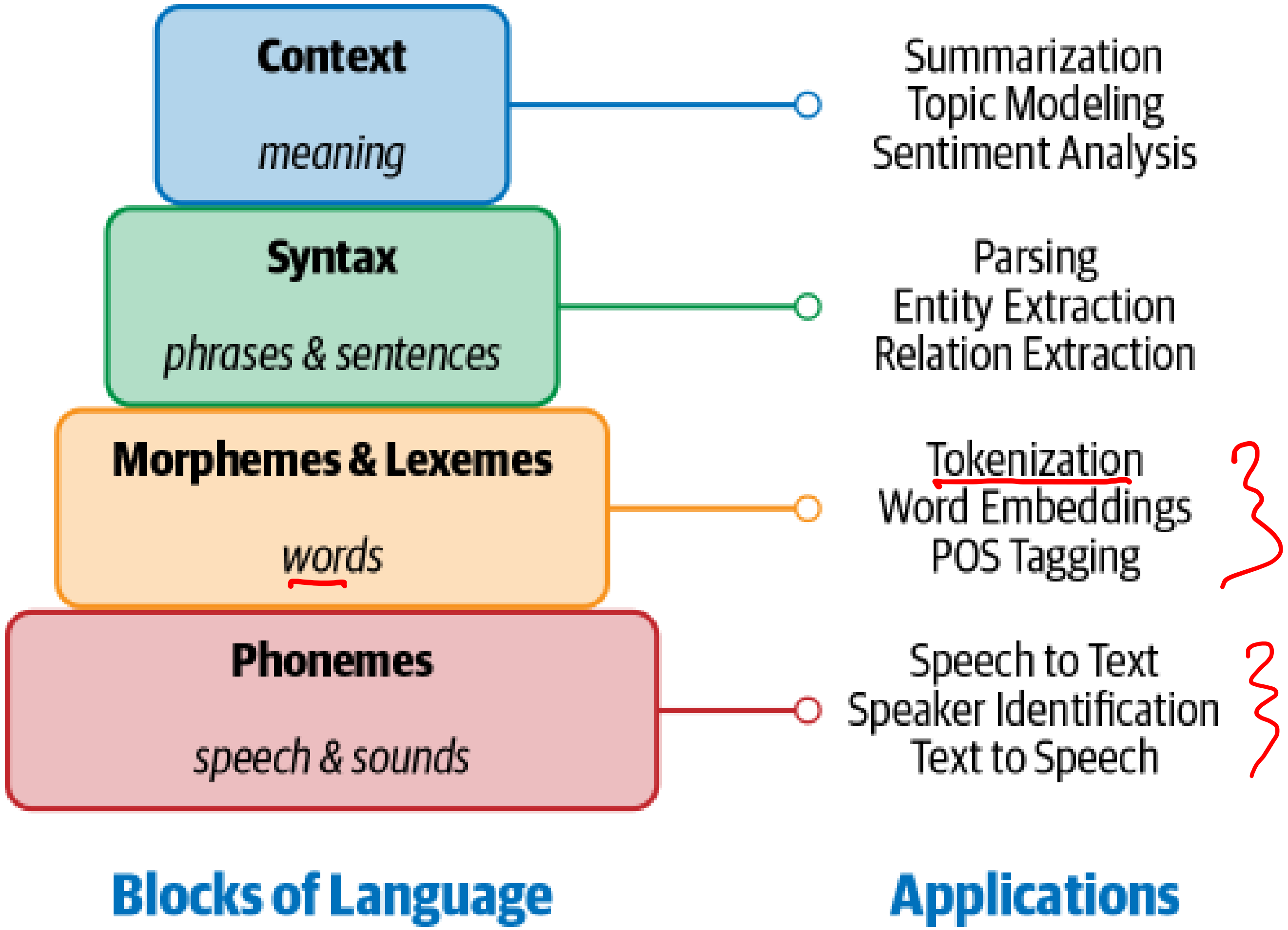


Fig 1-3, Vajjala et al

Two Representation Approaches

- Consider two sentences
  - “We prefer MCQ for the exam”
  - “For the exam, we prefer MCQ”
- Order of words don’t really matter here – we understand!
- How to represent word order is a fundamental question in NLP
- Two approaches
  - Sets – word order is discarded
    - Text is an unordered set of words
    - Called “Bag of words” Approach
  - Sequences
    - Process text in the order they appear like a timeseries
    - Sequence model
- Transformers – A hybrid approach that is technically order agnostic, but uses word position information and operate like a sequence model

Bag of Words Based on Occurence

	the	red	dog	cat	eats	food
1. the red dog →	1	1	1	0	0	0
2. cat eats dog →	0	0	1	1	1	0
3. dog eats food →	0	0	1	0	1	1
4. red cat eats →	0	1	0	1	1	0

When to use sequence models vs bag of words?

- Are bag of words model outdated?
- Have transformers taken over NLP?
- Not so soon. Each has its place.
- Use the following rule to determine which to use

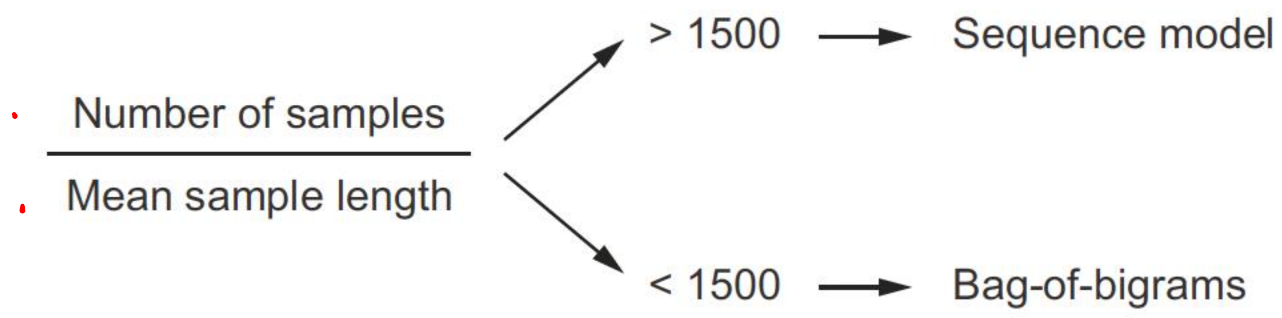


Figure 11.11 **A simple heuristic for selecting a text-classification model: the ratio between the number of training samples and the mean number of words per sample**

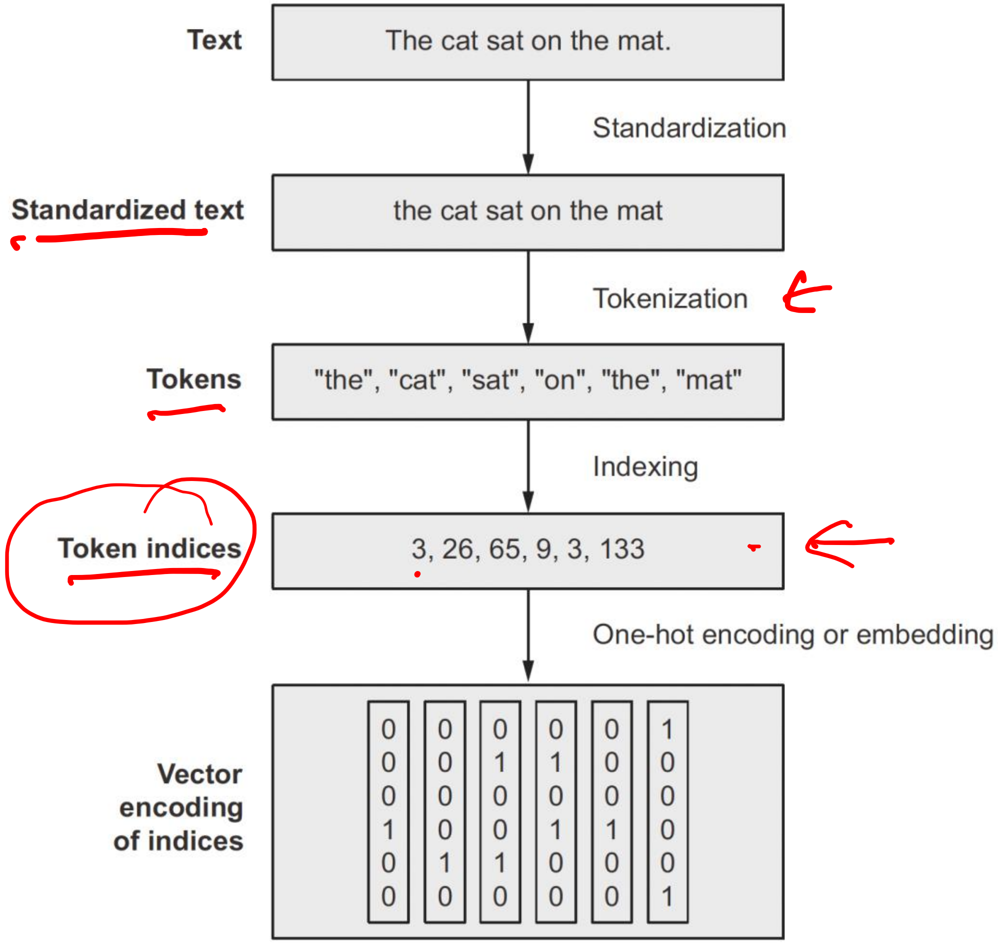
Refer: https://developers.google.com/machine-learning/guides/text-classification/step-2-5

Outline for Week01

- Part 01
  - Intro to NLP
  - Tasks in NLP
  - NLP Pipeline
  - Data Pre-processing, NLTK and Spacy
  - Two Representation Approaches
- Part 02
  - Text Vectorization
  - Word Embedding
  - Byte Pair Encoding Tokenization
  - Training Word Embedding

Data Pre-Processing for NLP

- DL Models are differentiable functions
- They take only numeric tensors as input – Cannot handle raw text
- Text Vectorization – Process of converting text to numbers is crucial to success of NLP
- Template for text vectorization
  - Standardize
    - Remove punctuation, convert to lowercase etc
  - Tokenize
    - Split text into units (tokens) – can be character, words, groups, sentences etc
  - Index
    - Given a numerical value to the tokens
- Finding a representation in numerical vector form is also called as embedding



Text Standardization

- General set of feature engineering tricks to ensure that your model doesn't need too many examples
- S1: "Couldn't this algo be correlated with an example of how it improvises learning?"
- S2: "Couldnt this Algorithm be related with example of how it improves learning?"
- Remove punctuation, capitalization and deal with common shortening, and sometimes perform stemming (to remove grammatical variations)
- Standardization is context specific and remove some information
  - Dealing with a QA task – then ? Is important

Text Tokenization

- Character level tokenization
  - Rare. Seen in specialized context text generation or speech recognition
- Word level tokenization
  - Typically used in sequence approach
- N-gram tokenization
  - Typically used in bag of words approach
- Byte-pair tokenization (GPT)

- Note about N-grams
  - N-gram tokenization and bag of words was the go-to method for language tasks (and still in many live applications)
  - They are mostly used in shallow models, Naïve Bayes classifiers
  - Deep models don’t need this feature engineering and can look at sequences and extracting representations in a hierarchical manner by themselves

Vocabulary

Byte pair encoding → used to build vocabulary

100k base

$\rightarrow$ 791, 382, 874, 240 $\Leftarrow W$

$$\begin{bmatrix}0\\ \vdots\\ 0\\ 0\\ 1\\ \vdots\\ 0\end{bmatrix}_{100k} \begin{bmatrix}0\\ \vdots\\ 0\\ \vdots\\ 0\end{bmatrix}_{100k} \begin{bmatrix}0\\ \vdots\\ 0\\ 1\\ \vdots\\ 0\end{bmatrix}_{100k} \begin{bmatrix}0\\ 0\\ \vdots\\ 0\end{bmatrix}$$

791st

100k × 4

$$\Rightarrow \underset{768\times 100k}{e}\ \underset{100k\times 4}{W} = \underset{768\times 4}{E} \Rightarrow [\;]\,[\;]\,[\;]\,[\;]$$

768

768 × 4

$$W_{4\times 100k}\ e_{100k\times 768} = E_{4\times 768}$$

weight matrix of the embedding layer

Deepak Subramani, deepakns@iisc.ac.in

3, 26, 65, 9 [MASK] ... seq_len = 15 (1000) context_length

20,000

max_token
vocab_size

$$\begin{bmatrix}0\\0\\1\\0\\0\\0\\\vdots\\0\end{bmatrix}\begin{bmatrix}0\\0\\0\\\vdots\\0\\1\\0\\0\\\vdots\\0\end{bmatrix}\begin{bmatrix}0\\\vdots\\1\\0\\\vdots\end{bmatrix}\begin{bmatrix}0\\0\\\vdots\\1\\\vdots\end{bmatrix}$$

20,000 one-hot vector

20,000 × 4 20 000 × 15

M $E_{emb\_dim \times vocab\_size}$ dense

$E_{256 \times 20000} * W_{20000 \times 15} = e_{256 \times 15}$

$E_{256 \times 1000} * W_{20000 \times 4} = e_{256 \times 4}$

Vocabulary Indexing and Text Vectorization

- To encode split tokens into numerical vectors, we use vocabulary indexing
  - Use a vocabulary and assign a unique number to that vocabulary
    - Vocabulary size is a hyperparameter
  - Do a one hot encoding of tokens
  - Map the one-hot encoding to a text vector to an embedding dimension
    - Embed_dim is a hyperparameter
- Advantages:
  - Allows all words to be represented uniquely across language tasks
  - It opens up possibilities of learning word vectors that have a “mathematical distance” between them that preserves meaningful relationships as understood by humans!

Sub word Embedding - Byte Pair Encoding

Vocabulary generation

- BPE powers modern DL based NLP
- Idea is simple, yet powerful
  - Start with UTF-8 bytes as tokens (256 number)
  - Identify frequently occurring byte-pair, and add it as a new token
  - Keep merging until you hit the desired vocabulary size (hyperparameter)

[UNK]

- Example:
  - Data: aaabdaaabac
  - aa occurs most frequently, so replace it with Z=aa to get ZabdZabac
  - ab occurs most frequently, so replace it with Y=ab to get ZYdZYac
  - ZY occurs most frequently, so replace it with X=ZY to get XdXac

Tabular data

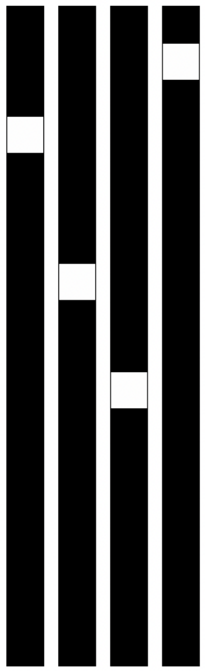
Pen 1

Pen 2

Word Embedding

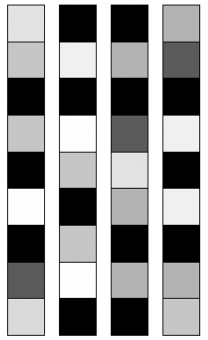
- One-Hot encoding of word vector makes an orthogonal words assumption
- This encoding considers that all words are independent of each other
- For words, this is a wrong assumption
- “Closet”, “Cupboard”, “Toilet” are not orthogonal words
  - Closet and Cupboard should be close or same vector in American English
  - Closet and Toilet should be close of same vector in Indian English
- What we want? – Geometric relationship between words should reflect the semantic relationship
  - L2 distance or cosine distance between words must reflect their “Semantic distance”
- Word Embeddings – vector representation of words that map human language into a structured geometric space

$\boxed{\text{1-hot}}$



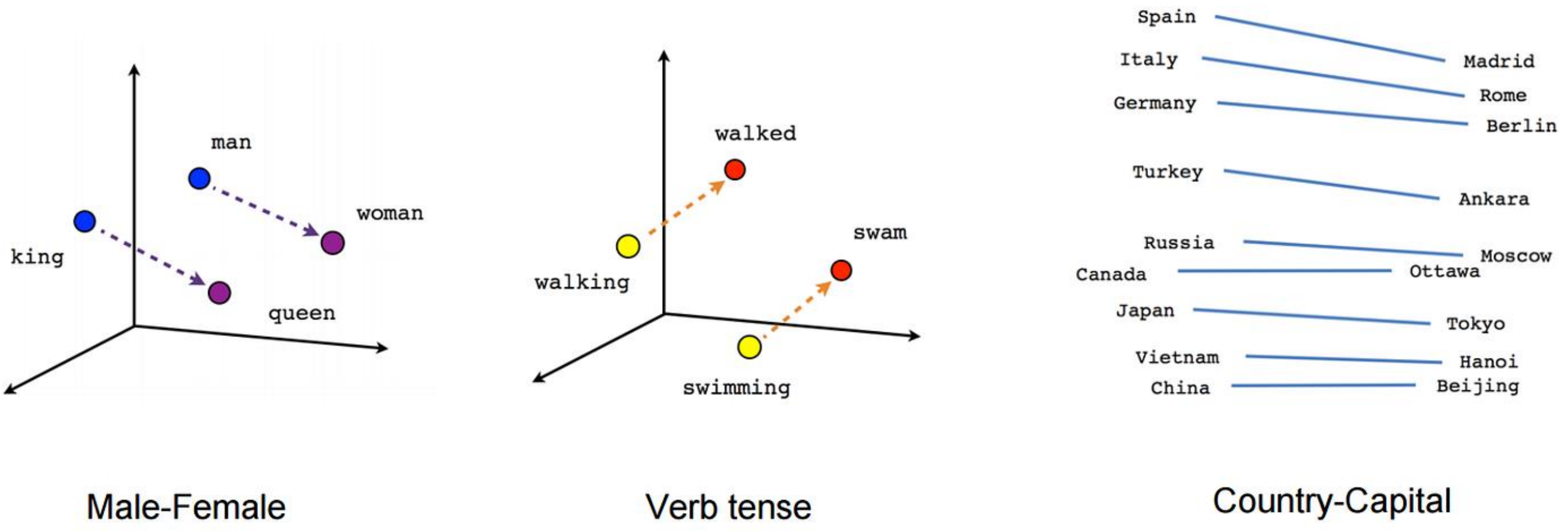
One-hot word vectors:
- Sparse
- High-dimensional
- Hardcoded

Word Embedding



Word embeddings:
- Dense
- Lower-dimensional
- Learned from data

Visualization of Word Vectors



Source: Gensim, Practice in AST

Embedding Layer

- Learn word embeddings jointly with the main task you care about
- Initialize to random word vectors (map)
- The embedding of a single word vector is a matrix vector multiplication
- The embedding matrix is learnt via backpropagation

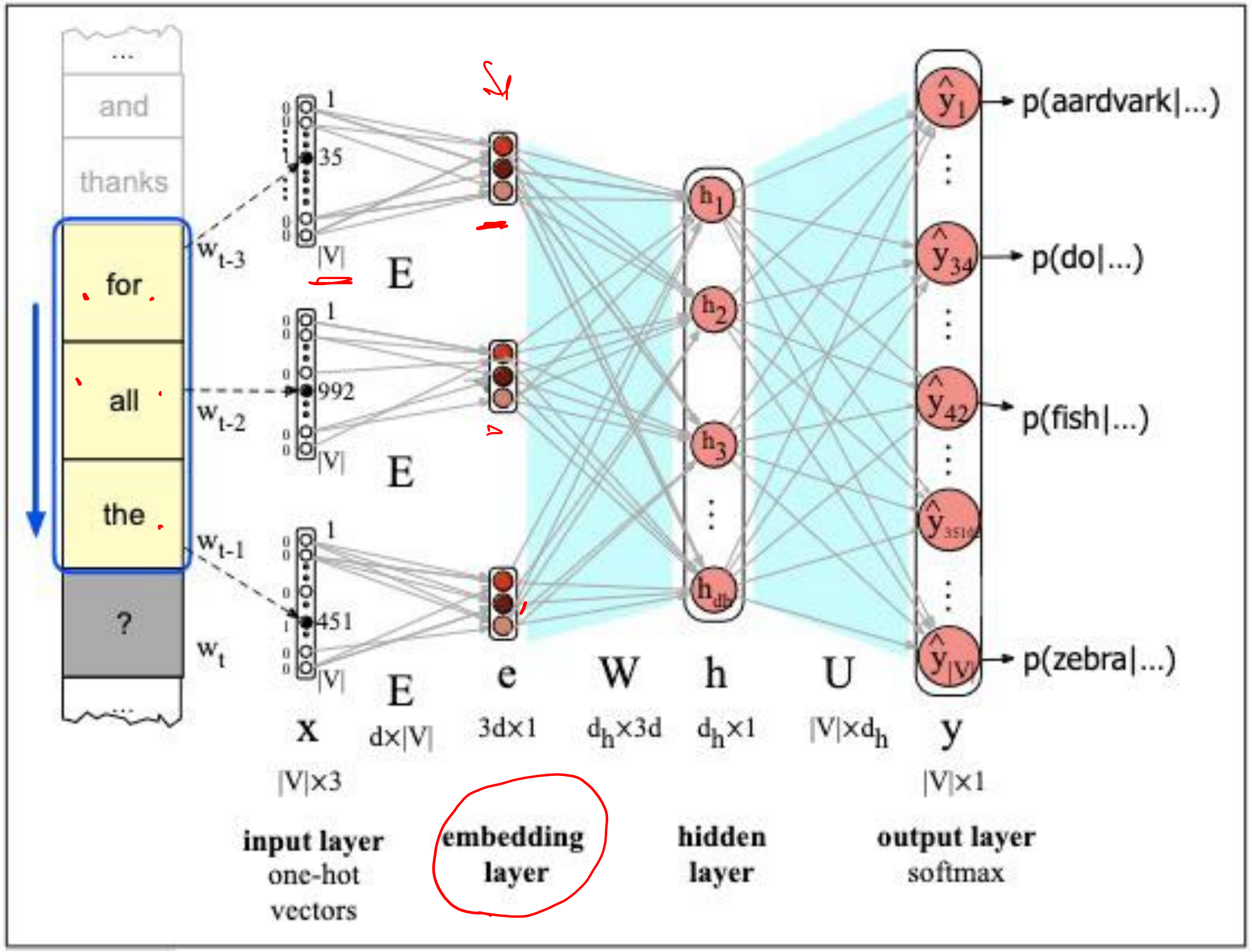
```
embedding_layer = layers.Embedding(input_dim=max_tokens, output_dim=256)
```

The Embedding layer takes at least two arguments: the number of possible tokens and the dimensionality of the embeddings (here, 256).

Word index ⟶ Embedding layer ⟶ Corresponding word vector

- Pre-trained embeddings can also be used such as Google Word2Vec, GloVe

Word Embedding – Pictorial Explanation



Types of Word Embeddings

- Contextual – Word representation vector depends on the location
  - ELMo
  - BERT
- Non Contextual – A word has the same representation vector irrespective of where it occurs in the sentence.
  - Word2Vec
  - GloVe
- ELMo and BERT are trained using transformer architecture

Model Name	Context Sensitive embeddings	Learnt representations
Word2vec	No	Words
Glove	No	Words
ELMo	Yes	Words
BERT	Yes	Subwords

ELMo – Embeddings from Language Models
BERT – Bidirectional Encoding Representations from Transformers
Glove – Global Vectors for Word Representation