Module-II (Part-2): Word Level Analysis

by:

Dr. Soumya Priyadarsini Panda

Sr. Assistant Professor

Dept. of CSE, SIT, Bhubaneswar

Morphological Parsing

- Study of word structure and formation of words from smaller units (morphemes)

- Goal is to discover the morphemes that build a given word

- Morphemes are the smallest meaning bearing units in a language.

Classes of Morphemes

- **Stems:** The main morpheme that contains the central meaning
- **Affixes:** Modify the meaning given by the stem
  - Prefix, suffix, etc

Example:

- "bread" consist of single morpheme
- "eggs" consist of two morpheme:
  - Egg
  - –s

- Job of morphological parser: identify the word "eggs" is the plural form of "egg"

Ways of word formation

1. Inflection:

- In inflectional form, the root word added with some grammatical morpheme forms a word that belongs to the same class as the original stem
- Example: pen (Noun), pen**s** (Noun)

2. Derivation:

- In derivation the word stem added with some grammatical morpheme forms a word that belongs to a different class
  - Example: teacher(Noun), teaching (Verb)

3. Compounding:

- Merging two or more words to form a new word
  - Exa: “desk” + “top”= desktop

Applications of morphological analysis and generation

- Spelling error correction

- Machine translation

Information sources for Morphological Parser

1. Lexicon:

List stems and affixes together with basic information

2. Morphotactics:

- Deals with ordering of morphemes to form a word
- Exa:
  - Valid word: “Rest-less-ness”
  - Invalid word: “rest-ness-less”

3. Orthographic rules:

- Spelling rules specifying the changes occurred by combining two morphemes
- Exa:

Spelling rule: y→ier changes

“Easy” → “easier”

Need of Morphological Analysis

- Morphological analysis can be avoided if an exhaustive lexicon is available

Exhaustive lexicon:

- Lists features for all the words formed from all the roots
- Limitations:
  - Heavy demand on memory
  - Fails to show the relationship between different roots having similar word forms
  - Not practical to list all possible word forms

Words and Word Classes

- Words are classified into categorized called Part-of-Speech (PoS) or word classes or lexical categories.

  Examples: Noun, Verb, Determiner, etc

- They can be categorized into 2 broad classes:
  - **Open word class:**
    - It can acquire new members
    - Example: Noun, verb (except auxiliary verb)
  - **Closed word class:**
    - It can't acquire new members
    - Example: preposition, auxiliary verb

Word Level Processing
Basic Concepts

Basic Concepts

corpus (plural corpora):

- A computer-readable collection of text or speech.

Example:

- **Brown corpus:** million-word collection of samples from 500 written English texts from different genres (newspaper, fiction, non-fiction, academic, etc.)

- Many NLP task requires a text/speech corpus to work on it.

Cont…

Lemma:

- A lemma is a set of lexical forms having the same stem, the same major part-of-speech, and the same word sense.

Example:

‘cat’ ‘cats’

Word forms:

- inflected/ derived/ compounded

Cont…

Words:

How many words in the below sentence?

He stepped out into the hall, was delighted to encounter a water brother.

- **13 words** (if punctuation marks not counted as words)
- **15 words** (if count punctuations)
- Whether period ("."), comma (","), etc can be considered in word counts depends on the task.

Cont…

- Whether period (“.”), comma (“,”), etc can be considered in word counts depends on the task.

- Punctuation is critical for:
  - finding boundaries of things (commas, periods, colons) and
  - identifying some aspects of meaning (question marks, exclamation marks, quotation marks

Cont…

Types:

- The number of distinct words in a corpus
- If the set of words in the vocabulary is V, the number of types is the word token vocabulary size |V|

Token:

- The total number N of running words

Cont…

Example:

They picnicked by the pool, then lay back on the grass and looked at the stars.

If punctuations are not considered,

No. of tokens=16

Types=14

Example

<table>
<tr><th><b>Corpus:</b></th><th><b>Tokens=N</b></th><th><b>Type= |V|</b></th></tr>
<tr><td>Brown corpus</td><td>1 million</td><td>38 thousand</td></tr>
<tr><td>COCA</td><td>440 million</td><td>2 million</td></tr>
<tr><td>Google N-grams</td><td>1 trillion</td><td>13 million</td></tr>
</table>

- When developing computational models for language processing from a corpus, the data requires to be pre processed or normalized before using it.

Text pre-processing

Text Normalization

- Every NLP task requires text normalization:
- The text normalization process commonly involves:
  - **Segmenting sentences**
  - **Tokenizing (segmenting) words**
  - **Normalizing word formats**

Sentence Segmentation

- Sentence segmentation is an important step in text processing.

- The most useful cues for segmenting a text into sentences are punctuation, like periods, question marks, and exclamation points.

- While !, ? are mostly unambiguous **period** “.” is very ambiguous
    - Sentence boundary .
    - Abbreviations like Inc. or Dr.
    - Numbers like .02% or 4.3

Cont…

- Sentence tokenization methods work by-
  - first deciding (based on rules or machine learning) whether
    - a period is part of the word or
    - is a sentence-boundary marker.

- An abbreviation dictionary can help determine whether the period is part of a commonly used abbreviation.
  - the dictionaries can be hand-built or machine learned

Tokenizing (segmenting) words

(i) Space-based tokenization:

- Can be used for languages that use space characters between words (Arabic, Cyrillic, Greek, Latin, etc)
- Segment off a token between instances of spaces

Example approaches to tokenize:

- Unix tools can be used for space-based tokenization, counting of tokens and sorting of tokens in alphabetic order on the corpus data
- NLTK (python tool for natural language processing)

Example

- Unix tools for space-based tokenization:
  - The "tr" command
  - Given a text file, output the word tokens and their frequencies

Unix command:

```
tr -sc ’A-Za-z’ ’\n’ < shakes.txt
```

- Output all word tokens in shakes.txt file into new lines

Issues in Tokenization

- Can't just blindly remove punctuation:
  - Ph.D.
  - prices ($45.55)
  - dates (01/02/06)
  - URLs (http://www.stanford.edu)
  - hashtags (#nlproc)
  - email addresses (someone@cs.colorado.edu)

- When should multiword expressions (MWE) be words?
  - New York, rock ’n’ roll

Cont…

(ii) Penn Treebank Tokenization standard

- This standard can separates clitics (doesn’t becomes does plus n’t), keeps hyphenated words together, and separates out all punctuation.

(iii) Byte-Pair Encoding for Tokenization:

- The token learner takes a raw training corpus (sometimes roughly pre-separated into words, for example by whitespace) and induces a vocabulary, a set of tokens.
- The token segmenter takes a raw test sentence and segments it into the tokens in the vocabulary.
- Three algorithms are widely used: byte-pair encoding, unigram language modeling, and Word Piece

Word Normalization

- Word normalization is the task of putting words/ tokens in a standard format.

Example:

U.S.A. or USA

Fed or fed

am, is, be, are

- **Case folding:** It is a kind of normalization that involves mapping everything to lower case.

Lemmatization

- Lemmatization is the task of determining that two words have the same root, despite their surface differences.

- The words: ‘am’, ‘are’, and ‘is’ have the shared lemma

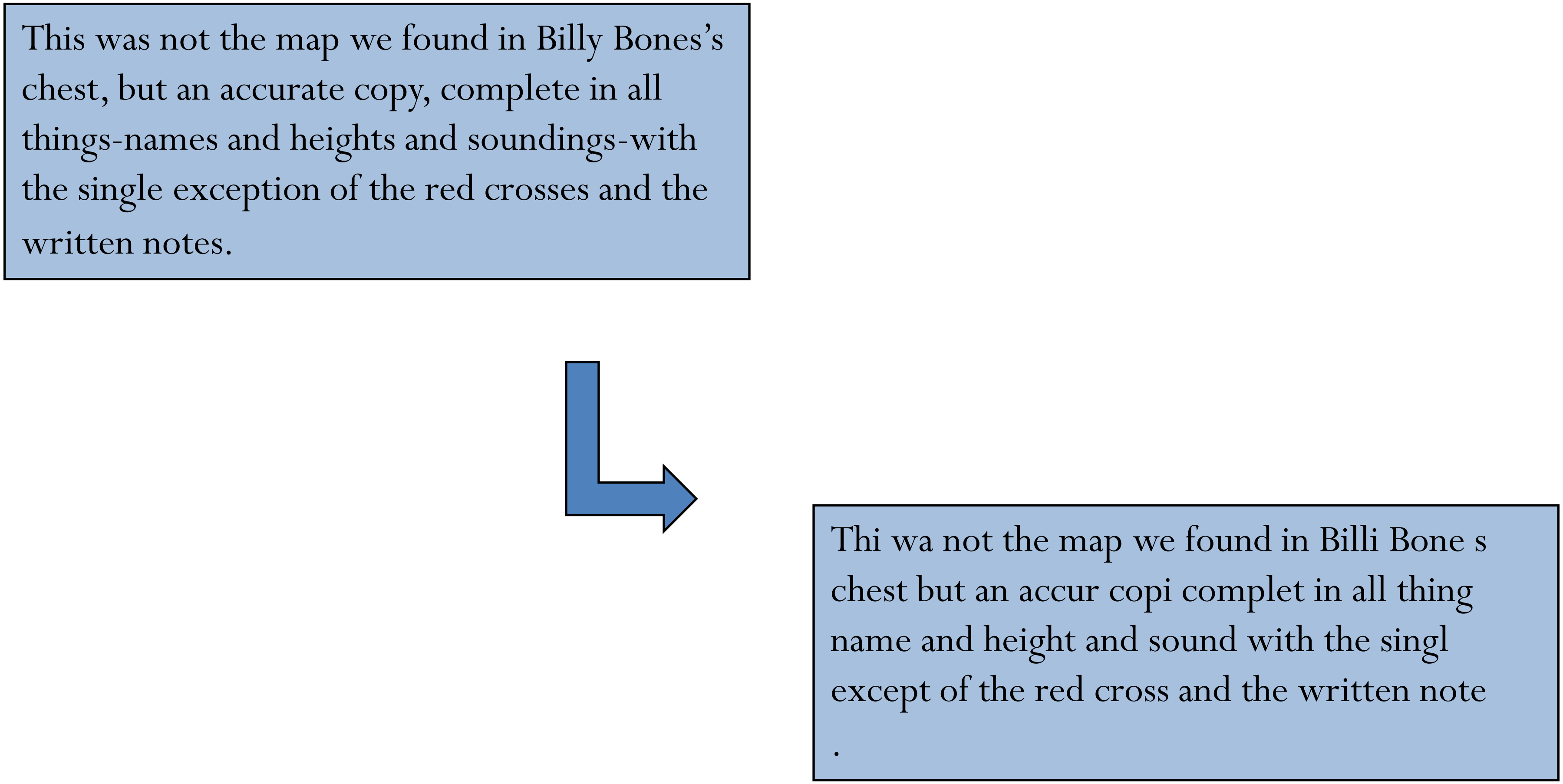
$$am, are, is \rightarrow be$$

$$car, cars, car's, cars' \rightarrow car$$

- Lemmatization is done by **Morphological Parsing**
  - Morphological Parsers parse ‘cats’ into two morphemes:
    - ‘cat’ and ‘s’

Stemming

- Reduce terms to stems, chopping off affixes crudely



This was not the map we found in Billy Bones's chest, but an accurate copy, complete in all things-names and heights and soundings-with the single exception of the red crosses and the written notes.

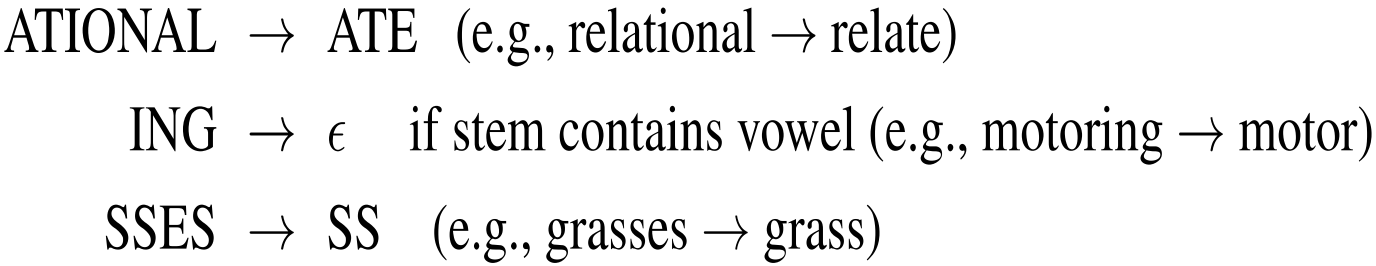
Thi wa not the map we found in Billi Bone s chest but an accur copi complet in all thing name and height and sound with the singl except of the red cross and the written note

.

Porter Stemmer

- Based on a series of rewrite rules run in series
  - A cascade, in which output of each pass fed to next pass

- Some sample rules:



$$\begin{array}{rcll}
\text{ATIONAL} & \rightarrow & \text{ATE} & \text{(e.g., relational} \rightarrow \text{relate)} \\
\text{ING} & \rightarrow & \epsilon & \text{if stem contains vowel (e.g., motoring} \rightarrow \text{motor)} \\
\text{SSES} & \rightarrow & \text{SS} & \text{(e.g., grasses} \rightarrow \text{grass)}
\end{array}$$

Stemming Vs. Lemmatization

- Stemming removes the affixes from the words to find the root words.

- Lemmatization uses vocabulary and morphological analysis of words to find the root word

Example:

Word	Stemming	Lemmatization
studied, studying, studies	study	study
caring	car	care
Better	-	good

Stop Word Removal

- Some systems ignore stop words

  - **Stop words:** very frequent words like *the* and *a*.
    - Sort the vocabulary by word frequency in training set
    - Call the top 10 or 50 words the **stopword list**.

    - Remove all stop words from both training and test sets
      - As if they were never there!

- But removing stop words in every situation doesn't usually help
  - So in practice most algorithms use **all** words and **don't** use stopword lists

Named Entities

- **Named entity**, in its core usage, means anything that can be referred to with a proper name.

- Most common 4 tags:
  - PER (Person): “Marie Curie”
  - LOC (Location): “New York City”
  - ORG (Organization): “Stanford University”
  - GPE (Geo-Political Entity): "Boulder, Colorado“

- Often multi-word phrases
- But the term is also extended to things that aren't entities:
  - dates, times, prices

Named Entity Recognition(NER)

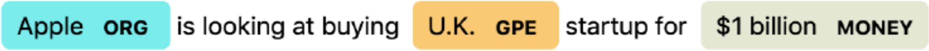
- NER is a subtask of **information extraction** that locates and classify named entities mentioned in unstructured text into pre-defined categories.

- Example categories:
  - person names, locations, time expressions, quantities, monetary values, etc.

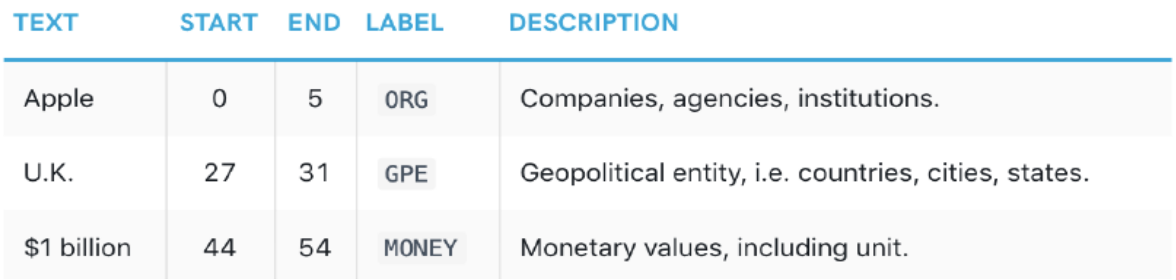
Example:

Input sentence: Apple is looking at buying U.K. startup for $1billion

Output from NER tagger:



Apple ORG is looking at buying U.K. GPE startup for $1 billion MONEY



TEXT	START	END	LABEL	DESCRIPTION
Apple	0	5	ORG	Companies, agencies, institutions.
U.K.	27	31	GPE	Geopolitical entity, i.e. countries, cities, states.
$1 billion	44	54	MONEY	Monetary values, including unit.

NER output

Citing high fuel prices, [$_{\text{ORG}}$ **United Airlines**] said [$_{\text{TIME}}$ **Friday**] it has increased fares by [$_{\text{MONEY}}$ **$6**] per round trip on flights to some cities also served by lower-cost carriers. [$_{\text{ORG}}$ **American Airlines**], a unit of [$_{\text{ORG}}$ **AMR Corp.**], immediately matched the move, spokesman [$_{\text{PER}}$ **Tim Wagner**] said. [$_{\text{ORG}}$ **United**], a unit of [$_{\text{ORG}}$ **UAL Corp.**], said the increase took effect [$_{\text{TIME}}$ **Thursday**] and applies to most routes where it competes against discount carriers, such as [$_{\text{LOC}}$ **Chicago**] to [$_{\text{LOC}}$ **Dallas**] and [$_{\text{LOC}}$ **Denver**] to [$_{\text{LOC}}$ **San Francisco**].

Why NER?

- Sentiment analysis: consumer’s sentiment toward a particular company or person?

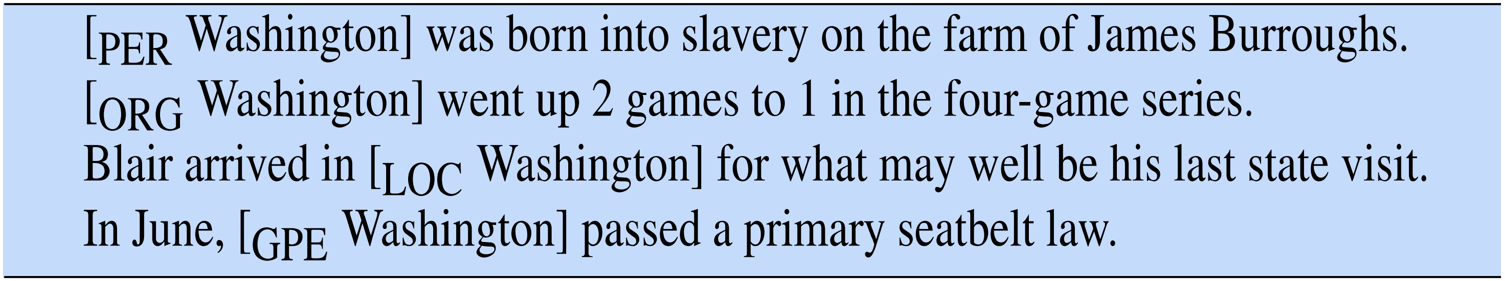
- Question Answering: answer questions about an entity?

- Information Extraction: Extracting facts about entities from text.

Why NER is hard

1) Segmentation
   - In POS tagging, no segmentation problem since each word gets one tag.
   - In NER we have to find and segment the entities!

2) Type ambiguity



[$_{\text{PER}}$ Washington] was born into slavery on the farm of James Burroughs.
[$_{\text{ORG}}$ Washington] went up 2 games to 1 in the four-game series.
Blair arrived in [$_{\text{LOC}}$ Washington] for what may well be his last state visit.
In June, [$_{\text{GPE}}$ Washington] passed a primary seatbelt law.

Standard algorithms for NER

- Supervised Machine Learning given a human-labeled training set of text annotated with tags

- Hidden Markov Models

- Conditional Random Fields (CRF)/ Maximum Entropy Markov Models (MEMM)

- Neural sequence models (RNNs or Transformers)

Some common Text Pre-processing Steps

- Tokenization
- Text Normalization
  - Removing punctuations like . , ! $( ) * % @, Removing URLs
  - Removing Stop words
  - Lower casing
- Stemming/ Lemmatization

Reference sources for details of implementation using python:

https://www.analyticsvidhya.com/blog/2021/06/text-preprocessing-in-nlp-with-python-codes/

https://towardsdatascience.com/text-preprocessing-in-natural-language-processing-using-python-6113ff5decd8

Text pre-processing in Sentiment Analysis Application

- Sentiment Analysis is the process of understanding if a given text is talking positively or negatively about a given subject.

- i.e. Sentiment analysis aims to estimate the sentiment polarity of a body of text based solely on its content.

- The sentiment polarity of text can be defined as-
  - a value that says whether the expressed opinion is:
    - positive (polarity=1), negative (polarity=0) when considered as a binary classification problem

- Sentiment may be categorized: positive(1) , negative (-1) or neutral (0)

Cont…

- Sentiment analysis may be considered as a multi class classification problem for varied emotions such as: happy, sad, angry, relaxed, stressed, fear, surprise, etc

- In the multi-label classification problem, an instance is associated with a subset of labels

Sentiment Analysis as a Binary Classification Problem

- Sentiment analysis can be expressed as the following classification problem:

- **Feature**: the string representing the input text

- **Target**: the text's **polarity** (0 or 1)
  - 1: if a positive sentiment
  - 0: if a negative sentiment

Cont…

- In order to be able to train a machine learning classifier, **numerical features are needed**.

- A succession of words, spaces, punctuation and sometimes emojis also are needed to be transformed into some numerical features that can be used in a learning algorithm.

Steps:

- **pre-processing:** Make the texts cleaner and easier to process

- **Vectorization:** Transform the cleaned texts into numerical vectors

Pre-Processing

- Assumes that the smallest unit of information in a text is the word (as opposed to the character).

- Represent texts as **word sequences**.

Example:

Text: This is a cat.

--> **Word Sequence:**

[‘this’, ‘is’, ‘a’, ‘cat’]

Vectorization

- Machine learning algorithms operate on a numeric feature space, expecting input as a two-dimensional array where rows are instances and columns are features.

- In order to perform machine learning on text, the documents/text are required to be transform into vector representations.

- This process is called **feature extraction** or **vectorization**

Cont...

- Text Vectorization is the process of converting text into numerical representation

Example techniques:

- **Bag of Words (BoW)**
- TF-IDF
- Word2Vec, etc

The Bag of Words Representation

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!

fairy
always
love
to
it
it
whimsical
it
I
and
seen
are
anyone
friend
happy
dialogue
recommend
adventure
who
sweet
of
satirical
movie
it
it
I
but
to
romantic
I
several
yet
again
it
the
humor
the
seen
would
to
scenes
I
the manages
the
fun
I
and
times
and
about
while
whenever
have
conventions
with

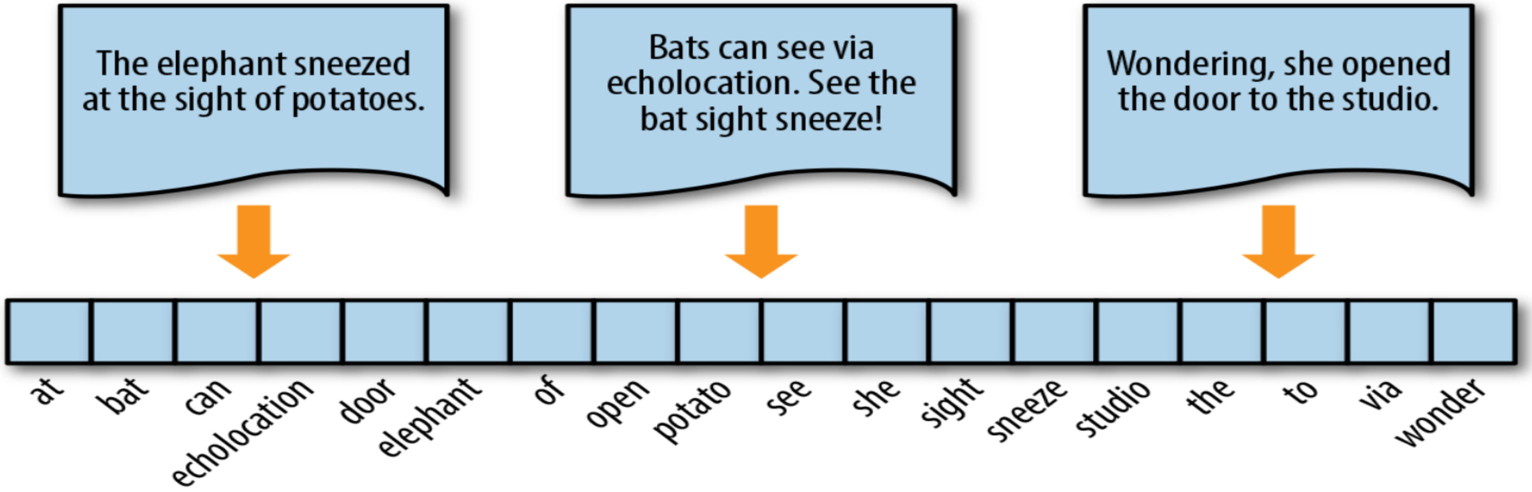
it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
…	…

Cont...

- The simplest text vectorization technique is Bag Of Words (BOW).

- It starts with a list of words called the vocabulary (this is often all the words that occur in the training data).

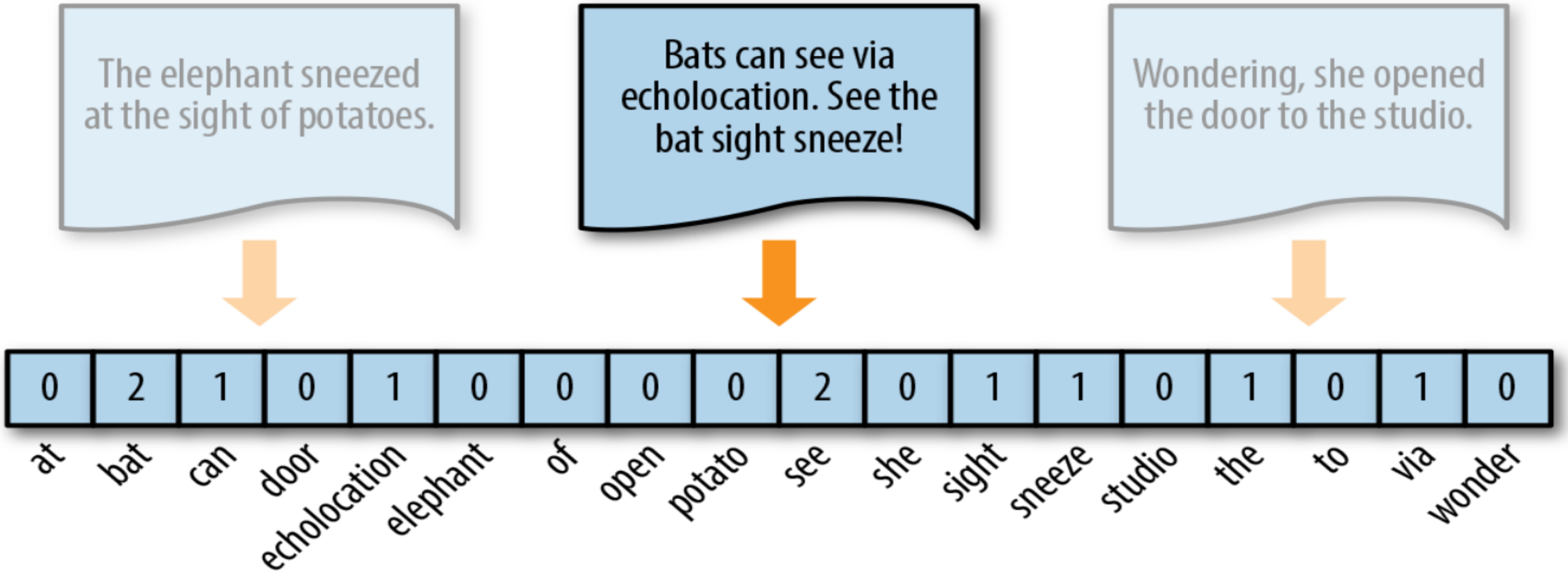
- Then, given an input text, it outputs a numerical vector which is simply the vector of word counts for each word of the vocabulary



The elephant sneezed at the sight of potatoes.
Bats can see via echolocation. See the bat sight sneeze!
Wondering, she opened the door to the studio.
at
bat
can
echolocation
door
elephant
of
open
potato
see
she
sight
sneeze
studio
the
to
via
wonder

Frequency Vector

- The simplest vector encoding model is to simply fill in the vector with the frequency of each word as it appears in the document



The elephant sneezed at the sight of potatoes.

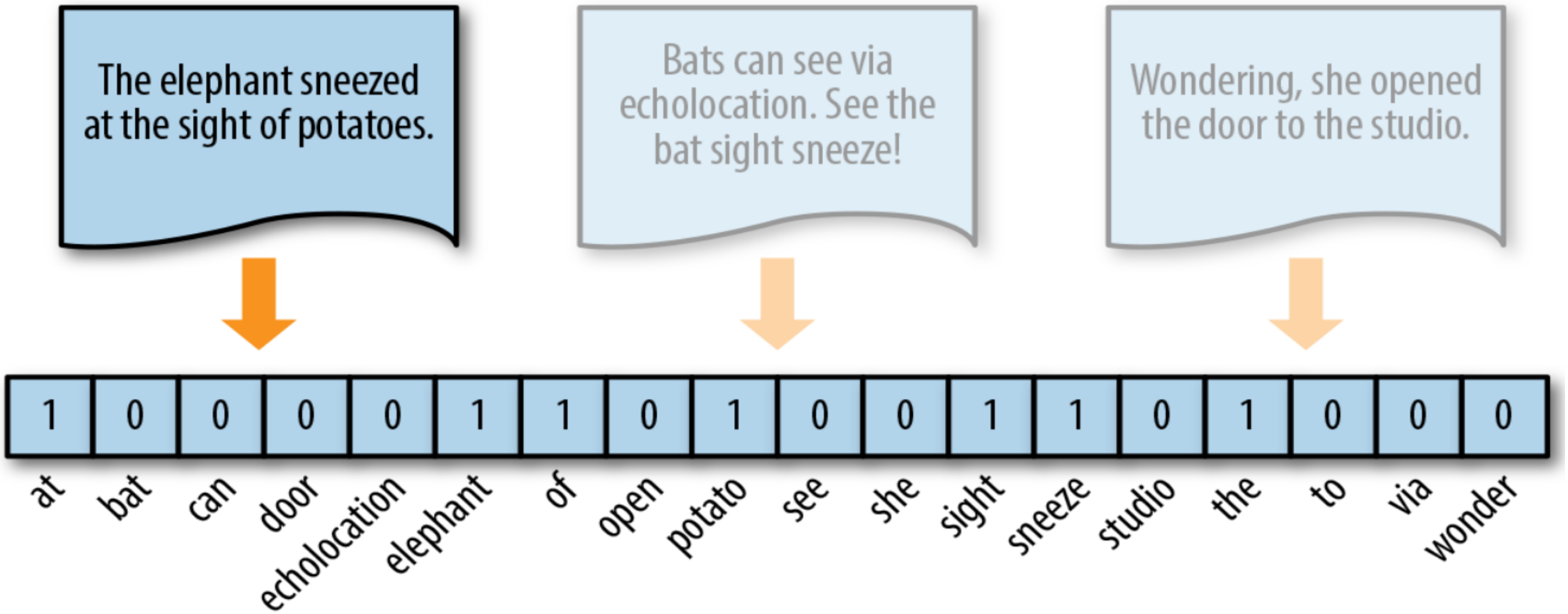
Bats can see via echolocation. See the bat sight sneeze!

Wondering, she opened the door to the studio.

at	bat	can	door	echolocation	elephant	of	open	potato	see	she	sight	sneeze	studio	the	to	via	wonder
0	2	1	0	1	0	0	0	0	2	0	1	1	0	1	0	1	0

One-Hot Encoding

- In frequency-based encoding methods tokens that occur very frequently are orders of magnitude more “significant” than other, less frequent ones
- one-hot encoding is a boolean vector encoding method that marks a particular vector index with a value of true (1) if the token exists in the document and false (0) if it does not



The elephant sneezed at the sight of potatoes.

Bats can see via echolocation. See the bat sight sneeze!

Wondering, she opened the door to the studio.

at	bat	can	door	echolocation	elephant	of	open	potato	see	she	sight	sneeze	studio	the	to	via	wonder
1	0	0	0	0	1	1	0	1	0	0	1	1	0	1	0	0	0

Example-1

Documents:

D_1: “Dog hates a cat. It loves to go out and play.”

D_2: “Cat loves to play with a ball.”

- We can build a corpus from above 2 documents just by combining it and the features will be all unique words:

  [‘and’, ‘ball’, ‘cat’, ‘dog’, ‘go’, ‘hates’, ‘it’, ‘loves’, ‘out’, ‘play’, ‘to’, ‘with’].
- Bag of Words takes a document from corpus and converts into a numeric vector by mapping each document word to a feature vector.
- Feature vector:

	and	ball	cat	dog	go	hate	it	loves	out	play	to	with
D_1	1	0	1	1	1	1	1	1	1	1	1	0
D_2	0	1	1	0	0	0	0	1	0	1	1	1

Cont…

- Using BOW is making the assumption that the more a word appears in a text, the more it is representative of its meaning.

- One thing to keep in mind is that the feature vectors that result from BOW are usually very large (exa: 80,000-dimensional vectors in this case).

- So we need to use simple algorithms that are efficient on a large number of features (e.g., Naive Bayes, linear SVM, or logistic regression)

Spelling Error Detection & Correction

Spelling Error Detection & Correction

- Detection and correction of spelling errors is an integral part of modern word processors.

- Given a sequence of characters corresponding to a misspelled word and an ordered list of possible correct words are to be produced.

Types of errors:

• Insertion errors :	thew
• Deletion errors:	th_
• Substitution errors:	thu
• Transposition errors:	hte

Spelling Error Patterns

- Spelling errors belongs to 2 different categories:

Non-word errors:

- When errors resulted in a word that doesn’t appear in a given lexicon or is not a valid word.

Example:

Graffe →Giraffe

Real word errors:

- When errors resulted in a word that is another actual word of the language.

  Example:

  with -> will

  their-> there

Minimum Edit Distance

- Much of natural language processing is concerned with measuring how similar two strings are.

- The minimum edit distance between two strings is the-
  - minimum number of operations **(insertion, deletion, substitution)** required to transform one string into another.

Example

The user typed “graffe”

Which is closest?

- graf
- graft
- grail
- giraffe

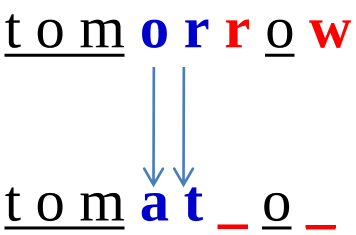
Cont…

Example-1: to convert the word “graffe” to “giraffe”, 1 insertion operation is required

Graffe →Giraffe

=> minimum edit distance = 1

Example-2: to convert the string ‘tomorrow’ to ‘tomato’:



t o m **o** **r** **r** o **w** //2 deletions

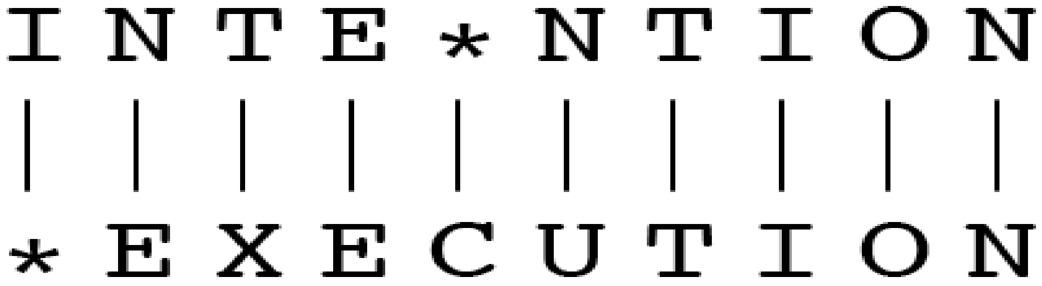
//2 substitutions

t o m **a** **t** _ o _

=> Minimum edit distance=4

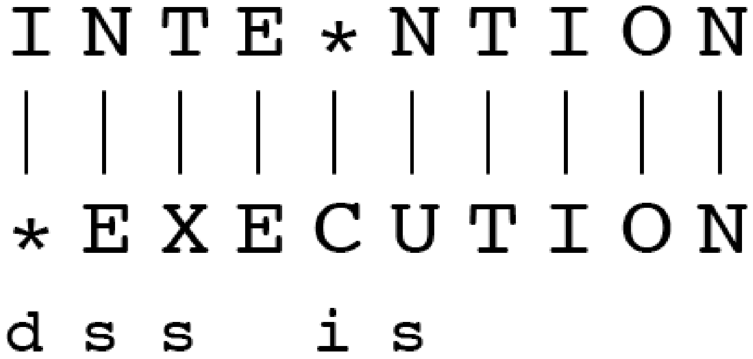
Example

- Two strings and their **alignment**:



```
I N T E * N T I O N
| | | | | | | | | |
* E X E C U T I O N
```

Example



```
I N T E * N T I O N
| | | | | | | | | |
* E X E C U T I O N
d s s   i s
```

- If each operation has cost of 1
  - Distance between these is 5

- If substitutions cost 2
  - Distance between them is 8

Part-of Speech (PoS) Tagging

- PoS tagging is the process of assigning a Part-of-Speech (noun, verb, etc) to each word in a sentence.

- The collection of tags used by a particular tagger is called a tag set.

- The input to the tagging algorithm are the sequence of words of a natural language sentence and specific tag set.

- The output is a single best PoS tag for each word

S->VP
VP->VP NP
NP-> DET NN
VP->VB
Book|VB , a|DET,
room| NN

Example:

Input sentence: book a room

Output: $[_{VP}\ [_{VB}\ \text{book}\ ]\ [_{NP}\ [_{DET}\ \text{a}]\ [_{NN}\ \text{room}]\ ]\ ]$

Cont…

- PoS tagging is an early step of text processing in many NLP applications like, machine translation, information extraction, information retrieval, etc

- **Types of PoS Tagging:**
  - **Rule-based taggers**
  - **Stochastic taggers**
  - **Hybrid tagger**

Rule-based Taggers

- Rule-based taggers uses hand coded rules to assign tags to words.

- These rules use a lexicon to obtain a list of candidate tags (PoS) and then use rules to discard incorrect tags.

Example-1:

The noun-verb ambiguity for the word "show" in the below sentence with the potential tag set {VB, NN} -

"The **show** must go on"

The ambiguity can be resolved by using the rule:

"IF preceding is a DET then eliminate VB tag"

=> show is tagged as NN

Cont…

- In addition to contextual information, many taggers use morphological information to disambiguate words

Example: TAGGIT is a rule-based tagger

Example-2: Rule: “IF word ends with –ing and preceding word is a verb, then label it as VB”

Advantages:

- Speed of tagging

Disadvantages:

- Rules are language specific, therefore it can be applied to only that language for which rules are applied.

Stochastic Taggers

- Stochastic taggers are data driven approaches that uses frequency based information to derive tag by referring a corpus.

- It disambiguate words based on the probability that a word occurs with a particular tag in a corpus.

- The unigram model can be used to assign probabilities .

- Example: HMM-based tagger is a stochastic tagger

Cont…

- Stochastic taggers uses a tagged training corpus and automatically calculates the frequency of the tags and find out the best tag based on the estimated frequencies.

- However it may result in incorrect tagging in some context.

Example: “book a room”

If frequency estimates from corpus:

Word	Type	#
book	NN	15
book	VB	5

=> tag assigned to the word ‘book’ will be NN

which is wrong in the context.

Cont…

- To perform more accurate predictions, the context information may be considered and bigram sequences may be used.

Advantages:

- Accuracy is more if training corpus is good.

Disadvantages:

- Requires pre-tagged corpus for training and in many languages pre tagged corpus is not available.

Hybrid Taggers

- It combines features of both rule-based and stochastic taggers.

- Like rule-based tagger, it uses rules to assign tags to words and like stochastic tagger it uses training corpus to introduce rules automatically from a tagged training corpus.

- Example: Brill tagger

Advantages:

- It produces more accurate results
- It automatically generates new rules

Disadvantages:

- Lack of annotated corpus in many languages.

Unknown Words

- Unknown words are words that doesn’t appear in the dictionary or training corpus.

- They can be handled in the tagging process by 2 methods:

  - Assign the most frequent tag to the unknown words

  - Initialize it to some open tag set and then disambiguate it using morphological information and probability values.

Standard algorithms for POS tagging

- Supervised Machine Learning Algorithms:
  - Hidden Markov Models
  - Conditional Random Fields (CRF)/ Maximum Entropy Markov Models (MEMM)
  - Neural sequence models (RNNs or Transformers)