

ACCA7026

NLP Foundations and Text Representation

Dr. Seena Joseph

Outcome: describe how raw text is cleaned and turned into numbers, and build a simple text classifier.

TOPIC

- 1 What NLP is, and why language is hard
 - 2 Cleaning text: segmentation, tokens, stopwords, stems and lemmas
 - 3 NLP tasks and chatbots
 - 4 Turning text into numbers: counts, TF-IDF, embeddings
 - 5 A first classifier in Python
-

How would you teach a computer to spot a positive review?

“A wonderful film with a moving story.” “A dull film with a weak plot.”

Write down two rules a program could follow.

Then write one review that would fool your rules.

What is natural language processing?

A field of AI that enables computers to understand, interpret and generate human language.

UNDERSTAND

Read text

Classify it, extract facts from it, find what it is about.

INTERPRET

Resolve meaning

Use context to decide which meaning of a word is intended.

GENERATE

Write text

Translate, summarise or answer in fluent language.

Why language is hard for computers

AMBIGUITY

One word, many meanings

“I went to the dock to ship my parcel.” “I went to the station to ship my parcel.” The same verb implies a different journey.

NEGATION

Small words flip meaning

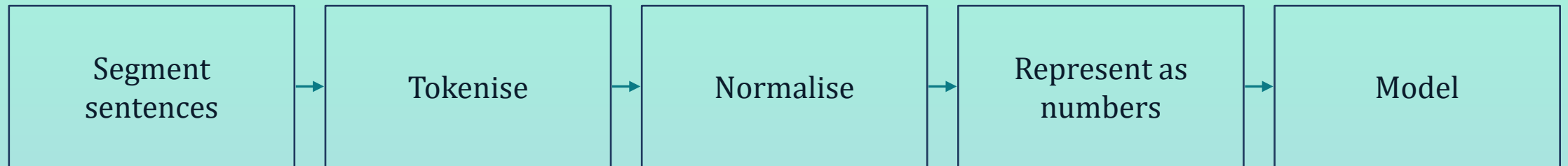
“Not bad” is praise. “Not a brilliant film” is criticism.

VARIATION

No fixed format

Slang, typos, abbreviations and new words appear constantly.

A classical NLP pipeline



Steps 1 to 3 prepare the text. Step 4 is the central problem: a model needs numbers.

Step 5 uses those numbers for classification, translation or generation.

Transformers (Next lecture) fold several of these steps into the model itself.

Sentence segmentation

“Dr. Smith arrived at 10 a.m. The meeting was productive! What did he say about the project?”

NAIVE: SPLIT AT EVERY FULL STOP

Breaks inside abbreviations

“Dr” “Smith arrived at 10 a” “m” “The meeting was productive! What did he say about the project?”

BETTER: USE A SEGMENTER

Three sentences

“Dr. Smith arrived at 10 a.m.” “The meeting was productive!” “What did he say about the project?”

Tokenisation

Split text into smaller units, called tokens, that can be counted and indexed.

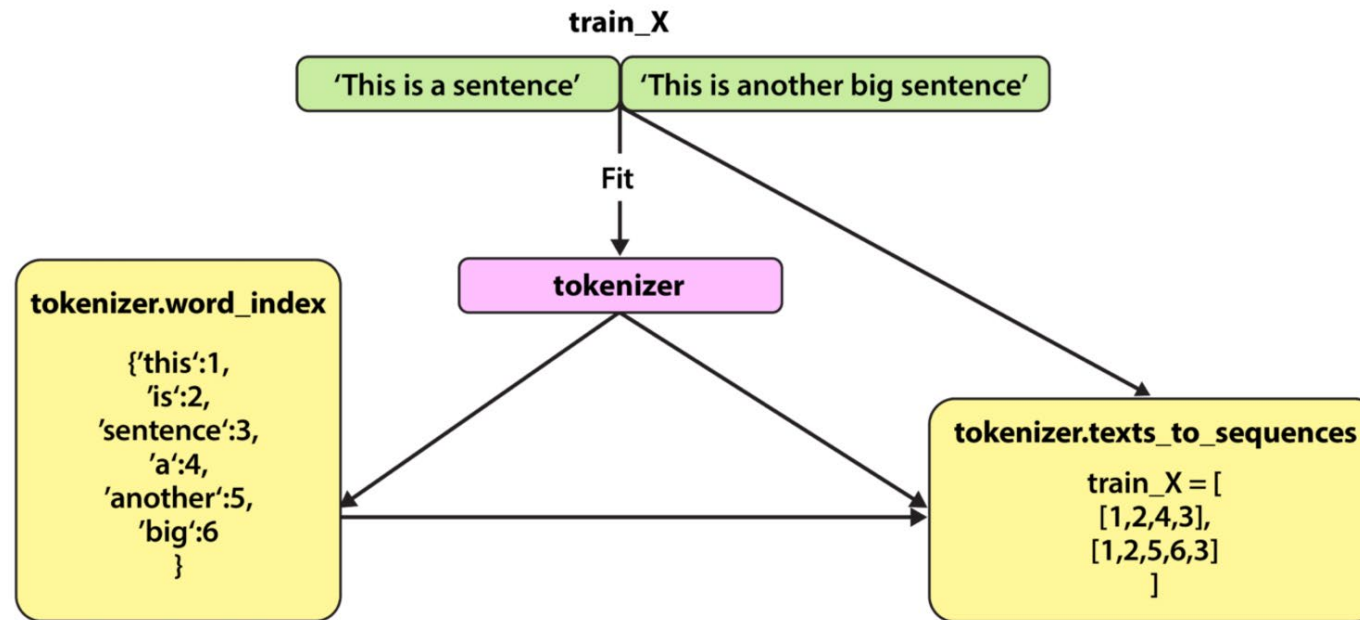
METHOD	RESULT FOR "DON'T PANIC, IT'S HERE!"
Split on spaces	Don't panic, it's here!
Word tokeniser	Do n't panic , it 's here !
Word index	Each distinct token gets a number, for example, Do 0 · n't 1 · panic 2 · , 3 · it 4 · 's 5 · here 6 · ! 7

Tokens can be words, phrases or characters. LLMs use subword tokens (Next Lecture).

Natural Language Processing (NLP)

Tokenisation

TOKENIZERS



Stopword removal

“The cat sat on the mat and looked at the sky.” → “cat sat mat looked sky”

IDEA

Drop very common words

Articles, conjunctions, prepositions and pronouns carry little topic information.

BENEFIT

Smaller, cleaner counts

Fewer features, and rare informative words stand out.

RISK

Some stopwords matter

Many lists contain “not” and “no”. Removing them can flip a sentiment.

Stemming and lemmatisation

WORD	STEM	LEMMA
running	run	run
studies	studi	study
studying	studi	study
happiness	happi	happiness
was	wa	be

Stemming chops endings by rule. Lemmatisation looks the word up in a dictionary. stemming chops endings by rule and may leave non-words; lemmatisation uses a dictionary and grammar to return a real base form.

Clean the sentence

“The staff were NOT helpful, and the queues were running very slowly!”

STEP	YOUR RESULT
1 Lowercase and tokenise	
2 Remove the stopwords: the, were, and, very	
3 Lemmatise the rest	

Then ask: did your cleaning keep the meaning? What would you change about the stopwords list?

Entities, relationships and concepts

ENTITY

A thing named in the text

“Show me hotels in Paris from 10 to 15 December with a pool.” Location: Paris. Check-in and check-out dates. Amenity: pool.

RELATIONSHIP

Entities linked together

“Smith’s birthday is on 2 January.” Person (Smith) is linked to date (2 January).

CONCEPT

An idea that is implied

“I will take the bus to work.” “Book me a train to London.” Concept: mode of transport.

Sentiment and emotion

NEGATIVE

NEUTRAL

POSITIVE



“A dull film with a weak plot.”

“The film is two hours long.”

“A wonderful, moving story.”

SENTIMENT

Where on the scale?

Negative, neutral or positive, sometimes with a score.

EMOTION

Which feeling?

Happiness, anger, fear and others.

What NLP is used for

TASK	INPUT → OUTPUT
Sentiment analysis	A review → negative, neutral or positive
Machine translation	Text in one language → the same meaning in another
Named entity recognition	Text → names, places and quantities, with positions
Grammatical error correction	A faulty sentence → a corrected sentence
Text generation	A prompt → new text in a chosen style
Toxicity classification	A comment → threat, insult, hate or none
Information retrieval	A query → the most relevant documents
Summarisation	A long text → a shorter text

Extractive and abstractive summarisation

Source: “Natural Language Processing enables machines to understand human language. It has many applications like sentiment analysis, translation, and summarisation.”

EXTRACTIVE

Select sentences

Copy the most important sentences unchanged:
“Natural Language Processing enables machines to understand human language.”

ABSTRACTIVE

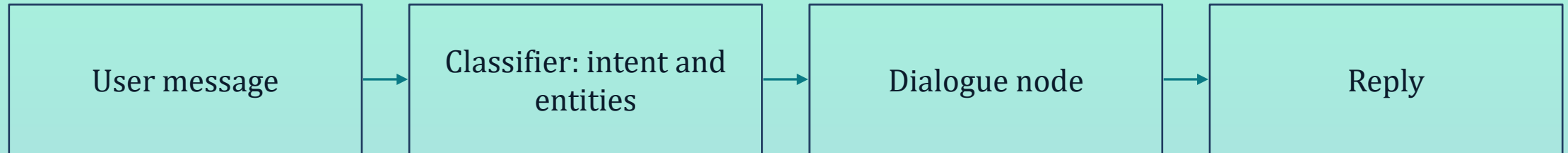
Write new sentences

“NLP helps machines comprehend human language, supporting tasks like sentiment analysis and summarisation.”

Extractive summarisation selects sentences or phrases that are already in the text and copies them unchanged. It works like a highlighter pen: the program scores each sentence for importance and keeps the top ones.

Abstractive summarisation writes new sentences that express the main ideas in different words. It works like a person who reads the text and then writes their own short version. It needs a language model that can generate text, normally a transformer.

How a chatbot works



INTENT

Why is the user here?

For example: renew a loan, report a lost card, find opening times.

ENTITIES

What details were given?

Item, date, place, name. They fill in the blanks of the answer.

DIALOGUE

A tree of IF-THEN nodes

Each node holds a statement and a list of expected replies.

Design the chatbot for a university library

USER MESSAGE	INTENT	ENTITIES	REPLY
“Is the library open on Sunday?”	Opening times	Day: Sunday	Sunday hours
“Can I renew the book I borrowed last week?”	Renew loan	Item: book; date: last week	Ask for the title
“Where is the Level 7 AI reading list?”	?	?	?
“I have lost my library card.”	?	?	?

Then list three more intents this chatbot needs, and one message it could not handle.

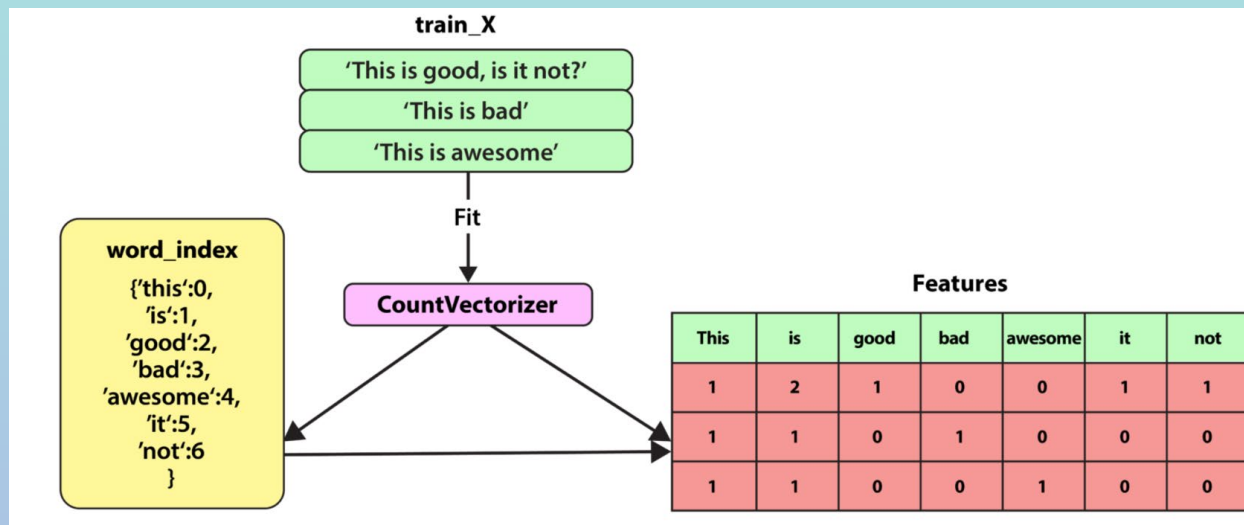
Natural Language Processing (NLP)

Feature Extraction

In NLP the feature extraction transform the text into a numerical representations that ML/DL algorithms can process.

Common Feature extraction techniques : Bag of Words and TF-IDF (Term Frequency Inverse Document Frequency)

Bag of Words: It counts the number of times each word or n-gram (combination of n words) appears in a document.



Bag of words

Count how often each vocabulary word appears in each document. Word order is discarded.

DOCUMENT	THE	FILM	WAS	GOOD	NOT
“the film was good”	1	1	1	1	0
“the film was not good”	1	1	1	1	1
“good good film”	0	1	0	2	0

Each row is a vector: the document becomes a point in a space with one axis per word.

The same idea works for n-grams, which count pairs or triples of words.

Natural Language Processing (NLP)

Feature Extraction

TF-IDF (Term Frequency Inverse Document Frequency):

TF-IDF, weights each word by its importance. Significance of the word is based on:

- **TF(word in a document)** – How important the word in the document?

TF (word in a document) = Number of occurrences of that word in document / Number of words in document.

- **IDF** – this measures how important is the word in the whole corpus? Or how unique the word across all.

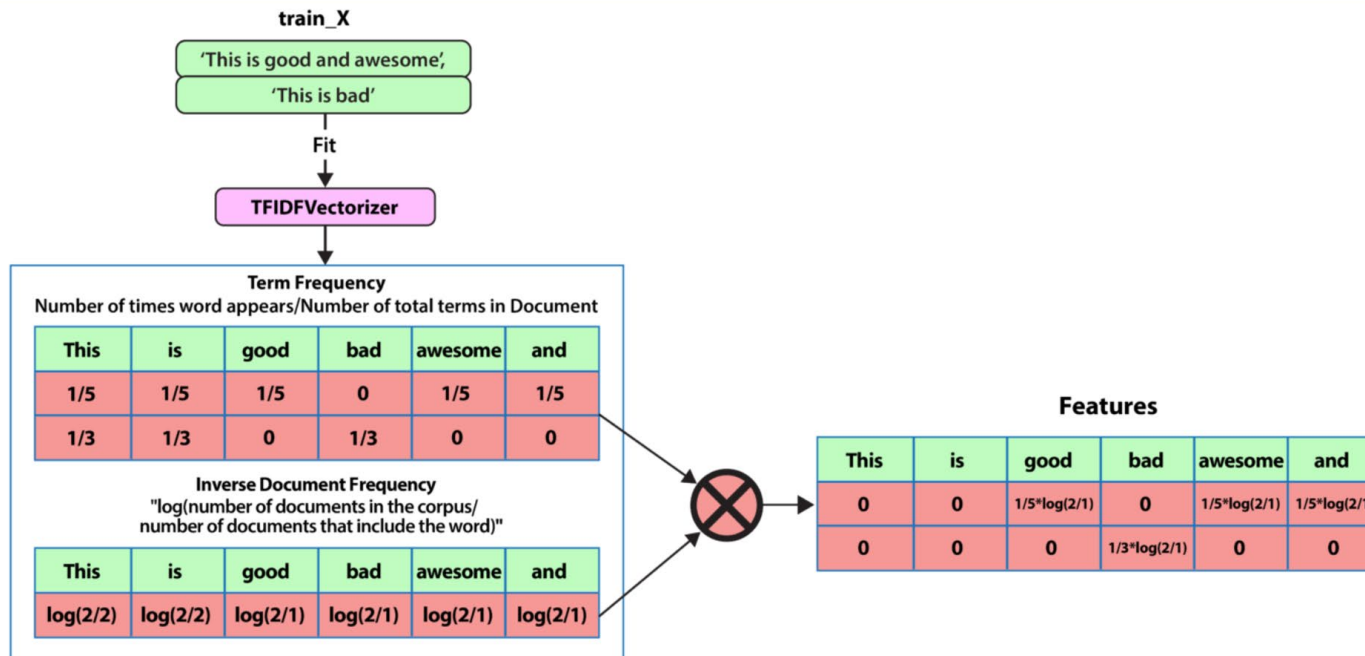
IDF(word in a corpus)= $\log(\text{number of documents in the corpus} / \text{number of documents that include the word})$

Natural Language Processing (NLP)

Feature Extraction

- TF-IDF score will be high if the word is rare and low if the word is common across the corpus. The TF-IDF score of a term is the product of TF and IDF.

TOKENIZERS: TERM FREQUENCY - INVERSE DOCUMENT FREQUENCY (TF-IDF)



Build the vectors

Vocabulary: the film was good not long

DOCUMENT

VECTOR

A “the film was long”

B “not good not good”

C “the long film was not good”

Which two documents are the most similar? Why?

Does “good not” give the same vector as “not good”?

TF-IDF

TF = count ÷ words in the document. IDF = $\ln(\text{documents} \div \text{documents containing the word})$.

Worked for “the film was not good” in the three-document set from the previous slides.

WORD	TF	DOCUMENTS WITH IT	IDF ($\ln(\text{Total docs/no of docs with words})$)	TF × IDF
the	0.2	2 of 3	0.41	0.081
film	0.2	3 of 3	0.00	0.000
was	0.2	2 of 3	0.41	0.081
not	0.2	1 of 3	1.10	0.220
good	0.2	3 of 3	0.00	0.000

A word found in every document scores zero. A rare word scores high.

What counts cannot see

ORDER

“Dog bites man” = “man bites dog”

Same words, same counts. N-grams (word pairs) recover some order.

MEANING

“Film” and “movie” are different columns

Synonyms share nothing. Unseen words get no value at all.

SIZE

Thousands of mostly-zero columns

Big vocabularies make large, sparse vectors.

To capture meaning, we need vectors that place related words close together.

Word embeddings: Word2Vec and GloVe

An embedding is a short list of numbers for each word, learned from the text it appears in. Instead of one column per word, each word gets a short list of numbers that places it on a map of meaning.

The problem	The idea	The result
In bag of words, “film” and “movie” are two separate columns. The computer sees no link between them.	Learn the numbers from lots of text. Words used in similar contexts get similar numbers. “I watched a great film last night” and “I watched a great movie last night” have the same words around the target, so film and movie end up close.	Close vectors mean similar use, which usually means similar meaning.

Words used in similar contexts get similar vectors: “a word is known by the company it keeps”.

Real embeddings have 50 to 300 dimensions.

Word embeddings: Word2Vec and GloVe (Global Vectors for Word Representation)

An embedding is a short list of numbers for each word, learned from the text it appears in. Instead of one column per word, each word gets a short list of numbers that places it on a map of meaning.

WORD	DIM 1	DIM 2	SIMILARITY TO “FILM”
film	0.90	0.20	1.00
movie	0.85	0.30	0.99
river	0.10	0.90	0.32

Words used in similar contexts get similar vectors: “a word is known by the company it keeps”.

Real embeddings have 50 to 300 dimensions.

Counts or embeddings?

A: “The film was excellent”

B: “The movie was superb”

1. Build bag-of-words vectors for A and B. How many words overlap?
2. Remove the stopwords (the, was). What is the overlap now?
3. Which representation would rate A and B as similar, and why?

The limit of static embeddings

Each word gets one vector, whatever the sentence around it.

PROBLEM

One vector for “bat”

“The bat flew out of the cave.” “She swung the bat.” The animal and the sports bat share a vector.

CAUSE

Fixed after training

The vector is looked up in a table.
The context cannot change it.

NEXT STEP

Contextual embeddings

Transformers compute a new vector for each word in each sentence (Next Lecture).

From features to a classifier

FEATURES	TYPICAL MODEL	WHERE
Bag of words or TF-IDF	Logistic regression, decision tree	Today
Embeddings	Neural network (CNN, RNN, LSTM, GRU)	Practical task
Pretrained language model	Fine-tune a transformer	Lectures 6 and 8

Same job in every row: map a text to a label.

Landmark NLP systems

YEAR

SYSTEM

IDEA

Activity : Complete this Table

Classify reviews with deep learning

STEP

- 1 Choose a labelled review dataset (for example Kaggle or Yelp) with enough reviews to train
 - 2 Preprocess the text, as in today's lecture
 - 3 Represent it: TF-IDF or embeddings
 - 4 Build a model: CNN, RNN, LSTM, GRU or an ensemble
 - 5 Train, then evaluate on held-out data
 - 6 Add a TF-IDF and logistic regression baseline to compare
-

Five questions

1. What does tokenisation produce?
2. Why can removing stopwords harm sentiment analysis?
3. A word appears in every document. What is its IDF?
4. Do “dog bites man” and “man bites dog” have the same bag-of-words vector?
5. Give one limit of static word embeddings.
6. Explain the difference between stemming and lemmatisation. Give one example of each.
7. State two limitations of a bag-of-words representation

Three things to remember

1

Clean with care

Segment, tokenise and normalise, but keep what carries meaning, such as “not”.

2

Text becomes numbers

Counts, TF-IDF or embeddings. Each step adds more meaning.

3

Order and context are the gap

Counts and static embeddings miss both. Lecture 6 addresses this.

Further reading

[NLP: A Complete Guide \(DeepLearning.AI\)](#)

[How to make a chatbot with Google Dialogflow](#)

P. A. Olujimi and A. Ade-Ibijola, “NLP techniques for automating responses to customer queries: a systematic review,” Discover Artificial Intelligence, vol. 3, no. 1, 2023, doi:10.1007/s44163-023-00065-5.

Activity script: [`NLPSample.ipynb`](#); Handout