

# Introduction, Tokens & Document Similarity

Why do computers need to work with text?

Jake Lever & Sean MacAvaney

1111101110101001111  
0100100100111101001001  
·0111101110·00100111011101  
100100100111.0·+·1·<001111  
11101110100100·00·110130·001  
·0100100111101110100100111011101  
0111010010010011101110101001111  
0100111010010011101110101001111  
0100111010010011101110101001111  
10100·00·0011101100·0·110110101  
111101110100100100111101110101001111  
0100100111101110100100100111101001001

Text  
as Data



University  
of Glasgow | School of  
Computing Science

# Welcome!



Today we will talk about:

- When do computers need to work with text?
- Overview of the course
  - What will we learn in this course
  - Course practicalities (e.g. labs, assessments, etc)
- How can computers find similar documents?

# Text As Data lecturers this year



**Dr Jake Lever**

*Teaching Honours (4074) / MSci (5096)*  
Research: Biomedical Text Mining



**Dr Sean MacAvaney**

*Teaching MSc (5106)*  
Research: Information Retrieval & Natural Language Processing



# Why make computers work with text?

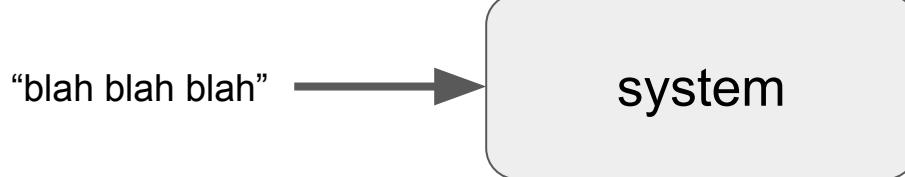
- Humans interact with one another through language
- Language can be represented as text
- This makes text a convenient interface between humans and computers



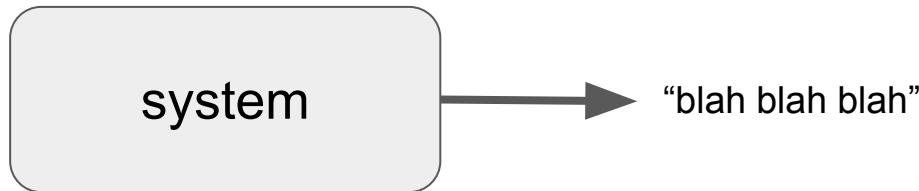


# Computers may use text as input or output or both

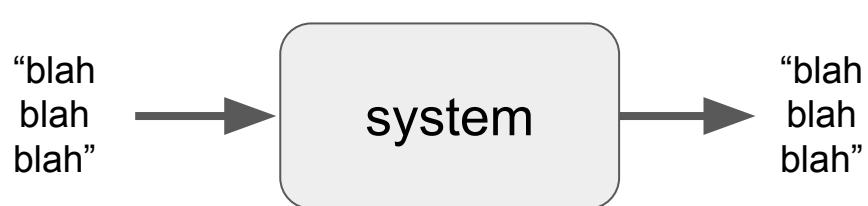
As Input



As Output

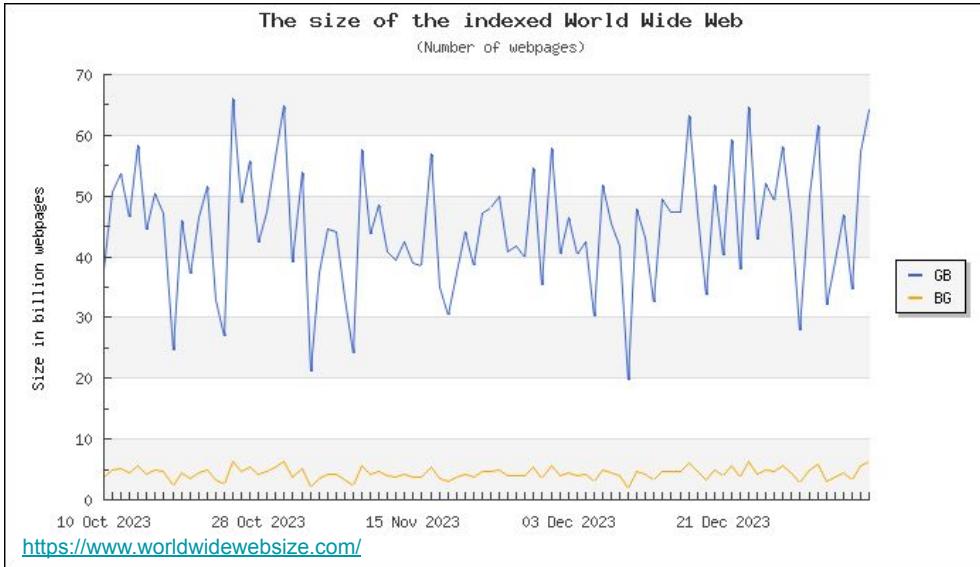


Or Both Input  
and Output?





# Text data is ever growing and we need to work with it



## Every minute

- Email users send ~200M messages (<https://www.statista.com/statistics/456500/daily-number-of-e-mails-worldwide/>)
- Google receives ~6M search queries (<https://seo.ai/blog/how-many-people-use-google>)



# Text data is unstructured, so hard to process!

- Structured data - think tables or databases
  - Column headings provide labels on the meaning of each cell
- Unstructured text - a section of text with no labels to provide meaning

“With 67 million people, France is the second most populous country in the European Union. Its capital is the beautiful city of Paris and since 2002, the country has used the Euro. Germany also uses...”

Country	Capital	Currency	Population
France	Paris	Euro	67,000,000
Germany	Berlin	Euro	83,000,000
Canada	Ottawa	Canadian Dollar	38,000,000

**Unstructured data**

**Structured data**



# Think-Pair-Share: Comparing structured/unstructured

"With 67 million people, France is the second most populous country in the European Union. Its capital is the beautiful city of Paris and since 2002, the country has used the Euro. Germany also uses..."

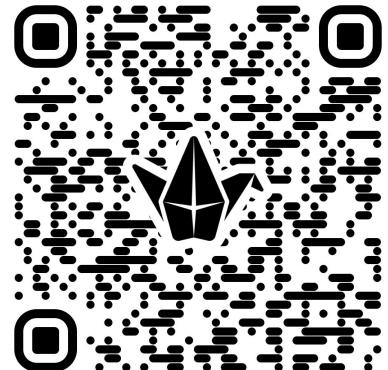
Country	Capital	Currency	Population
France	Paris	Euro	67,000,000
Germany	Berlin	Euro	83,000,000
Canada	Ottawa	Canadian Dollar	38,000,000

## Unstructured data

In your group, discuss the following and add your notes to the Padlet

- You are building a search system.
  - Why might you prefer to have structured data?
  - Why might you prefer to have unstructured data?

## Structured data



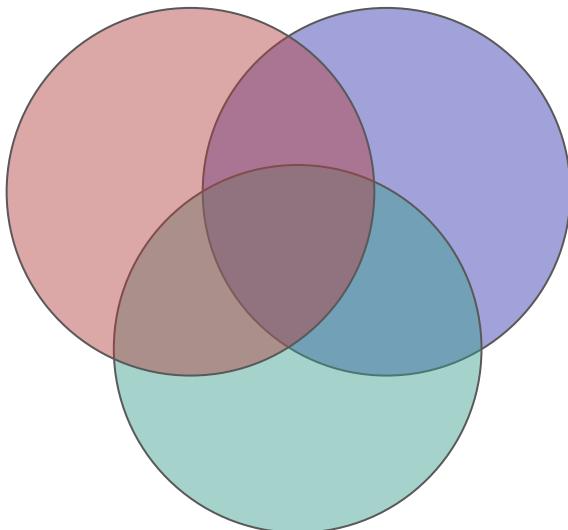
[https://padlet.com/jakelever/tad2025\\_1](https://padlet.com/jakelever/tad2025_1)



# Names of this field

Just so you are aware: This area of computer science has a few names that overlap:

- Natural language processing (NLP)
- Computational linguistics
- Text Analytics



# Language follows rules. Can't we decode those rules?

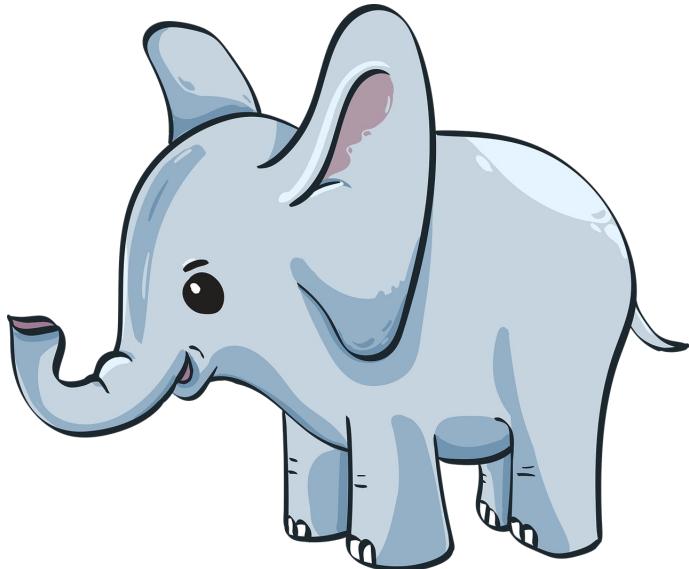
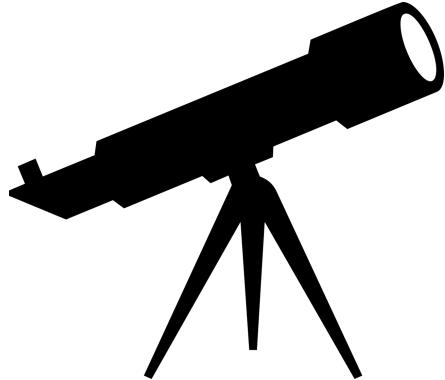


Dealing with text can't be that hard. Language follows rules that I learnt in school. I'll write those rules with some pattern matching and job done!

No, language is more than just rules (though they help). It uses prior knowledge and reasoning.



Language is beautiful and also a massive pain

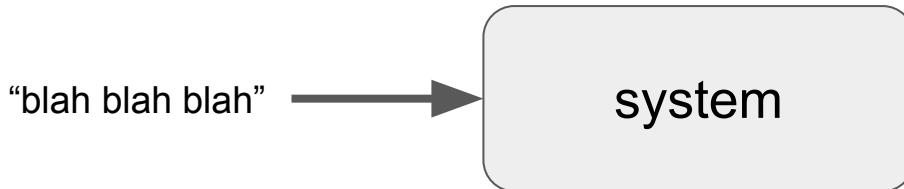


“I saw the **elephant** with my telescope”

*Who has the telescope?*



# Examples applications using Text as Input



Google News search results for "James Webb Space Telescope":

- Full coverage: James Webb Space Telescope was launched into space
- Top coverage:
  - THE VERGE**: James Webb Space Telescope: what to expect from the NASA mission (23 Dec)
  - SatelliteProME.com**: James Webb Space Telescope sunshield tensioning begins today (Yesterday)
  - NewScientist**: Nasa's Webb telescope is a joy. But it's the private ventures that push at limits | Martin Rees (Yesterday)
  - TweakTown**: NASA drops update on James Webb Space Telescope unfolding (4 hours ago)

News Aggregation



Search Tools

Draft saved

Recipients

Subject

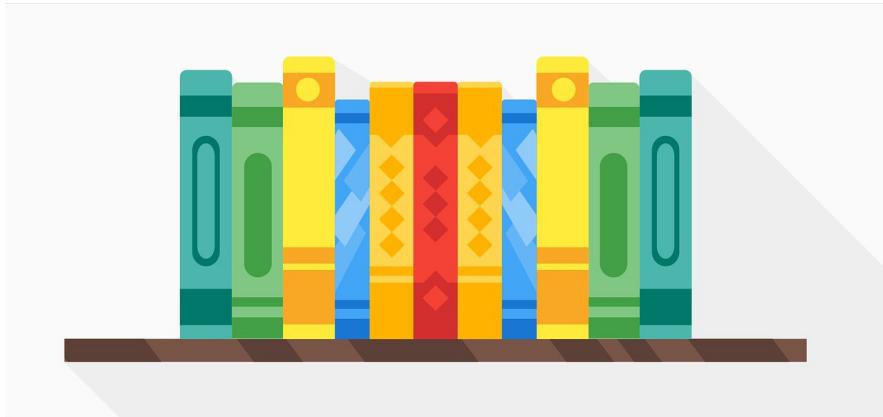
Dear Dr Smith, I hope this email finds you well

Email Suggestions



# Examples of text inputs

- Documents
- Tweets
- Voice commands
- Search queries
- Web pages
- Medical records
- Books
- so much more





# Example Task: Document similarity

Can we find other Reddit posts that are similar to the one below?

## Anyone tried Irn Bru?

It's a Scottish drink and it's banned some countries and I was wondering if anyone here has tried it. It has quite a unique taste and it's not something you'd forget quickly. You either love it or hate it I think.

 29 Comments

 Award

 Share

 Save

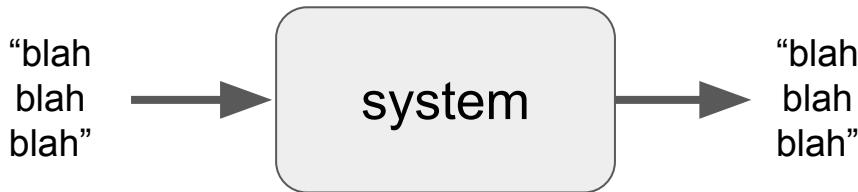
 Hide

 Report

100% Upvoted



# Example applications using Text as Input and Output



Assistants (e.g. Siri, Alexa)

*“Où est la bibliothèque?”*



*“Where is the library?”*

Machine Translation

Draft saved

Recipients

Subject

Dear Dr Smith, I hope this email finds you well

Email Suggestions

We will touch on text as output - but the main focus  
of this course is taking text as input!



# Basic text output: Generating numbers with rules

Some data (e.g. numbers) can be turned into their textual form with a number of rules

3 → “three”

43 → “forty-three”

143 → “one hundred and forty-three”

```
✓ [1] !pip install inflect  
3s
```

Requirement already satisfied: inflect in /usr/local/lib/python3.7/dist-packages (2.1.0)

```
✓ [2] import inflect  
0s  
p = inflect.engine()  
p.number_to_words(1234)
```

'one thousand, two hundred and thirty-four'



# Advanced text output: Creative writing



[STARGATE ATLANTIS](#) [STARGATE SG-1](#)

## Stargate: Cast Reunite To Read a Google AI Written Script

Is AI a friend or foe when it comes to sci-fi writing?

Posted: 19 Oct 2021 6:00 pm

## I Wrote a Book with GPT-3 AI in 24 Hours — And Got It Published

In early 2021 I signed up for the GPT-3 beta program to see how good it is. A few days later I had co-authored *Aum Golly* — a book of AI poems on humanity. A few months later it was published. This is what it means for writers and publishers.

<https://www.ign.com/articles/stargate-google-ai-script>  
<https://medium.com/swlh/i-wrote-a-book-with-gpt-3-ai-in-24-hours-and-got-it-published-93cf3c96f120>



Jukka Aalho

Follow



Jun 12, 2021 · 8 min read ★





# Examples of text output: An AI-written text adventure game

Xaxas is a world of peace and prosperity. It is a land in which all races live together in harmony. The gnomes build their machines and live among the humans. Elves run the academies in which people come to learn. Ogres have no qualms living in the same town as humans or gnomes. Yet, all is not well.

You are Silvera, a female gnome paladin. You are at home in the beautiful city of Deccos living among the humans and making sure that they don't act out of line or try to change secret messages the gnomes spread to the homes and businesses in the town. You walk through the village square when suddenly a brawl breaks out between a group of humans and a group of gnomes. A few of these gnomes were sitting on a wooden cart and apparently they decided to take matters into their own hands. You happen to be passing by at the exact moment that they decided to break out the weapons.



Do

What do you do?



<https://play.aidungeon.io>



# Text as Data has a long exciting history as a research area

- Building with linguistics research
  - How does language work?
  - How do we learn language?
- Tied to computational performance
  - New CPUs and GPUs have enabled new advances
- The Internet provides an incredible source of example text
- Deep learning is changing the whole approach!
- The methods are sufficiently mature for high-profile products like ChatGPT, Alexa, etc.



**You chose the right time to start studying it!**



# New amazing abilities with language

New language systems are trained by asking them to complete a sentence

It showed 9 o'clock on my \_\_\_\_\_

# Text researchers became obsessed with Muppet characters



**BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding**

Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova  
Google AI Language  
[jacobdevlin,mingweichang,kentonl,kristout@google.com](mailto:jacobdevlin,mingweichang,kentonl,kristout@google.com)

**Abstract**

We introduce a new language representation model called **BERT**, which stands for Bidirectional Encoder Representations from Transformers. Unlike most language representation models (Peters et al., 2018a; Radford et al., 2018), BERT is designed to pre-train language representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications.

BERT is conceptually simple and empirically powerful. It outperforms state-of-the-art models on eleven natural language processing tasks, including pushing the GLUE score to 80.5% (7.7% point absolute improvement), MultiNLPI accuracy to 86.7% (4.6% absolute improvement), and SQuAD 1.1 question answering Test F1 to 93.2 (1.5 point absolute improvement) and SQuAD v2.0 Test F1 to 83.1 (5.1 point absolute improvement).

## 1 Introduction

Language model pre-training has been shown to effectively improve many natural language processing tasks (Dai and Le, 2015; Peters et al., 2018a; Radford et al., 2018; Howard and Ruder, 2018). These include sentence-level tasks such as natural language inference (Bowman et al., 2015; Williams et al., 2018) and paraphrasing (Dolan and Brockett, 2005), which aim to predict the relationships between sentences by analyzing them holistically, as well as token-level tasks such as named entity recognition (NER) and reading comprehension, where models are required to produce fine-grained output at the token level (Tjong Kim Sang and De Meulder, 2003; Rajpurkar et al., 2016).

4171  
Proceedings of NAACL-HLT 2019, pages 4171–4186  
Minneapolis, Minnesota, June 2 – June 7, 2019. ©2019 Association for Computational Linguistics

The ELMo and BERT deep learning approaches developed a new model for deep learning that could succeed at several different problems



# The GPT-4 Hype

OpenAI's GPT-4 deep learning model showed incredible new abilities

Examples at:

<https://platform.openai.com/docs/examples>



## Emoji Translation

Transform

Natural Language

Translate regular text into emoji text.

### Prompt

#### SYSTEM

You will be provided with text, and your task is to translate it into emojis. Do not use any regular text. Do your best with emojis only.

#### USER

Artificial intelligence is a technology with great promise.

### Sample response





# Github Copilot

Language models can work with code as well as written language.

See <https://copilot.github.com/>

The screenshot shows a GitHub Copilot interface. At the top, there are three tabs: 'draw\_scatterplot.js' (selected), 'draw\_scatterplot.py', and 'draw\_scatterplot.ipynb'. The main area displays Python code for drawing a scatterplot:

```
1 import matplotlib.pyplot as plt
2
3 def draw_scatterplot(x_values, y_values):
4     plt.scatter(x_values, y_values, s=20)
5     plt.title("Scatter Plot")
6     plt.xlabel("x values")
7     plt.ylabel("y values")
8     plt.show()
```

A blue box highlights the line 'plt.scatter(x\_values, y\_values, s=20)'. In the bottom left corner, there is a blue button labeled 'Copilot' with a person icon.

## Prompt

```
# Python 3
def remove_common_prefix(x, prefix, ws_prefix):
    x["completion"] = x["completion"].str[len(prefix):]
    if ws_prefix:
        # keep the single whitespace as prefix
        x["completion"] = " " + x["completion"]
    return x

# Explanation of what the code does

#
```

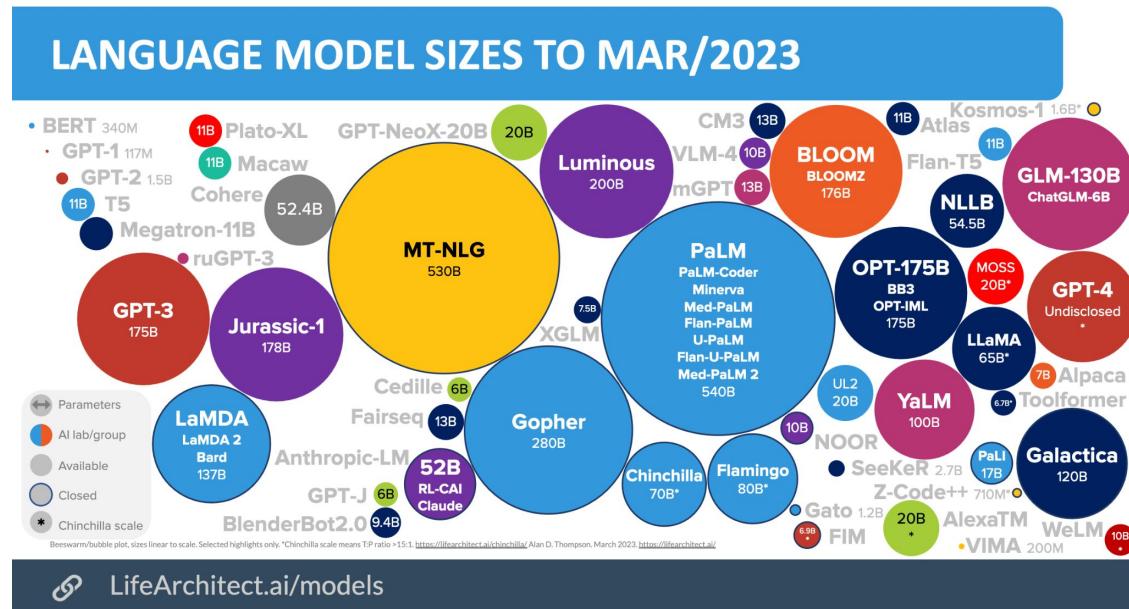
## Sample response

The code above is a function that takes a dataframe and a prefix as input and returns a dataframe with the prefix removed from the completion column.



# This is a fast moving field - GPT/BERT were in 2018!

- In the past, bigger (and more complex) models haven't translated into better performance
- With transformer-based models, bigger (e.g. more parameters & data) seems to be better, for now.
- But bigger models come with huge costs: training, computational, data, environmental, etc.





# There is a lot of hype about AI “understanding” text

FORTUNE

Most Popular



GM's Impala ad is a viral hit—and it could become a patriotic Christmas classic



PAID CONTENT  
Why a design-first mentality is critical for fintech  
FROM INVESTCLOUD



TECH • ARTIFICIAL INTELLIGENCE

## DeepMind debuts massive language A.I. that approaches human-level reading comprehension

BY JEREMY KAHN

December 8, 2021 4:00 PM GMT

- Be skeptical of claims of human-level abilities
- Most systems may be very good at specific task, but research on reliable generalisation is ongoing
- Models that appear to generalise are often already trained on the task (among many others)

BBC Jake



Home

News

Sport

Weather

iPlayer

Sounds

# NEWS

Home | Coronavirus | Climate | UK | World | Business | Politics | Tech | Science | Health | Family & Education

Technology

## Computer AI passes Turing test in 'world first'

© 9 June 2014



<https://fortune.com/2021/12/08/deepmind-gopher-nlp-ultra-large-language-model-beats-gpt-3/>

<https://www.bbc.co.uk/news/technology-27762088>

# Playing with Text Generation



Technology

## ChatGPT: New AI chatbot has everyone talking to it

7 December 2022



GETTY IMAGES

<https://chat.openai.com>

JA

Write a poem about natural language processing



Natural language processing,  
A field both complex and fair,  
It helps us understand,  
The words that people share.

From chatbots to translators,  
It's used in many a tool,  
To make our lives easier,  
And communication more cool.

It breaks down language barriers,  
And helps us all connect,  
A world united by words,  
Is a world that we can respect.

So let's embrace NLP,  
And all it has to give,  
With its help, we'll go far,  
And continue to live and thrive.

<https://www.bbc.co.uk/news/technology-63861322>



# Summary of Text as Data Introduction

- Computers are now frequently using text as input and output
  - Inputs can be search queries, documents to process, voice commands, etc
  - Outputs can be text summaries, creative writing and more!
- The amount of text data is growing at a substantial rate every day
- Deep learning has had some recent big impact on the field with transformers
- These new systems have impressive abilities both in interpreting and generating text
- This field is VERY fast moving

---

# Course Details

# Syllabus Overview



- **Representing text:** traditional methods using sparse vectors
- **Clustering:** how do you find a document similar to another?
- **Principles of language modelling:** the probability of one word following another
- **Classifying text:** for example, sentiment analysis
- **Representing text with dense numerical vectors:** word2vec and contextual methods
- **Principles of transformers and deep learning**
- **Named entity recognition & linking:** Find words that describe specific people, objects, locations, etc
- **Extracting relations:** How are the 2+ entities in the text related?
- **Coreference:** Making links between mentions that are referring to the same thing
- **Ethical & societal considerations:** Are we okay with the applications and implications of these technologies?
- **Large language models:** What innovations give generative AI amazing abilities?

# Technical Content



- Python
- Google Colab (a branch of Jupyter Notebook)
- Specific libraries
  - spacy
  - scikit-learn
  - transformers (from HuggingFace)





# The Moodle

- Will contain links to:
  - Lecture slides
  - Recordings (soon after each lecture) & no password
  - Labs
- Coursework and Exam information (in the future)
- Place for any announcements

Honours/MSci: <https://moodle.gla.ac.uk/course/view.php?id=38671>

# Teams



- Feel free to post questions to the Teams channel
- Various resources will be shared through it so keep an eye on it

**If you are not a member,  
contact me and I can add you**





# Textbook and Other Materials

No required textbook for this course

Optionally, you can read more in:

- Speech and Language Processing (3rd ed. draft)
  - Dan Jurafsky and James H. Martin
  - <https://web.stanford.edu/~jurafsky/slp3/>
- scikit-learn documentation is very good!
  - [https://scikit-learn.org/stable/user\\_guide.html](https://scikit-learn.org/stable/user_guide.html)



# Grading

Weight	Assessed Exercise
20%	Coding Coursework
80%	Exam

**Honours (COMPSCI4074)**

Weight	Assessed Exercise
20%	Coding Coursework
10%	Literature Review
70%	Exam

**MSci (COMPSCI5096)**

- Assessed Exercises
  - Different weighting for Honours or MSci
  - Due in early March
  - Coding coursework (20%)
  - Literature review coursework (10% for Masters students only)
- Exam
  - In May
  - More challenging for MSci

# Labs



**Location:** Boyd Orr 720 (H, MSci)

**Time:** Tuesday mornings  
(9am, 10am or 11am - check your enrollment)

- There is a mini-lab (Lab 0) online to get used to Colab and the data.
- Material in labs is examinable, including knowledge of basic libraries



# Office Hours



I will be available on **Thursdays at 4pm-5pm**. You can come to my office (SAWB 331 - directions in the Moodle) or message me on Teams to chat online.

I'm also happy to chat after the lecture for a short time.

I will not be able to respond quickly to questions outside lecture, lab times and office hours





# Bug Bounty

Report any mistake in  
lectures/labs/coursework/etc as an  
issue on GitHub:

[https://github.com/jakelever/textasdata\\_bugbounty](https://github.com/jakelever/textasdata_bugbounty)

Please keep it polite.

Prizes at end of semester for most  
bugs reported: 1st (£30), 2nd (£20)  
and 3rd (£10).

Link is on the Moodle.



# Course Updates this Year



1. Lots of small tweaks
  - Small fixes to labs
  - Remove a few confusing parts of labs
2. Change in amount of course content?
  - This is a content-heavy course!
3. Updates to LLM content (later in course)

*No big changes this year*

# Important Idea: Different approaches are good for different problems



- This course teaches a number of different approaches to text problems
- No approach is the best for all problems
- Each approach has strengths and weaknesses
  - Often accuracy versus computational cost
- To know which is the best for your problem, you need to set up an experiment so you can compare methods





# Learning Outcomes

Describe classical models for textual representations such as the one-hot encoding, bag-of-words models, and sequences with language modelling.

Identify potential applications of text analytics in practice.

Describe various common techniques for classification, clustering and topic modelling, and select the appropriate machine learning task for a potential document processing application.

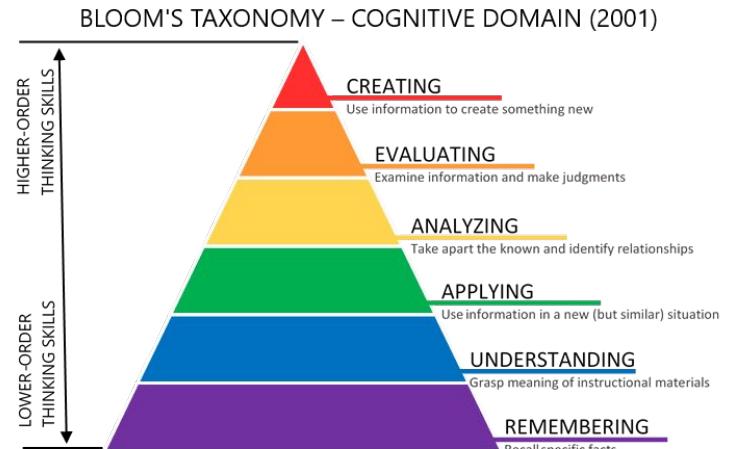
Represent data as features to serve as input to machine learning models.

Deploy unsupervised and machine learned approaches for document/text analytics tasks.

Critically analyze and critique recent developments in natural language and text processing academic literature.

Assess machine learning model quality in terms of relevant error metrics for document processing tasks, in an appropriate experimental design.

Evaluate and explain the appropriate application of recent research developments to real-world problems.





# Types of Feedback



- Discussions during lab time (from me and our TAs)
- Labtest system in labs you can use to check your progress
- Written feedback on the coursework submissions



# Organisation Summary

- Lectures on Tuesday from 1pm til 3pm
- Labs on Tuesday mornings
  - On Google Colab
  - Not marked
- Coursework due in early March
  - Plus literature review for MSci students
- Exam in May
- Office Hours at Thursdays at 4pm

---

**BREAK!**

---

# Document Similarity

# Example Task: Document similarity

Can we find other Reddit posts that are similar to the one below?

## Anyone tried Irn Bru?

It's a Scottish drink and it's banned some countries and I was wondering if anyone here has tried it. It has quite a unique taste and it's not something you'd forget quickly. You either love it or hate it I think.

 29 Comments

 Award

 Share

 Save

 Hide

 Report

100% Upvoted



# Think-Pair-Share: Approaches to document similarity

1. Which of the two posts (A or B) is closest to our post of interest? Why?
2. How might you code it?

**Post of Interest**

Anyone tried Irn Bru?

**Option A**

It's a Scottish drink and it's banned some countries and I was wondering if anyone here has tried it. It has quite a unique taste and it's not something you'd forget quickly. You either love it or hate it I think.

29 Comments    Award    Share    Save    Hide    Report

100% Upvoted

**What is the worst or some of the worst sodas you have drunk**

**Option B**

The absolute worst soda ive ever had that i can remember is probaly the new mystery fanta or watermelon+strawberry tango some other ones include mango coke, sugar free irn bru (but xtra is nice)

35 Comments    Award    Share    Save    Hide    Report

100% Upvoted

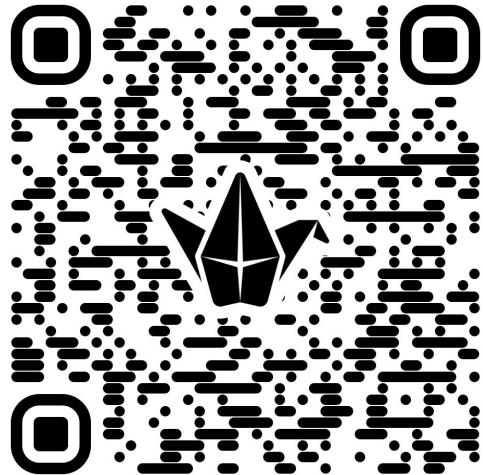
I once had a box of tea that I believe was Scottish Highland black tea. Can anyone recommend me a tea along those lines?

**Option B**

I'd like to repurchase this tea but I cannot find the exact box and don't want to buy something I am not sure will be good!

6 Comments    Award    Share    Save    Hide    Report

80% Upvoted



[https://padlet.com/jakelever/tad2025\\_2](https://padlet.com/jakelever/tad2025_2)

# Possible Solution



**If two posts have overlap, we hypothesise that they are talking about a similar subject.**

**The more overlap, the higher similarity**



# Data cleaning is often necessary

Text documents often contain a lot of formatting, links and other content that needs to be removed or reformatted (depending on what you need).

- Bold, italics, etc may need to be removed
- URLs in links may be need to be removed
  - Or could be valuable information depending on what you want to do
- Tables could be removed in whole, or just the formatting



111



111



r/NintendoSwitch

Full list of upcoming games on the Nintendo Switch (US) (Updated 1/1/2022)

[Discussion](#)

Console exclusives (games that are also on PC and/or mobile, but not on other consoles) in *Italics*.  
Nintendo exclusives (games that are only on Nintendo platforms) in **bold**.

For those looking at this list and not sure what's likely to be noteworthy, I have compiled [a page for noteworthy releases in January](#). Please give them a look if you want to see what games are likely to be some of the month's highlights!

As for the full list of upcoming games, here you go:

Games	Release date	Date confirmed by
Crumble	1/3/22	<a href="#">Nintendo.com</a>
Freddi Fish 3: The Case of the Stolen Conch Shell	1/3/22	<a href="#">Nintendo.com</a>
Putt-Putt Travels Through Time	1/3/22	<a href="#">Nintendo.com</a>
Arcadia Fallen	1/5/22	<a href="#">Nintendo.com</a>
Breakneck City	1/5/22	<a href="#">Nintendo.com</a>
McDroid	1/5/22	<a href="#">Nintendo.com</a>
Demon Gaze Extra	1/6/22	<a href="#">Official Website</a>
Enir Drumster Bear 2: He Who Bears	1/6/22	<a href="#">Nintendo.com</a>

# How do we measure overlap between two documents/posts?

1. Individual letters?
2. Short sequences of letters?
3. Words?



# 1. Individual letters?

Computers represent text as raw numbers for each character. Do two posts with similar underlying data have similar meaning?

I	R	N		B	R	U
73	82	78	32	66	82	85

*ASCII representation of “IRN BRU”*

**Not really.** The order and grouping of letters is key.

After all, “forty five” and “over fifty” share the same letters but have different meanings



# Text Encodings: ASCII and Unicode

To be experts in text data, you should be aware of ASCII and Unicode:

**ASCII:** 128 possible characters (in one byte)

- A-Z, a-z, 0-9, standard punctuation, various control characters (e.g. new line)

But what about: á, λ and 😊? Enter: **Unicode!**

**Unicode:** A variable-byte (2-4 bytes) encoding

- Encodes the alphabets of many languages
- Emojis, maths symbols and lots more
- Is frequently updated with new characters (v16.0 released in Sept 2024)

Other encodings exist but **ASCII** and **Unicode** are the important ones.



# Unicode issues can cause problems

Being unaware of Unicode can lead to odd character errors (known as mojibake):  
“The Mona Lisa doesn’t have eyebrows.”

Python has the “encoding” parameter when loading files. UTF-8 (the normal Unicode option) is default on Mac and Linux, but not on Windows.

```
▶ with open('unicode_file.txt', encoding='utf-8') as f:  
    contents = f.read()
```

A detailed Unicode explainer for when Unicode causes you problems:  
<https://docs.python.org/3/howto/unicode.html>



# Neat use of individual letters: Detecting other languages



- Different languages use different letters at different frequencies
- This property can be used to detect other languages (though there are better ways 😊)
- Frequencies of pairs/triples (e.g. n-grams) of letters can provide more data
- However, individual letters are not typically used for text analysis



## 2. Sequences of letters? (also known as character n-grams)

Individual letters aren't very useful, but some sequences can be quite specific: e.g. "irn"

Bi-grams (n=2) are pairs of neighbouring characters

e.g. "ir", "rn", "n ", " b", "br" "ru"

Tri-grams (n=3) are triples of neighbouring characters

e.g. "irn", "rn ", "r b", " br", "bru"

You can extract n-grams by running a moving window across the text (see right)

It's a Scottish drink and it's  
It's a Scottish drink and it's

*Sliding window showing  
character 4-grams*



# Strengths & Weaknesses of Character N-grams

## Strengths

- More specific than individual letters, e.g. tri-grams can capture short words
- Easy to implement

## Weaknesses

- Using small n (e.g. bigrams) won't be specific enough for many words
- Large n may create n-grams that are very rare
- Same n-gram can come from different words with different meanings
  - “ong” is in “wrong” and “strong”

### 3. Words

Given a section of text, how do we split it up into “words”?

## Tokenization

- Splitting a section of text into individual **tokens**
- A token (or term) is the technical name for a meaningful sequence of characters
- A token isn't exactly a word.
  - A word may be broken into separate tokens if it meaningfully makes sense (e.g. “don’t” is often split into “do” and “n’t”)
  - Tokens may also be punctuation, numbers, some whitespace or other spans of text
- Other languages can have very different tokenization challenges:
  - 授人以鱼不如授人以渔。

How would you tokenize text in English?



# Tokenizing with some handmade rules

Example sentence: “John’s father didn’t have £100.”

**Idea 1:** Splitting with whitespace (spaces, tabs, etc)

John's	father	didn't	have	£100.
--------	--------	--------	------	-------

**Idea 2:** Splitting with whitespace + punctuation (.,;!:#\$%, etc)

John	'	s	father	didn	'	t	have	£	100	.
------	---	---	--------	------	---	---	------	---	-----	---

**Ideal Tokenization:** Requires some handcrafted rules to get:

John	's	father	did	n't	have	£	100	.
------	----	--------	-----	-----	------	---	-----	---



# Tokenizing with Spacy

Good tokenization requires a lot of hand-crafted rules. Fortunately there are many good implementations out there. In labs, we will use Spacy!

# spaCy

Other popular toolkits:

- Stanford CoreNLP
- nltk

```
[1] import spacy  
  
[2] nlp = spacy.load("en_core_web_sm")  
  
[3] doc = nlp("John's father didn't have £100.")  
    for token in doc:  
        print(token)
```

```
John  
's  
father  
did  
n't  
have  
£  
100  
.
```



# Summary of Data for Measuring Overlap

What should we measure the overlap between posts with?

## 1. Individual letters?

No. Letters don't provide enough information

## 2. Sequences of letters (character n-grams)?

Maybe. Has some problems but can be very efficient

## 3. Words (or really tokens)?

Yes. Words capture meaning better than character n-grams. You could use word n-grams to capture extra information



# What is our post about?

Some words are more important for representing the subject of the post

Anyone tried **Irn Bru**?

It's a **Scottish drink** and it's **banned** some **countries** and I was **wondering** if anyone here has **tried** it. It has quite a **unique taste** and it's not something you'd **forget quickly**. You either **love** it or **hate** it I think.



# What is our post **NOT** about?

Some common words do not convey much meaning by themselves

**Anyone** tried Irn Bru?

It's a Scottish drink **and it's** banned **some** countries **and I was** wondering **if anyone here has** tried **it. It has** quite **a** unique taste **and it's not something you'd** forget quickly. **You either** love **it or hate it I think.**



# Filtering using stopwords

A common practice is to remove words with minimal meaning.

These words are known as “stopwords”.  
There are many lists (e.g. right) but none are definitive.

**BUT:** Some important phrases contain common words, e.g. “The Who”, “to be or not to be”

**Fun history:** The first standardized list of stopwords was proposed by C.J. Van Rijsbergen in ~1975. He founded the Information Retrieval group at the University of Glasgow.

A	CANNOT	INTO	OUR	THUS
ABOUT	CO	IS	OURS	TO
ABOVE	COULD	IT	OURSELVES	TOGETHER
ACROSS	DOWN	ITS	OUT	TOO
AFTER	DURING	ITSELF	OVER	TOWARD
AFTERWARDS	EACH	LAST	OWN	TOWARDS
AGAIN	EG	LATTER	PER	UNDER
AGAINST	EITHER	LATTERLY	PERHAPS	UNTIL
ALL	ELSE	LEAST	RATHER	UP
ALMOST	ELSEWHERE	LESS	SAME	UPON
ALONE	ENOUGH	LTD	SEEM	US
ALONG	ETC	MANY	SEEMED	VERY
ALREADY	EVEN	MAY	SEEMING	VIA
ALSO	EVER	ME	SEEMS	WAS
ALTHOUGH	EVERY	MEANWHILE	SEVERAL	WE
ALWAYS	EVERYONE	MIGHT	SHE	WELL
AMONG	EVERYTHING	MORE	SHOULD	WERE
AMONGST	EVERWHERE	MOREOVER	SINCE	WHAT
AN	EXCEPT	MOST	SO	WHATEVER
AND	FEW	MOSTLY	SOME	WHEN
ANOTHER	FIRST	MUCH	SOMEHOW	WHENCE
ANY	FOR	MUST	SOMEONE	WHENEVER
ANYHOW	FORMER	MY	SOMETHING	WHERE
ANYONE	FORMERLY	MYSELF	SOMETIMES	WHEREAFTER
ANYTHING	FROM	NAMELY	SOMETIMES	WHEREAS
ANYWHERE	FURTHER	NEITHER	SOMEWHERE	WHEREBY
ARE	HAD	NEVER	STILL	WHEREIN
AROUND	HAS	NEVERTHELESS	SUCH	WHEREUPON
AS	HAVE	NEXT	THAN	WHEREVER
AT	HE	NO	THAT	WHETHER
BE	HENCE	NOBODY	THE	WHITHER
BECAME	HER	NONE	THEIR	WHICH
BECAUSE	HERE	NOONE	THEM	WHILE
BECOME	HEREAFTER	NOR	THEMSELVES	WHO
BECOMES	HEREBY	NOT	THEN	WHOEVER
BECOMING	HEREIN	NOTHING	THENCE	WHOLE
BEEN	HEREUPON	NOW	THERE	WHOM
BEFORE	HERS	NOWHERE	THEREAFTER	WHOSE
BEFOREHAND	HERSELF	OF	THEREBY	WHY
BEHIND	HIM	OFF	THEREFORE	WILL
BEING	HIMSELF	OFTEN	THEREIN	WITH
BELOW	HIS	ON	THEREUPON	WITHIN
BESIDE	HOW	ONCE	THESE	WITHOUT
BESIDES	HOWEVER	ONE	THEY	WOULD
BETWEEN	I	ONLY	THIS	YET
BEYOND	IE	ONTO	THOSE	YOU
BOTH	IF	OR	THOUGH	YOUR
BUT	IN	OTHER	THROUGH	YOURS
BY	INC	OTHERS	THROUGHOUT	YOURSELF
CAN	INDEED	OTHERWISE	THRU	YOURSSELVES

---

Breather!



Two words only map together if they match exactly!

**Problem:** The two sentences below don't overlap at all by individual words

“I loved IRN BRU”

“He loves irn bru”

**Case (“IRN” and “irn”):** A common practice is to make text case insensitive by converting all text to lowercase.

But: the case of a word may give important information.

**Word Forms (“loved” and “loves”):** We need to process words to a canonical version by removing suffixes so that they match (e.g. loved -> love)

But: you may lose some meaning



# Breaking words into stems and affixes

**Stems:** The core meaning-bearing units (e.g. love)

For verbs, think of the infinitive (to **love**)

**Affixes:** Bits and pieces that adhere to stems (e.g. -d or -s)

Often grammatical additions for conjugation or tense

Stems and affixes are known broadly as morphemes.

**Morpheme:** Small meaningful units that make up words

We want the stems.  
We need **STEMMING!**



# Stemming

**Definition:** Process for reducing inflected words to their stem or root form

Often rule-based and removes common suffixes (e.g. -d, -ing, -s, etc).

Generally works on single words with no context.

Can make mistakes but often they aren't that bad. (e.g. computing -> comput)





# The Porter Stemmer - a popular stemming algorithm

## Step 1a

sses	→ ss	caresses	→ caress
ies	→ i	ponies	→ poni
ss	→ ss	caress	→ caress
s	→ Ø	cats	→ cat

## Step 2 (for long stems)

ational	→ ate	relational	→ relate
izer	→ ize	digitizer	→ digitize
ator	→ ate	operator	→ operate
...			

## Step 1b

(*v*)ing	→ Ø	walking	→ walk
		sing	→ sing
(*v*)ed	→ Ø	plastered	→ plaster
...			

## Step 3 (for longer stems)

al	→ Ø	revival	→ reviv
able	→ Ø	adjustable	→ adjust
ate	→ Ø	activate	→ activ
...			

*This is just a sample of the large rule set*



# Weakness of Stemming: Ignores meaning

Two words can have different meaning but end up with the stem:

emergency → emerg

emerge      → emerg



# Lemmatization is stemming + (a bit of) meaning

Two words can have different meaning but end up with the stem:

emergency → emerg

emerge → emerg

**Lemmatization** is slightly smarter than basic stemming. It may use the part of speech (e.g. verb/noun) to get some sense of meaning and apply different rules

emergency → emergency

emerge → emerge

Ideally two words have the same lemma if:

1. Same **stem**
2. Same **part-of-speech** (e.g. noun, verb, etc)
3. Essentially the same meaning

# Context of words matter

Words don't work independently and neighbouring words can change their meaning.

“really **like**” and “don’t **like**” have different meanings!

How can we capture that?



# Word n-grams can capture additional meaning

You can go beyond individual words into pairs or triples of words.

Word bi-grams could capture “irn bru” as two words instead of “irn” and “bru” individually

Word bi-grams and tri-grams are fairly powerful but can be computationally expensive.

About ~7 million words in English wikitionary

It's a Scottish drink and it's banned some  
It's a Scottish drink and it's banned some

*Sliding window showing word  
3-grams*



# Summary of problems and solutions

1. Individual characters aren't very helpful
  - Use words
2. Need to split text into words - actually quite challenging to do cleanly
  - Tokenizers use set of rules to split using whitespace, punctuation and other rules
  - Use the Spacy parser (generally good)
3. Some words do not add useful information
  - Use stopwords to filter out common words
  - Use other metrics to give stronger weight to “likely important” rarer words
4. Words with same meaning may not match exactly
  - e.g. swim != swimming
  - Use stemming and lemmatization
5. Some words only keep meaning with context of neighbouring word(s)
  - Use bi-grams, tri-grams, etc



# Summary of a standard pipeline

1. Data cleaning
  - e.g. converting web page to plain text (by removing links, tables, etc)
2. Tokenizing
  - May also separate sentences
3. Normalize the words
  - Using stemming/lemmatization to get a normalized form of words
4. Stopwords removal

---

# Document similarity using set similarity

# Problem Statement (again) and overlap idea

Can we find other Reddit posts that are similar to the one below?

## Anyone tried Irn Bru?

It's a Scottish drink and it's banned some countries and I was wondering if anyone here has tried it. It has quite a unique taste and it's not something you'd forget quickly. You either love it or hate it I think.

29 Comments

Award

Share

Save

Hide

Report

100% Upvoted

If two posts have substantial overlap of words, we hypothesise that they are talking about a similar subject



# Comparing documents with sets of tokens

A set is: a group of **unique** items

A document can be represented as a set of the tokens in it.

Tokens could be stemmed, etc as in previous section

Some characteristics of sets:

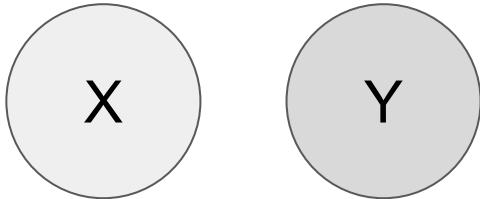
- No concept of item frequency (so words that occur many times in a document are treated the same as one that appears once)
- Very efficient computation when comparing sets
- Python (and most other languages) have sets as a standard data structure

```
tokens_in_document = ['irn', 'bru', 'is', 'great', '.', 'i', 'always', 'drink', 'irn', 'bru', '.']
set_of_tokens = set(tokens_in_document)
set_of_tokens
{'.', 'always', 'bru', 'drink', 'great', 'i', 'irn', 'is'}
```



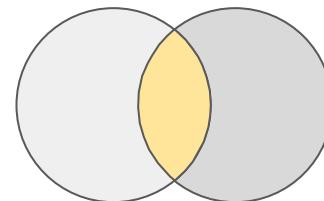
# Set operations

$|X|$  = number of elements in a set X



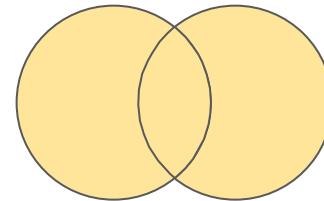
$X \cap Y$  = intersection of sets X and Y

*items that are in X and Y*



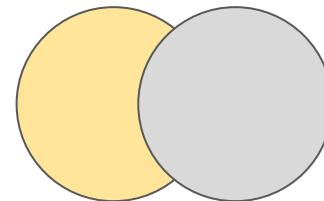
$X \cup Y$  = union of sets X and Y

*items that are in X or Y*



$X - Y$  = set difference of X and Y

*items that are in X but not Y*





# Similarity measures

$\text{sim}(X, Y)$  is a function that calculates a similarity measure where  $X$  and  $Y$  are the sets of tokens for two documents to be compared

$\text{sim}(X, Y)$  should be **high** when  $X$  and  $Y$  are **similar**

$\text{sim}(X, Y)$  should be **low** when  $X$  and  $Y$  are **dissimilar**

There is not one definitive similarity measure.  
Let's explore a few!



## Number of overlapping words

$$\text{sim}(X, Y) = |X \cap Y|$$

where X and Y are sets of words for two documents,  $\cap$  is in the intersection and  $|Z|$  is the count of elements in a set Z.

- Can argue that two documents are similar if they contain many of the same tokens
- **But:** needs to be normalized (i.e. so that  $\text{sim}(X, Y)$  is between 0 and 1). A long document that contains a large selection of words would match with lots of documents so need to factor in the number of tokens in each token set.



# Overlap Coefficient

$$\text{sim}(X, Y) = \frac{|X \cap Y|}{\min(|X|, |Y|)}$$

where X and Y are sets of words for two documents,  $\cap$  is in the intersection and  $|.|$  is the count of elements in a set.

*Meaning: The % of unique tokens in the smaller document that appears in the larger document*

## Improvement:

- Now using the number of unique tokens in X and Y to normalize the measure
- Bounded between zero and one
- 0 means no overlapping tokens
- 1 means that one set is a subset of the other



# Sørensen–Dice Coefficient

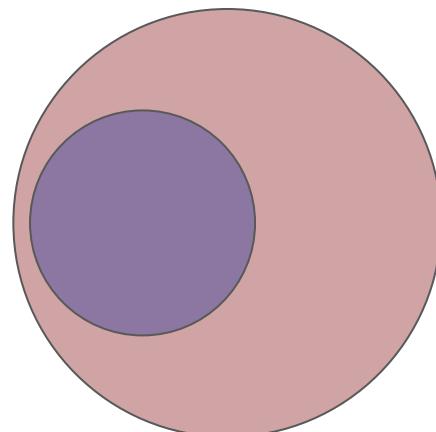
$$\text{sim}(X, Y) = \frac{2 |X \cap Y|}{|X| + |Y|}$$

Properties:

- Bounded between 0 and 1
- 0 means no overlap and 1 means perfect overlap

Difference with Overlap Coefficient

- Will only be 1 if the two sets are exactly matching, not if one is a subset of the other





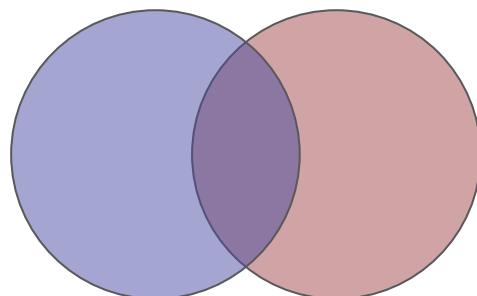
# Jaccard Similarity

$$\text{sim}(X, Y) = \frac{|X \cap Y|}{|X \cup Y|}$$

The number of overlapping tokens divided by the number of unique tokens across the two documents

## Properties

- Bounded between 0 and 1 (like Sørensen–Dice)
- Very popular





# Jaccard Similarity

$$\text{sim}(X, Y) = \frac{|X \cap Y|}{|X \cup Y|}$$

Difference with Sørensen–Dice Coefficient

- Jaccard **distance** ( $1 - \text{sim}(X, Y)$ ) satisfies the triangle inequality such that:
  - $\text{dist}(X, Z) \leq \text{dist}(X, Y) + \text{dist}(Y, Z)$
- This makes Jaccard a metric
- Sørensen–Dice does not satisfy this - making it a semi-metric

## Properties of a metric

- Distance between X and X is zero :  $d(X, X) = 0$
- Positive :  $d(X, Y) > 0$  where  $X \neq Y$
- Symmetric :  $d(X, Y) = d(Y, X)$
- Triangle inequality holds :  $d(X, Z) \leq d(X, Y) + d(Y, Z)$



# Think-Pair-Share: Calculating Jaccard Similarity

Calculate the Jaccard Similarity for the two “documents” (assume tokenizing on whitespace, lowercasing and ignoring punctuation):

1. IRN BRU rocks! I drink IRN BRU daily!
2. irn bru is bad. sprite is better



Use the code 5500 8824  
on [menti.com](https://menti.com)

$$\text{sim}(X, Y) = \frac{|X \cap Y|}{|X \cup Y|}$$



# Example using Jaccard Similarity

X (Unique tokens in first document)

{ "irn", "bru", "rocks", "i", "drink", "daily" }

Y (Unique tokens in second document)

{ "irn", "bru", "is", "bad", "sprite", "better" }

Jaccard Similarity:

$$\frac{|X \cap Y|}{|X \cup Y|} = \frac{|\{\text{"irn", "bru"}\}|}{|\{\text{"irn", "bru", "rocks", "i", "drink", "daily", "is", "bad", "sprite", "better"}\}|} = \frac{2}{10} = \frac{1}{5}$$



# Summary of measures of set similarity

- Documents can be represented as sets of tokens
- Set similarity measures can be used to calculate whether two documents are similar or dissimilar
- Measures explored:
  - Overlap Coefficient
  - Sørensen–Dice Coefficient
  - Jaccard Similarity
- Weaknesses
  - All words are treated equally - but some should be more important
  - Word frequencies are ignored
  - Two different words with similar meaning cannot match

# Next Lab: Tokenizing text and document similarity



The next lab will involve tokenizing Reddit posts and using set similarity measures to find similar posts

