

Text Processing, Tasks, and Corpora

Nathan Schneider ~ 16 January 2024

<https://people.cs.georgetown.edu/nschneid/cosc5402/>

Text Processing

Basics of working with text

Basics of working with text

- Computers represent text as strings of **characters**
 - Letters, numbers, punctuation, symbols, emojis, diacritics/combining characters, spaces

Basics of working with text

- Computers represent text as strings of **characters**
 - Letters, numbers, punctuation, symbols, emojis, diacritics/combining characters, spaces
- At the machine level, a string is stored as a sequence of **bytes**, which are interpreted as integers → code points → characters in some **encoding**
 - e.g. Unicode defines 0x0B90 to be “ஐ” (Tamil letter Ai)
 - UTF-8 is a way of encoding Unicode characters as bytes

Basics of working with text

- Computers represent text as strings of **characters**
 - Letters, numbers, punctuation, symbols, emojis, diacritics/combining characters, spaces
- At the machine level, a string is stored as a sequence of **bytes**, which are interpreted as integers → code points → characters in some **encoding**
 - e.g. Unicode defines 0x0B90 to be “ஐ” (Tamil letter Ai)
 - UTF-8 is a way of encoding Unicode characters as bytes
- A group of characters used to write in a language is called a **script**
 - e.g., Latin characters for European languages; Hangul for Korean; Devanagari for Hindi; CJK Ideographs

Basics of working with text

- Computers represent text as strings of **characters**
 - Letters, numbers, punctuation, symbols, emojis, diacritics/combining characters, spaces
- At the machine level, a string is stored as a sequence of **bytes**, which are interpreted as integers → code points → characters in some **encoding**
 - e.g. Unicode defines 0x0B90 to be “ஐ” (Tamil letter Ai)
 - UTF-8 is a way of encoding Unicode characters as bytes
- A group of characters used to write in a language is called a **script**
 - e.g., Latin characters for European languages; Hangul for Korean; Devanagari for Hindi; CJK Ideographs
- **Text editors** provide functionality for working with (plain) text files

Find & replace

- In a text editor, or in Python, there are ways to find/replace exact string matches
- If you don't know the exact string you're looking for, you can use a **regular expression**

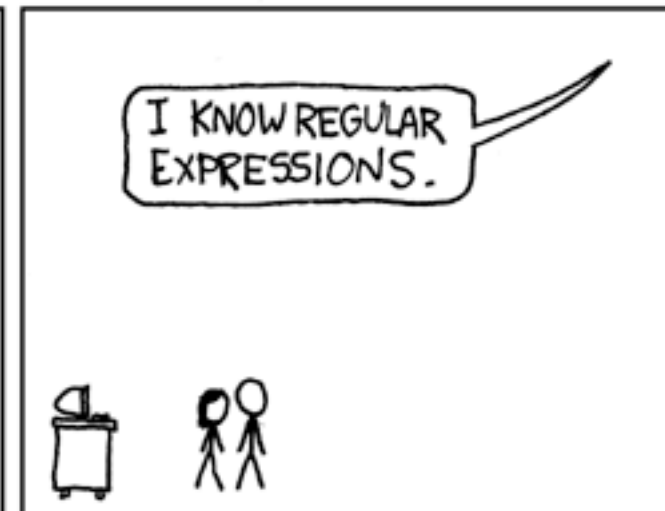
- In a text exact s

WHENEVER I LEARN A NEW SKILL I CONCOCT ELABORATE FANTASY SCENARIOS WHERE IT LETS ME SAVE THE DAY.

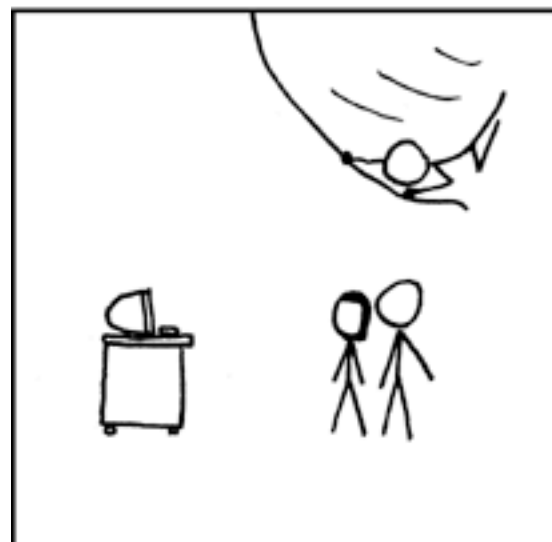


replace

- If you don't use a re



you can



Find & replace

- In a text editor, or in Python, there are ways to find/replace exact string matches
- If you don't know the exact string you're looking for, you can use a **regular expression**
- Regular expression patterns allow for **disjunctions** and **wildcards**, so the pattern may match more than one particular string
 - Pattern to find all English articles ("a", "an", "the")?
 - Pattern to check whether a string is an email address?

Regular expressions: (pro|con)s

Regular expressions: (pro|con)s

- Simple pattern language.
 - (though some differences by implementation, e.g. Python vs. Perl)

Regular expressions: (pro|con)s

- Simple pattern language.
 - (though some differences by implementation, e.g. Python vs. Perl)
- Deterministic—the expression either matches the string or it doesn't!
 - No ambiguity in the correct behavior (of course your expression may have a bug)

Regular expressions: (pro|con)s

- Simple pattern language.
 - (though some differences by implementation, e.g. Python vs. Perl)
- Deterministic—the expression either matches the string or it doesn't!
 - No ambiguity in the correct behavior (of course your expression may have a bug)
- Requires a very precise statement in terms of characters/substrings.
 - “names of people”, “sentences about chemistry” would be hard to express as a regex

Documents/sentences/words

Documents/sentences/words

- For more sophisticated NLP, we often want to break documents down into sentences, and sentences into words.
 - (In linguistics, “word” is very hard concept to define precisely across languages. But for now, think roughly “groups of characters separated by spaces in English text”).

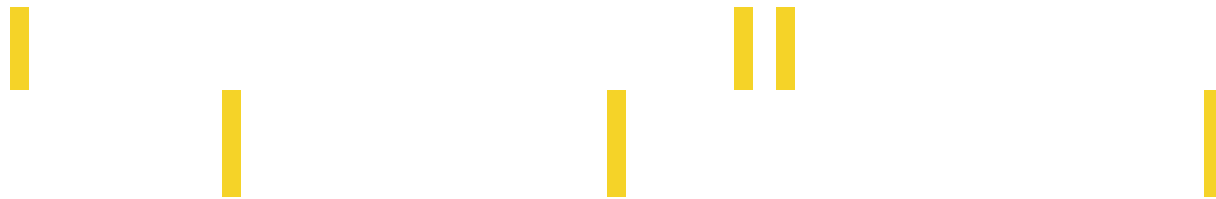
Documents/sentences/words

- For more sophisticated NLP, we often want to break documents down into sentences, and sentences into words.
 - (In linguistics, “word” is very hard concept to define precisely across languages. But for now, think roughly “groups of characters separated by spaces in English text”.)
- (Word) tokenization and sentence tokenization
 - The split units are called **tokens**.
 - *Token* also refers to an instance of a word, in contrast to its **type** (the word in general).

Tokenization

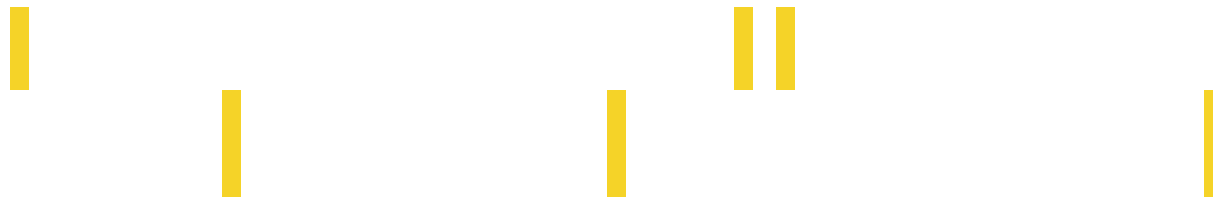
Tokenization

Daniels made several appearances as C-3PO on numerous TV shows and commercials, notably on a Star Wars-themed episode of The Donny and Marie Show in 1977, Disneyland's 35th Anniversary.



Tokenization

Daniels made several appearances as C-3PO on numerous TV shows and commercials, notably on a Star Wars-themed episode of The Donny and Marie Show in 1977, Disneyland's 35th Anniversary.



Tokenization

Daniels made several appearances as C-3PO on numerous TV shows and commercials, notably on a Star Wars-themed episode of The Donny and Marie Show in 1977, Disneyland's 35th Anniversary.



Daniels made several appearances as C-3PO on numerous TV shows and commercials, notably on a Star Wars-themed episode of The Donny and Marie Show in 1977, Disneyland's 35th Anniversary.

Tokenization

Daniels made several appearances as C-3PO on numerous TV shows and commercials, notably on a Star Wars-themed episode of The Donny and Marie Show in 1977, Disneyland's 35th Anniversary.



Daniels made several appearances as C-3PO on numerous TV shows and commercials, notably on a Star Wars-themed episode of The Donny and Marie Show in 1977, Disneyland's 35th Anniversary.

- For most kinds of writing, regular expressions can do this pretty well, most of the time.
 - Language/regional/stylistic differences in use of spaces, punctuation, etc.
 - Python library for English: `nltk.word_tokenize(string)`

Sentence Tokenization/ Splitting: Not always easy

In April 1938, Bernarr A. Macfadden, publisher of *Liberty* magazine stepped in, offering a prize of \$1,000 to the winning composer, stipulating that the song must be of simple “harmonic structure”, “within the limits of [an] untrained voice”, and its beat in “march tempo of military pattern”.

The contest rules required the winner to submit his entry in written form, and Crawford immediately complied. However his original title, *What Do You think of the Air Corps Now?*, was soon officially changed to *The Army Air Corps*.

[https://en.wikipedia.org/wiki/The_U.S._Air_Force_\(song\)](https://en.wikipedia.org/wiki/The_U.S._Air_Force_(song))

Preprocessing

- Tokenization is a kind of **preprocessing** that can be done to a piece of text.
- Depending on the data source and application, it may help to perform
 - **data cleaning:** remove irrelevant material like page numbers in an OCR'd text, or irrelevant documents like spam/advertising
 - **normalization:** collapsing different types of spelling variation such as capitalization (colour → color; U.S. → US → us)
 - **lemmatization:** mapping words of a language to a canonical form (the “dictionary entry” or **lemma**): {eat, eats, eating, ate, eaten} → eat

Tasks

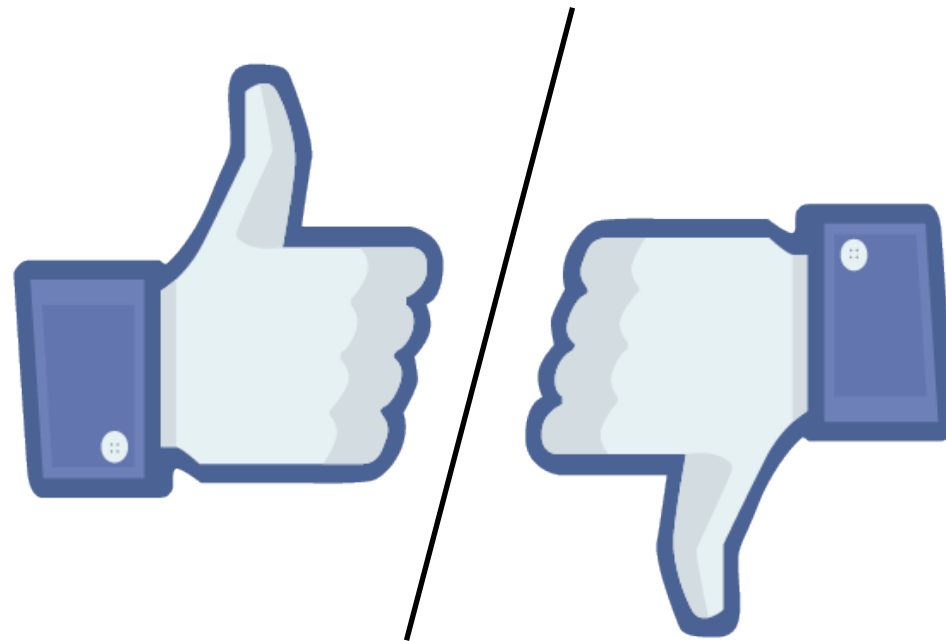
- Recall that NLP covers a range of problems, ranging from core linguistic analysis to practical applications:

Applications & Core Tasks

- Machine Translation
 - Information Retrieval
 - Question Answering
 - Dialogue Systems
 - Information Extraction
 - Summarization
 - Sentiment Analysis
 - ...
- Language modeling/text generation
 - Part-of-speech tagging
 - Syntactic parsing
 - Named entity recognition
 - Coreference resolution
 - Word sense disambiguation
 - Semantic role labeling
 - ...

- We measure progress in NLP by evaluating systems designed to perform various **tasks**.
 - benchmarking
- Different tasks target different languages, parts of language, and applications.
- By way of example, we'll focus on the task of...

Sentiment Analysis



Goal: Predict the **opinion** expressed in a piece of text.

E.g., **+** or **-**. (Or a rating on a scale.)

Sentiment Analysis

Filled with horrific dialogue, laughable characters, a laughable plot, and really no interesting stakes during this film, "Star Wars Episode I: The Phantom Menace" is not at all what I wanted from a film that is supposed to be the huge opening to the segue into the fantastic Original Trilogy. The positives include the score, the sound effects, and most of the

KJ Proulx (/user/id/896976177/)

★ Super Reviewer

Sentiment Analysis

Filled with horrific dialogue, laughable characters, a laughable plot, and really no interesting stakes during this film, "Star Wars Episode I: The Phantom Menace" is not at all what I wanted from a film that is supposed to be the huge opening to the segue into the fantastic Original Trilogy. The positives include the score, the sound effects, and most of the

KJ Proulx (/user/id/896976177/)
★ Super Reviewer

Extraordinarily faithful to the tone and style of the originals, The Force Awakens brings back the Old Trilogy's heart, humor, mystery, and fun. Since it is only the first piece in a new three-part journey it can't help but feel incomplete. But everything that's already there, from the stunning visuals, to the thrilling action sequences, to the charismatic new characters,

Matthew Samuel Mirliani (/user/id/896467979/)
★ Super Reviewer

Sentiment Analysis



Filled with horrific dialogue, laughable characters, a laughable plot, and really no interesting stakes during this film, "Star Wars Episode I: The Phantom Menace" is not at all what I wanted from a film that is supposed to be the huge opening to the segue into the fantastic Original Trilogy. The positives include the score, the sound effects, and most of the

KJ Proulx (/user/id/896976177/)

★ Super Reviewer



Extraordinarily faithful to the tone and style of the originals, The Force Awakens brings back the Old Trilogy's heart, humor, mystery, and fun. Since it is only the first piece in a new three-part journey it can't help but feel incomplete. But everything that's already there, from the stunning visuals, to the thrilling action sequences, to the charismatic new characters,

Matthew Samuel Mirliani (/user/id/896467979/)

★ Super Reviewer

Sentiment Analysis

★★

Filled with horrific dialogue, laughable characters, a laughable plot, and really no interesting stakes during this film, "Star Wars Episode I: The Phantom Menace" is not at all what I wanted from a film that is supposed to be the huge opening to the segue into the fantastic Original Trilogy. The positives include the score, the sound effects, and most of the

KJ Proulx (/user/id/896976177/)

★ Super Reviewer

★★★★★

Extraordinarily faithful to the tone and style of the originals, The Force Awakens brings back the Old Trilogy's heart, humor, mystery, and fun. Since it is only the first piece in a new three-part journey it can't help but feel incomplete. But everything that's already there, from the stunning visuals, to the thrilling action sequences, to the charismatic new characters,

Matthew Samuel Mirliani (/user/id/896467979/)

★ Super Reviewer

[RottenTomatoes.com](https://www.rottentomatoes.com) + intuitions about positive/negative cue words

**So, you want to build a
sentiment analyzer**

So, you want to build a sentiment analyzer

Questions to ask yourself:

So, you want to build a sentiment analyzer

Questions to ask yourself:

1. What is the input for each prediction? (sentence? full review text? text+metadata?)

So, you want to build a sentiment analyzer

Questions to ask yourself:

1. What is the input for each prediction? (sentence? full review text? text+metadata?)
2. What are the possible outputs? (+ or -)

So, you want to build a sentiment analyzer

Questions to ask yourself:

1. What is the input for each prediction? (sentence? full review text? text+metadata?)
2. What are the possible outputs? (+ or -)
3. How will it decide?

So, you want to build a sentiment analyzer

Questions to ask yourself:

1. What is the input for each prediction? (sentence? full review text? text+metadata?)
2. What are the possible outputs? (+ or -)
3. How will it decide?
4. How will you measure its effectiveness?

So, you want to build a sentiment analyzer

Questions to ask yourself:

1. What is the input for each prediction? (sentence? full review text? text+metadata?)
2. What are the possible outputs? (+ or -)
3. How will it decide?
4. How will you measure its effectiveness?

The last one, at least, requires data!

BEFORE you build a system, choose a dataset for evaluation!

Why is data-driven evaluation important?

- Good science requires controlled experimentation.
- Good engineering requires benchmarks.
- Your intuitions about typical inputs are probably wrong.

Sometimes you want multiple evaluation datasets: e.g., one for **development** as you hack on your system, and one reserved for final **testing**.

Corpora

Corpora

- A dataset with many instances of language use is called a **corpus** (pl. *corpora*).
 - ▶ < Latin 'body'
 - ▶ It can be a carefully curated sample for studying how language works.
 - ▶ Or, for building and evaluating NLP systems.

Corpora

- A dataset with many instances of language use is called a **corpus** (pl. *corpora*).
 - ▶ < Latin 'body'
 - ▶ It can be a carefully curated sample for studying how language works.
 - ▶ Or, for building and evaluating NLP systems.
- Where can you get one?
 - ▶ A precompiled dataset, e.g. for NLP/machine learning contests
 - ▶ Create your own, e.g. by scraping websites

Annotations

To evaluate and compare sentiment analyzers, we need reviews with **gold labels** (+ or −) attached. These can be

- derived automatically from the original data artifact (**metadata** such as star ratings), or
- added by a human annotator who reads the text
 - Issue to consider/measure: How consistent are human annotators? If they often have trouble deciding or agreeing, how can this be addressed?

More on these issues later in the course!

An evaluation measure

Once we have a dataset with gold (correct) labels, we can give the text of each review as input to our system and measure how often its output matches the gold label.

Simplest measure:

$$\text{accuracy} = \frac{\# \text{ correct}}{\# \text{ total}}$$

More measures later in the course!

Catching our breath

We now have:

- ✓ a definition of the sentiment analysis task (inputs and outputs)
- ✓ a way to measure a sentiment analyzer (accuracy on gold data)

So we need:

- an algorithm for predicting sentiment

A simple sentiment classification algorithm

Use a **sentiment lexicon** to count positive and negative words:

Positive:

absolutely	beaming	calm
adorable	beautiful	celebrated
accepted	believe	certain
acclaimed	beneficial	champ
accomplish	bliss	champion
achieve	bountiful	charming
action	bounty	cheery
active	brave	choice
admire	bravo	classic
adventure	brilliant	classical
affirm	bubbly	clean
...		...

Negative:

abysmal	bad	callous
adverse	banal	can't
alarming	barbed	clumsy
angry	belligerent	coarse
annoy	bemoan	cold
anxious	beneath	collapse
apathy	boring	confused
appalling	broken	contradictory
atrocious		contrary
awful		corrosive
		corrupt
		...

From <http://www.enchantedlearning.com/wordlist/>

Simplest rule: Count positive and negative words in the text. Predict whichever is greater.

Some possible problems with simple counting

1. Hard to know whether words that *seem* positive or negative tend to actually be used that way.
 - sense ambiguity
 - sarcasm/irony
 - text could mention expectations or opposing viewpoints, in contrast to author's actual opinion
2. Opinion words may be describing (e.g.) a character's attitude rather than an evaluation of the film.
3. Some words act as semantic modifiers of other opinion-bearing words/phrases, so interpreting the full meaning requires sophistication:

I **can't** stand this movie

VS.

I **can't** believe how great this movie is

What if we have more data?

Perhaps corpora can help address the first objection:

1. Hard to know whether words that *seem* positive or negative tend to actually be used that way.

A data-driven method: Use **frequency counts** to ascertain which words tend to be positive or negative.

NLTK

The Natural Language Toolkit (<http://nltk.org>) is a Python library for NLP. NLTK

- is open-source, community-built software
- was designed for teaching NLP: simple access to datasets, reference implementations of important algorithms
- contains wrappers for using (some) state-of-the-art NLP tools in Python

It will help if you familiarize yourself with Python **strings** and methods/libraries for manipulating them.

(If you are familiar with Python 2.7, know that strings and Unicode are handled differently in Python 3.)

Using an NLTK corpus

```
>>> from nltk.corpus import movie_reviews
>>> movie_reviews.words()
[u'plot', u':', u'two', u'teen', u'couples', u'go', ...]
>>> movie_reviews.sents()
[[u'plot', u':', u'two', u'teen', u'couples', u'go',
  ↳ u'to', u'a', u'church', u'party', u',', u'drink',
  ↳ u'and', u'then', u'drive', u'.'], [u'they',
  ↳ u'get', u'into', u'an', u'accident', u'.'], ...]
>>> print('\n'.join(' '.join(sent) for sent in
  ↳ movie_reviews.sents()[ :5]))
plot : two teen couples go to a church party , drink
  ↳ and then drive .
they get into an accident .
one of the guys dies , but his girlfriend continues to
  ↳ see him in her life , and has nightmares .
what ' s the deal ?
watch the movie and " sorta " find out .
```


Using an NLTK corpus: word frequencies

```
>>> from nltk import FreqDist
>>> f = FreqDist(movie_reviews.words())
>>> f.most_common()[:20]
[(u',', 77717), (u'the', 76529), (u'.' , 65876), (u'a',
↳ 38106), (u'and', 35576), (u'of', 34123), (u'to',
↳ 31937), (u'""', 30585), (u'is', 25195), (u'in',
↳ 21822), (u's', 18513), (u'"""', 17612), (u'it',
↳ 16107), (u'that', 15924), (u'-' , 15595), (u')',
↳ 11781), (u'(', 11664), (u'as', 11378), (u'with',
↳ 10792), (u'for', 9961)]
```

Using an NLTK corpus: word frequencies

```
>>> f = FreqDist(w for w in movie_reviews.words() if
    ↪ any(c.isalpha() for c in w))
>>> f.most_common()[:20]
[(u'the', 76529), (u'a', 38106), (u'and', 35576),
 ↪ (u'of', 34123), (u'to', 31937), (u'is', 25195),
 ↪ (u'in', 21822), (u's', 18513), (u'it', 16107),
 ↪ (u'that', 15924), (u'as', 11378), (u'with',
 ↪ 10792), (u'for', 9961), (u'his', 9587), (u'this',
 ↪ 9578), (u'film', 9517), (u'i', 8889), (u'he',
 ↪ 8864), (u'but', 8634), (u'on', 7385)]
```

Using an NLTK corpus: categories

```
>>> movie_reviews.categories()
[u'neg', u'pos']
>>> fpos =
    ↪ FreqDist(movie_reviews.words(categories='pos'))
>>> fneg =
    ↪ FreqDist(movie_reviews.words(categories='neg'))
>>> fMoreNeg = fneg - fpos
>>> fMoreNeg.most_common()[:20]
[(u'movie', 721), (u't', 700), (u'i', 685), (u'bad',
    ↪ 673), (u'?', 631), (u'",', 628), (u'have', 421),
    ↪ (u'!', 399), (u'no', 350), (u'plot', 321),
    ↪ (u'there', 318), (u'if', 301), (u'*', 286),
    ↪ (u'this', 282), (u'so', 267), (u'why', 250),
    ↪ (u'just', 221), (u'only', 219), (u'worst', 210),
    ↪ (u'even', 207)]
```

What if we have more data?

Perhaps corpora can help address the first objection:

1. Hard to know whether words that *seem* positive or negative tend to actually be used that way.

A data-driven method: Use frequency counts from a **training corpus** to ascertain which words tend to be positive or negative.

- Why separate the training and test data (held-out test set)? Because otherwise, it's just data analysis; no way to estimate how well the system will do on new data in the future.

ChatGPT Activity



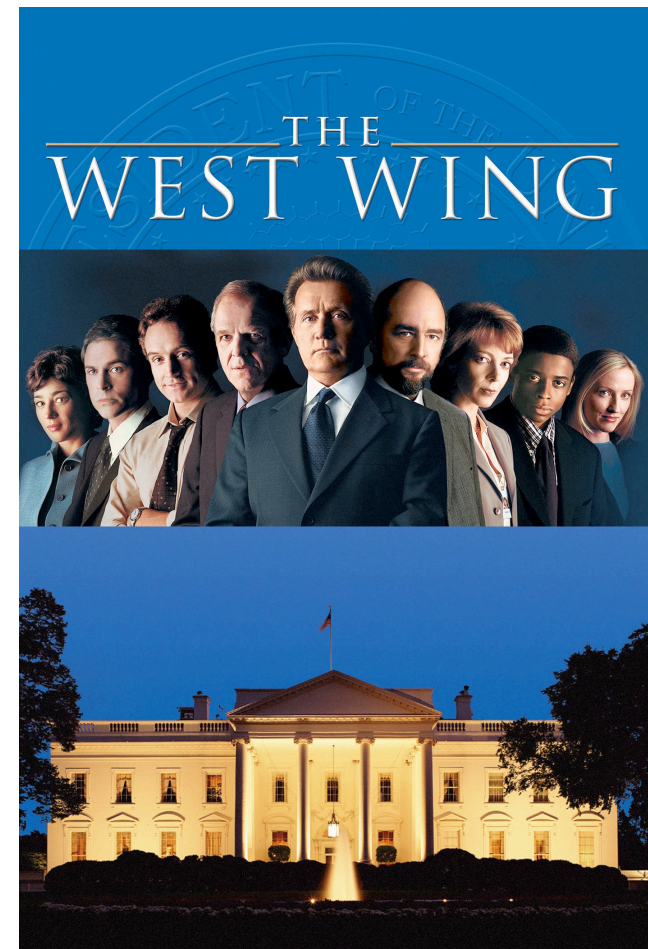
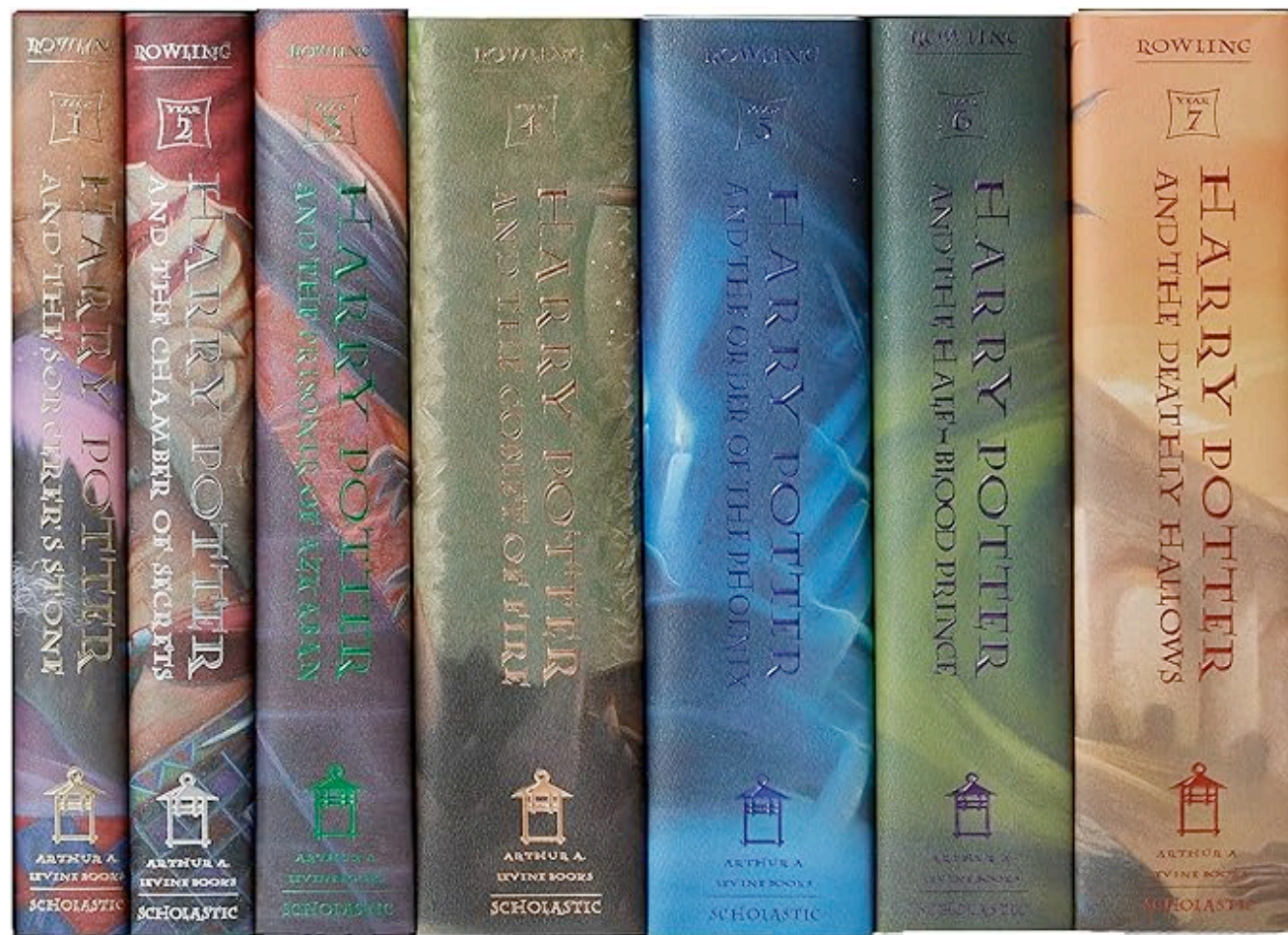
ChatGPT

- ChatGPT is a well-known system that responds to text prompts with generated text (or code).
 - It is a testament to the power of corpus data.
- Who has used it? What for?

Strengths/Weaknesses

1. Ask some factual questions with specific correct answers (e.g. birthday of a celebrity). Does the system provide a correct answer?
2. Engage in a dialogue about a favorite book, film, or research paper of your choice.
 - Can the system provide an accurate summary?
 - Are the answers on topic? Consistent?
 - If you ask about details, are the answers correct?
 - What happens if you ask a trick question?
3. Ask it to provide information in a creative form, e.g. lyrics in the style of your favorite singer.

Nathan vs. ChatGPT



Caution


Caution

- It is hard to avoid anthropomorphizing systems like ChatGPT.


Caution

- It is hard to avoid anthropomorphizing systems like ChatGPT.
 - They produce fluent text, which is something we tend to associate with human intelligence.


Caution

- It is hard to avoid anthropomorphizing systems like ChatGPT.
 - ▶ They produce fluent text, which is something we tend to associate with human intelligence.
 - ▶ The system was trained on massive amounts of text on the web produced by humans, and remixes them to generate new strings. “Stochastic ”


Caution

- It is hard to avoid anthropomorphizing systems like ChatGPT.
 - ▶ They produce fluent text, which is something we tend to associate with human intelligence.
 - ▶ The system was trained on massive amounts of text on the web produced by humans, and remixes them to generate new strings. “Stochastic ”
 - ▶ The system is impressive and (in some respects) useful!


Caution

- It is hard to avoid anthropomorphizing systems like ChatGPT.
 - They produce fluent text, which is something we tend to associate with human intelligence.
 - The system was trained on massive amounts of text on the web produced by humans, and remixes them to generate new strings. “Stochastic ”
 - The system is impressive and (in some respects) useful!
- But would you trust ChatGPT to make an important decision/document?

Caution

- It is hard to avoid anthropomorphizing systems like ChatGPT.
 - ▶ They produce fluent text, which is something we tend to associate with human intelligence.
 - ▶ The system was trained on massive amounts of text on the web produced by humans, and remixes them to generate new strings. “Stochastic ”
 - ▶ The system is impressive and (in some respects) useful!
- But would you trust ChatGPT to make an important decision/document?
 - ▶ Much of the generated text contains incorrect details, and it is easy to trick the system. Fluency ≠ a mind at work.

Caution

- **It is hard to avoid anthropomorphizing systems like ChatGPT.**
 - They produce fluent text, which is something we tend to associate with human intelligence.
 - The system was trained on massive amounts of text on the web produced by humans, and remixes them to generate new strings. “Stochastic ”
 - The system is impressive and (in some respects) useful!
- **But would you trust ChatGPT to make an important decision/document?**
 - Much of the generated text contains incorrect details, and it is easy to trick the system. Fluency ≠ a mind at work.
 - Beware industry hype! (“Generative AI” is the new “blockchain”)

“Intelligent” image generation?



YOU

Make a wheres Waldo picture



CHATGPT



Here's your "Where's Waldo" style picture. Have fun trying to spot the character!

ALT