Introduction to NLP

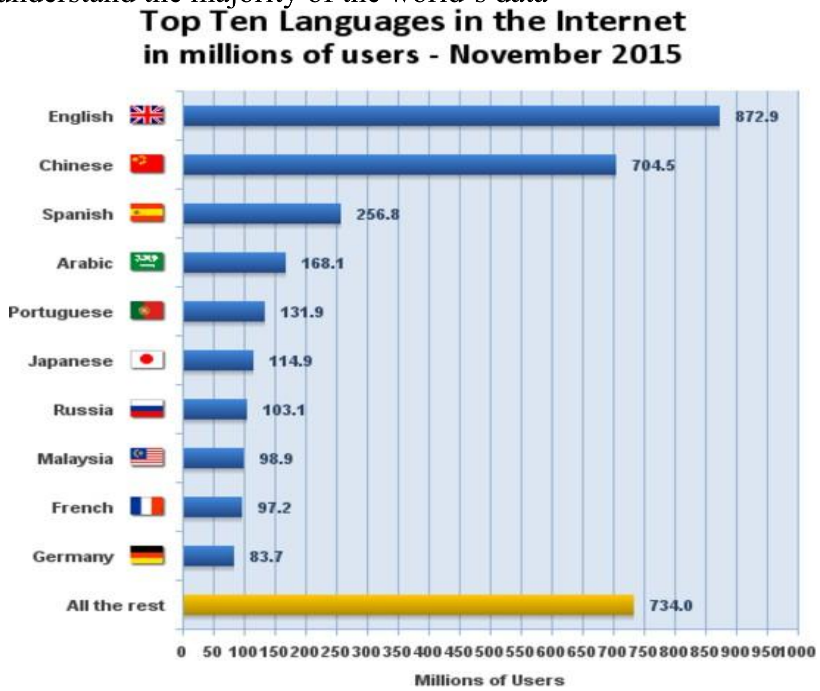
Reference Books

1. Daniel Jurafsky and James H. Martin. 2009. Speech and Language Processing: An Introduction to Natural Language Processing, Speech Recognition, and Computational Linguistics. 2nd edition. Prentice-Hall.
2. Christopher D. Manning and Hinrich Schütze. 1999. Foundations of Statistical Natural Language Processing. MIT Press.

UNIT-I: Introduction, Regular Expressions, Text Normalization, Edit Distance: Words, Corpora, Text Normalization, Word Normalization, Lemmatization and Stemming, Sentence Segmentation, The Minimum Edit Distance Algorithm.

WHY STUDY NLP?

- Text is the largest repository of human knowledge
- news articles, web pages, scientific articles, patents, emails, government documents ....
- Tweets, Facebook posts, comments, Quora ...
- [1] You could not understand the majority of the world's data



[1] Source: Internet world statistics

What is NLP?

Fundamental and Scientific Goal

- Deep understanding of broad language

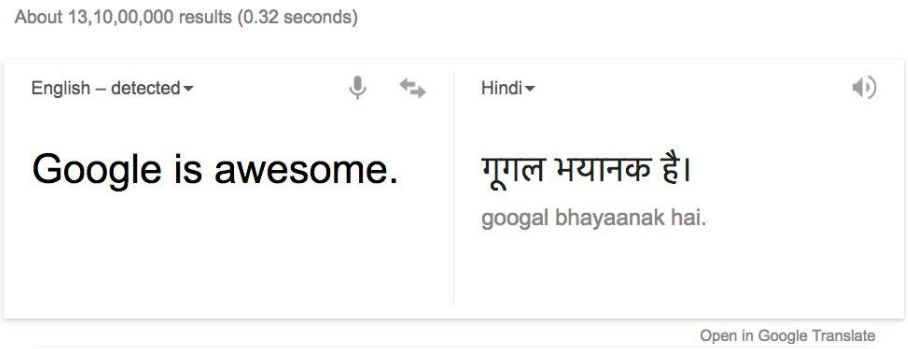
Engineering Goal

- Design, implement, and test systems that process natural languages for practical applications

Natural Language Processing (NLP) is a subfield of artificial intelligence and computational linguistics concerned with the interactions between computers and human (natural) languages. It involves the development of algorithms and systems that enable computers to understand, interpret, generate, and respond to text and speech in a manner that is both meaningful and useful.

This definition encompasses a wide range of tasks including syntactic analysis (e.g., part-of-speech tagging, parsing), semantic analysis (e.g., word sense disambiguation, semantic role labeling), and pragmatic tasks (e.g., dialogue systems, sentiment analysis).

What do we do in NLP?
Goals can be very ambitious: Good quality translation



About 13,10,00,000 results (0.32 seconds)

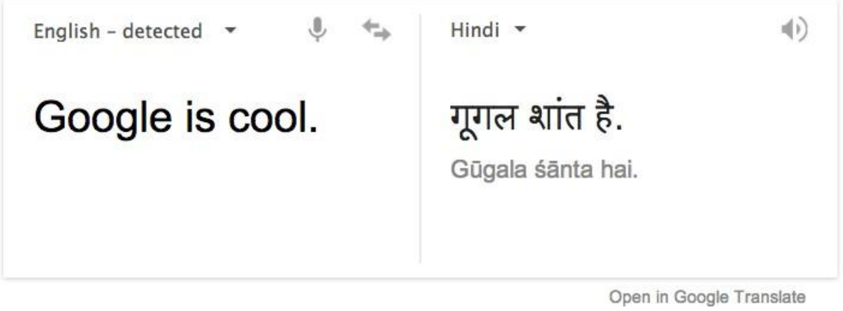
English – detected ▾ | Hindi ▾

Google is awesome.

गूगल भयानक है।

googal bhayaanak hai.

Open in Google Translate



English - detected ▾ | Hindi ▾

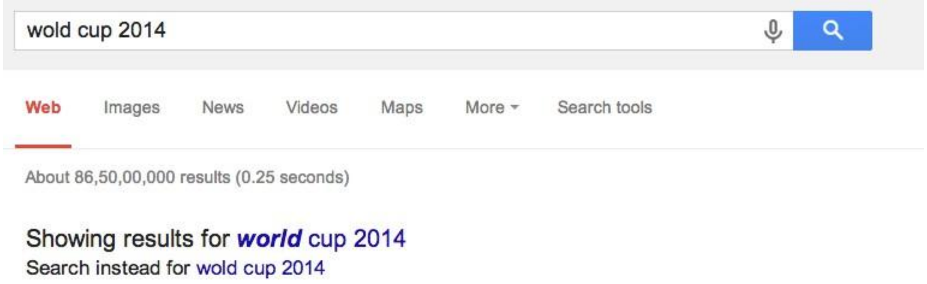
Google is cool.

गूगल शांत है.

Gūgala śānta hai.

Open in Google Translate

Well, even humans have made blunders:
Pepsi Chinese blunder
“Come alive with the Pepsi Generation”, when translated into Chinese meant, “Pepsi brings your relatives back from the dead.”
KFC’s Chinese blunder
KFC’s slogan, “Finger lickin’ good”, when translated into Chinese meant “We’ll eat your fingers off.”
Well, even humans ...
Goals can be very ambitious: Open Domain Chatbots
And Goals Can be Practical: Auto Completion

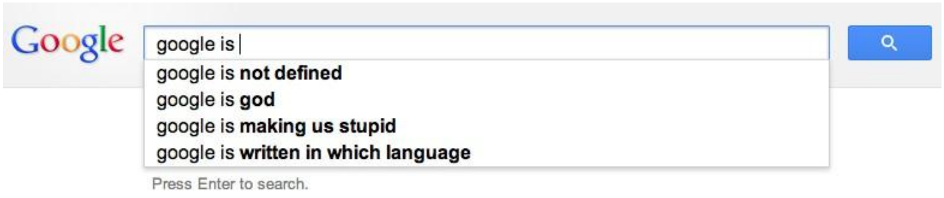


wold cup 2014

About 86,50,00,000 results (0.25 seconds)

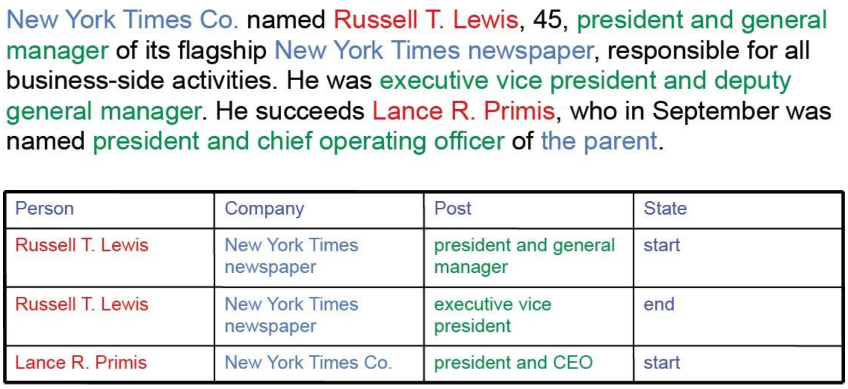
Showing results for ***world*** cup 2014
Search instead for wold cup 2014

And Goals can be Practical: Search Engines



google is|
google is **not defined**
google is **god**
google is **making us stupid**
google is **written in which language**
Press Enter to search.

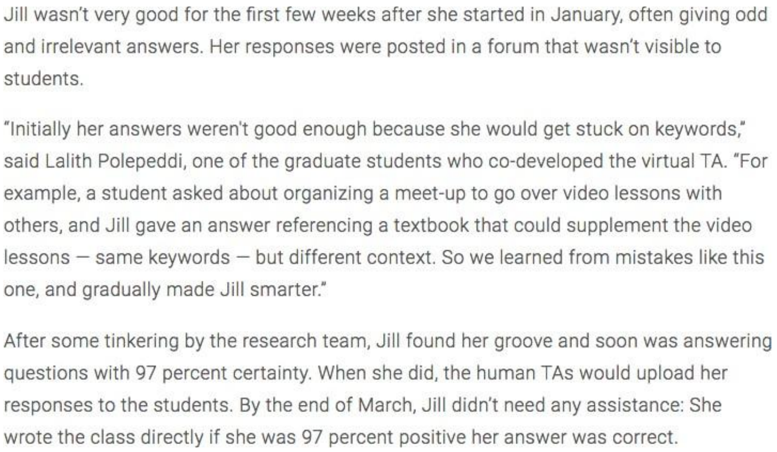
And Goals can be Practical: Information Extraction



New York Times Co. named Russell T. Lewis, 45, president and general manager of its flagship New York Times newspaper, responsible for all business-side activities. He was executive vice president and deputy general manager. He succeeds Lance R. Primis, who in September was named president and chief operating officer of the parent.

Person	Company	Post	State
Russell T. Lewis	New York Times newspaper	president and general manager	start
Russell T. Lewis	New York Times newspaper	executive vice president	end
Lance R. Primis	New York Times Co.	president and CEO	start

And Goals can be Practical: Domain-specific Chatbots



Jill wasn't very good for the first few weeks after she started in January, often giving odd and irrelevant answers. Her responses were posted in a forum that wasn't visible to students.

"Initially her answers weren't good enough because she would get stuck on keywords," said Lalith Polepeddi, one of the graduate students who co-developed the virtual TA. "For example, a student asked about organizing a meet-up to go over video lessons with others, and Jill gave an answer referencing a textbook that could supplement the video lessons — same keywords — but different context. So we learned from mistakes like this one, and gradually made Jill smarter."

After some tinkering by the research team, Jill found her groove and soon was answering questions with 97 percent certainty. When she did, the human TAs would upload her responses to the students. By the end of March, Jill didn't need any assistance: She wrote the class directly if she was 97 percent positive her answer was correct.

[1]http://www.news.gatech.edu/2016/05/09/artificial-intelligence-course-creates-ai-teaching-assist

And Goals can be Practical: Sentiment Analysis

Other Goals

- Spam detection
- Machine Translation services on the Web
- Text Summarization
- . . .

Natural Language Technology not yet perfect

But still good enough for several useful applications

WHY IS NLP HARD?

Lexical Ambiguity

- **Will Will will Will's will?**

  The sentence **"Will Will will Will's will?"** is a classic example of **lexical ambiguity**, where multiple meanings of the word "will" (as a **noun**, **verb**, and **proper noun**) are used. Here are several **interpretations** of this sentence:

  Basic Disambiguation:

- **Will (1)** – proper noun (a person's name)
- **will (2)** – verb (to bequeath or leave something in a will)
- **Will (3)** – proper noun (another or the same person's name)
- **will (4)** – noun (a legal document for inheritance)

Possible Interpretations:

1. **Interpretation 1:**
   Will (a person) bequeath the contents of Will's (another person's) will (legal document)?
   - Translation: *Is the person named Will going to execute or carry out the terms of the will belonging to another person named Will?*
2. **Interpretation 2:**
   Will (a person) create or legally draft the will of another person named Will?
   - Here, "will" is used in the sense of **writing or preparing** a will.
3. **Interpretation 3:**
   Is Will (the person) going to make legal arrangements to pass on the contents of Will's will (legal document)?
   - Here, there could be two different people named Will, or the same Will with a recursive will (though impractical, grammatically plausible).

To help parse the ambiguity, the sentence can be punctuated and spaced differently:

Will [subject] Will [verb] Will's [possessive noun] will [object]?

= *Will (person) will (verb) Will's (possessive) will (noun)?*

Humorous or Philosophical Interpretation:

- *Can Will (person) use his free will to will Will's (another Will) will?*
  - Playing on the multiple meanings of "will" including **intention**, **action**, and **document**.

Exercises

- **Rose rose to put rose roes on her rows of roses.**
- **Buffalo buffalo Buffalo buffalo buffalo buffalo Buffalo buffalo.**

  → **Buffaloes from Buffalo, NY, whom buffaloes from Buffalo bully, bully buffaloes from Buffalo.**

Language ambiguity: Structural

- The man saw the boy with the binoculars.

  The sentence **"The man saw the boy with the binoculars"** is a classic example of **structural (syntactic) ambiguity**, where the sentence structure allows for multiple interpretations due to the ambiguous attachment of the prepositional phrase **"with the binoculars"**.

Two Main Interpretations:

1. Instrumental Interpretation

"The man used the binoculars to see the boy."

- Structure:
  - **[The man] [saw] [the boy] [with the binoculars]**
  - Prepositional phrase **"with the binoculars"** modifies the **verb phrase "saw"** (i.e., *how* the man saw the boy).
- Parse Tree Fragment:

```
├── V: saw
├── NP: the boy
└── PP: with the binoculars  ← attaches to VP
```

2. Attributive Interpretation

"The man saw the boy who had the binoculars."

- Structure:
  - **[The man] [saw] [the boy with the binoculars]**
  - Prepositional phrase **"with the binoculars"** modifies the **noun phrase "the boy"** (i.e., *which* boy the man saw).
- Parse Tree Fragment:

```
NP
├── Det: the
├── N: boy
└── PP: with the binoculars  ← attaches to NP
```

Summary Table:

Interpretation	**Prepositional Phrase Modifies**	**Meaning**
Instrumental	Verb phrase ("saw")	The man used binoculars to see the boy.
Attributive (Descriptive)	Noun phrase ("the boy")	The boy had the binoculars.

Exercises

- **Flying planes can be dangerous.**
- **Hole found in the room wall; police are looking into it**.

Language imprecision and vagueness

- It is very warm here.
  ??? here implies? Which place? Which country? …..It is not clear.
- Q: Did your mother call your aunt last night?
  A: I'm sure she must have.

But that's the fun part of it

Why is the teacher wearing sun-glasses?

...

Because the class is so bright.

In above sentence who is bright? Is class bright or student bright?

Ambiguities:

News Headlines

1. Hospitals Are Sued by 7 Foot Doctors
2. Stolen Painting Found by Tree
3. Teacher Strikes Idle Kids
   Why will teacher strike idle kids…so its Techer Strikes; Idle Kids

Ambiguity is pervasive

Find at least 5 meanings of this sentence:

I made her duck

- I cooked duck for her
- I cooked duck belonging to her
- I created the (artificial) duck, she owns
- I caused her to quickly lower her head or body
- I waved my magic wand and turned her into a duck

Syntactic Category

- 'Duck' can be a noun or verb
- 'her' can be a possessive ('of her') or dative ('for her') pronoun

Word Meaning

- 'make' can mean 'create' or 'cook'

Grammar

make can be

- **Transitive:** (verb with a noun direct object)
- **Ditransitive:** (verb has 2 noun objects)
- **Action-transitive:** (verb has a direct object + verb)

Phonetics

- I'm eight or duck
- I'm aid her duck

Ambiguity is Explosive

- I saw the man with the telescope. 2 parses (I Saw with telescope – 1 parse and I saw a man with telescope- another parse)
- I saw the man on the hill with the telescope. 5 parses

- I saw the man on the hill in Texas with the telescope. 14 parses
- I saw the man on the hill in Texas with the telescope at noon. 42 parses
- I saw the man on the hill in Texas with the telescope at noon on Monday. 132 parses

Why is Language Ambiguous?

- The goal in the production and comprehension of natural language is efficient communication.
- Allowing resolvable ambiguity
  -)permits shorter linguistic expressions
  -)avoids language being overly complex
- Language relies on people's ability to use their knowledge and inference abilities to properly resolve ambiguities

Natural Languages vs. Computer Languages

- Ambiguity is the primary difference between natural and computer languages.
- Formal programming languages are designed to be unambiguous
  -) Formal programming languages can be defined by a grammar that produces a unique parse for each sentence in the language.
- Programming languages are also designed for efficient (deterministic) parsing.

Why else is NLP hard?

I DON'T UNDERSTAND A WORD YOUNG PEOPLE SAY THESE DAYS.
C U - I'LL TXT U L8R :)
Chris Madden

Why is NLP hard?

…………….Target is?????

Why Else Is NLP Hard?

Non-standard English

Great job @justinbieber! Were SOO PROUD of what youve accomplished! U taught us 2 #neversaynever & you yourself should never give up either

Segmentation Issues

the New York-New Haven Railroad

the [New] [York-New] [Haven] [Railroad]
the [New York]-[New Haven] [Railroad]

Dealing with Idioms

- dark horse
- Ball in your court
- Burn the midnight oil

neologisms

- Unfriend
- retweet
- Google/Skype/photoshop

Why is NLP hard?

New Senses of a word

- That's sick dude!
- Giants ... multinationals, conglomerates, manufacturers

Tricky Entity Names

- Where is A Bug's Life playing ...
- Let It Be was recorded ...

What we do in NLP?

Tools Required

- Knowledge about language Knowledge about the world
- A way to combine knowledge resources

How is it generally done?

- Probabilistic models built from language data
  -) *P*("maison" → "house") is high
  -) P(I saw a van) > P(eyes awe of an)
- Extracting rough text features does half the job.

Regular Expressions

Refer to class notes discussed problems-

CLASSICAL NLP VS MODERN NLP

A Conceptual Comparison

Introduction

Natural Language Processing (NLP) has evolved significantly over time. Classical NLP relied on rule-based and statistical models, whereas modern NLP leverages deep learning and transformer-based architectures.

Approach & Techniques

Classical NLP:

- Rule-based and Statistical models
- Handcrafted linguistic rules
- Methods: HMM, CRF, SVM, Naïve Bayes

Modern NLP:

- Deep learning-based approaches
- Self-learning neural networks
- Methods: Transformers (BERT, GPT), RNNs, CNNs

Feature Representation

Classical NLP:

- Handcrafted features (TF-IDF, N-grams, POS tagging)
- Requires domain expertise

Modern NLP:

- Automatic feature learning using deep networks
- Word embeddings (Word2Vec, GloVe, BERT embeddings)

Scalability & Data Requirements

Classical NLP:

- Works well with small datasets
- Performance limited by manual feature engineering

Modern NLP:

- Requires large datasets for pre-training
- Pre-trained models can be fine-tuned with minimal data

Interpretability & Explainability

Classical NLP:

- More interpretable due to rule-based structures
- Features like TF-IDF and N-grams are understandable

Modern NLP:

- Black-box nature due to deep learning models
- Explainability tools (SHAP, attention visualization)

Generalization & Adaptability

Classical NLP:

- Task-specific models
- Poor generalization across different domains

Modern NLP:

- Pre-trained models generalize well
- Few-shot and zero-shot learning capabilities

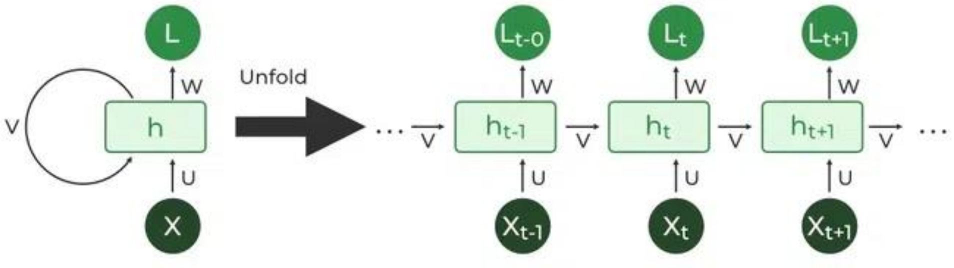
Applications & Evolution

Classical NLP:

- Machine Translation: Rule-based, Statistical MT
- Sentiment Analysis: Naïve Bayes, SVM with TF-IDF

Modern NLP:

- Machine Translation: Transformer-based (Google Translate)
- Sentiment Analysis: BERT-based models



Conclusion

Classical NLP relied on rule-based and statistical methods, requiring extensive manual effort. Modern NLP leverages deep learning, particularly transformers, for improved generalization and scalability.]

EMPIRICAL LAWS

Function Words vs. Content Words

Function words have little lexical meaning but serve as important elements to the structure of sentences.

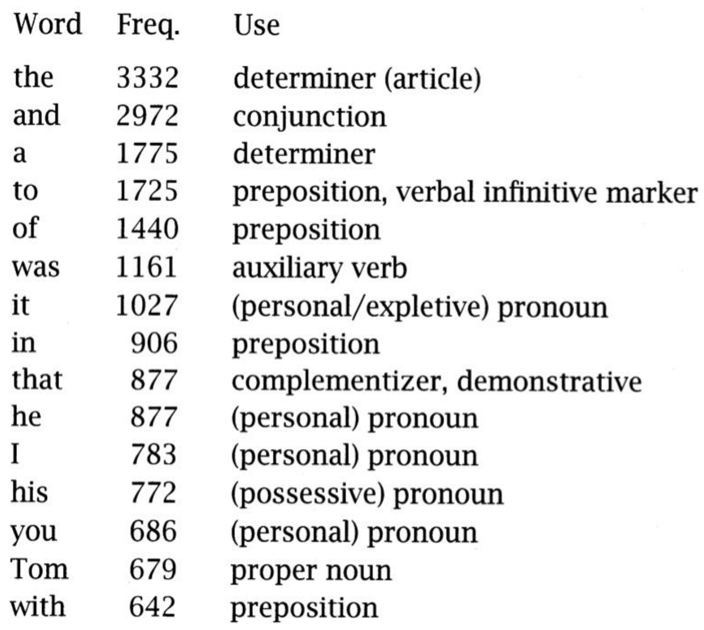
Example

- The winfy prunkilmonger from the glidgement mominkled and brangified all his levensers vederously.
- Glop angry investigator larm blonk government harassed gerfritz infuriated sutbor pumrog listeners thoroughly.

Function words are closed-class words

prepositions, pronouns, auxiliary verbs, conjunctions, grammatical articles, particles etc.

Most Common Words in Tom Sawyer



Word	Freq.	Use
the	3332	determiner (article)
and	2972	conjunction
a	1775	determiner
to	1725	preposition, verbal infinitive marker
of	1440	preposition
was	1161	auxiliary verb
it	1027	(personal/expletive) pronoun
in	906	preposition
that	877	complementizer, demonstrative
he	877	(personal) pronoun
I	783	(personal) pronoun
his	772	(possessive) pronoun
you	686	(personal) pronoun
Tom	679	proper noun
with	642	preposition

- The list is dominated by the little words of English, having important grammatical roles.
- These are usually referred to as *function words*, such as determiners
- The one really exceptional word is *Tom*, whose frequency reflects the text chosen.
- How many words are there in this text?

Type vs. Tokens

Type-Token distinction

Type-token distinction is a distinction that separates a concept from the objects which are particular instances of the concept

Type/Token Ratio

- The type/token ratio (TTR) is the ratio of the number of different words (types) to the number of running words (tokens) in a given text or corpus.
- This index indicates how often, on average, a new ‘word form’ appears in the text or corpus.

Comparison Across Texts

Mark Twain’s Tom Sawyer

- 71,370 word tokens
- 8,018 word types
- TTR = 0.112

Complete Shakespeare work

- 884,647 word tokens
- 29,066 word types
- TTR = 0.032

Empirical Observations on Various Texts

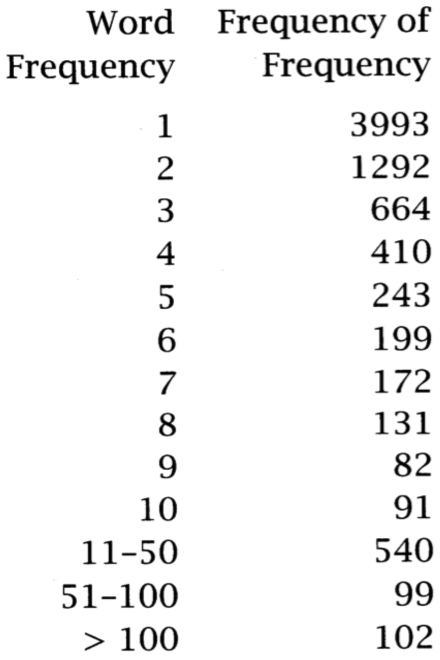
Comparing Conversation, academic prose, news, fiction

- TTR scores the lowest value (tendency to use the same words) in conversation.
- TTR scores the highest value (tendency to use different words) in news.
- Academic prose writing has the second lowest TTR.

Not **a valid measure of ‘text complexity’ by itself**

- The value varies with the size of the text.
- For a valid measure, a running average is computed on consecutive 1000-word chunks of the text.

Word Distribution from Tom Sawyer



Word Frequency	Frequency of Frequency
1	3993
2	1292
3	664
4	410
5	243
6	199
7	172
8	131
9	82
10	91
11–50	540
51–100	99
> 100	102

TTR = 0.11 ⇒ Words occur on average 9 times each.
But words have a very uneven distribution.

Most words are rare

- 3993 (50%) word types appear only once
- They are called *happax legomena*
  (Greek for 'read only once')

But common words are very common

- 100 words account for 51% of all tokens of all text

Zipf's Law

Count the frequency of each word type in a large corpus List the word types in decreasing order of their frequency

Zipf's Law

A relationship between the frequency of a word (f) and its position in the list (its rank r).

$$\boldsymbol{f \propto \frac{1}{r}}$$

or, there is a constant k such that

f .r = k

i.e. the 50th most common word should occur with 3 times the frequency of the 150th most common word.

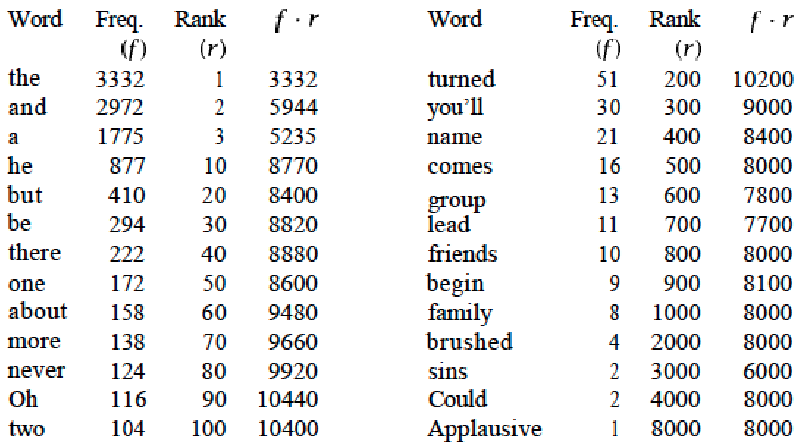
Let

- p_r denote the probability of word of rank *r*
- *N* denote the total number of word occurrences

$$Pr = \frac{f}{N} = \frac{A}{r}$$

The value of *A* is found closer to 0.1 for corpus

Empirical Evaluation from Tom Sawyer



Word	Freq.	Rank	$f \cdot r$	Word	Freq.	Rank	$f \cdot r$
	(f)	(r)			(f)	(r)	
the	3332	1	3332	turned	51	200	10200
and	2972	2	5944	you'll	30	300	9000
a	1775	3	5235	name	21	400	8400
he	877	10	8770	comes	16	500	8000
but	410	20	8400	group	13	600	7800
be	294	30	8820	lead	11	700	7700
there	222	40	8880	friends	10	800	8000
one	172	50	8600	begin	9	900	8100
about	158	60	9480	family	8	1000	8000
more	138	70	9660	brushed	4	2000	8000
never	124	80	9920	sins	2	3000	6000
Oh	116	90	10440	Could	2	4000	8000
two	104	100	10400	Applausive	1	8000	8000

Zipf's Other Laws

Correlation: Number of meanings and word frequency

The number of meanings *m* of a word obeys the law:

$m \propto \sqrt{f}$

Given the First law

$$m \propto \frac{\sqrt{1}}{r}$$

Empirical Support

- Rank ≈ 10000, average 2.1 meanings
- Rank ≈ 5000, average 3 meanings
- Rank ≈ 2000, average 4.6 meanings

Correlation: Word length and word frequency

Word frequency is inversely proportional to their length.

Impact of Zipf's Law

The Good part

Stopwords account for a large fraction of text, thus eliminating them greatly reduces the number of tokens in a text.

The Bad part

Most words are extremely rare and thus, gathering sufficient data for meaningful statistical analysis is difficult for most words.

Vocabulary Growth

How does the size of the overall vocabulary (number of unique words) grow with the size of the corpus?

Heaps' Law

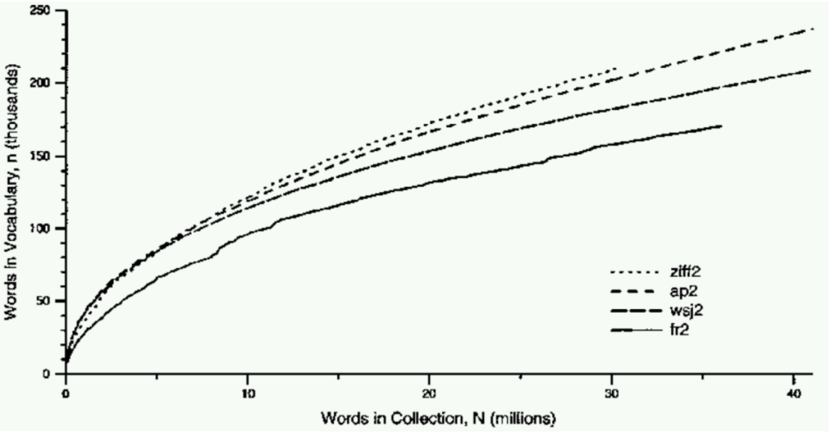
Let |V| be the size of vocabulary and N be the number of tokens.

$|V| = KN^{\beta}$

Typically

- K ≈ 10-100
- $\beta \approx 0.4$ - 0.6 (roughly square root)

Heaps' Law: Empirical Evidence



TEXT PRE-PROCESSING

Text processing: tokenization

What is Tokenization?

Tokenization is the process of segmenting a string of characters into words.

Depending on the application in hand, you might have to perform sentence segmentation as well.

Sentence Segmentation

The problem of deciding where the sentences begin and end.

Challenges Involved

- While '!', '?' are quite unambiguous
- Period "." is quite ambiguous and can be used additionally for
  -) Abbreviations (Dr., Mr., m.p.h.)
  -) Numbers (2.4%, 4.3)

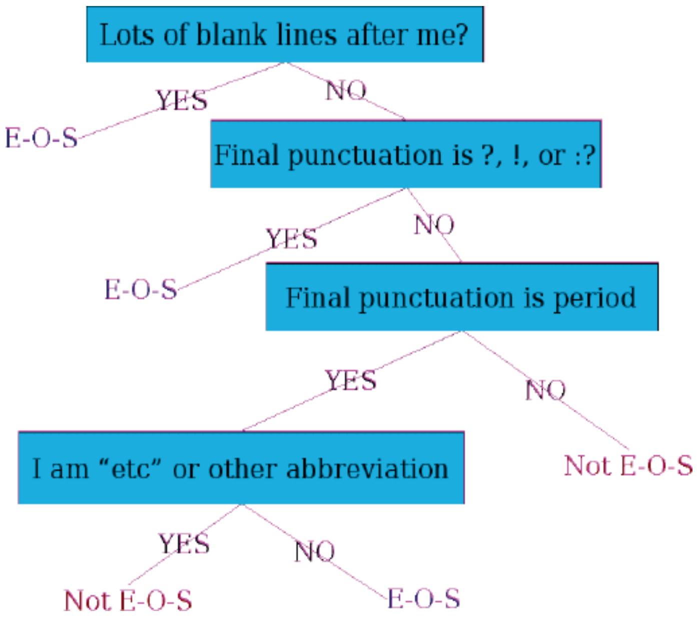
Approach: build a binary classifier

For each "."

- Decides EndOfSentence/NotEndOfSentence
- Classifiers can be: hand-written rules, regular expressions, or machine learning

Sentence Segmentation: Decision Tree Example

Decision Tree: Is this word the end-of-sentence (E-O-S)?


Lots of blank lines after me?
YES
NO
E-O-S
Final punctuation is ?, !, or :?
YES
NO
E-O-S
Final punctuation is period
YES
NO
I am "etc" or other abbreviation
Not E-O-S
YES
NO
Not E-O-S
E-O-S

Other Important Features

- Case of word with ".": Upper, Lower, Cap, Number
- Case of word after ".": Upper, Lower, Cap, Number
- Numeric Features
    -) Length of word with "."
    -) Probability (word with "." occurs at end-of-sentence)
    -) Probability (word after "." occurs at beginning-of-sentence)

Implementing Decision Trees

- Just an if-then-else statement
- Choosing the features is more important
- For numeric features, thresholds are to be picked
- With increasing features including numerical ones, difficult to set up the structure by hand
- Decision Tree structure can be learned using machine learning over a training corpus

Basic Idea

Usually works top-down, by choosing a variable at each step that best splits the set of items.

Popular algorithms: ID3, C4.5, CART

Other Classifiers

The questions in the decision tree can be thought of as features, that could be exploited by any other classifier:

- Support Vector Machines
- Logistic regression
- Neural Networks

Word Tokenization

What is Tokenization?

Tokenization is the process of segmenting a string of characters into words.

I have a can opener; but I can't open these cans.

Word Token

- An occurrence of a word
- For the above sentence, 11 word tokens.

Word Type

- A different realization of a word
- For the above sentence, 10 word types.

Tokenization in practice

- NLTK Toolkit (Python)
- Stanford CoreNLP (Java)
- Unix Commands

Word Tokenization

Issues in Tokenization

- Finland’s → Finland Finlands Finland’s ?
- What’re, I’m, shouldn’t → What are, I am, should not ? San Francisco → one token or two?
- m.p.h. → ??

For information retrieval, use the same convention for documents and queries

Handling Hyphenation

Hyphens can be

End-of-Line Hyphen

Used for splitting whole words into part for text justification.

This paper describes MIMIC, an adaptive mixed initia-tive spoken dialogue system that provides movie show-time information.

Lexical Hyphen

Certain prefixes are offen written hyphenated, e.g. co-, pre-, meta-, multi-, etc.

Sententially Determined Hyphenation

Mainly to prevent incorrect parsing of the phrase. Some possible usages:

- Noun modified by an ‘ed’-verb: *case-based, hand-delivered*
- Entire expression as a modifier in a noun group: *three-to-five-year direct marketing plan*

Language Specific Issues:

French and German

French

l’ensemble: want to match with un ensemble

German

Noun coumpounds are not segmented

- Lebensversicherungsgesellschaftsangestellter
- ‘life insurance company employee’
- Compound splitter required for German information retrieval

Chinese and Japanese

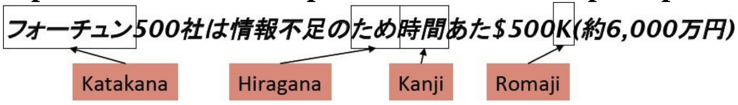
No space between words

莎拉波娃现在居住在美国东南部的佛罗里达。

莎拉波娃 现在 居住 在 美国 东南部 的 佛罗里达

Sharapova now lives in US southeastern Florida

Japanese: further complications with multiple alphabets intermingled.


フォーチュン500社は情報不足のため時間あた$500K(約6,000万円)

Sanskrit

सत्यम्ब्रूयात्प्रियम्ब्रूयान्नब्रूयात्सत्यमप्रियम्प्रियञ्चनानृतम्ब्रूयादेषधर्मःसनातनः

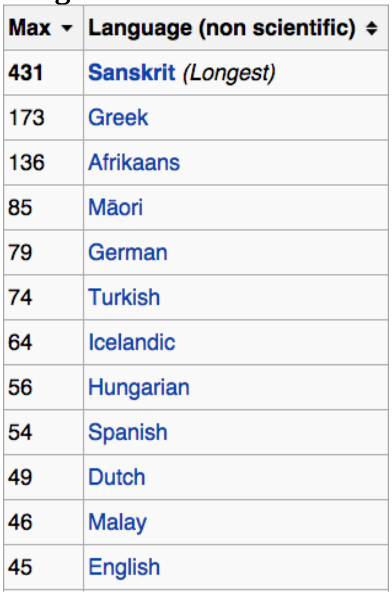
satyam. bru¯ ya¯ tpriyam. bru¯ ya¯ nnabru¯ ya¯ tsatyamapriyam. priyam. cana¯ n.rtambru¯ ya¯ d- es. adharmah. sana¯ tanah. .

“One should tell the truth, one should say kind words; one should neither tell harsh truths, nor flattering lies; this is a rule for all times.”

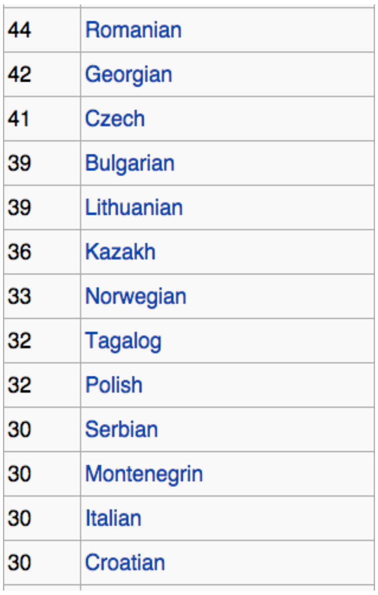
Segmented Text:

satyam bru¯ ya¯ t priyam bru¯ ya¯ t na bru¯ ya¯ t satyam apriyam priyam ca na an.rtam bru¯ ya¯ t es. ah. dharmah. sana¯ tanah. .

Longest Words



Max ▾	Language (non scientific) ⬍
431	Sanskrit *(Longest)*
173	Greek
136	Afrikaans
85	Māori
79	German
74	Turkish
64	Icelandic
56	Hungarian
54	Spanish
49	Dutch
46	Malay
45	English



44	Romanian
42	Georgian
41	Czech
39	Bulgarian
39	Lithuanian
36	Kazakh
33	Norwegian
32	Tagalog
32	Polish
30	Serbian
30	Montenegrin
30	Italian
30	Croatian

Compound word composed of 431 letters, from the Varada¯ mbika¯ Parin˙aya Campu¯ by Tirumala¯ mba

निरन्तरान्धकारिता-दिगन्तर-कन्दलदमन्द-सुधारस-बिन्दु-सान्द्रतर-घनाघन-वृन्द-सन्देहकर-स्यन्दमान-मकरन्द-बिन्दु-बन्धुरतर-माकन्द-तरु-कुल-तल्प-कल्प-मृदुल-सिकता-जाल-जटिल-मूल-तल-मरुवक-मिलदलघु-लघु-लय-कलित-रमणीय-पानीय-शालिका-बालिका-करार-विन्द-गलन्तिका-गलदेला-लवङ्ग-पाटल-घनसार-कस्तूरिकातिसौरभ-मेदुर-लघुतर-मधुर-शीतलतर-सलिलधारा-निराकरिष्णु-तदीय-विमल-विलोचन-मयूख-रेखापसारित-पिपासायास-पथिक-लोकान्

Word Tokenization in
Chinese or Sanskrit

Also called ‘Word Segmentation’.

Greedy Algorithm for Chinese

Maximum Matching (Greedy Algorithm)

- Start a pointer at the beginning of the string
- Find the largest word in dictionary that matches the string starting at pointer
- Move the pointer over the word in string

Think of the cases when word segmentation would be required for English Text.

Finding constituent words in a compound hashtags: #ThankYouSachin, #musicmonday etc.

Text Segmentation for Sanskrit

General assumption behind the design

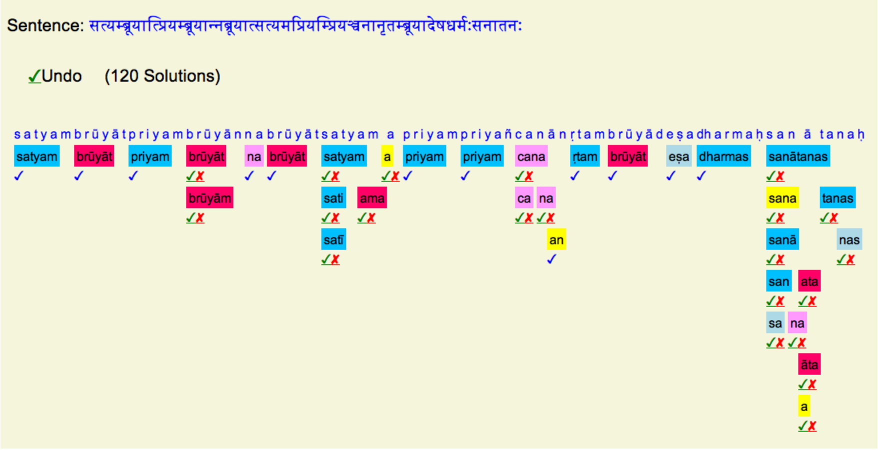
Sentences from Classical Sanskrit may be generated by a regular relation R of the Kleene closure W* of a regular set W of words over a finite alphabet Σ.

- W: vocabulary of (inflected) words (padas) and
- R: sandhi

Analysis of a sentence

A candidate sentence w is analyzed by inverting relation R to produce a finite sequence w_1, w_2,...w_n of word forms, together with a proof that
$w \in R(w_1 \cdot w_2 ... \cdot w_n)$.

Word Segmentation in Sanskrit



Sentence: सत्यम्ब्रूयात्प्रियम्ब्रूयान्नब्रूयात्सत्यमप्रियम्प्रियश्चनानृतम्ब्रूयादेषधर्मःसनातनः

✓Undo (120 Solutions)

satyambrūyātpriyambrūyānnabrūyātsatyam a priyampriyañcanānṛtambrūyādeṣadharmaḥsan ā tanaḥ

Normalization

Why to "normalize"?

Indexed text and query terms must have the same form.

- U.S.A. and USA should be matched
- We implicitly define equivalence classes of terms

Case Folding

- Reduce all letters to lower case
- Possible exceptions (Task dependent):
) Upper case in mid sentence, may point to named entities (e.g. General Motors)
) For MT and inforamtion extraction, some cases might be helpful (*US* vs. *us*)

Lemmatization

- Reduce inflections or variant forms to base form:
) am, are, is → be
) car, cars, car's, cars' → car
- Have to find the correct dictionary headword form

Morphology

Morphology studies the internal structure of words, how words are built up from smaller meaningful units called morphemes

Morphemes are divided into two categories

- Stems: The core meaning bearing units
- Affixes: Bits and pieces adhering to stems to change their meanings and grammatical functions
) Prefix: un-, anti-, etc (a-, ati-, pra- etc.)
) Suffix: -ity, -ation, etc (-taa, -ke, -ka etc.)
) Infix: '*n*' in '*vindati*' (he knows), as contrasted with *vid* (to know).

Stemming

- Reducing terms to their stems, used in information retrieval
- Crude chopping of affixes

) language dependent

) *automate(s), automatic, automation* all reduced to *automat*

for example compressed and compression are both accepted as equivalent to compress.

➡

for exampl compress and compress ar both accept as equival to compress

Porter's algorithm:Stemming(reference)

Step 1a

- sses → ss (caresses → caress)
- ies → i (ponies → poni)
- ss → ss (caress → caress) s
- → φ (cats → cat)

Step 1b

- (*v*)ing → φ (walking → walk, king → king)
- (*v*)ed → φ (played → play)

Step 2

- ational → ate (relational → relate)
- izer → ize (digitizer → digitize) ator
- → ate (operator → operate)

Step 3

- al → φ (revival → reviv)
- able → φ (adjustable → adjust)
- ate → φ (activate → activ)

MINIMUM EDIT DISTANCE

Definition of Minimum Edit Distance

How similar are two strings?

- Spell correction
- The user typed "graffe"

  Which is closest?
  - graf
  - graft
  - grail
  - giraffe
- Computational Biology
- Align two sequences of nucleotides

  AGGCTATCACCTGACCTCCAGGCCGATGCCC
  TAGCTATCACGACCGCGGTCGATTTGCCCGAC
- Resulting alignment:

  -AGGCTATCACCTGACCTCCAGGCCGA--TGCCC---
  TAG-CTATCAC--GACCGC—GGTCGATTTGCCCGAC
- Also for Machine Translation, Information Extraction, Speech Recognition

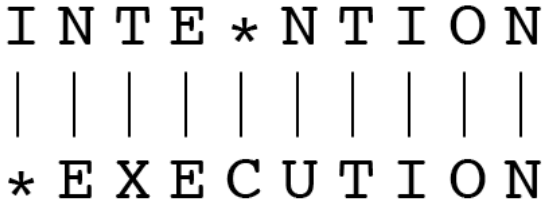
Edit Distance

- The minimum edit distance between two strings
- Is the minimum number of editing operations

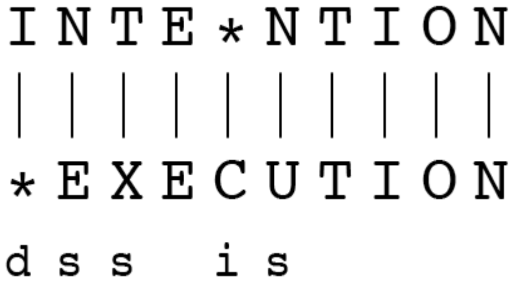
- Insertion
    - Deletion
    - Substitution
- Needed to transform one into the other

Minimum Edit Distance

- Two strings and their **alignment**:



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



```
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 (Levenshtein)
    - Distance between them is 8

Alignment in Computational Biology

- Given a sequence of bases
  AGGCTATCACCTGACCTCCAGGCCGATGCCC
  TAGCTATCACGACCGCGGTCGATTTGCCCGAC
- An alignment:
  -AGGCTATCACCTGACCTCCAGGCCGA--TGCCC---
  TAG-CTATCAC--GACCGC--GGTCGATTTGCCCGAC
- Given two sequences, align each letter to a letter or gap

Other uses of Edit Distance in NLP

- Evaluating Machine Translation and speech recognition

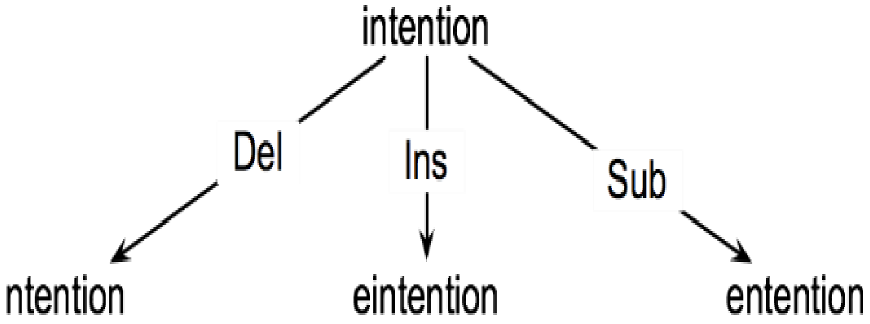
```
R Spokesman confirms    senior government adviser was shot
H Spokesman said    the senior           adviser was shot dead
            S     I         D                       I
```

- Named Entity Extraction and Entity Coreference
    - IBM Inc. announced today
    - IBM profits
    - Stanford President John Hennessy announced yesterday
    - for Stanford University President John Hennessy

How to find the Min Edit Distance?

- Searching for a path (sequence of edits) from the start string to the final string:
    - **Initial state**: the word we're transforming
    - **Operators**: insert, delete, substitute
    - **Goal state**: the word we're trying to get to

    - **Path cost**: what we want to minimize: the number of edits



Minimum Edit as Search

- But the space of all edit sequences is huge!
    - We can’t afford to navigate naïvely
    - Lots of distinct paths wind up at the same state.
        - We don’t have to keep track of all of them
        - Just the shortest path to each of those revisted states.

Defining Min Edit Distance

- For two strings
    - X of length n
    - Y of length m
- We define D(i,j)
    - the edit distance between X[1..i] and Y[1..j]
    - i.e., the first i characters of X and the first j characters of Y
- The edit distance between X and Y is thus D(n,m)

Definition of Minimum Edit Distance

Computing Minimum Edit Distance

Dynamic Programming for Minimum Edit Distance

- Dynamic programming: A tabular computation of D(*n*,*m*)
- Solving problems by combining solutions to subproblems.
- Bottom-up
    - We compute D(i,j) for small *i,j*
    - And compute larger D(i,j) based on previously computed smaller values
    - i.e., compute D(*i,j*) for all *i* (0 < *i* < n) and *j* (0 < j < m)

Defining Min Edit Distance (Levenshtein)

- Initialization

  D(i,0) = i
  D(0,j) = j

- Recurrence Relation*:*

```
For each i = 1…M
    For each j = 1…N
```

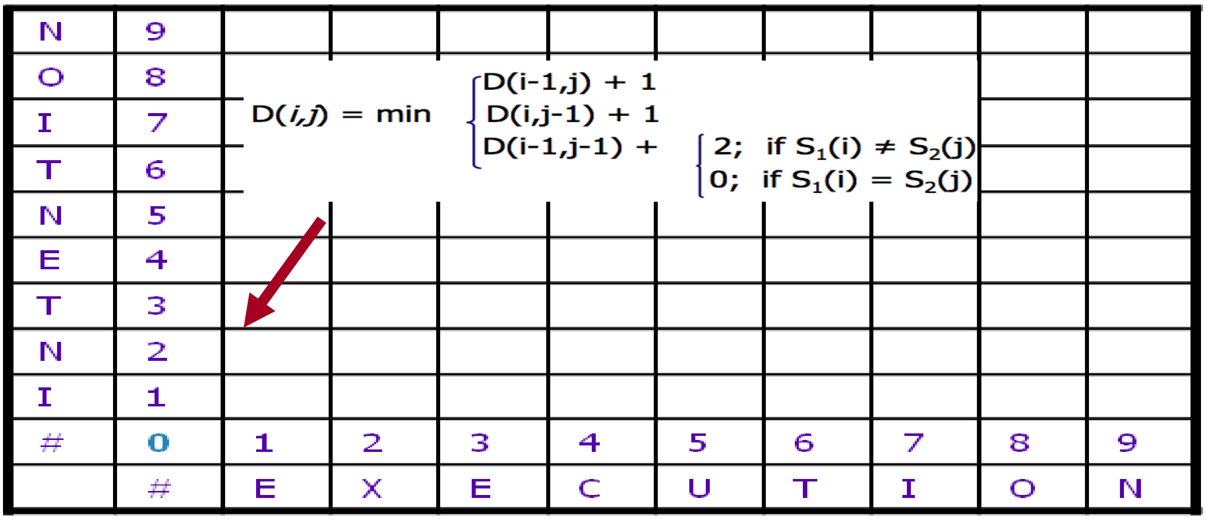
$$D(i,j)=\min\begin{cases}D(i-1,j)+1\\ D(i,j-1)+1\\ D(i-1,j-1)+\begin{cases}2;\ \text{if } X(i)\neq Y(j)\\ 0;\ \text{if } X(i)=Y(j)\end{cases}\end{cases}$$

- Termination*:*

D(N,M) is distance

The Edit Distance Table

N	**9**									
O	**8**									
I	**7**									
T	**6**									
N	**5**									
E	**4**									
T	**3**									
N	**2**									
I	**1**									
#	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
	#	**E**	**X**	**E**	**C**	**U**	**T**	**I**	**O**	**N**



$$D(i,j) = \min \begin{cases} D(i-1,j) + 1 \\ D(i,j-1) + 1 \\ D(i-1,j-1) + \begin{cases} 2; & \text{if } S_1(i) \neq S_2(j) \\ 0; & \text{if } S_1(i) = S_2(j) \end{cases} \end{cases}$$

N	9									
O	8									
I	7									
T	6									
N	5									
E	4									
T	3									
N	2									
I	1									
#	0	1	2	3	4	5	6	7	8	9
	#	E	X	E	C	U	T	I	O	N

N	**9**									
O	**8**									
I	**7**									
T	**6**									
N	**5**									
E	**4**									
T	**3**									
N	**2**									
I	**1**									
#	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**

	#	**E**	**X**	**E**	**C**	**U**	**T**	**I**	**O**	**N**

$$D(i,j) = \min \begin{cases} D(i-1,j) + 1 \\ D(i,j-1) + 1 \\ D(i-1,j-1) + \begin{cases} 2; & \text{if } S_1(i) \neq S_2(j) \\ 0; & \text{if } S_1(i) = S_2(j) \end{cases} \end{cases}$$

N	9	8	9	10	11	12	11	10	9	**8**
O	8	7	8	9	10	11	10	9	8	9
I	7	6	7	8	9	10	9	8	9	10
T	6	5	6	7	8	9	8	9	10	11
N	5	4	5	6	7	8	9	10	11	10
E	4	3	4	5	6	7	8	9	10	9
T	3	4	5	6	7	8	7	8	9	8
N	2	3	4	5	6	7	8	7	8	7
I	1	2	3	4	5	6	7	6	7	8
#	**0**	1	2	3	4	5	6	7	8	9
	#	E	X	E	C	U	T	I	O	N

Computing Minimum Edit Distance
Backtrace for Computing Alignments
Computing alignments:

- Edit distance isn't sufficient
  - We often need to **align** each character of the two strings to each other
- We do this by keeping a "backtrace"
- Every time we enter a cell, remember where we came from
- When we reach the end,
  - Trace back the path from the upper right corner to read off the alignment

Edit Distance

N	9									
O	8									
I	7									
T	6									
N	5									

E	4									
T	3									
N	2									
I	1									
#	**0**	1	2	3	4	5	6	7	8	9
	#	E	X	E	C	U	T	I	O	N

$$D(i,j) = \min \begin{cases} D(i-1,j) + 1 \\ D(i,j-1) + 1 \\ D(i-1,j-1) + \begin{cases} 2; & \text{if } S_1(i) \neq S_2(j) \\ 0; & \text{if } S_1(i) = S_2(j) \end{cases} \end{cases}$$

MinEdit with Backtrace

n	9	↓ 8	↙←↓ 9	↙←↓ 10	↙←↓ 11	↙←↓ 12	↓ 11	↓ 10	↓ 9	↙ **8**	
o	8	↓ 7	↙←↓ 8	↙←↓ 9	↙←↓ 10	↙←↓ 11	↓ 10	↓ 9	↙ **8**	← 9	
i	7	↓ 6	↙←↓ 7	↙←↓ 8	↙←↓ 9	↙←↓ 10	↓ 9	↙ **8**	← 9	← 10	
t	6	↓ 5	↙←↓ 6	↙←↓ 7	↙←↓ 8	↙←↓ 9	↙ **8**	← 9	← 10	←↓ 11	
n	5	↓ 4	↙←↓ 5	↙←↓ 6	↙←↓ 7	↙←↓ **8**	↙←↓ 9	↙←↓ 10	↙←↓ 11	↙↓ 10	
e	4	↙ 3	← 4	↙← **5**	← **6**	← 7	←↓ 8	↙←↓ 9	↙←↓ 10	↓ 9	
t	3	↙←↓ 4	↙←↓ **5**	↙←↓ 6	↙←↓ 7	↙←↓ 8	↙ 7	←↓ 8	↙←↓ 9	↓ 8	
n	2	↙←↓ **3**	↙←↓ 4	↙←↓ 5	↙←↓ 6	↙←↓ 7	↙←↓ 8	↓ 7	↙←↓ 8	↙ 7	
i	**1**	↙←↓ 2	↙←↓ 3	↙←↓ 4	↙←↓ 5	↙←↓ 6	↙←↓ 7	↙ 6	← 7	← 8	
#	**0**	1	2	3	4	5	6	7	8	9	
	#	**e**	**x**	**e**	**c**	**u**	**t**	**i**	**o**	**n**	

Adding Backtrace to Minimum Edit Distance

Base conditions: Termination:

$D(i,0) = i$ $D(0,j) = j$ $D(N,M)$ is distance

Recurrence Relation:

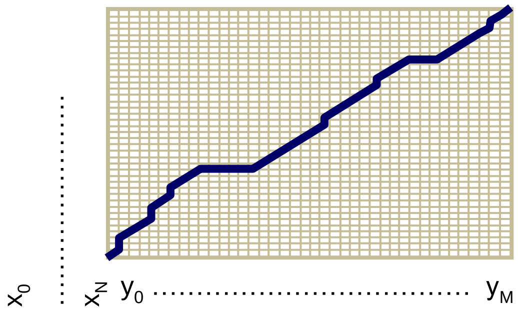
For each $i = 1 \ldots M$

For each $j = 1 \ldots N$

$$D(i,j) = \min \begin{cases} D(i-1,j) + 1 & \text{deletion} \\ D(i,j-1) + 1 & \text{insertion} \\ D(i-1,j-1) + \begin{cases} 2; \text{ if } X(i) \neq Y(j) \\ 0; \text{ if } X(i) = Y(j) \end{cases} & \text{substitution} \end{cases}$$

$$ptr(i,j) = \begin{cases} \text{LEFT} & \text{insertion} \\ \text{DOWN} & \text{deletion} \\ \text{DIAG} & \text{substitution} \end{cases}$$

The Distance Matrix



Slide adapted from Serafim Batzoglou

Every non-decreasing path

from (0,0) to (M, N)

corresponds to

an alignment

An optimal alignment is composed of optimal subalignments

- Two strings and their **alignment**:

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

Performance

- Time:
  O(nm)
- Space:
  O(nm)
- Backtrace
  O(n+m)

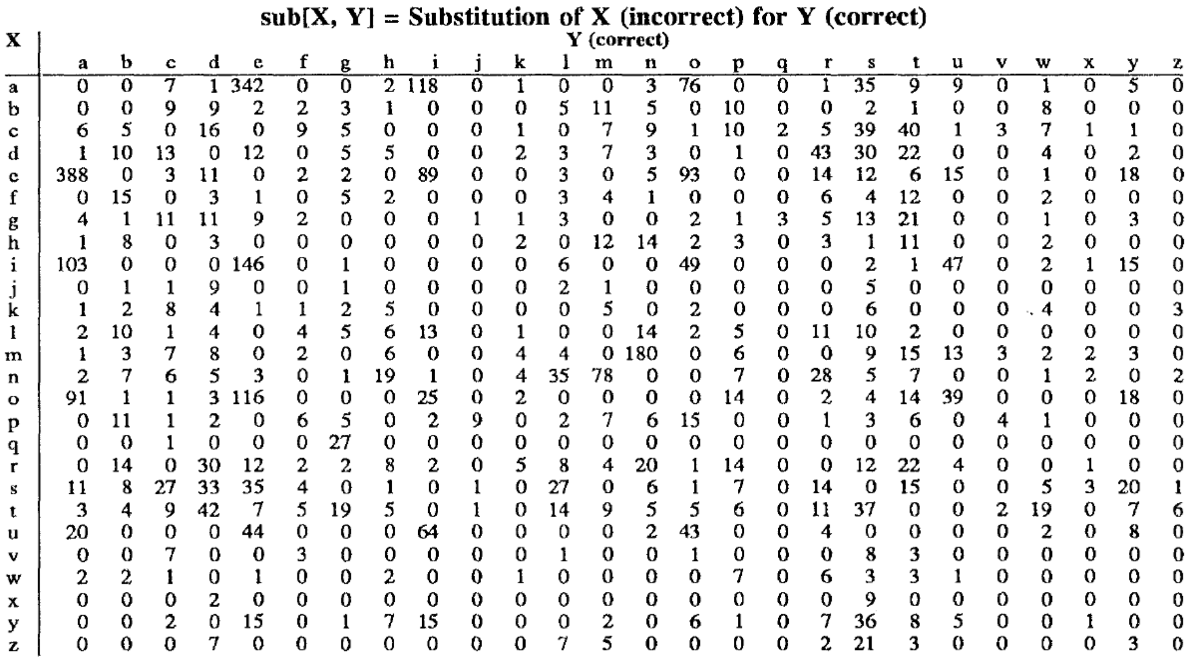
Backtrace for Computing Alignments

Weighted Minimum Edit Distance

Weighted Edit Distance

- Why would we add weights to the computation?
  - Spell Correction: some letters are more likely to be mistyped than others
  - Biology: certain kinds of deletions or insertions are more likely than others

Confusion matrix for spelling errors



sub[X, Y] = Substitution of X (incorrect) for Y (correct)

X	Y (correct)																									
	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a	0	0	7	1	342	0	0	2	118	0	1	0	0	3	76	0	0	1	35	9	9	0	1	0	5	0
b	0	0	9	9	2	2	3	1	0	0	0	5	11	5	0	10	0	0	2	1	0	0	8	0	0	0
c	6	5	0	16	0	9	5	0	0	0	1	0	7	9	1	10	2	5	39	40	1	3	7	1	1	0
d	1	10	13	0	12	0	5	5	0	0	2	3	7	3	0	1	0	43	30	22	0	0	4	0	2	0
e	388	0	3	11	0	2	2	0	89	0	0	3	0	5	93	0	0	14	12	6	15	0	1	0	18	0
f	0	15	0	3	1	0	5	2	0	0	0	3	4	1	0	0	0	6	4	12	0	0	2	0	0	0
g	4	1	11	11	9	2	0	0	0	1	1	3	0	0	2	1	3	5	13	21	0	0	1	0	3	0
h	1	8	0	3	0	0	0	0	0	0	2	0	12	14	2	3	0	3	1	11	0	0	2	0	0	0
i	103	0	0	0	146	0	1	0	0	0	0	6	0	0	49	0	0	0	2	1	47	0	2	1	15	0
j	0	1	1	9	0	0	1	0	0	0	0	2	1	0	0	0	0	0	5	0	0	0	0	0	0	0
k	1	2	8	4	1	1	2	5	0	0	0	0	5	0	2	0	0	0	6	0	0	0	4	0	0	3
l	2	10	1	4	0	4	5	6	13	0	1	0	0	14	2	5	0	11	10	2	0	0	0	0	0	0
m	1	3	7	8	0	2	0	6	0	0	4	4	0	180	0	6	0	0	9	15	13	3	2	2	3	0
n	2	7	6	5	3	0	1	19	1	0	4	35	78	0	0	7	0	28	5	7	0	0	1	2	0	2
o	91	1	1	3	116	0	0	0	25	0	2	0	0	0	0	14	0	2	4	14	39	0	0	0	18	0
p	0	11	1	2	0	6	5	0	2	9	0	2	7	6	15	0	0	1	3	6	0	4	1	0	0	0
q	0	0	1	0	0	0	27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	0	14	0	30	12	2	2	8	2	0	5	8	4	20	1	14	0	0	12	22	4	0	0	1	0	0
s	11	8	27	33	35	4	0	1	0	1	0	27	0	6	1	7	0	14	0	15	0	0	5	3	20	1
t	3	4	9	42	7	5	19	5	0	1	0	14	9	5	5	6	0	11	37	0	0	2	19	0	7	6
u	20	0	0	0	44	0	0	0	64	0	0	0	0	2	43	0	0	4	0	0	0	0	2	0	8	0
v	0	0	7	0	0	3	0	0	0	0	0	1	0	0	1	0	0	0	8	3	0	0	0	0	0	0
w	2	2	1	0	1	0	0	2	0	0	1	0	0	0	0	7	0	6	3	3	1	0	0	0	0	0
x	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9	0	0	0	0	0	0	0
y	0	0	2	0	15	0	1	7	15	0	0	0	2	0	6	1	0	7	36	8	5	0	0	1	0	0
z	0	0	0	7	0	0	0	0	0	0	0	7	5	0	0	0	0	2	21	3	0	0	0	0	3	0

Weighted Min Edit Distance

- Initialization:

  D(0,0) = 0

  D(i,0) = D(i-1,0) + del[x(i)]; $1 < i \le N$

  D(0,j) = D(0,j-1) + ins[y(j)]; $1 < j \le M$

- Recurrence Relation:

$$D(i,j) = \min \begin{cases} D(i-1,j) & + \text{del}[x(i)] \\ D(i,j-1) & + \text{ins}[y(j)] \\ D(i-1,j-1) & + \text{sub}[x(i),y(j)] \end{cases}$$

- Termination:

  D(N,M) is distance

Where did the name, dynamic programming, come from?

…The 1950s were not good years for mathematical research. [the] Secretary of Defense …had a pathological fear and hatred of the word, research…

I decided therefore to use the word, “**programming**”.

I wanted to get across the idea that this was dynamic, this was multistage… I thought, let’s … take a word that has an absolutely precise meaning, namely **dynamic**… it’s impossible to use the word, **dynamic**, in a pejorative sense. Try thinking of some combination that will possibly give it a pejorative meaning. It’s impossible.

Thus, I thought dynamic programming was a good name. It was something not even a Congressman could object to.”

Richard Bellman, “Eye of the Hurricane: an autobiography” 1984.