Why people care about spelling

- People want to appear to be educated.
- Misspellings can cause misunderstandings and real-life problems:
  - For example:
    - Did you see her god yesterday? It’s a big golden retriever.
    - This will be a fee [free] concert.
  - 1991 Bell Atlantic & Pacific Bell telephone network outages were partly caused by a typographical error: A 6 in a line of computer code was supposed to be a D. “That one error caused the equipment and software to fail under an avalanche of computer-generated messages.” (Wall Street Journal, Nov. 25, 1991)

Why people care about spelling (cont.)

- Standard spelling makes it easy to organize words and text:
  - e.g., Without standard spelling, how would you look up things in a lexicon or thesaurus?
  - e.g., Optical character recognition software can use knowledge about standard spelling to recognize scanned words even for barely legible input.
- Standard spelling makes it possible to provide a single text, which is accessible to a wide range of readers (different backgrounds, speaking different dialects, etc.).
- Using standard spelling is associated with being well-educated, i.e., is used to make a good impression in social interaction.

Detection vs. Correction

- There are two distinct tasks:
  - error detection = simply find the misspelled words
  - error correction = correct the misspelled words
- e.g., It might be easy to tell that ater is a misspelled word, but what is the correct word? water? later? after?
  ⇒ Depends on what we want to do with our results as to what we want to do.
  Note, though, that detection is a prerequisite for correction.

What causes errors?

- Keyboard mistypings
- Phonetic errors
- Knowledge problems

Who cares about spelling?

A communicative act, it doesn’t matter in what order the letters in a word are, the only important thing is the frist and last letter be at the right place. The rest can be a toat mes and you can stil read it would poutborm. This is bescue the huumn mind does not raed ervry letter by istlef, but the word as a whole.

(See http://www.mrc-cbu.cam.ac.uk/personal/matt.davis/Cmbridge/ for the story behind this supposed research report.)

A docto has airitted the maglisheus of a tagenee caecn pintaet who deid aetfr a hatopsl durg blendur.

How are spell checkers used?

- interactive spell checkers = spell checker detects errors as you type.
  - It may or not make suggestions for correction.
  - Requires a “real-time” response (i.e., must be fast)
- It is up to the human to decide if the spell checker is right or wrong.
- If there are a list of choices, we may not require 100% accuracy in the corrected word
- automatic spell correctors = spell checker runs on a whole document, finds errors, and corrects them
  - A much more difficult task.
  - A human may or may not proofread the results later.

Keyboard mistypings

- run-on errors = two separate words become one
  - e.g., the fuzz becomes thefuzz
- split errors = one word becomes two separate words
  - e.g., equalization becomes equali zation
Note that the resulting items might still be words!
  - e.g., a tollway becomes atoll way
Keyboard mistypings (cont.)

- **Word replacement:** replacing one word with some
  - e.g., Jack becomes Hack since h and j are next to each other on a typical American keyboard

Physical similarity

- similarity of shape, e.g., mistaking two physically similar letters when typing up something handwritten
  - e.g., tight for fight

More examples for phonetic errors

1. a. death in Venice
   b. deaf in Venice
2. a. give them an ice bucket
   b. give them a nice bucket
3. a. the stuffy nose
   b. the stuff he knows
4. a. the biggest hurdle
   b. the biggest turtle
5. a. a Coke and a danish
   b. a coconut danish

Phonetic errors

- **Phonetic errors** = errors based on the sounds of a language (not necessarily on the letters)
  - **HOMOPHONES** = two words which sound the same
    - e.g., red/read (past tense), cite/site/site, they’re/their/there
  - ** Spoonerisms** = switching two letters/sounds around
    - e.g., Pleating and humming

Knowledge problems

- not knowing a word and guessing its spelling (can be phonetic)
  - e.g., sientist
- not knowing a rule and guessing it
  - e.g., Do we double a consonant for ing words?
  
  jog → joning
  
  Jake → jokking

What makes spelling correction difficult?

- **Tokenization:** What is a word?
- **Inflection:** How are some words related?
- **Productivity of language:** How many words are there?

How we handle these issues determines how we build a dictionary.

Inflection

- A word in English may appear in various guises due to word inflections = word endings which are fairly systematic for a given part of speech
  - plural noun ending: the boy + s → the boys
  - past tense verb ending: walk + ed → walked
- This can make spell-checking hard:
  - There are exceptions to the rules: mans, runned
  - There are words which look like they have a given ending, but they don’t: Hans, deed

Productivity

- part of speech change: nouns can be verbified
  - tabled is a new verb coined after the noun table
- **Calvin:** I like to verb words.
- **Hobbes:** What?
- **Calvin:** I take nouns and adjectives and use them as verbs. Remember when “access” was a thing? Now, it's something you do. It got verbed. Verbing weirds language.
- **Hobbes:** Maybe we can eventually make language a complete impediment to understanding.
- **morphological productivity:** prefixes and suffixes can be added
  - e.g., I can speak of un-email-able for someone who you can’t reach by email
- **words entering and exiting the lexicon, e.g.:**
  - thou, or spleet ‘split’ (Hamlet III.2.10) are on their way out
  - SARS and wifi (or wi-fi) have entered recently
Non-word error detection

- **Non-word error detection** is essentially the same thing as **word recognition** – splitting up "words" into true words and non-words.

- **How is non-word error detection done?**
  - using a dictionary (construction and lookup)
  - n-gram analysis

Dictionary lookup

- **Dictionary lookup** = lookup a potential word in the dictionary (how do you do this quickly?)

N-gram analysis

- **An n-gram** here is a string of n letters.
  - a 1-gram (unigram)
  - at 2-gram (bigram)
  - ate 3-gram (trigram)
  - late 4-gram

How do we store and use n-gram information?

- Store the number of times an n-gram appears (like in Language Identification). But, maybe we just want to know if an n-gram is possible.

- We could have a list of possible and impossible n-grams (1 = possible, 0 = impossible):
  - on 1
  - ket 0
  - police 1
  - asdf 0

- Any word which has a 0 for any substring is a misspelled word.

- Problems with such an approach:
  - Information is repeated (po is in police)
  - Requires a lot of computer storage space
  - Inefficient (slow) when looking up every string

Bigram array

- Instead, we can define a **bigram array** = information stored in a tabular fashion.

An example, for the letters k, l, m, with examples in parentheses

```
  ... k l m ...
  ...
  k 0 1 (tackle) 1 (Hackman)
  l 1 (elk) 1 (hello) 1 (alms)
  m 0 0 1 (hammer)
```

- The first letter of the bigram is given by the vertical letters (i.e., down the side), the second by the horizontal ones (i.e., across the top).

- This is a **non-positional bigram array** = the array 1's and 0's apply for a string found anywhere within a word (beginning, 4th character, ending, etc.).

Positional bigram array

- To store information specific to the beginning, the end, or some other position in a word, we can use a **positional bigram array** = the array only applies for a given position in a word.

Here's the same array as before, but now only applied to word endings:

```
  ... k l m ...
  ...
  k 0 0 0
  l 1 (elk) 1 (half) 1 (elm)
  m 0 0 0
```
Having discussed how errors can be detected, we want to know how to correct these misspelled words:

- The most common method is isolated-word error correction – correcting words without taking context into account.
- Note: This technique can only handle errors that result in non-words.
- Knowledge about what is a typical error helps in finding correct word.

### Rule-based methods

One can generate correct spellings by writing rules:

- Common misspelling rewritten as correct word:
  - e.g., hte → the
- Rules
  - based on inflections:
    - e.g., ViCing → ViCIng
    - (where V = vowel and C = consonant)
  - based on other common spelling errors (such as keyboard effects or common transpositions):
    - e.g., CaC → Cac
    - e.g., Cie → Cei

### Similarity key techniques

- Problem: How can we find a list of possible corrections?
- Solution: Store words in different boxes in a way that puts the similar words together.

- Example:
  1. Start by storing words by their first letter (first letter effect),
     - e.g., punct starts with the code P
  2. Then assign numbers to each letter
     - e.g., 0 for vowels, 1 for b, p, f, v (all bilabials), and so forth, e.g., punct → PS02
  3. Then throw out all zeros and repeated letters,
     - e.g., PS02 → PS2
  4. Look for real words within the same box,
     - e.g., punk is also in the PS2 box.

### Probabilistic methods

Two main probabilities are taken into account:

- transition probabilities = probability (chance) of going from one letter to the next.
  - e.g., What is the chance that a will follow p in English?
    - That will follow p?
- confusion probabilities = probability of one letter being mistaken (substituted) for another (can be derived from a confusion matrix)
  - e.g., What is the chance that q is confused with p?

Useful to combine probabilistic techniques with dictionary methods

### Minimum edit distance

- In order to rank possible spellings corrections, it can be useful to calculate the minimum edit distance.
- Minimum number of operations it would take to convert one word into another.
- For example, we can take the following five steps to convert junk to haku:
  1. junk → juk (deletion)
  2. juk → huk (substitution)
  3. huk → hku (transposition)
  4. hku → haku (insertion)
  5. huku → haku (insertion)

- But is this the minimal number of steps needed?
To be able to compute the edit distance of two words at all, we need to ensure there is a finite number of steps. This can be accomplished by:

- requiring that letters cannot be changed back and forth a potentially infinite number of times, i.e., we limit the number of changes to the size of the material we are presented with, the two words.

- Idea: Never deal with a character in either word more than once.

- Result:
  - In the worst case, we delete each character in the first word and then insert each character of the second word.
  - The worst case edit distance for two words is \( \text{length}(\text{word} 1) + \text{length}(\text{word} 2) \)

---

### Computing edit distances
**Figuring out the worst case**

- The graph is **acyclic** = for any given node, it is impossible to return to that node by following the arcs.

- We can add identifiers to the states, which allows us to define a **topological order**:

---

### Computing edit distances
**Adding numbers to the example graph**

- The path is **acyclic** for any given node, it is impossible to return to that node by following the arcs.

- We can add identifiers to the states, which allows us to define a **topological order**:

---

### Computing edit distances
**Computing edit distances**

- To calculate minimum edit distance, we set up a directed, acyclic graph, a set of nodes (circles) and arcs (arrows).

- Horizontal arcs correspond to deletions, vertical arcs correspond to insertions, and diagonal arcs correspond to substitutions (and a letter can be “substituted” for itself).

- Instead of assuming the same cost for all operations, in reality one will use different costs, e.g., for the first character or based on the confusion probability.

---

### Computing edit distances
**Computing costs to the arcs of the example graph**

- We need to add the costs involved to the arcs.

- In the simplest case, the cost of deletion, insertion, and substitution is 1 each (and substitution with the same character is free).

---

### Grammar correction—what does it correct?

- Syntactic errors = errors in how words are put together in a sentence: the order or form of words is incorrect, i.e., ungrammatical.

- **Local** syntactic errors: 1-2 words away
  - e.g., *The study was conducted mainly be John Black.*
  - A verb is where a preposition should be.

- **Long-distance** syntactic errors: (roughly) 3 or more words away
  - e.g., *The kids who are most upset by the little totem is going home early.*
  - Agreement error between subject kids and verb is
**More on grammar correction**

- Semantic errors = errors where the sentence structure sounds okay, but it doesn’t really mean anything.
  - e.g., They are leaving in about fifteen minutes to go to her.

  → *minuets* and *minutes* are both plural nouns, but only one makes sense here.

There are many different ways in which grammar correctors work, two of which we’ll focus on:

- **Bigram model** (bigrams of words)
- **Rule-based model**

**Beyond regular expressions**

- But what about correcting the following:
  - *A baseball teams were successful.*
  - We should change *A* to *The*, but a simple regular expression doesn’t work because we don’t know where the word teams might show up.
  - *A wildly overpaid, horrendous baseball teams were successful.* (Five words later; change needed.)
  - *A player on both my teams was successful.* (Five words later; no change needed.)

- We need to look at how the sentence is constructed in order to build a better rule.

**Syntax**

- **Syntax** = the study of the way that sentences are constructed from smaller units.
  - There cannot be a “dictionary” for sentences since there is an infinite number of possible sentences:
    - (6) The house is large.
    - (7) John believes that the house is large.
    - (8) Mary says that John believes that the house is large.

  There are two basic principles of sentence organization:
  - **Linear order**
  - **Hierarchical structure (Constituency)**

**Bigram grammar correctors**

We could also look at **bigrams**: now we are talking about bigrams of words, i.e., two words appearing next to each other.

- **Question**: Given the previous word, what is the probability of the current word?
  - e.g., given these, we have a 5% chance of seeing *reports* and a 0.001% chance of seeing *report* (these report cards).
  - Thus, we will change report to reports.

- But there’s a major problem: we may hardly ever see these reports, so we won’t know the probability of that bigram.

- **(Partial) Solution**: use bigrams of parts of speech.
  - e.g., What is the probability of a noun given that the previous word was an adjective?

**Rule-based grammar correctors**

We can write regular expressions to target specific error patterns. For example:

- **To a certain extend, we have achieved our goal**.
  - Match the pattern some or certain followed by extend, which can be done using the regular expression some|certainextend.
  - Change the occurrence of extend in the pattern to extend.

- Naber (2003) uses 56 such rules to build a grammar corrector which works nearly as well as that in commercial products.

**Constituency**

- What are the “meaningful units” of a sentence like *Many executives eat at really fancy restaurants*?
  - Many executives
  - really fancy
  - really fancy restaurants

  → at really fancy restaurants

- We refer to these meaningful groupings as **constituents** of a sentence.

  → There are many tests to determine what a constituent is, but we will not concern ourselves with them here.

**Hierarchical structure**

- Constituents can appear within other constituents. We can represent this in a bracket form or in a **syntactic tree**.
  - Constituents shown through brackets:
    - *[[Many executives] [eat [at [[really fancy] restaurants]]]]*
  - Constituents displayed as a tree:

```
       a
      / \
     b   c
    /     \    
   d       e    
  /         /   
 f restaurants
```

**Categories**

- We would also like some way to say that *Many executives and really fancy restaurants are the same type of grouping, or constituent, whereas at really fancy restaurants seems to be something else.*

  → For this, we will talk about different **categories**
    - Lexical
    - Phrasal
Determining lexical categories

How do we determine which category a word belongs to?
- Distribution: Where can these kinds of words appear in a sentence?
  - e.g., Nouns like mouse can appear after articles ("determiners") like the, while a verb like eat cannot.
- Morphology: What kinds of word prefixes/suffixes can a word take?
  - e.g., Verbs like walk can take a ed ending to mark them as past tense. A noun like mouse cannot.

Phrasal categories (cont.)

- Susan
- students
- you
- most dogs
- some children
- a huge, lovable bear
- my friends from Brazil
- the people that we interviewed

Since all of these contain nouns, we consider these to be noun phrases, abbreviated with NP.

Phrasal categories

What about phrases? Can we assign them categories? We can also look at their distribution and see which ones behave in the same way.
- The joggers ran through the park.

What other phrases can we put in place of The joggers?

Phrase Structure Rules

- We can give rules for building these phrases. That is, we want a way to say that a determiner and a noun make up a noun phrase, but a verb and an adverb do not.
- Phrase structure rules are a way to build larger constituents from smaller ones.
  - e.g., S → NP VP
  This says:
  - A sentence (S) constituent is composed of a noun phrase (NP) constituent and a verb phrase (VP) constituent. (hierarchy)
  - The NP must precede the VP. (linear order)

Some other English rules

- NP → Det N (the cat, a house, this computer)
- NP → Det AdjP N (the happy cat, a really happy house)
  - For phrase structure rules, as shorthand parentheses are used to express that a category is optional.
  - We thus can compactly express the two rules above as one rule:
    - NP → Det (AdjP) N
  - Note that this is different and has nothing to do with the use of parentheses in regular expressions.
- AdjP → (Adv) Adj (really happy)
- VP → V (laugh, run, eat)
- VP → V NP (love John, hit the wall, eat cake)
- VP → V NP NP (give John the ball)
- PP → P NP (to the store, at John, in a New York minute)
- NP → NP PP (the cat on the stairs)
The man in the kitchen drives a truck.

16 errors 17 errors
generative
bottom-up
The discussion is based on Markus Dickinson (to
For a discussion of the confusion matrix, cf. Mark D.
A baseball teams were successful.
It specifies that each rule must have:
\[ \text{terminals} = \text{actual words} \]
A CFG tries to capture a natural language completely.
Why "context-free"? Because these rules make no reference to any context surrounding them. i.e. you can't say “PP → P NP” when there is a verb phrase (VP) to the left.

Is this really how spell checkers work?
As far as we know, yes, but:
- Many spell checkers are proprietary and the way they work is kept secret; we don’t know how they work exactly, which hampers research and thereby progress.
- Others, such as aspell and ispell, are open source spell checkers, meaning that anyone can
  - contribute to their further development, and
  - see how they work, which makes it possible to understand exactly what they will and what they won’t catch.
  (cf. http://aspell.sourceforge.net/ and

Context-free grammars
- A context-free grammar (CFG) is essentially a collection of phrase structure rules.
  - It specifies that each rule must have:
    - a left-hand side (LHS): a single non-terminal element → (phrasal and lexical categories)
    - a right-hand side (RHS): a mixture of non-terminal and terminal elements

Parsing
So, using these phrase structure (context-free) rules, parse
a sentence = assign a structure to a sentence. Do you parse top-down or bottom-up (or a mixture)?
- top-down: build a tree by starting at the top (i.e. \( S → NP \ VP \)) and working down the tree.
- bottom-up: build a tree by starting with the words at the bottom and working up to the top.
There are many, many parsing techniques out there.

Properties of Phrase Structure Rules
- generative = a schematic strategy that describes a set of sentences completely.
- potentially (structurally) ambiguous = have more than one analysis
(13) We need more intelligent leaders.
(14) Paraphrases:
  a. We need leaders who are more intelligent.
  b. Intelligent leaders? We need more of them!
- hierarchical = categories have internal structure; they aren’t just linearly ordered.
- recursive = property allowing for a rule to be reapplied (within its hierarchical structure).
  e.g., \( NP \rightarrow NP PP \)
PP → P NP
The property of recursion means that the set of potential sentences in a language is infinite.

Writing grammar correction rules
So, with context-free grammars, we can now write some correction rules, which we will just sketch here.
- A baseball teams were successful.
A followed by PLURAL NP: change A → The
- John at the taco.
The structure of this sentence is NP PP, but that doesn’t make up a whole sentence. We need a verb somewhere.

A Poem on the Dangers of Spell Checkers
Michael Livingston
(http://www.courses.rochester.edu/livingston/guide/phonix.html)
Eye halve a spelling chequer
It came with my pea sea.
It plainly marques four my revue
Miss steaks eye kin knot sea.
Eye strike a key and type a word
Weather eye am wrong oar write
It shows me strait a weigh.
As soon as a mist ache is maid
It nose bee fore two long
And eye can put the error rite
Its rare lea ever wrong.
Eye have run this poem throw it
I am shore your pleased two no
Its letter perfect awl the weigh
My chequer tolled me sew.