HG8003 Technologically Speaking:
The intersection of language and technology.

Representing Language

Francis Bond
Division of Linguistics and Multilingual Studies
http://www3.ntu.edu.sg/home/fcbond/
bond@ieee.org

Lecture 2
Location: LT8
# Schedule

<table>
<thead>
<tr>
<th>Lec.</th>
<th>Date</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>01-16</td>
<td>Introduction, Organization: Overview of NLP; Main Issues</td>
</tr>
<tr>
<td>2</td>
<td>01-23</td>
<td>Representing Language</td>
</tr>
<tr>
<td>3</td>
<td>02-06</td>
<td>Representing Meaning</td>
</tr>
<tr>
<td>4</td>
<td>02-13</td>
<td>Words, Lexicons and Ontologies</td>
</tr>
<tr>
<td>5</td>
<td>02-20</td>
<td>Text Mining and Knowledge Acquisition</td>
</tr>
<tr>
<td>6</td>
<td>02-27</td>
<td>Structured Text and the Semantic Web</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Quiz</strong></td>
</tr>
<tr>
<td>7</td>
<td>03-13</td>
<td>Citation, Reputation and PageRank</td>
</tr>
<tr>
<td>8</td>
<td>03-20</td>
<td>Introduction to MT, Empirical NLP</td>
</tr>
<tr>
<td>9</td>
<td>03-27</td>
<td>Analysis, Tagging, Parsing and Generation</td>
</tr>
<tr>
<td>10</td>
<td>Video</td>
<td>Statistical and Example-based MT</td>
</tr>
<tr>
<td>11</td>
<td>04-03</td>
<td>Transfer and Word Sense Disambiguation</td>
</tr>
<tr>
<td>12</td>
<td>04-10</td>
<td>Review and Conclusions</td>
</tr>
<tr>
<td>Exam</td>
<td>05-06</td>
<td>17:00</td>
</tr>
</tbody>
</table>

> Video week 10
Introduction

- Review
- Representing Text
- Representing Speech
- Relating written and spoken language
Review of week 1
Natural language is ambiguous

We need to resolve this ambiguity for many tasks

- Humans are good at this task
- Machines find it hard
Layers of Language

There are many layers of linguistic analysis

1. Phonetics & Phonology (sound)
2. Morphology (sub-word)
3. Syntax (grammar/structure)
4. Semantics (sentence meaning)
5. Pragmatics (contextual meaning)
If we want to do anything with language, we need a way to represent language.

We can interact with the computer in several ways:

- write or read text
- speak or listen to speech

The computer has to have some way to represent either text (writing) or speech (sound)
Text
Representing Text

➤ Analog representation
  ➢ continuous, hard to reproduce, index and transmit

➤ Digital Representation
  ➢ discrete, easy to reproduce, index and transmit
What is writing?

A system of more or less permanent marks used to represent an utterance in such a way that it can be recovered more or less exactly without the intervention of the utterer.

*Peter T. Daniels, The World's Writing Systems*

The earliest intersection of language and technology.
The Origins of Writing

➤ Writing was invented independently in at least three places:

➢ Mesopotamia
➢ China
➢ Mesoamerica

Possibly also Egypt and Indus.

➤ The written records are incomplete

➤ Gradual development from pictures/tallies
Follow the money

Before 2700, writing is only accounting.

- Temple and palace accounts
- Gold, Wheat, Sheep

How it developed

- One token per thing (in a clay envelope)
- One token per thing in the envelope and marked on the outside
- One mark per thing
- One mark and a symbol for the number
- Finally symbols for names

Clay Tokens and Envelope  

Clay Tablet
What is represented by writing?

- **Phonemes:** /maɪ dɔɡ laɪks ævəkədoʊz/  
  (45)

- **Syllables:** (maɪ) (dɔɡ) (laɪks) (æ)(və)(ka)(doz)  
  (10,000+)

- **Morphemes:** maɪ/me + ’s dɔɡ laɪk + s ævəkədo + z  
  (100,000+)

- **Words:** *my dog likes avocados*  
  (200,000+)

- **Concepts:** speaker poss canine fond avacado+PL  
  (400,000+)
Writing systems used for human languages

There are three major types of writing systems:

- Alphabetic
- Syllabic
- Logographic
Alphabetic systems

➤ **Alphabets** (phonemic alphabets)
  ➢ represent all sounds, i.e., consonants and vowels
  ➢ Examples: Etruscan, Latin, Cyrillic, Runic, International Phonetic Alphabet

➤ **Abjads** (consonant alphabets)
  ➢ represent consonants only (sometimes plus selected vowels; vowel diacritics generally available)
  ➢ Examples: Arabic, Aramaic, Hebrew
Alphabet example: Russian

The Cyrillic alphabet is used to write many languages, mainly Slavic. Here is the set used for Russian.

(from: http://www.omniglot.com/writing/russian.htm)
Abjad example: Phoenician

An abjad used to write Phoenician, created between the 18th and 17th centuries BC; assumed to be the forerunner of the Greek and Hebrew alphabet.

(from: http://www.omniglot.com/writing/phoenician.htm)
A note on the letter-sound correspondence

- Alphabets use letters to encode sounds (consonants, vowels).

- But the correspondence between spelling and pronunciation in many languages is quite complex, i.e., not a simple one-to-one correspondence.

- Example: English

  - same spelling – different sounds: ough: *ought, cough, tough, through, though, hiccough*
  - silent letters: *knee, knight, knife, debt, psychology, mortgage*
  - one letter – multiple sounds: *exit*
  - multiple letters – one sound: *the, revolution*
  - alternate spellings: *jail or gaol*
Syllabic systems

➤ Syllabic alphabets (Alphasyllabaries)

➤ writing systems with symbols that represent a consonant with a vowel, but the vowel can be changed by adding a diacritic (= a symbol added to the letter).

➤ Examples: Balinese, Javanese, Tibetan, Tamil, Thai, Tagalog

➤ Syllabaries

➤ writing systems with separate symbols for each syllable of a language

➤ Examples: Cherokee, Ethiopic, Cypriot, Ojibwe, Hiragana (Japanese)
Syllabic alphabet example: Lao

Script developed in the 14th century to write the Lao language, based on an early version of the Thai script, which was developed from the Old Khmer script, which was itself based on Mon scripts.

Example for vowel diacritics around the letter k:
Syllabic alphabet example: Hiragana

Script developed in 10th century from Chinese characters. 52 characters.
Logographic writing systems

- **Logographs** (also called Logograms):
  - Pictographs (Pictograms): originally pictures of things, now stylized and simplified.
  - Ideographs (Ideograms): representations of abstract ideas
  - Compounds: combinations of two or more logographs.
  - Semantic-phonetic compounds: symbols with a meaning element (hints at meaning) and a phonetic element (hints at pronunciation).

- **Examples:** Chinese, Japanese, Mayan, Vietnamese, Ancient Egyptian
## Development of Chinese character horse

<table>
<thead>
<tr>
<th>Type of Characters</th>
<th>Descriptions</th>
</tr>
</thead>
</table>
| 金文 | Bronze script  
Jin wen  
15th - 11th centuries B.C.E. |
| 甲骨文 | Oracle-bone script  
Jia gu wen  
12th - 11th centuries B.C.E. |
| 大篆 | Large-seal script  
da chuan  
c. 6th century B.C.E. |
| 小篆 | Small-seal script  
xiao chuan  
2nd century B.C.E. |
| 賽書 | Clerical script  
lì shu  
2nd century C.E. |
| 楷書 | Standard script  
k'ai shu  
since c. 4th century C.E. |
| 行書 | Running script  
Xing shu  
since c. 4th century C.E. |
| 草書 | Cursive script  
Chao shu  
since c. 4th century C.E. |
Logograph writing system example: Chinese

- **Pictographs**
  - woman
  - child
  - mouth
  - sun
  - moon
  - mountain
  - river
  - pig
  - eye
  - heart
  - rain
  - field
  - tree
  - turtle

- **Ideographs**
  - one
  - two
  - three
  - above
  - below
  - middle
  - strength
  - convex
  - concave

- **Compounds of Pictographs/Ideographs**
  - good
  - peaceful
  - bright
  - home/family
  - thought
  - prison
  - thunder
  - man/male

Representing Language
97% of Chinese characters are phonetic compounds!
An example from Ancient Egyptian

\[ msh \text{ (crocodile)} = m + s + h \text{ crocodile} = \text{cat (miw)} \]
A writing system with an unusual realization

➤ Tactile

➤ Braille is a writing system that makes it possible to read and write through touch; primarily used by the (partially) blind.
➤ It uses patterns of raised dots arranged in cells of up to six dots in a 3 x 2 configuration.
➤ Each pattern represents a character, but some frequent words and letter combinations have their own pattern.
### Braille alphabet

#### Basic letters

<table>
<thead>
<tr>
<th>•</th>
<th>⠅</th>
<th>⠆</th>
<th>⠇</th>
<th>⠈</th>
<th>⠉</th>
<th>⠊</th>
<th>⠋</th>
<th>⠌</th>
<th>⠍</th>
<th>⠎</th>
<th>⠏</th>
<th>⠐</th>
<th>⠑</th>
<th>⠒</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b</td>
<td>c</td>
<td>d</td>
<td>e</td>
<td>f</td>
<td>g</td>
<td>h</td>
<td>i</td>
<td>j</td>
<td>k</td>
<td>l</td>
<td>m</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>⠎</th>
<th>⠏</th>
<th>⠐</th>
<th>⠑</th>
<th>⠒</th>
<th>⠓</th>
<th>⠔</th>
<th>⠕</th>
<th>⠖</th>
<th>⠗</th>
<th>⠘</th>
<th>⠙</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>o</td>
<td>p</td>
<td>q</td>
<td>r</td>
<td>s</td>
<td>t</td>
<td>u</td>
<td>v</td>
<td>w</td>
<td>x</td>
<td>y</td>
</tr>
</tbody>
</table>

#### Accented letters

| ⠐ | ⠑ | ⠒ | ⠓ | ⠔ | ⠕ | ⠖ | ⠗ | ⠘ | ⠙ | ⠐ | ⠑ | ⠒ | ⠓ | ⠔ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| à | â | ä/æ | è | é | ê | è | ì | î |

| ⠐ | ⠑ | ⠒ | ⠓ | ⠔ | ⠕ | ⠖ | ⠗ | ⠘ | ⠙ | ⠐ | ⠑ | ⠒ | ⠓ | ⠔ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| í | ò | ô | ö/œ | ù | û | ü | ç |
### Words and abbreviations

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>but</th>
<th>can</th>
<th>do</th>
<th>every</th>
<th>from</th>
<th>go</th>
<th>have</th>
<th>just</th>
<th>knowledge</th>
<th>like</th>
<th>more</th>
<th>not</th>
</tr>
</thead>
<tbody>
<tr>
<td>Braille</td>
<td>people</td>
<td>quite</td>
<td>rather</td>
<td>so</td>
<td>that</td>
<td>us</td>
<td>very</td>
<td>will</td>
<td>it</td>
<td>you</td>
<td>as</td>
<td>and</td>
<td>for</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>of</th>
<th>the</th>
<th>with</th>
<th>child/ch</th>
<th>gh</th>
<th>shall/sh</th>
<th>this/th</th>
<th>which/wh</th>
<th>ed</th>
<th>er</th>
<th>out/ou</th>
<th>ow</th>
<th>bb</th>
</tr>
</thead>
</table>

|       | cc    | dd    | en    | gg; were | in   | st     | ing    | ar     |       |        |

Braille terminals (refreshable Braille displays) push the pins up in real time
I learned to read Braille a while back, and I’ve noticed that the messages on signs don’t always match the regular text.

(Cartoon from http://xkcd.com/315/)

Braille Secrets
There is never a simple correspondence between a writing system and a language.

For example, English uses the Roman alphabet, but Arabic numerals (e.g., 3 and 4 instead of III and IV).

Even when a new alphabet is designed, pronunciation changes.
Comparison of writing systems

The pros and cons of each type of system depend on a variety of factors:

**Accuracy:** Can every word be written down accurately?

**Learnability:** How long does it take to learn the system?

**Cognitive ability:** Are some systems unnatural? (e.g. Does dyslexia show that alphabets are unnatural?)

**Language-particular differences:** English has thousands of possible syllables; Japanese has very few in comparison (52)

**Connection to history/culture:** Is there meaning in the system beyond its use as a writing system?
Encoding written language on a computer

➢ Information on a computer is stored in **bits**.

➢ A bit is either on (= 1, yes) or off (= 0, no).

➢ A list of 8 bits makes up a byte, e.g., 01001010

➢ Just like with the base 10 numbers we’re used to, the order of the bits in a byte matters:

   ➢ Big Endian: most important bit is leftmost (the standard way of doing things)
   ➢ Little Endian: most important bit is rightmost (only used on Intel machines)
How much information in a byte?

➢ Every bit encodes two states (1 or 0)

➢ $n$ bits encodes $2^n$ states

➢ $2 \times 2 \times 2 \times 2 \ldots n$ times

➢ So 8 bits encodes $2^8$ or 256 things

$\begin{array}{cccccccc}
0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\
64 & 8 & 2 & & & & & =74
\end{array}$

There are only 10 types of people in the world: those who understand binary, and those who don’t.
(http://en.wikipedia.org/wiki/Mathematical_joke)
An encoding standard: ASCII

➤ With 256 possible characters, we can store:
  ➤ every single letter used in English,
  ➤ plus all the things like commas, periods, space bar, percent sign (%), back space, and so on.

➤ ASCII = the American Standard Code for Information Interchange
  ➤ 7-bit code for storing English text
  ➤ 7 bits = 128 possible characters.
  ➤ The numeric order reflects alphabetic ordering.
The ASCII chart

Codes 1–31 are used for control characters (backspace, return, tab, . . . ).

<table>
<thead>
<tr>
<th>Code</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>032</td>
<td>.</td>
</tr>
<tr>
<td>033</td>
<td>!</td>
</tr>
<tr>
<td>034</td>
<td>&quot;</td>
</tr>
<tr>
<td>035</td>
<td>#</td>
</tr>
<tr>
<td>036</td>
<td>$</td>
</tr>
<tr>
<td>037</td>
<td>%</td>
</tr>
<tr>
<td>038</td>
<td>&amp;</td>
</tr>
<tr>
<td>039</td>
<td>'</td>
</tr>
<tr>
<td>040</td>
<td>(</td>
</tr>
<tr>
<td>041</td>
<td>)</td>
</tr>
<tr>
<td>042</td>
<td>*</td>
</tr>
<tr>
<td>043</td>
<td>+</td>
</tr>
<tr>
<td>044</td>
<td>,</td>
</tr>
<tr>
<td>045</td>
<td>-</td>
</tr>
<tr>
<td>046</td>
<td>.</td>
</tr>
<tr>
<td>047</td>
<td>/</td>
</tr>
<tr>
<td>048</td>
<td>0</td>
</tr>
<tr>
<td>049</td>
<td>1</td>
</tr>
<tr>
<td>050</td>
<td>2</td>
</tr>
<tr>
<td>051</td>
<td>3</td>
</tr>
<tr>
<td>052</td>
<td>4</td>
</tr>
<tr>
<td>053</td>
<td>5</td>
</tr>
<tr>
<td>054</td>
<td>6</td>
</tr>
<tr>
<td>055</td>
<td>7</td>
</tr>
<tr>
<td>056</td>
<td>8</td>
</tr>
<tr>
<td>057</td>
<td>9</td>
</tr>
<tr>
<td>058</td>
<td>:</td>
</tr>
<tr>
<td>059</td>
<td>;</td>
</tr>
<tr>
<td>060</td>
<td>&lt;</td>
</tr>
<tr>
<td>061</td>
<td>=</td>
</tr>
<tr>
<td>062</td>
<td>&gt;</td>
</tr>
<tr>
<td>063</td>
<td>?</td>
</tr>
<tr>
<td>064</td>
<td>@</td>
</tr>
<tr>
<td>065</td>
<td>A</td>
</tr>
<tr>
<td>066</td>
<td>B</td>
</tr>
<tr>
<td>067</td>
<td>C</td>
</tr>
<tr>
<td>068</td>
<td>D</td>
</tr>
<tr>
<td>069</td>
<td>E</td>
</tr>
<tr>
<td>070</td>
<td>F</td>
</tr>
<tr>
<td>071</td>
<td>G</td>
</tr>
<tr>
<td>072</td>
<td>H</td>
</tr>
<tr>
<td>073</td>
<td>I</td>
</tr>
<tr>
<td>074</td>
<td>J</td>
</tr>
<tr>
<td>075</td>
<td>K</td>
</tr>
<tr>
<td>076</td>
<td>L</td>
</tr>
<tr>
<td>077</td>
<td>M</td>
</tr>
<tr>
<td>078</td>
<td>N</td>
</tr>
<tr>
<td>079</td>
<td>O</td>
</tr>
<tr>
<td>080</td>
<td>P</td>
</tr>
<tr>
<td>081</td>
<td>Q</td>
</tr>
<tr>
<td>082</td>
<td>R</td>
</tr>
<tr>
<td>083</td>
<td>S</td>
</tr>
<tr>
<td>084</td>
<td>T</td>
</tr>
<tr>
<td>085</td>
<td>U</td>
</tr>
<tr>
<td>086</td>
<td>V</td>
</tr>
<tr>
<td>087</td>
<td>W</td>
</tr>
<tr>
<td>088</td>
<td>X</td>
</tr>
<tr>
<td>089</td>
<td>Y</td>
</tr>
<tr>
<td>090</td>
<td>Z</td>
</tr>
<tr>
<td>091</td>
<td>[</td>
</tr>
<tr>
<td>092</td>
<td>\</td>
</tr>
<tr>
<td>093</td>
<td>]</td>
</tr>
<tr>
<td>094</td>
<td>^</td>
</tr>
<tr>
<td>095</td>
<td>_</td>
</tr>
<tr>
<td>096</td>
<td>`</td>
</tr>
<tr>
<td>097</td>
<td>a</td>
</tr>
<tr>
<td>098</td>
<td>b</td>
</tr>
<tr>
<td>099</td>
<td>c</td>
</tr>
<tr>
<td>100</td>
<td>d</td>
</tr>
<tr>
<td>101</td>
<td>e</td>
</tr>
<tr>
<td>102</td>
<td>f</td>
</tr>
<tr>
<td>103</td>
<td>g</td>
</tr>
<tr>
<td>104</td>
<td>h</td>
</tr>
<tr>
<td>105</td>
<td>i</td>
</tr>
<tr>
<td>106</td>
<td>j</td>
</tr>
<tr>
<td>107</td>
<td>k</td>
</tr>
<tr>
<td>108</td>
<td>l</td>
</tr>
<tr>
<td>109</td>
<td>m</td>
</tr>
<tr>
<td>110</td>
<td>n</td>
</tr>
<tr>
<td>111</td>
<td>o</td>
</tr>
<tr>
<td>112</td>
<td>p</td>
</tr>
<tr>
<td>113</td>
<td>q</td>
</tr>
<tr>
<td>114</td>
<td>r</td>
</tr>
<tr>
<td>115</td>
<td>s</td>
</tr>
<tr>
<td>116</td>
<td>t</td>
</tr>
<tr>
<td>117</td>
<td>u</td>
</tr>
<tr>
<td>118</td>
<td>v</td>
</tr>
<tr>
<td>119</td>
<td>w</td>
</tr>
<tr>
<td>120</td>
<td>x</td>
</tr>
<tr>
<td>121</td>
<td>y</td>
</tr>
<tr>
<td>122</td>
<td>z</td>
</tr>
<tr>
<td>123</td>
<td>{</td>
</tr>
<tr>
<td>124</td>
<td></td>
</tr>
<tr>
<td>125</td>
<td>}</td>
</tr>
<tr>
<td>126</td>
<td>~</td>
</tr>
</tbody>
</table>
What if 127 characters isn’t enough?

➢ Local Variants

[092] Japanese ASCII: Yen (¥) instead of backslash (\)
[035] UK ASCII: Pounds Sterling (£) instead of hash (#).

➢ Transliteration

➢ Multi-byte encodings
Transliteration

➢ Use ASCII, and fake the missing letters
  ➢ ue for ü, oe for ö, . . .

➢ Volapuk replaces Cyrillic letters with Latin ones in order to look the same as typed or handwritten Cyrillic letters.

  1. Replace ”the same” letters: a, e, K, M, T, o, y.
  2. Replace similar-looking letters: Γ with 2 (handwritten resemblance) or ρ, . . .
  3. Replace all other non-obvious hard-to-represent characters; there are many options for each letter: Φ with qp or 0. The choice for each letter depends on the preferences of the individual user.

➢ These transliterations are hard to read
Different coding systems

➢ Extended ASCII (use 256 characters)

➢ Other encodings
  ➢ ISO 8859-1: includes extra letters for French, German, Spanish, …
  ➢ ISO 8859-5: Cyrillic alphabet
  ➢ ISO 8859-7: Greek alphabet
  ➢ ISO 8859-8: Hebrew alphabet

➢ But you can only have one encoding at a time!
  ➢ You can’t have both Greek and Russian

➢ 256 characters is not enough for many languages
Multi-byte Encodings

➢ Use more bytes


➢ An ASCII character is represented by one byte, with the first bit 0.
➢ A character from JIS-X-0208 (code set 1) is represented by two bytes, both with the first bit 1.
  * This includes Hiragana, Katakana and most Chinese Characters.
➢ A character from JIS-X-0212 (code set 3) is represented by three bytes, the first being 0x8F, and the second two both with the first bit 1.
  * This includes many more Chinese characters.

This encoding scheme allows the easy mixing of 7-bit ASCII and 8-bit Japanese.
Example of EUC-JP

犬 は  d o g  だ 。  EOF
b8a4  a4cf  64  6f  67  a4c0  a1a3  0a

➢  Written in hexadecimal: 0123456789ABCDEF

➢  0 = 0000 = 0 (0 + 0 + 0 + 0)
➢  1 = 0001 = 1 (0 + 0 + 0 + 1)
➢  2 = 0010 = 2 (0 + 0 + 2 + 0)

...  ➢  A = 1010 = 10 (8 + 0 + 2 + 0)

...  ➢  E = 1110 = 14 (8 + 4 + 2 + 0)
➢  F = 1111 = 15 (8 + 4 + 2 + 1)

➢  Bit one = 1 ⇒ > 8
Problems with stateless encodings

➤ Shift-JIS is stateless

➤ Consider the following:

剣 道
8C95 93B9
白 血 病
9492 8C8C 9561

➤ 剣 8C95 matches across character boundaries
  * but you don’t want to match it here

➤ When you delete a character, you need to know how many bytes it is
Still more problems

➢ EUC-JP is stateful so it can’t fit all characters
   ➢ using one bit to show state, so only: $2^{14} = 16,384$

➢ You need to know what the encoding is:
   ➢ ”æ—‡å—œŒ—ä[]” (文字化け)

Much more in:
Unicode solves many of these problems

“Unicode provides a unique number for every character, no matter what the platform, no matter what the program, no matter what the language.” (www.unicode.org)
How big is Unicode?

➤ Version 3.2 has codes for 95,221 characters from alphabets, syllabaries and logographic systems.

➤ Uses 32 bits (4 bytes)
   Can represent \(2^{32} = 4,294,967,296\) characters.

➤ 4 billion possibilities for each character?

➤ That takes a lot of space on the computer!
   ➤ Four times as much as ASCII
Compact encoding of Unicode characters

- UTF-32 (32 bits): direct representation

- UTF-16 (16 bits): \(2^{16} = 65,536\) (subset!)

- UTF-8 (variable width encoding)

  
  \[
  \begin{array}{cccc}
  \text{U+0000-U+007F} & 0\text{x} & 0\text{x} & \text{ASCII} \\
  \text{U+0080-U+07FF} & 110\text{y} & \text{10} & \text{Alphabets/Syllabaries} \\
  \text{U+0800-U+FFFF} & 1110\text{y} & \text{10y} & \text{Logographs} \\
  \text{U+10000-U+10FFFF} & 11110\text{z} & \text{10z} & \text{Room to expand} \\
  \end{array}
  \]

- First byte says how many will follow

  Nice consequence: ASCII text is in a valid UTF-8 encoding.
How do we type everything in?

➢ Use a keyboard tailored to your specific language

➢ e.g. Highly noticeable how much slower your English typing is when using a Danish-designed keyboard.

➢ Use a processor that allows you to switch between different character systems.
  ➢ e.g. Type in Cyrillic characters on your English keyboard.

➢ Use combinations of characters.
  ➢ e.g. An e followed by an ’ might result in an é.

➢ Pick and choose from a table of characters.
Unwritten languages

Many languages have never been written down. Of the 6912 spoken languages, approximately 3000 have never been written down.

Some examples:

➢ Salar, a Turkic language in China.

➢ Gugu Badhun, a language in Australia.

➢ Southeastern Pomo, a language in California

On going work in adding alphabets, often by Bible translators!
You can remove a lot of information and still understand

For example, with no vowels, spaces or segmentation

FACTSRSTRNFTN

Obama, close GTMO now!

It is much easier if you know the meaning

Redundancy is important if there is noise

There is normally a lot of noise, so all natural languages are redundant
Another Example

Before the addition of the check-bits in Hamming’s code we - intuitively - dealing with pure information. Extra symbols did not change the amount of information that was being conveyed so we say that this was redundant.

➢ Redundancy is useful
Efficient Representation

➤ Language is also efficient in its coding

➤ Consider the most common 20 words of English:

  the, of, and, a, to, in, is, you, that, it, he, was, for, on, are, as, with, his, they, I

➤ They are all short!

➤ Frequent expressions are shortened
today not on this day

➤ This makes the overall text shorter
Playing with Writing: Acrostics

To the Members of the California State Assembly: I am returning Assembly Bill 1176 without my signature.

For some time now I have lamented the fact that major issues are overlooked while many unnecessary bills come to me for consideration. Water reform, prison reform, and health care are major issues my Administration has brought to the table, but the Legislature just kicks the can down the alley.

Yet another legislative year has come and gone without the major reforms Californians overwhelmingly deserve. In light of this, and after careful consideration, I believe it is unnecessary to sign this measure at this time.

Sincerely, Arnold Schwarzenegger

Playing with Writing: Acrostics

To the Members of the California State Assembly: I am returning Assembly Bill 1176 without my signature.

For some time now I have lamented the fact that major issues are overlooked while many unnecessary bills come to me for consideration. Water reform, prison reform, and health care are major issues my Administration has brought to the table, but the Legislature just kicks the can down the alley.

Yet another legislative year has come and gone without the major reforms Californians overwhelmingly deserve. In light of this, and after careful consideration, I believe it is unnecessary to sign this measure at this time.

Sincerely, Arnold Schwarzenegger

"My goodness. What a coincidence," a shocked, shocked Schwarzenegger spokesman Aaron McLear is quoted by the Associated Press as saying. "I suppose when you do so many vetoes, something like this is bound to happen."
Playing with Writing: Hanzi

目田 氏王 mùtián shīwáng “eye-field clan-king”
Playing with Writing: Graphemic Puns

➢ 目田 氏王 mùtián shīwáng “eye-field clan-king”

➢ 自由 民主 zìyóu mínzhǔ “freedom democracy”
  ➢ Just like freedom and democracy but missing a bit
  ➢ Not caught by censors (at first)


*Language Log* http://languagelog.ldc.upenn.edu/nll/?p=2614
Speech
Representing Speech

➢ The need for speech representation

➢ Storing sound

➢ Transcriptions
The need for speech

➢ We want to be able to encode any spoken language

➢ What if we want to work with an unwritten language?
➢ What if we want to examine the way someone talks and don’t have time to write it down?

➢ Many applications for encoding speech:

➢ Building spoken dialogue systems, i.e. speak with a computer (and have it speak back).
➢ Helping people sound like native speakers of a foreign language.
➢ Helping speech pathologists diagnose problems
What does speech look like?

We can transcribe (write down) the speech into a phonetic alphabet.

- It is very expensive and time-consuming to have humans do all the transcription.

- To automatically transcribe, we need to know how to relate the audio signal to the individual sounds that we hear.

- We need to know:
  - some properties of speech
  - how to measure these speech properties
  - how these measurements correspond to sounds we hear
What makes representing speech hard?

- Sounds run together, and it’s hard to tell where one sound ends and another begins.

- People say things differently from one another:
  - People have different dialects
  - People have different size vocal tracts

- Hand written text shares similar problems
People say things differently across time: What we think of as one sound is not always (usually) said the same

ccoatirculation = sounds affecting the way neighboring sounds are said
e.g. k is said differently depending on if it is followed by ee or by oo.

What we think of as two sounds are not always all that different.
e.g. The s in see is acoustically very similar to the sh in shoe
Articulatory properties: How it’s produced

➤ We could talk about how sounds are produced in the vocal tract, i.e. articulatory phonetics

➤ place of articulation (where): [t] vs. [k]
➤ manner of articulation (how): [t] vs. [s]
➤ voicing (vocal cord vibration): [t] vs. [d]

➤ But unless the computer is modeling a vocal tract, we need to know acoustic properties of speech which we can quantify.
Measuring sound

- Sound is actually a continuous wave

- We store data at each discrete point, in order to capture the general pattern of the sound

- **Sampling Rate**: how many times in a given second we extract a moment of sound; measured in samples per second. Sound is continuous, but we have to store data in a discrete manner.
Signal sampling representation.
Sampling rate

The higher the sampling rate, the better quality the recording ... but the more space it takes.

➢ Speech needs at least 8000 samples/second, but most likely 16,000 or 22,050 Hz will be used nowadays.

➢ The rate for CDs is 44,100 samples/second (or Hertz (Hz))

Now, we can talk about what we need to measure
Acoustic properties: What it sounds like

➤ Sound waves = “small variations in air pressure that occur very rapidly one after another”

➤ The main properties we measure:

➤ speech flow: rate of speaking, number and length of pauses (seconds)
➤ amplitude (loudness): amount of energy (decibels)
➤ frequency = how fast the sound waves are repeating (cycles per second, i.e. Hertz)
  * pitch = how high or low a sound is
  * In speech, there is a fundamental frequency, or pitch, along with higher-frequency overtones.

Researchers also look at things like intonation, i.e., the rise and fall in pitch.
Speech Sample

Pitch track, transcription, spectogram and audio waveform.
Measurement-sound correspondence

➢ How dark is the picture? → How loud is the sound?
   ➢ We measure this in decibels.

➢ Where are the lines the darkest? → Which frequencies are the loudest and most important?
   ➢ We can measure this in terms of Hertz, and it tells us what the vowels are.

➢ Speech signals are very different from text.
   ➢ No words!
Applications of speech encoding

- Mapping sounds to symbols (alphabet), and vice versa, has some very practical uses.
  - **Automatic Speech Recognition (ASR):** sounds to text
  - **Text-to-Speech Synthesis (TTS):** texts to sounds
- These are not easy tasks.
- Text-to-Speech Synthesis is somewhat easier.
Automatic Speech Recognition (ASR)

- Automatic speech recognition = process by which the computer maps a speech signal to text.

- Uses/Applications:
  - Dictation
  - Dialogue systems
  - Telephone conversations
  - People with disabilities – e.g. a person hard of hearing could use an ASR system to get the text (closed captioning)
  - Spying (many agencies run ASR on phone conversations and search for keywords)
  - Indexing audio data
Steps in an ASR system

1. Digital sampling of speech

2. **Acoustic signal processing** = converting the speech samples into particular measurable units

3. Recognition of sounds, groups of sounds, and words

   May or may not use more sophisticated analysis of the utterance to help. e.g., a [t] might sound like a [d], and so word information might be needed (more on this later)
Kinds of ASR systems

Different kinds of systems, with an accuracy-robustness tradeoff:

➢ **Speaker dependent**: works for a single speaker

➢ **Speaker independent**: work for any speaker of a given variety of a language, e.g. American English

➢ A common type of system starts general, but learns:

   ➢ **Speaker adaptive** = start as independent but begin to adapt to a single speaker to improve accuracy
   ➢ Adaptation may simply be identifying what type of speaker a person is and then using a model for that type of speaker
Kinds of ASR systems

➢ Differing sizes and types of vocabularies

➢ from tens of words to tens of thousands of words
➢ normally very domain-specific, e.g., flight vocabulary

➢ continuous speech vs. isolated-word systems:

➢ **continuous speech systems** = words connected together and not separated by pauses
➢ **isolated-word systems** = single words recognized at a time, requiring pauses to be inserted between words
  * easier to find the endpoints of words
  * harder to use
How good are the systems?

- Dictation: 1-2% Word Error Rate
  - speakers who match the training data
  - after system adaptation
  - a clean noise environment (e.g. quiet office or laboratory space).

- Noisy room, multiple speakers 50+% WER
The first successful wide spread testing in NLP

- Compare your output to a reference
- Calculate the number of substitutions, deletions and insertions to make them match (Minimum Edit Distance)
- Normalize by dividing by the length of the reference

\[ WER = \frac{S+D+I}{N} \]

Reference: I want to recognize speech today
System: I want wreck a nice peach today
Eval: D S I I S

\[ WER = \frac{2+1+2}{6} = 0.83 \]
Some properties of WER

- Correlates well with the task
- Reducing WER is always a good thing
- A WER of 0 implies perfect results (assuming the reference is correct)
- $WER > .95$ considered the minimum to be useful
- Competitions were held to see who could get the lowest WER
  - Speech Recognition had 10 years of rapid improvement
  - It has slowed down now
How good are the systems?

<table>
<thead>
<tr>
<th>Task</th>
<th>Vocab</th>
<th>WER (%)</th>
<th>WER (%) adapted</th>
</tr>
</thead>
<tbody>
<tr>
<td>Digits</td>
<td>11</td>
<td>0.4</td>
<td>0.2</td>
</tr>
<tr>
<td>Dialogue (travel)</td>
<td>21,000</td>
<td>10.9</td>
<td>—</td>
</tr>
<tr>
<td>Dictation (WSJ)</td>
<td>5,000</td>
<td>3.9</td>
<td>3.0</td>
</tr>
<tr>
<td>Dictation (WSJ)</td>
<td>20,000</td>
<td>10.0</td>
<td>8.6</td>
</tr>
<tr>
<td>Dialogue (noisy, army)</td>
<td>3,000</td>
<td>42.2</td>
<td>31.0</td>
</tr>
<tr>
<td>Phone Conversations</td>
<td>4,000</td>
<td>41.9</td>
<td>31.0</td>
</tr>
</tbody>
</table>

Results of various DARPA competitions (from Richard Sproat’s slides)
Why is it so difficult?

➢ Speaker variability
  ➢ Gender
  ➢ Dialect/Foreign Accent
  ➢ Individual Differences: Physical differences; Language differences (idiolect)

➢ Many, many rare events
  ➢ 300 out of 2000 diphones in the core set for the AT&T NextGen system occur only once in a 2-hour speech database
Rare events are frequent

➢ Collect about 10,000,000 character 4-grams, from English newswire text, merging upper and lower case — 60 distinct characters including space.

➢ 197,214 lines of text.

➢ Of these, 14,317 (7%) contain at least one 4-gram that only occurs once in 10,000,000.

➢ Increase it to 5-grams: 21% of lines contain at least one 5-gram that only occurs once in 10,000,000.
What is an $n$-gram?

- An $n$-gram is a chunk of $n$ things: most often words, but could be characters, letters, morphemes, stems, . . .

- Approximation of language: information in $n$-grams tells us something about language, but doesn’t capture the structure.

- Efficient: finding and using every, e.g., two-word collocation in a text is quick and easy to do.

- $n$-grams help a variety of NLP applications, including word prediction.
  - We can predict the next word of an utterance, based on the previous...

- unigram, bigram, trigram, 4-gram, . . .
Text-to-Speech Synthesis (TTS)

- Could just record a voice saying phrases or words and then play back those words in the appropriate order.

- This won’t work for, e.g., dialogue systems where speech is generated on the fly.

- Or can break the text down into smaller units
  1. Convert input text into phonetic alphabet (ambiguous mapping)
  2. Synthesize phonetic characters into speech

- To synthesize characters into speech, people have tried:
  - using a model based on frequencies, the loudness, etc.
  - using a model of the vocal tract and human speech production
In some sense, TTS really is the reverse process of ASR.

Since we know what frequencies correspond to which vowels, we can play those frequencies to make it sound like the right vowel.

However, sounds are always different (across time, across speakers).

One way to generate speech is to have a database of speech and to use the diphones, i.e., two-sound segments, to generate sounds.

Diphones help with the context-dependence of sounds.
It’s hard to be natural

When trying to make synthesized speech sound natural, we encounter the same problems as what makes speech encoding hard:

- The same sound is said differently in different contexts.
- Different sounds are sometimes said nearly the same.
- Different sentences have different intonation patterns.
- Lengths of words vary depending on where in the sentence they are spoken.

1. The car crashed into the tree.
2. It’s my car.
3. Cars, trucks, and bikes are vehicles.
Speech to Text to Speech

If we convert speech to text and then back to speech, it should sound the same.

➢ But at the conversion stages, there is information loss.

➢ To avoid this loss would require a lot of memory and knowledge about what exact information to store.

➢ The process is thus irreversible.

➢ In fact, people can’t say the same sentence exactly the same way either!
Relating Speech to Text

➢ Speech can have
  ➢ loudness
  ➢ intonation

➢ Text can have
  ➢ Different fonts, styles and size
  ➢ Colour
  ➢ Position
  ➢ Explicit markup

➢ Both combine with other modes
  ➢ gestures
  ➢ pictures
Speech vs Writing (1): Time-bound

➢ Speech
  ➢ time-bound
  ➢ dynamic, transient
  ➢ normally direct between a speaker and a known addressee

➢ Writing
  ➢ space-bound
  ➢ static, permanent
  ➢ normally indirect with the addressee unknown

Speech vs Writing (2): Spontaneous

➤ Speech

➢ no lag between production and reception
➢ hard to plan complex constructions
⇒ repetitions, rephrasing, comment clauses
➢ Sentence boundaries often unclear

➤ Writing

➢ lag between production and reception
➢ readers can reread and analyse in depth
⇒ careful organization and compact expressions
➢ Sentence (and paragraph, . . . ) boundaries are clear (English!)
Speech vs Writing (3): Face-to-Face

➤ **Speech**

➢ Extralinguistic cues are common (facial expressions, gestures)
➢ Immediate feedback (back channel)
➢ Deictic expressions are common (referring to the situation)
  *that one, you, now, over there*

➤ **Writing**

➢ Different extralinguistic possibilities (fonts, color, pictures)
➢ No immediate feedback
➢ Fewer deictic expressions
Speech vs Writing (4): Loosely Structured

➤ **Speech**

➤ Contractions are common: *isn’t, he’s*
➤ Long coordinate sentences are common
➤ Informal vocabulary: *thingamajig, whatsit*
➤ Obscenity more common

➤ **Writing**

➤ Subordination more common (relative clauses)
➤ Longer sentences (can be multipage)
➤ Some items rarely pronounced: 
\[
H(p) = - \sum_{x \in X} p(x) \log_2 p(x)
\]
Speech vs Writing (5): Socially Interactive

➢ Speech

➢ Well suited to social functions
  * greetings
  * maintaining social relationships
  * expression attitudes and opinions
  Much use of prosody and non-verbal features

➢ Writing

➢ Suited to recording facts and communicating ideas
➢ Easier to scan
➢ Tables demonstrate relations between things
➢ Text can be read at one’s own pace
Speech vs Writing (6): Immediately Revisable

➢ Speech

➢ You can rephrase at once, based on feedback
➢ Errors once spoken can’t be withdrawn
➢ Interruptions and overlap is common

➢ Writing

➢ You can remove errors without the speaker ever seeing them
➢ Once published errors can only be withdrawn through revisions
➢ Interruptions are not visible
Speech vs Writing (7): Prosodically Rich

➢ Speech
  ➢ Prosody
    intonation; loudness; tempo; rhythm, pause tone of voice

➢ Writing
  ➢ Pages, lines, capitalization, spatial organization
  ➢ Punctuation (?!.)
  ➢ Tables, graphs, formulae
### Comparison of speed for different modalities

<table>
<thead>
<tr>
<th>Activity</th>
<th>Speed (wpm)</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reading</td>
<td>300</td>
<td>200 (proof reading)</td>
</tr>
<tr>
<td>Writing</td>
<td>31</td>
<td>21 (composing)</td>
</tr>
<tr>
<td>Speaking</td>
<td>150</td>
<td></td>
</tr>
<tr>
<td>Hearing</td>
<td>150</td>
<td>210 (speeded up)</td>
</tr>
<tr>
<td>Typing</td>
<td>33</td>
<td>19 (composing)</td>
</tr>
</tbody>
</table>

(English, computer science students, various studies)

Speed in words per minute (one word is 6 characters)

- ➢ Reading >> Speaking/Hearing >> Typing
  - ⇒ Speech for input
  - ⇒ Text for output
There are many ways to represent text
Some are easier to encode than others
Efficient representation is not always the goal
Speech signals are very different from text
It is non-trivial to convert between them
Acknowledgments

➤ Many slides inspired by Marcus Dickinson

➤ Much of the information on writing systems is taken from the great site http://www.omniglot.com

➤ Pictures from wikipedia and the open clipart library