Formal Models of Language

Paula Buttery

Dept of Computer Science & Technology, University of Cambridge

For communication, information has to be **transmitted**

Goal: To optimise, in terms of **throughput** and **accuracy**, the communication of messages in the presence of a **noisy channel**

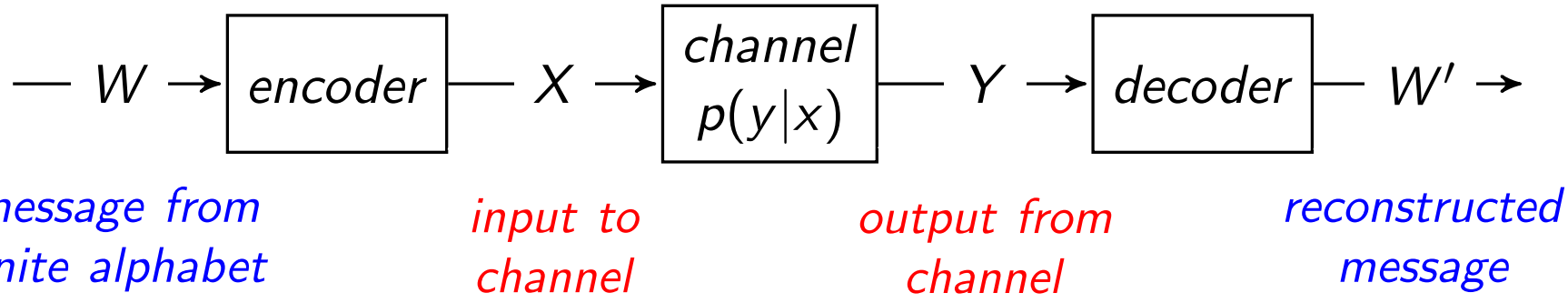
There is a trade off between:

- **compression**: making the most efficient code by removing all the redundancy
- **accuracy**: adding redundant information so that the input can still be recovered despite the presence of noise

Today we will:

- formalise the noisy channel more carefully
- look at some implications for natural language evolution
- see how the noisy channel model has inspired a framework for solving problems in Natural Language Processing.

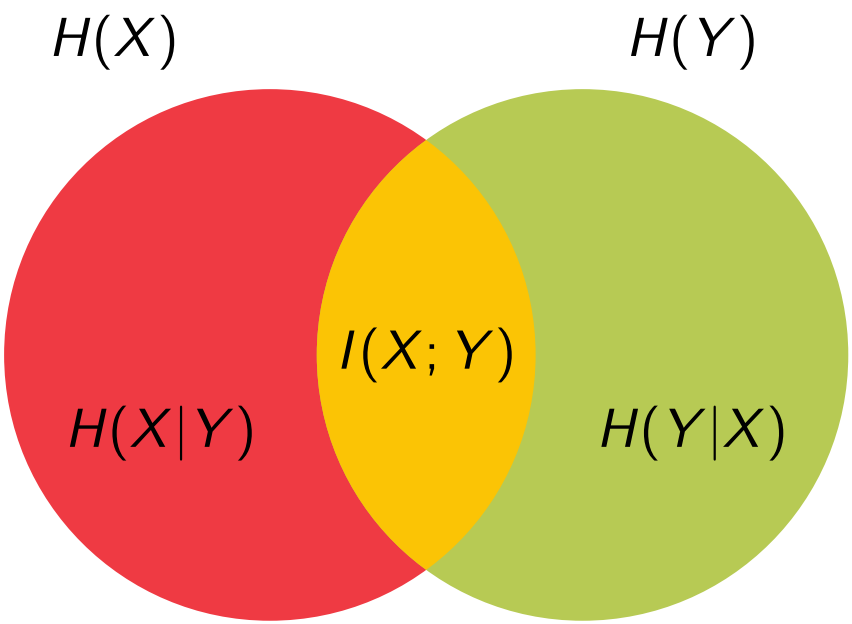
Transmission can be modelled using a **noisy channel**



- message should be efficiently encoded but with enough redundancy for the decoder to detect and correct errors
- the output depends probabilistically on the input
- the decoder finds the mostly likely original message given the output received

Mutual information: the information Y contains about X

- **Mutual Information** $I(X;Y)$ is a measure of the reduction in uncertainty of one random variable due to knowing about another
- Can also think of $I(X;Y)$ being the amount of information one random variable contains about another



$H(X)$ average information of input

$H(Y)$ average information in output

$H(X|Y)$ the uncertainty in (extra information needed for) X given Y is known

$I(X;Y)$ the mutual information; the information in Y that tells us about X

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) \\ &= H(Y) - H(Y|X) \end{aligned}$$

Channel **capacity** is determined by **mutual information**

- The capacity of a channel is the maximum of the mutual information of X and Y over all input distributions of the input $p(X)$

$$C = \max_{p(X)} I(X;Y)$$

- C is the rate we can transmit information through a channel with an arbitrarily low probability of not being able to recover the input from the output
- As long as transmission rate is less than C we don't need to worry about errors (optimal rate is C)
- If transmission rate exceeds C then we need to slow down (e.g. by inserting a *that*—last lecture)
- In practical applications we reach the channel capacity by designing an encoding for the input that maximises mutual information.

What might this mean for the evolution of natural languages?

Piantadosi et al.—ambiguity has a communicative benefit

- If we are trying to maximise mutual information why has natural language evolved to be so ambiguous?

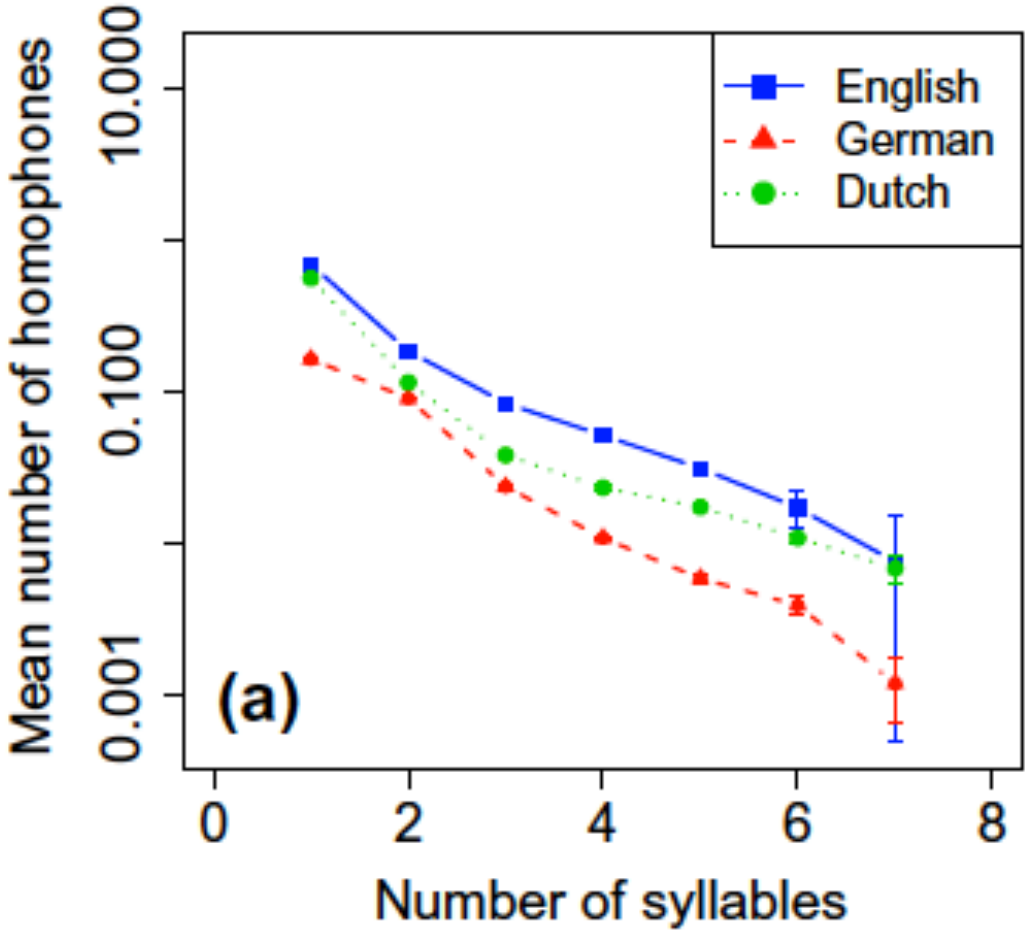
- Efficient communication systems will necessarily be globally ambiguous when context is informative about meaning.

- Notice that **ambiguity is not an issue in normal language** use and overloaded linguistic units are only ambiguous out of context:
  - Alice wanted **to** cry
  - Alice went **to** the garden
  - Alice saw **two** rabbits
  - Dinah saw some rabbits **too**.

- It is optimal to overload simple units for efficient transmission (we can assign the short efficient codes more than once and re-use them)

Piantadosi et al.—ambiguity has a communicative benefit

Some evidence to support the argument found in corpora: shorter words have more meanings



Implication: there must be enough information in the context to allow for the ambiguity in the simple units as well as any other noise in the channel.

Gibson et al.—a noisy channel can account for word order

Word order can provide context that is informative about meaning—this might account for observed word order in the world's languages

Most languages (out of 1,056 studied) exhibit one of two word orders:

- **subject-verb-object** (SVO) — 41% of languages

  the girl chases the rabbit (e.g. English)
- **subject-object-verb** (SOV) — 47% of languages

  the girl the rabbit chases (e.g. Japanese)

- For interest, 8% exhibit **verb-subject-object** (VSO) e.g. Welsh and Irish and 96% of languages have the subject before the object

Gibson et al.—noisy channel account of word order

- Experimental observations:
  - English speakers (SVO) were shown animations of simple events and asked to describe them using only gestures
  - For events in which a human acts on an inanimate object most participants use SOV despite being SVO speakers (e.g. *girl boy kicks*)
  - For events in which a human acts on another human most participants use SVO (e.g. *girl kicks boy*)
  - Preference in each case is around 70%
- Previous experiments show human preference for linguistic recapitulation of old information before introducing new information
- This might explain SOV gestures for SVO speakers—the verb is new information the people/objects are not.
- So why still use SVO for the animate-animate events? And why is English SVO?

Gibson et al.—noisy channel account of word order

Argument is that SVO ordering has a better chance of preserving information over a noisy channel.

- Consider the scenario of a girl kicking a boy
- Now let one of the nouns get lost in transmission.
- If the word order is SOV (*the girl the boy kicks*), the listener receives either:
  the girl kicks or *the boy kicks*
- If the word order is SVO (*the girl kicks the boy*) the listener receives either:
  the girl kicks or *kicks the boy*
- In the SVO case more information has made it through the noisy channel (preserved in the position relative to the verb)

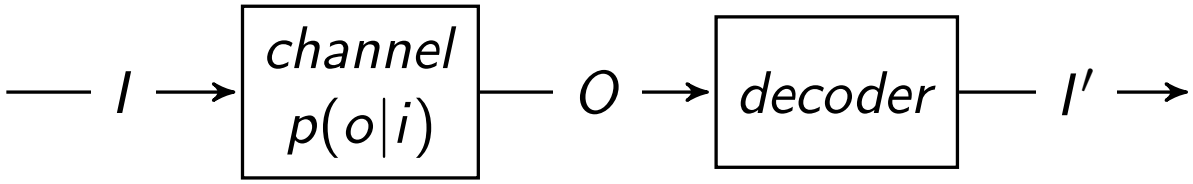
Gibson et al.—noisy channel account of word order

Further evidence for the argument is presented from the finding that there is a correlation between word order and case markings.

- Case marking means that words change depending on their syntactic function: e.g. *she* (subject), *her* (object)
- Case marking is rare in SVO languages (like English) and more common in SOV languages
- Suggestion is that when there are other information cues as to which noun is subject and which is object speakers can default to any natural preference for word order.

In Natural Language Processing, however, our starting point is **after** the evolutionary natural language encoding.

Noisy channel inspired an NLP problem-solving **framework**



- Many problems in NLP can be framed as trying to find the most likely input given an output:
$$I' = \underset{i}{\text{argmax}}\, p(i|o)$$
- $p(i|o)$ is often difficult to estimate directly and reliably, so use Bayes' theorem:
$$p(i|o) = \frac{p(o|i)p(i)}{p(o)}$$
- Noting that $p(o)$ will have no effect on argmax function:
$$I' = \underset{i}{\text{argmax}}\, p(i|o) = \underset{i}{\text{argmax}}\, p(i)p(o|i)$$
- $p(i)$ is the probability of the input (a **language model**)
- $p(o|i)$ is the **channel probability** (the probability of getting an output from the channel given the input)

SMT is an intuitive (non-SOTA) example of noisy channel

We want to translate a text from English to French:

- In statistical machine translation (SMT) noisy channel model assumes that original text is in French
- We pretend the original text has been through a noisy channel and come out as English (the word *hello* in the text is actually *bonjour* corrupted by the channel)
- To recover the French we need to decode the English:

$$f' = \underset{f}{\operatorname{argmax}}\, p(f|e) = \underset{f}{\operatorname{argmax}}\, p(f)p(e|f)$$

SMT is an intuitive (now historic) example of noisy channel

Recover the French by decoding the English: $f' = \underset{f}{\operatorname{argmax}}\, p(f)p(e|f)$

- $p(f)$ is the **language model**.
  - ensures **fluency** of the translation (usually a very large n-gram model)
- $p(e|f)$ is the **translation model**.
  - ensures **fidelity** of the translation (derived from very large parallel corpora)

Noisy channel framework influenced many applications

Speech Recognition: recover word sequence by decoding the speech signal

$$words' = \underset{words}{\text{argmax}}\, p(words)p(speech\_signal|words)$$

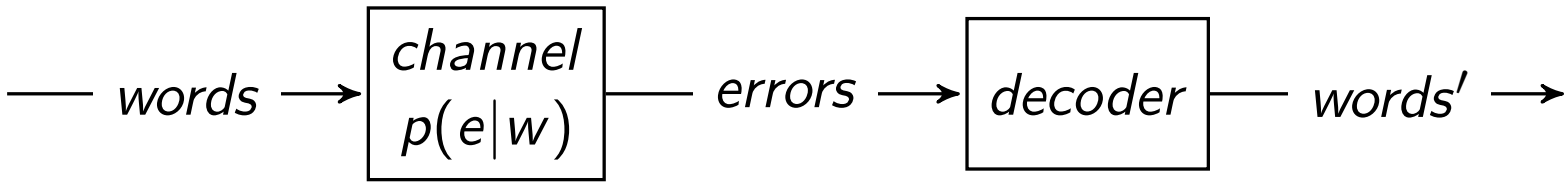
- $p(words)$ is the **language model** (n-gram model)
- $p(speech\_signal|words)$ is the **acoustic model**.

Noisy channel framework influenced many applications

Part-of-Speech Tagging:

$$tags' = \underset{tags}{\operatorname{argmax}}\, p(tags)p(words|tags)$$

Optical Character Recognition:



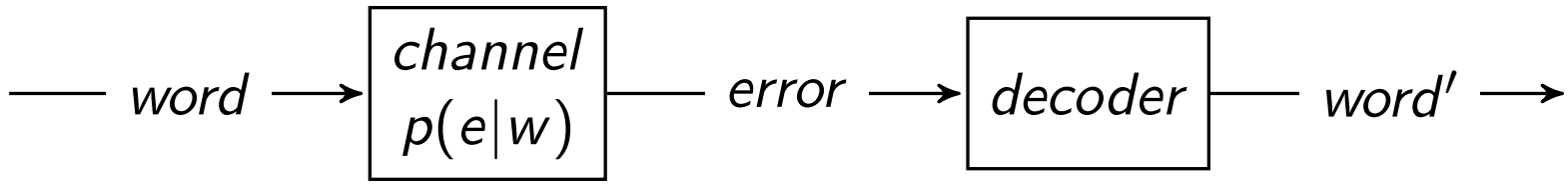
$$words' = \underset{words}{\operatorname{argmax}}\, p(words)p(errors|words)$$

Spelling can be corrected using the noisy channel method

- There are two types of spelling error:
  - **non-word** errors: *Alcie* instead of *Alice*
  - **real-word** errors: *three* instead of *there*, *there* instead of *their*
- For illustration we will show how to use a noisy channel model to correct non-word errors
- Non-word correction is a significant problem:
  - 26%: of web queries *Wang et al.*
  - 13%: errors when asked to retype rather than backspace *Whitelaw et al.*
- **Detection** of non-word errors is easy (they fail to appear in a dictionary)
- The best candidate for **correction** will be the item that maximises the noisy channel equation.

Spelling can be corrected using the noisy channel method

Spelling correction:



$$word' = \underset{word}{\operatorname{argmax}}\, p(word)p(error|word)$$

- $p(word)$ can be obtained from a corpus
- $p(error|word)$ can be modelled using minimum text edit distance or minimum pronunciation distance (the probability of the edit)

Spelling can be corrected using the noisy channel method

- Damerau-Levenshtein is edit distance model that counts: *insertions*. *deletions*, *substitutions*, *transpositions*

error	candidate correction	corrected letters	error letters	type
acress	actress	t		deletion
acress	cress	-	a	insertion
acress	caress	ca	ac	transposition
acress	access	c	r	substitution
acress	across	o	e	substitution
acress	acres	-	s	insertion
acress	acres	-	s	insertion

- 80% of errors are within edit distance 1
- Almost all errors within edit distance 2

Spelling can be corrected using the noisy channel method

- $p(error|word)$ may be calculated from confusion tables created from error annotated training data
- e.g. substitution(x,w) confusion matrix

	a	b	c	d	e	...
a	0	0	7	1	342	...
b	0	0	9	9	2	...
c	6	5	0	16	0	...
d	1	10	13	0	12	...
e	338	0	3	11	0	...

- if misspelled word is $x = x_1, x_2 ... x_m$
- and corrected word is $w = w_1, w_2 ... w_n$
- If proposed edit at x_i is a substitution $p(x|w) = \frac{substitution(x_i, w_i)}{count(w_i)}$
- similar equations for a *deletion*, *insertion* and *transposition*

Spelling can be corrected using the noisy channel method

- For typo *acress* chosen word is

$$= \underset{word}{\operatorname{argmax}}\, p(word|error) = \underset{word}{\operatorname{argmax}}\, p(word)p(error|word)$$

candidate	corr	err	$x\|w$	$p(x\|w)$	p(w)	$p(x\|w)p(w)10^9$
actress	t	-	c|ct	.000117	.0000231	2.7
cress	-	a	a|#	.00000144	.000000544	.00078
caress	ca	ac	ac|ca	.00000164	.00000170	.0028
access	c	r	r|c	.000000209	.0000916	.019
across	o	e	e|o	.0000093	.000299	2.8
acres	-	s	es|e	.0000321	.0000318	1.0
acres	-	s	ss|s	.0000342	.0000318	1.0

Noisy channel could be used to model comprehension

- For many natural language applications, noisy channel models have been surpassed by data hungry sequence-to-sequence neural models (more in *NLP* course next year)
- Natural language communication is an area where the model might still yield research insights
- Classic assumption in sentence processing: input to the parser is an error-free sequence of words (e.g. Hale and Yngve)
- This assumption is problematic because we are communicating across a noisy channel
- The ultimate interpretation of a sentence should depend on the proximity of plausible alternatives under the noise model
- This could be modelled in terms of insertions and deletions (just like spelling correction)...