Spelling Correction and the Noisy Channel.

- Spelling correction :-
- **Non-Word Spelling** correction is the detection and correction of spelling errors that result in non-words (like graffe for giraffe).

- **Real Word spelling** correction :- Detecting and correcting spelling errors even if they accidentally result in an actual real word of English (real-word errors).

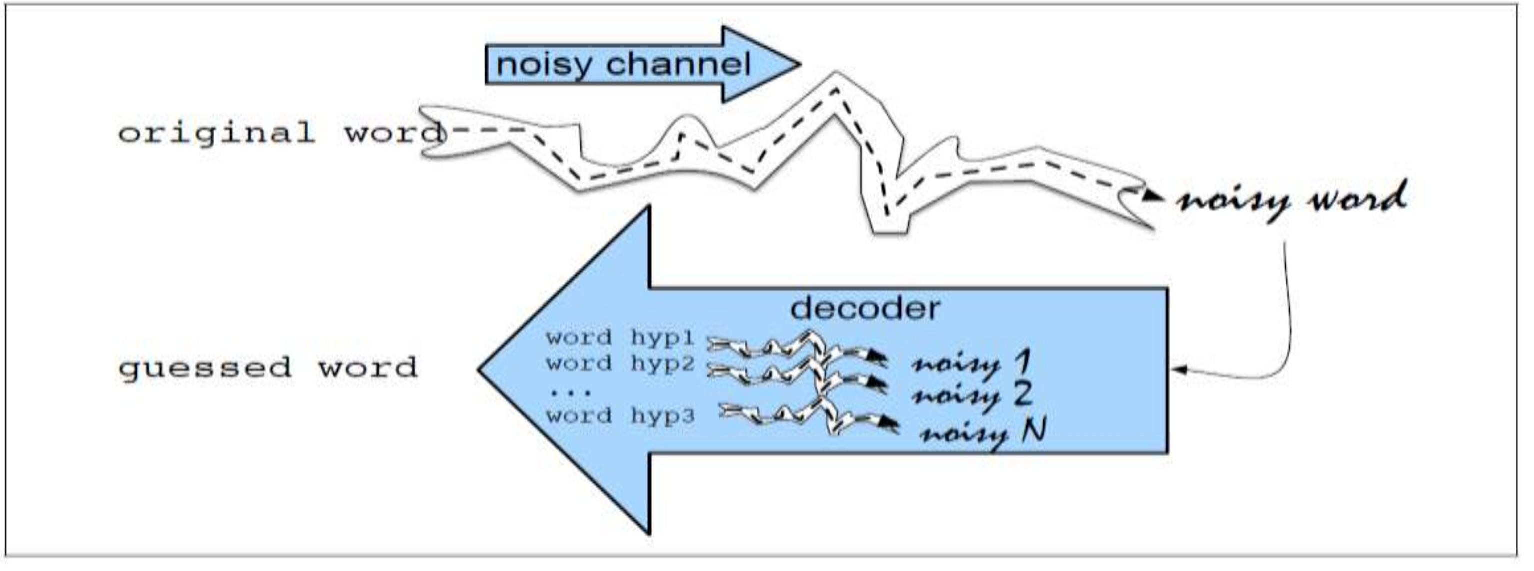
- **Typographical Errors :-** Errors in (insertion, deletion, transposition) that accidentally produce a real word (e.g., there for three).

- **Cognitive Errors** where the writer substituted the wrong spelling of a homophone or near-homophone (e.g., dessert for desert, or piece for peace).

- **Non-word Errors** are detected by looking for any word not found in a dictionary. For example, the misspelling graffe above would not occur in a dictionary.

Intution of Noisy Channel

- Is to treat the misspelled word as if a correctly spelled word had been "distorted" by being passed through a noisy communication channel.
- Channel introduces "noise" in the form of substitutions or other changes to the letters, making it hard to recognize the "true" word.



Modeling

$$\hat{w} = \underset{w \in V}{\operatorname{argmax}} P(w|x)$$

This noisy channel model is a kind of Bayesian inference.

An observation ***x*** (a misspelled word) and our job is to find the word ***w*** that generated this misspelled word.

Out of all possible words in the vocabulary **V** we want to find the word **w** such that **P(w|x)** is highest.

$$\hat{w} = \underset{w \in V}{\operatorname{argmax}} \frac{P(x|w)P(w)}{P(x)}$$

$$\hat{w} = \underset{w \in V}{\operatorname{argmax}} P(x|w)\,P(w)$$

- Apply the noisy channel approach to correcting non-word spelling errors by taking any word not in our spell dictionary, generating a list of **candidate words**, ranking them according to **Noisy Channel Model** equation, and picking the highest-ranked one.

$$\hat{w} = \underset{w \in C}{\operatorname{argmax}} \quad \overbrace{P(x|w)}^{\text{channel model}} \quad \overbrace{P(w)}^{\text{prior}}$$

		Transformation			
		Correct	Error	Position	
Error	**Correction**	**Letter**	**Letter**	**(Letter #)**	**Type**
acress	actress	t	—	2	deletion
acress	cress	—	a	0	insertion
acress	caress	ca	ac	0	transposition
acress	access	c	r	2	substitution
acress	across	o	e	3	substitution
acress	acres	—	s	5	insertion
acress	acres	—	s	4	insertion

- The first stage of the algorithm proposes **candidate corrections by finding words that have a similar spelling to the input word**.

- Analysis of spelling error data
  - The majority of spelling errors consist of a single-letter change.
  - The simplifying assumption that these candidates have an edit distance of 1 from the error word.

- The **prior probability of each correction P(w)** is the language model probability of the word (w) in context, which can be computed using any language model.

w	**count(w)**	**p(w)**
actress	9,321	.0000231
cress	220	.000000544
caress	686	.00000170
access	37,038	.0000916
across	120,844	.000299
acres	12,874	.0000318

- we estimate the likelihood $P(x|w)$, also called **the channel model.**

additional: addional, additonal

environments: enviornments, enviorments, enviroments

preceded: preceeded

- **del[x;y]: count(xy typed as x)**
- **ins[x;y]: count(x typed as xy)**
- **sub[x;y]: count(x typed as y)**
- **trans[x;y]: count(xy typed as yx)**

$$P(x|w) = \begin{cases} \dfrac{\text{del}[x_{i-1}, w_i]}{\text{count}[x_{i-1} w_i]}, \text{ if deletion} \\ \dfrac{\text{ins}[x_{i-1}, w_i]}{\text{count}[w_{i-1}]}, \text{ if insertion} \\ \dfrac{\text{sub}[x_i, w_i]}{\text{count}[w_i]}, \text{ if substitution} \\ \dfrac{\text{trans}[w_i, w_{i+1}]}{\text{count}[w_i w_{i+1}]}, \text{ if transposition} \end{cases}$$

Candidate Correction	Correct Letter	Error Letter	x|w	P(x|w)
actress	t	-	c|ct	.000117
cress	-	a	a|#	.00000144
caress	ca	ac	ac|ca	.00000164
access	c	r	r|c	.000000209
across	o	e	e|o	.0000093
acres	-	s	es|e	.0000321
acres	-	s	ss|s	.0000342

Candidate Correction	Correct Letter	Error Letter	x|w	P(x|w)	P(w)	10^9*P(x|w)P(w)
actress	t	-	c|ct	.000117	.0000231	2.7
cress	-	a	a|#	.00000144	.000000544	0.00078
caress	ca	ac	ac|ca	.00000164	.00000170	0.0028
access	c	r	r|c	.000000209	.0000916	0.019
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

"stellar and versatile **acress** whose combination of sass and glamour has defined her. . . ".

- P(actress | versatile) = 0.000021
- P(across | versatile) = 0.000021
- P(whose | actress) = 0.0010
- P(whose | across) = 0.000006

- It is important to use larger language models than unigrams.

$$\begin{aligned} \mathrm{P}(\text{“versatile actress whose”}) &= .000021 * .0010 = 210 \times 10^{-10} \\ \mathrm{P}(\text{“versatile across whose”}) &= .000021 * .000006 = 1 \times 10^{-10} \end{aligned}$$

Classification Problem.

ACTUAL VALUES		POSITIVE	NEGATIVE
	POSITIVE	**TP**	**FN**
	NEGATIVE	**FP**	**TN**

$$Precision = \frac{TP}{TP + FP} \qquad Recall = \frac{TP}{TP + FN}$$

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

		Predicted		
		Greyhound	Mastiff	Samoyed
Actual	Greyhound	P[illegible]	P[illegible]	P[illegible]
	Mastiff	P[illegible]	P[illegible]	P[illegible]
	Samoyed	P[illegible]	P[illegible]	P[illegible]