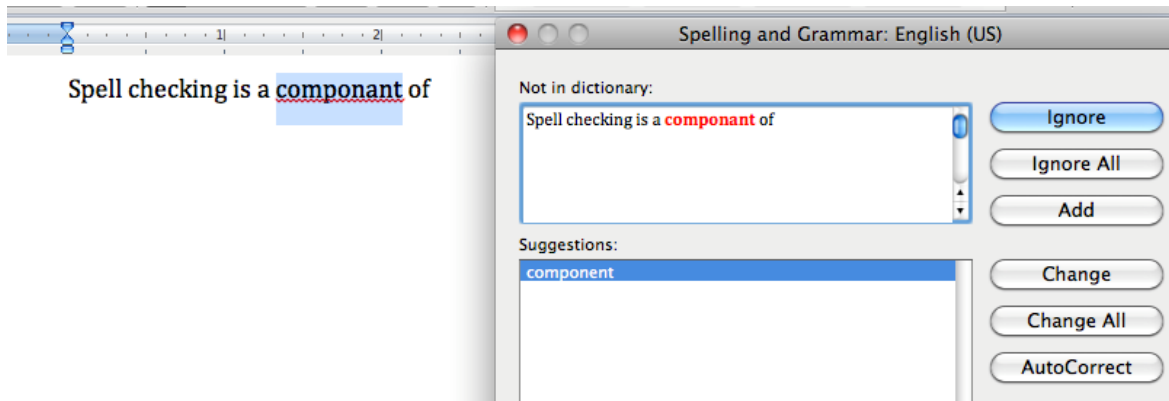






# Applications for spelling correction

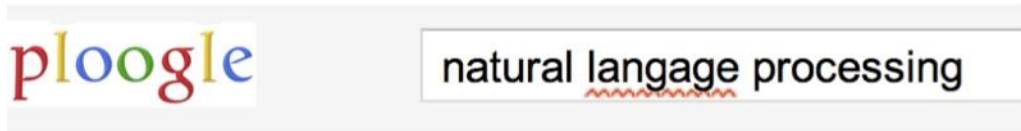
## Word processing



## Phones



## Web search



Showing results for [natural language processing](#)  
Search instead for [natural language processing](#)



# Spelling Tasks

- Spelling Error Detection
- Spelling Error Correction:
  - Autocorrect
    - hte → the
  - Suggest a correction
  - Suggestion lists



# Types of spelling errors

- Non-word Errors
  - *graffe* → *giraffe*
- Real-word Errors
  - Typographical errors
    - *three* → *there*
  - Cognitive Errors (homophones)
    - *piece* → *peace*,
    - *too* → *two*



# Rates of spelling errors

**26%:** Web queries [Wang et al. 2003](#)

**13%:** Retyping, no backspace: [Whitelaw et al. English&German](#)

**7%:** Words corrected retyping on phone-sized organizer

**2%:** Words uncorrected on organizer [Soukoreff & MacKenzie 2003](#)

**1-2%:** Retyping: [Kane and Wobbrock 2007](#), [Gruden et al. 1983](#)



# Non-word spelling errors

- Non-word spelling error detection:
  - Any word not in a **dictionary** is an error
  - The larger the dictionary the better
- Non-word spelling error correction:
  - Generate **candidates**: real words that are similar to error
  - Choose the one which is best:
    - Shortest weighted edit distance
    - Highest noisy channel probability



# Real word spelling errors

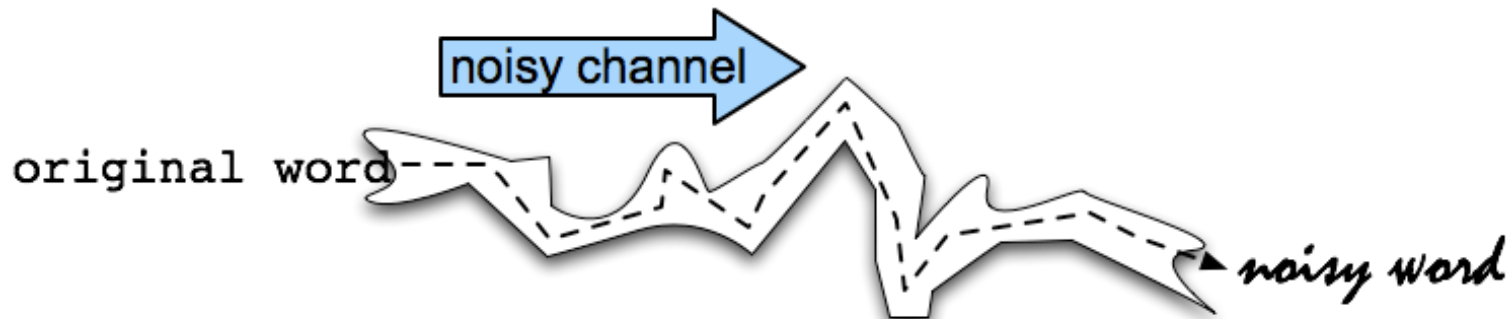
- For each word  $w$ , generate candidate set:
  - Find candidate words with similar *pronunciations*
  - Find candidate words with similar *spelling*
  - Include  $w$  in candidate set
- Choose best candidate
  - Noisy Channel
  - Classifier







# Noisy Channel Intuition





# Noisy Channel

- We see an observation  $x$  of a misspelled word
- Find the correct word  $w$

$$\begin{aligned}\hat{w} &= \operatorname{argmax}_{w \in V} P(w | x) \\ &= \operatorname{argmax}_{w \in V} \frac{P(x | w)P(w)}{P(x)} \\ &= \operatorname{argmax}_{w \in V} P(x | w)P(w)\end{aligned}$$



# History: Noisy channel for spelling proposed around 1990

- **IBM**

- Mays, Eric, Fred J. Damerau and Robert L. Mercer. 1991. Context based spelling correction. *Information Processing and Management*, 23(5), 517–522

- **AT&T Bell Labs**

- Kernighan, Mark D., Kenneth W. Church, and William A. Gale. 1990. A spelling correction program based on a noisy channel model. Proceedings of COLING 1990, 205-210



# Non-word spelling error example

acress



# Candidate generation

- Words with similar spelling
  - Small edit distance to error
- Words with similar pronunciation
  - Small edit distance of pronunciation to error



# Damerau-Levenshtein edit distance

- Minimal edit distance between two strings, where edits are:
  - Insertion
  - Deletion
  - Substitution
  - Transposition of two adjacent letters



# Words within 1 of access

Error	Candidate Correction	Correct Letter	Error Letter	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



# Candidate generation

- 80% of errors are within edit distance 1
- Almost all errors within edit distance 2
- Also allow insertion of **space** or **hyphen**
  - `thisidea` → `this idea`
  - `inlaw` → `in-law`





# Language Model

- Use any of the language modeling algorithms we've learned
- Unigram, bigram, trigram
- Web-scale spelling correction
  - Stupid backoff



# Unigram Prior probability

Counts from 404,253,213 words in Corpus of Contemporary English (COCA)

word	Frequency of word	P(word)
actress	9,321	.0000230573
gress	220	.0000005442
caress	686	.0000016969
access	37,038	.0000916207
across	120,844	.0002989314
acres	12,874	.0000318463



# Channel model probability

- **Error model probability, Edit probability**
- *Kernighan, Church, Gale 1990*
- *Misspelled word  $x = x_1, x_2, x_3 \dots x_m$*
- *Correct word  $w = w_1, w_2, w_3, \dots, w_n$*
- $P(x|w)$  = probability of the edit
  - (deletion/insertion/substitution/transposition)



# Computing error probability: confusion matrix

`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)`

Insertion and deletion conditioned on previous character



# 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	5	0	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	6	0	0	0	4	0	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



# Generating the confusion matrix

- [Peter Norvig's list of errors](#)
- [Peter Norvig's list of counts of single-edit errors](#)



# Channel model

Kernighan, Church, Gale 1990

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



# Channel model for access

Candidate Correction	Correct Letter	Error Letter	$x w$	$P(x word)$
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	.00000093
acres	-	s	es e	.0000321
acres	-	s	ss s	.0000342





# Noisy channel probability for access

Candidate Correction	Correct Letter	Error Letter	$x w$	$P(x word)$	$P(word)$	$10^9 * P(x w)P(w)$
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 probability for across

Candidate Correction	Correct Letter	Error Letter	$x w$	$P(x word)$	$P(word)$	$10^9 * P(x w)P(w)$
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
<b>across</b>	<b>o</b>	<b>e</b>	<b>e o</b>	<b>.0000093</b>	<b>.000299</b>	<b>2.8</b>
acres	-	s	es e	.0000321	.0000318	1.0
acres	-	s	ss s	.0000342	.0000318	1.0



# Using a bigram language model

- "a stellar and versatile **acress** whose combination of sass and glamour..."
- Counts from the Corpus of Contemporary American English with add-1 smoothing
- $P(\text{actress}|\text{versatile}) = .000021$     $P(\text{whose}|\text{actress}) = .0010$
- $P(\text{across}|\text{versatile}) = .000021$     $P(\text{whose}|\text{across}) = .000006$
- $P(\text{"versatile actress whose"}) = .000021 * .0010 = 210 \times 10^{-10}$
- $P(\text{"versatile across whose"}) = .000021 * .000006 = 1 \times 10^{-10}$



# Using a bigram language model

- "a stellar and versatile **actress** whose combination of sass and glamour..."
- Counts from the Corpus of Contemporary American English with add-1 smoothing
- $P(\text{actress}|\text{versatile}) = .000021$     $P(\text{whose}|\text{actress}) = .0010$
- $P(\text{across}|\text{versatile}) = .000021$     $P(\text{whose}|\text{across}) = .000006$
- $P(\text{"versatile actress whose"}) = .000021 * .0010 = 210 \times 10^{-10}$
- $P(\text{"versatile across whose"}) = .000021 * .000006 = 1 \times 10^{-10}$



# Evaluation

- Some spelling error test sets
  - [Wikipedia's list of common English misspelling](#)
  - [Aspell filtered version of that list](#)
  - [Birkbeck spelling error corpus](#)
  - [Peter Norvig's list of errors \(includes Wikipedia and Birkbeck, for training or testing\)](#)

# Spelling Correction and the Noisy Channel



Real-Word Spelling  
Correction



# Real-word spelling errors

- ...leaving in about fifteen ***minuets*** to go to her house.
- The design ***an*** construction of the system..
- Can they ***lave*** him my messages?
- The study was conducted mainly ***be*** John Black.
- 25-40% of spelling errors are real words [Kukich 1992](#)



# Solving real-world spelling errors

- For each word in sentence
  - Generate *candidate set*
    - the word itself
    - all single-letter edits that are English words
    - words that are homophones
- Choose best candidates
  - Noisy channel model
  - Task-specific classifier



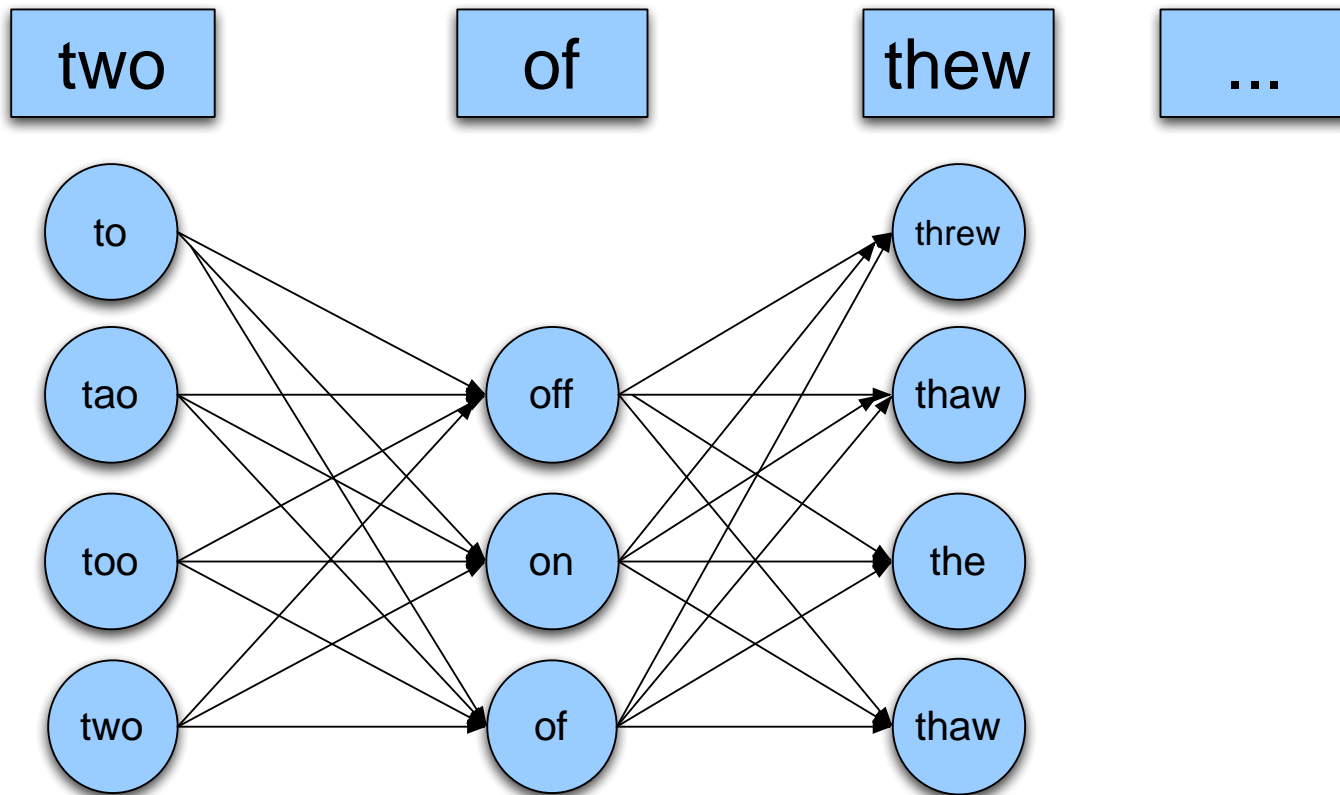


# Noisy channel for real-word spell correction

- Given a sentence  $w_1, w_2, w_3, \dots, w_n$
- Generate a set of candidates for each word  $w_i$ 
  - $\text{Candidate}(w_1) = \{w_1, w'_1, w''_1, w'''_1, \dots\}$
  - $\text{Candidate}(w_2) = \{w_2, w'_2, w''_2, w'''_2, \dots\}$
  - $\text{Candidate}(w_n) = \{w_n, w'_n, w''_n, w'''_n, \dots\}$
- Choose the sequence  $W$  that maximizes  $P(W)$

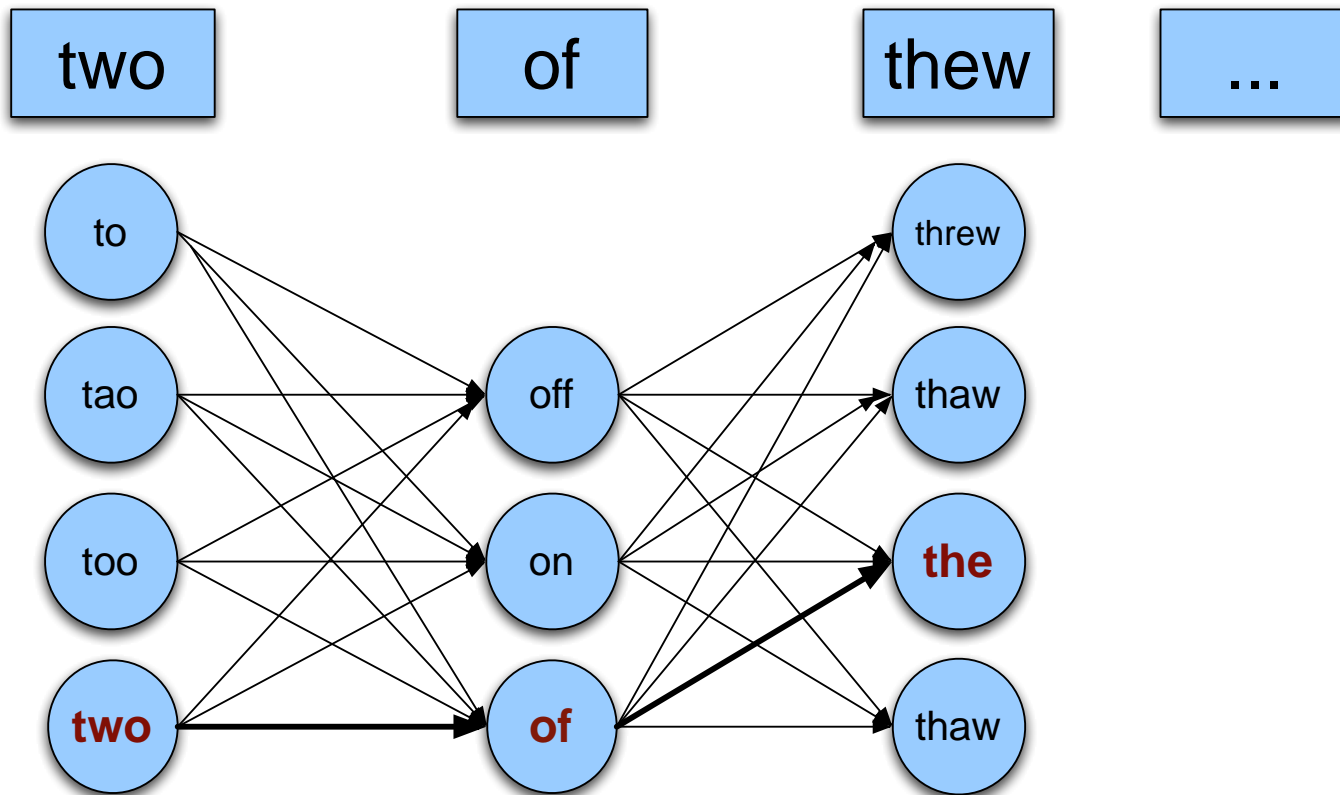


# Noisy channel for real-word spell correction





# Noisy channel for real-word spell correction





# Simplification: One error per sentence

- Out of all possible sentences with one word replaced
  - $w_1, w''_2, w_3, w_4$     **two off thew**
  - $w_1, w_2, w'_3, w_4$     **two of the**
  - $w'''_1, w_2, w_3, w_4$     **too of thew**
  - ...
- Choose the sequence  $W$  that maximizes  $P(W)$



# Where to get the probabilities

- Language model
  - Unigram
  - Bigram
  - Etc
- Channel model
  - Same as for non-word spelling correction
  - Plus need probability for no error,  $P(w | w)$



# Probability of no error

- What is the channel probability for a correctly typed word?
- $P(\text{"the"} | \text{"the"})$
- Obviously this depends on the application
  - .90 (1 error in 10 words)
  - .95 (1 error in 20 words)
  - .99 (1 error in 100 words)
  - .995 (1 error in 200 words)



# Peter Norvig's "thew" example

x	w	x w	$P(x w)$	$P(w)$	$10^9 P(x w)P(w)$
thew	the	ew e	0.000007	0.02	144
thew	thew		0.95	0.00000009	90
thew	thaw	e a	0.001	0.0000007	0.7
thew	threw	h hr	0.000008	0.000004	0.03
thew	thwe	ew we	0.000003	0.00000004	0.0001







# HCI issues in spelling

- If very confident in correction
  - Autocorrect
- Less confident
  - Give the best correction
- Less confident
  - Give a correction list
- Unconfident
  - Just flag as an error



# State of the art noisy channel

- We never just multiply the prior and the error model
- Independence assumptions  $\rightarrow$  probabilities not commensurate
- Instead: Weigh them

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

- Learn  $\lambda$  from a development test set



# Phonetic error model

- Metaphone, used in GNU aspell
  - Convert misspelling to metaphone pronunciation
    - “Drop duplicate adjacent letters, except for C.”
    - “If the word begins with 'KN', 'GN', 'PN', 'AE', 'WR', drop the first letter.”
    - “Drop 'B' if after 'M' and if it is at the end of the word”
    - ...
  - Find words whose pronunciation is 1-2 edit distance from misspelling’s
  - Score result list
    - Weighted edit distance of candidate to misspelling
    - Edit distance of candidate pronunciation to misspelling pronunciation



# Improvements to channel model

- Allow richer edits (Brill and Moore 2000)
  - ent→ant
  - ph→f
  - le→al
- Incorporate pronunciation into channel (Toutanova and Moore 2002)



# Channel model

- Factors that could influence  $p(\text{misspelling} | \text{word})$ 
  - The source letter
  - The target letter
  - Surrounding letters
  - The position in the word
  - Nearby keys on the keyboard
  - Homology on the keyboard
  - Pronunciations
  - Likely morpheme transformations



# Nearby keys





# Classifier-based methods for real-word spelling correction

- Instead of just channel model and language model
- Use many features in a classifier (next lecture).
- Build a classifier for a specific pair like:

whether/weather

- “cloudy” within +/- 10 words
- \_\_\_ to VERB
- \_\_\_ or not