Bahria University
Department of Computer Science

Adv. Natural Language Processing

Lecture 7

Instructor: Dr. Muhammad Asfand-e-yar

Previous Lecture

- Introduction to N – Grams
- Estimating N-Grams Probabilities
- Evaluation and Perplexity
- Generalization and Zeros
- Laplace Smoothing (Add 1)
- Interpolation and Backoff
- Kneser-Ney Smoothing

Today's Lecture

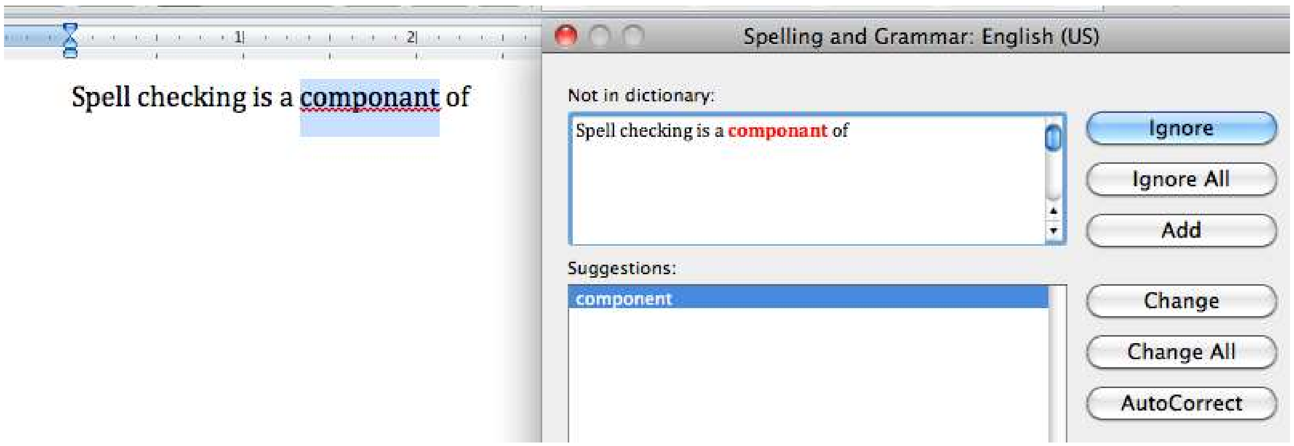
- Spelling Corrections Task
- Noisy Channel Model of Spelling
- Real World Spelling Corrections
- State of the Art Systems

Spelling Correction and the Noisy Channel

Spelling Correction Task

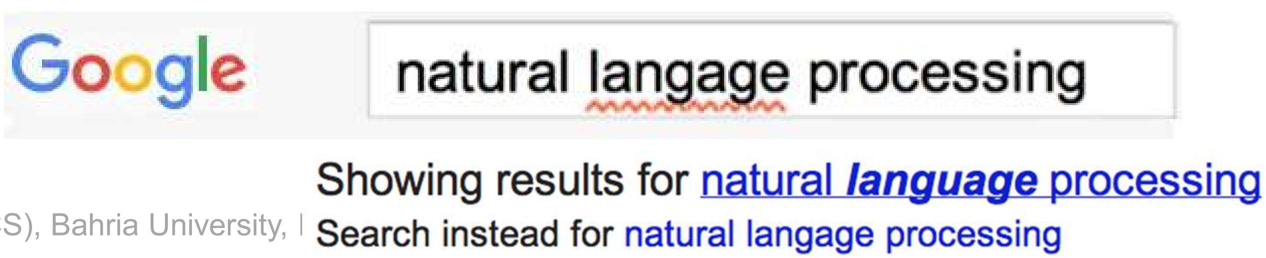
Applications for spelling correction

Word processing



Phones

Web search



Spelling Tasks

1. Spelling Error Detection

2. Spelling Error Correction:
    - Autocorrect
        - *hte* → *the*
    - Suggest a correction
    - Suggestion lists

Types of Spelling Errors

1. Non-word Errors

   graffe → *giraffe*

2. Real-word Errors
   a. Typographical Errors
      - *their* → *there*

   b. 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 smart devices.

2%: Words uncorrected on organizer: Soukoreff &MacKenzie 2003

1-2%: Retyping: Kane and Wobbrock 2007, Gruden et al. 1983

1. 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

2. 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

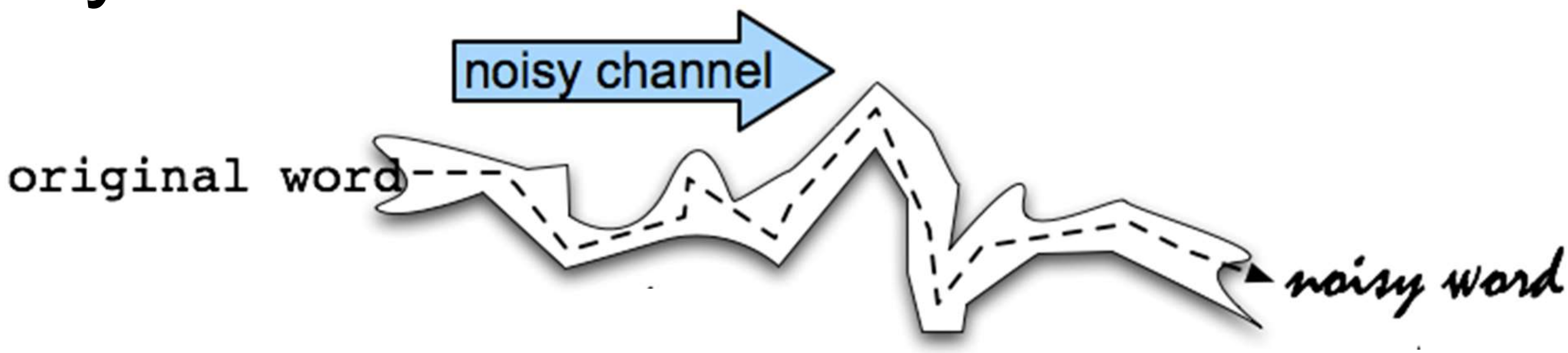
Choose best candidate

1. Noisy Channel
2. Classifier

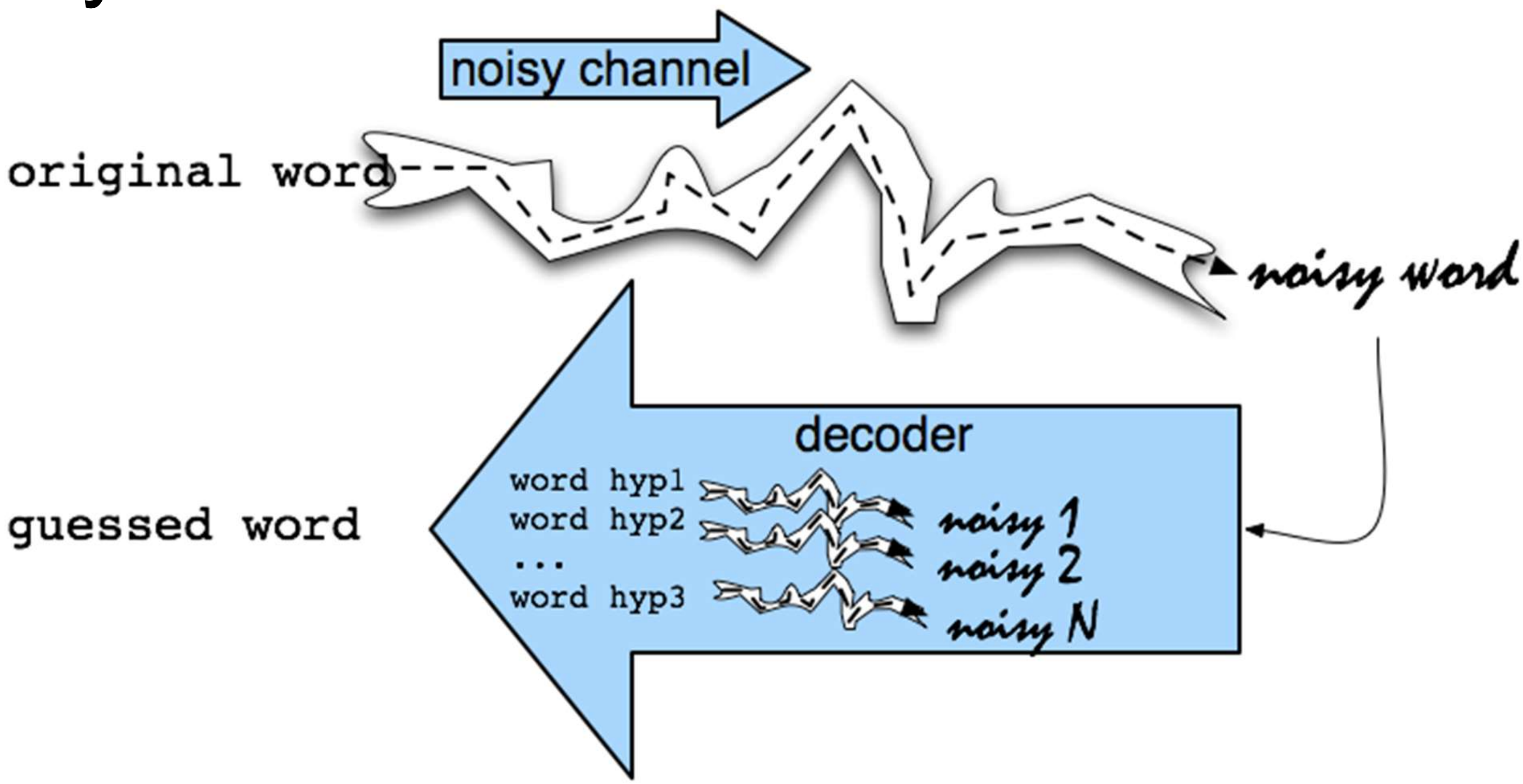
Spelling Correction and the Noisy Channel

Noisy Channel Model of Spelling

Noisy Channel Intuition



Noisy Channel Intuition



Noisy Channel

We see an observation *x* of a misspelled word

Find the correct word *w*

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

The best probable word

The best word out of vocabulary

Maximizing the best word Probability of word *w* to the given misspelled word *x*

Noisy Channel

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

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

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

The Bayes' rule (from Bayesian Classification) will be used to break down the probability *P(a|b)*

$$P(a|b) = \frac{P(b|a)P(a)}{P(b)}$$

We can simplify by dropping the denominator P(x).
Why P(x) is dropped?
Since we are choosing a potential correction word out of all words, we will be computing ***P(x|w)P(w) / P(x)*** for each word.
But P(x) doesn't change for each word.
Therefore, we can choose the word that maximizes this simpler formula.

Noisy Channel

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

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

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

Then it means that maximizing is depended on the two things according to formula;

1) P(x | w); i.e. likely hood (MLE)
2) P(w); i.e. Prior

The expression "P(x | w)" is called the <u>Channel Model</u>, which is also called Error Model

The expression "P(w)" is called the <u>Language Model</u> as seen before, i.e. the probability of the correct word

History: Noisy Channel for Spelling (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 `acress`

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
cress	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 $x = x_1, x_2, x_3 \ldots x_m$

Correct word $w = w_1, w_2, w_3, \ldots, w_n$

P(x|w) = probability of the edit

(deletion/insertion/substitution/transposition)

Computing Error Probability: confusion matrix

To construct the Channel Model a confusion matrix is to be created.

```
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 (incorrect) \ 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	0	5	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	0	6	0	0	0	4	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

You can also generate the confusion matrix table by yourself. The Peter Norvig collected errors from Wikipedia and other online resources. So, you can construct the matrix from the given list of errors.

Peter Norvig's list of errors

http://norvig.com/ngrams/

Peter Norvig's list of counts of single-edit errors

Channel Model

Kernighan, Church, Gale 1990

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

x_i is the error word.
w_i is the correct word.
w_{i-1} is the previous to the correct word.

Channel Model for acress

Channel Model

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	.0000093
acres	-	s	es|e	.0000321
acres	-	s	ss|s	.0000342

Noisy Channel Probability for `acress`

Candidate Correction	Correct Letter	Error Letter	x|w	P(x|word)	P(word)	10^9 *P(x|w)P(w)
				Channel Model	Language Model	To make readable
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 `acress`

To make readable

Candidate Correction	Correct Letter	Error Letter	x|w	P(x|word)	P(word)	10^9 *P(x|w)P(w)
				Channel Model	Language Model	
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

Using a Bigram Language Model

"a stellar and versatile **acress** whose combination of sass and glamour…"

- We checked in previous slides by Unigram
- Now we will check through Bigram

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(actress|versatile)=.000021; P(whose|actress) = .0010

P(across|versatile) =.000021; P(whose|across) = .000006

P("versatile actress whose") = .000021 x .0010 = 210 x 10^{-10}

P("versatile across whose") = .000021 x .000006 = 1 x 10^{-10}

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(actress|versatile)=.000021; P(whose|actress) = .0010

P(across|versatile) =.000021; P(whose|across) = .000006

P("versatile actress whose") = .000021 x .0010 = 210 x 10^{-10}

P("versatile across whose") = .000021 x .000006 = 1 x 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)

You can develop your training set from any of these sets and test sets to check your model that how it works.