EECS 595

Natural Language Processing

Lecture 3: Text Classification and Sentiment Analysis

Instructor: Joyce Chai

Last lecture on N-grams

- What is N-Gram?
- How to calculate N-Gram using MLE?
- How to address zero counts problem using smoothing?

Shannon's Method

- Assigning probabilities to sentences is all well and good, but it's not terribly illuminating . A more interesting task is to turn the model around and use it to generate random sentences that are *like* the sentences from which the model was derived.
- Generally attributed to Claude Shannon.

Shannon's Method

- Sample a random bigram (<s>, w) according to its probability
- Now sample a random bigram (w, x) according to its probability
  - Where the prefix w matches the suffix of the first.
- And so on until we randomly choose a (y, </s>)
- Then string the words together
- <s> I
  I want
  want to
  to eat
  eat Chinese
  Chinese food
  food </s>

Approximating Shakespeare

- As we increase the value of N, the accuracy of the n-gram model increases
- Generating sentences with random unigrams...
  - Every enter now severally so, let
  - Hill he late speaks; or! a more to leg less first you enter
- With bigrams...
  - What means, sir. I confess she? then all sorts, he is trim, captain.
  - Why dost stand forth thy canopy, forsooth; he is this palpable hit the King Henry.

- Trigrams
  - Sweet prince, Falstaff shall die.
  - This shall forbid it should be branded, if renown made it empty.
- Quadrigrams
  - What! I will go seek the traitor Gloucester.
  - Will you not tell me who I am?

- There are 884,647 tokens, with 29,066 word form types, in about a one million word Shakespeare corpus
- Shakespeare produced 300,000 bigram types out of 844 million possible bigrams: so, 99.96% of the possible bigrams were never seen (have zero entries in the table).
- Quadrigrams worse: What's coming out looks like Shakespeare because it ***is*** Shakespeare.

N-Gram Training Sensitivity

- If we repeated the Shakespeare experiment but trained on a Wall Street Journal corpus, there would be little overlap in the output
- This has major implications for corpus selection or design

Unigram

Months the my and issue of year foreign new exchange's september were recession exchange new endorsed a acquire to six executives

Bigram

Last December through the way to preserve the Hudson corporation N. B. E. C. Taylor would seem to complete the major central planners one point five percent of U. S. E. has already old M. X. corporation of living on information such as more frequently fishing to keep her

Trigram

They also point to ninety nine point six billion dollars from two hundred four oh six three percent of the rates of interest stores as Mexico and Brazil on market conditions

Google N-Gram Release

http://googleresearch.blogspot.com/2006/08/all-our-n-gram-are-belong-to-you.html

All Our N-gram are Belong to You

By Peter Norvig - 8/03/2006 11:26:00 AM

Posted by Alex Franz and Thorsten Brants, Google Machine Translation Team

Here at Google Research we have been using word n-gram models for a variety of R&D projects, such as statistical machine translation, speech recognition, spelling correction, entity detection, information extraction, and others. While such models have usually been estimated from training

to share this enormous dataset with everyone. We processed 1,024,908,267,229 words of running text and are publishing the counts for all 1,176,470,663 five-word sequences that appear at least 40 times. There are 13,588,391 unique words, after discarding words that appear less than 200 times.

Google N-Gram Release

- serve as the incoming 92
- serve as the incubator 99
- serve as the independent 794
- serve as the index 223
- serve as the indication 72
- serve as the indicator 120
- serve as the indicators 45
- serve as the indispensable 111
- serve as the indispensible 40
- serve as the individual 234

Google Caveat

- Remember the lesson about test sets and training sets... Test sets should be similar to the training set (drawn from the same distribution) for the probabilities to be meaningful.
- So... The Google corpus is fine if your application deals with arbitrary English text on the Web.
- If not then a smaller domain specific corpus is likely to yield better results.
- More: https://books.google.com/ngrams/info
- https://pypi.org/project/google-ngram-downloader/

Unknown Words

- But once we start looking at test data, we'll run into words that we haven't seen before (pretty much regardless of how much training data you have).
- With an Open Vocabulary task
  - Create an unknown word token <UNK>
  - Training of <UNK> probabilities
    - Create a fixed lexicon L, of size V
      - From a dictionary or
      - A subset of terms from the training set
    - At text normalization phase, any training word not in L changed to <UNK>
    - Now we count that like a normal word
  - At test time
    - Use UNK counts for any word not in training

Evaluation

- How do we know if our models are any good?
  - And in particular, how do we know if one model is better than another.
- Well Shannon's game gives us an intuition.
  - The generated texts from the higher order models sure look better. That is, they sound more like the text the model was obtained from.
  - But what does that mean? How do we quantify that?

Evaluation

- Standard method
  - Train parameters of our model on a **training set**.
  - Look at the models performance on some new data
    - This is exactly what happens in the real world; we want to know how our model performs on data we haven't seen
  - So use a **test set**. A dataset which is different than our training set, but is drawn from the same source
  - Then we need an **evaluation metric** to tell us how well our model is doing on the test set.
- What could be an intuitive but expensive way?
  - Extrinsic evaluation: speech recognition
  - Expensive
- Intrinsic evaluation: Perplexity

Perplexity

- Perplexity is the probability of the test set (assigned by the language model), normalized by the number of words:

$$PP(W) = P(w_1 w_2 ... w_N)^{-\frac{1}{N}}$$

$$= \sqrt[N]{\frac{1}{P(w_1 w_2 ... w_N)}}$$

- For bigrams: $PP(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i \mid w_{i-1})}}$

Minimizing perplexity is the same as maximizing probability

  - **The best language model is one that best predicts an unseen test set**

Lower perplexity means a better model

- Training 38 million words, test 1.5 million words, WSJ

N-gram Order	Unigram	Bigram	Trigram
Perplexity	962	170	109

Take away

- N-Grams played a crucial role in NLP for a long time.
  - The idea and concepts (e.g., smoothing) etc. are still important for similar problems in other fields.
- In practice, N-Gram has been recently replaced by neural language models trained on large amount of data.

Text Classification and Sentiment Analysis

Positive or negative movie review?

- unbelievably disappointing

- Full of zany characters and richly applied satire, and some great plot twists

- this is the greatest screwball comedy ever filmed

- It was pathetic. The worst part about it was the boxing scenes.

What is the subject of this article?

MEDLINE Article

Brain and Cognition

Syntactic frame and verb bias in aphasia: Plausibility judgments of undergoer-subject sentences

?

MeSH Subject Category Hierarchy

- Antogonists and Inhibitors
- Blood Supply
- Chemistry
- Drug Therapy
- Embryology
- Epidemiology
- …

Text Classification

- Assigning subject categories, topics, or genres
- Spam detection
- Authorship identification
- Age/gender identification
- Sentiment analysis
- …

Text Classification: definition

- *Input*:
  - a document d
  - a fixed set of classes $C = \{c_1, c_2, \ldots, c_J\}$

- *Output*: a predicted class $c \in C$

Classification Methods: Hand-coded rules

- Rules based on combinations of words or other features
  - spam: black-list-address OR (“dollars” AND“have been selected”)
- Accuracy can be high
  - If rules carefully refined by expert
- But building and maintaining these rules is expensive

Classification Methods: Supervised Machine Learning

- *Input:*
  - a document *d*
  - a fixed set of classes $C = \{c_1, c_2, \ldots, c_J\}$
  - A training set of *m* hand-labeled documents $(d_1,c_1),\ldots,(d_m,c_m)$
- *Output:*
  - a learned classifier $\gamma:d \rightarrow c$

Classification Methods: Supervised Machine Learning

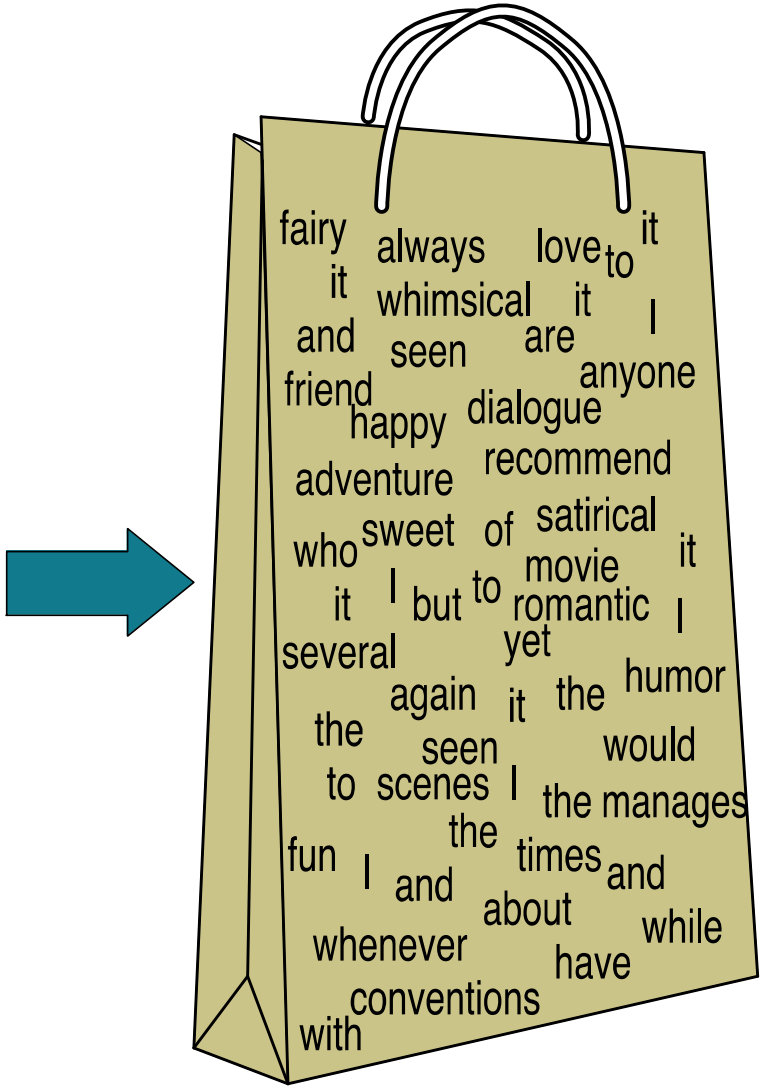
- Any kind of classifier
  - Naïve Bayes
  - Logistic regression
  - Neural networks
  - Support vector machines
  - …

Naïve Bayes Intuition

- Simple ("naïve") classification method based on Bayes rule
- Relies on very simple representation of document
  - Bag of words

The Bag of Words Representation

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!



it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
…	…

The bag of words representation

$\gamma($

seen	2
sweet	1
whimsical	1
recommend	1
happy	1
...	...

$)=c$

Bayes' Rule Applied to Documents and Classes

- For a document *d* and a class *c*

$$P(c \mid d) = \frac{P(d \mid c)P(c)}{P(d)}$$

Naïve Bayes Classifier (I)

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(c \mid d)$$

MAP is "maximum a posteriori" = most likely class

$$= \underset{c \in C}{\operatorname{argmax}} \frac{P(d \mid c)P(c)}{P(d)}$$

Bayes Rule

$$= \underset{c \in C}{\operatorname{argmax}} P(d \mid c)P(c)$$

Dropping the denominator

Naïve Bayes Classifier (II)

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(d \mid c) P(c)$$

$$= \underset{c \in C}{\operatorname{argmax}} P(x_1, x_2, \ldots, x_n \mid c) P(c)$$

Document d represented as features x1..xn

Naïve Bayes Classifier (IV)

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(x_1, x_2, \ldots, x_n \mid c) P(c)$$

$O(|X|^n \bullet |C|)$ parameters

How often does this class occur?

Could only be estimated if a very, very large number of training examples was available.

We can just count the relative frequencies in a corpus

Naïve Bayes Independence Assumptions

$$P(x_1, x_2, \ldots, x_n \mid c)$$

- **Bag of Words assumption**: Assume position doesn't matter
- **Conditional Independence**: Assume the feature probabilities $P(x_i \mid c_j)$ are independent given the class *c.*

$$P(x_1, \ldots, x_n \mid c) = P(x_1 \mid c) \bullet P(x_2 \mid c) \bullet P(x_3 \mid c) \bullet \ldots \bullet P(x_n \mid c)$$

Naïve Bayes Classifier

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(x_1, x_2, \ldots, x_n \mid c) P(c)$$

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}} P(c_j) \prod_{x \in X} P(x \mid c)$$

Naïve Bayes Learning

- From training corpus, extract *Vocabulary*

- Calculate $P(c_j)$ terms
  - For each c_j in C do

    $docs_j \leftarrow$ all docs with class $=c_j$ $\quad P(c_j) \leftarrow \dfrac{|docs_j|}{|\text{total \# documents}|}$

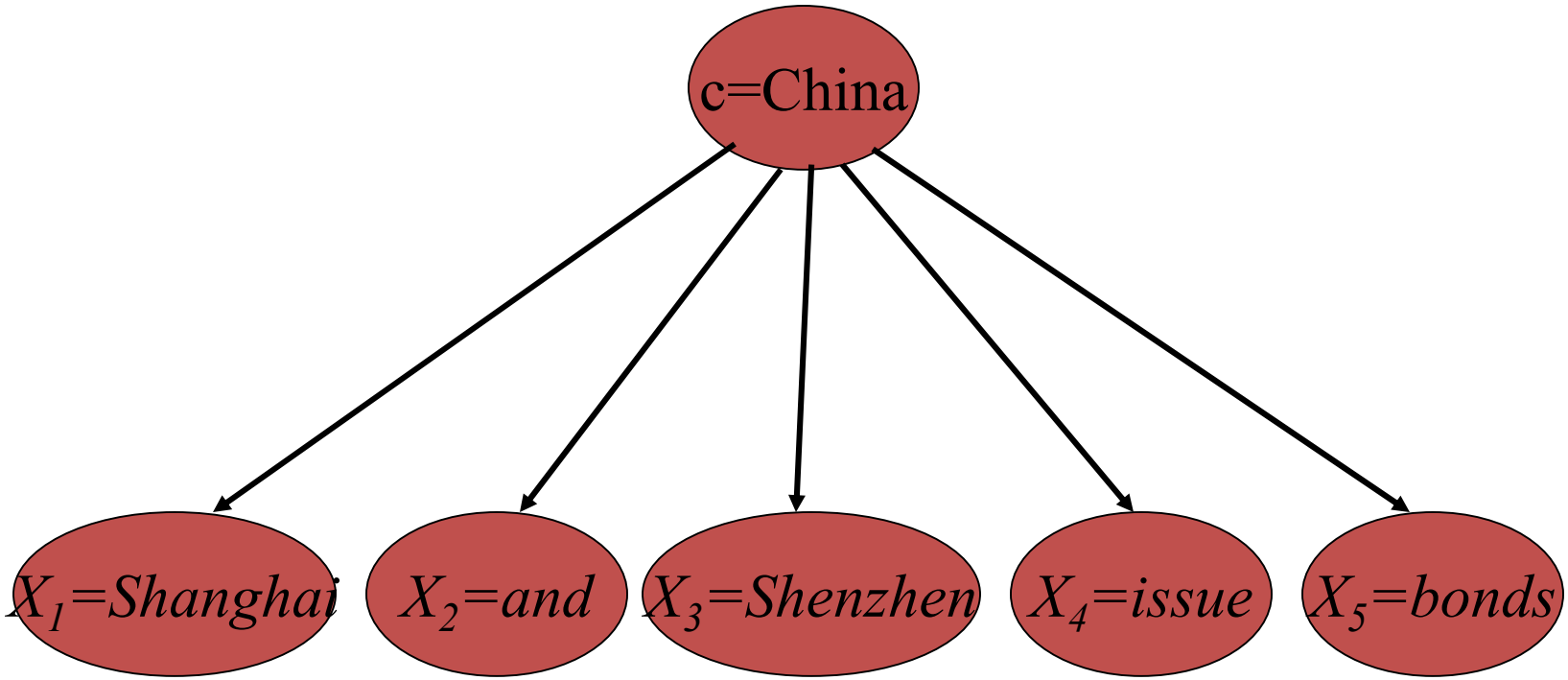
- Calculate $P(w_k \mid c_j)$ terms
  - $Text_j \leftarrow$ single doc containing all $docs_j$
  - For each word w_k in *Vocabulary*

    $n_k \leftarrow$ # of occurrences of w_k in $Text_j$

    n is the total number of words in $Text_j$

$$P(w_k \mid c_j) \leftarrow \frac{n_k + \alpha}{n + \alpha |Vocabulary|}$$

Generative Model for Naïve Bayes



Naïve Bayes and Language Modeling

- Naïve bayes classifiers can use any sort of feature
  - URL, email address, dictionaries, network features
- But if, as in the previous slides
  - We use **only** word features
  - we use **all** of the words in the text (not a subset)
- Then
  - Naïve bayes has an important similarity to language modeling.

Each class = a unigram language model

- Assigning each word: P(word | c)
- Assigning each sentence: P(s|c)= Π P(word|c)

Class *pos*

0.1	I
0.1	love
0.01	this
0.05	fun
0.1	film
…	

I	love	this	fun	film
0.1	0.1	0.01	0.05	0.1

P(s | pos) = 0.0000005

Naïve Bayes as a Language Model

- Which class assigns the higher probability to s?

Model pos	
0.1	I
0.1	love
0.01	this
0.05	fun
0.1	film

Model neg	
0.2	I
0.001	love
0.01	this
0.005	fun
0.1	film

I	love	this	fun	film
0.1	0.1	0.01	0.05	0.1
0.2	0.001	0.01	0.005	0.1

$P(s|pos) > P(s|neg)$

	Doc	Words	Class
Training	1	Chinese Beijing Chinese	c
	2	Chinese Chinese Shanghai	c
	3	Chinese Macao	c
	4	Tokyo Japan Chinese	j
Test	5	Chinese Chinese Chinese Tokyo Japan	?

$$\hat{P}(c) = \frac{N_c}{N} \qquad \hat{P}(w \mid c) = \frac{count(w,c)+1}{count(c)+|V|}$$

Priors: $P(c) = ¾$; $P(j) = ¼$

Conditional Probabilities:

P(Chinese|*c*) = (5+1) / (8+6) = 6/14 = 3/7
P(Tokyo|*c*) = (0+1) / (8+6) = 1/14
P(Japan|*c*) = (0+1) / (8+6) = 1/14
P(Chinese|*j*) = (1+1) / (3+6) = 2/9
P(Tokyo|*j*) = (1+1) / (3+6) = 2/9
P(Japan|*j*) = (1+1) / (3+6) = 2/9

Choosing a class:

$P(c|d5) \propto 3/4 * (3/7)^3 * 1/14 * 1/14 \approx 0.0003$

$P(j|d5) \propto 1/4 * (2/9)^3 * 2/9 * 2/9 \approx 0.0001$

Naïve Bayes in Spam Filtering

- SpamAssassin Features:
  - Mentions Generic Viagra
  - Online Pharmacy
  - Mentions millions of (dollar) ((dollar) NN,NNN,NNN.NN)
  - Phrase: impress ... girl
  - From: starts with many numbers
  - Subject is all capitals
  - HTML has a low ratio of text to image area
  - One hundred percent guaranteed
  - Claims you can be removed from the list
  - 'Prestigious Non-Accredited Universities'
  - http://spamassassin.apache.org/tests_3_3_x.html

Underflow Prevention: log space

- Multiplying lots of probabilities can result in floating-point underflow.
- Since log(*xy*) = log(*x*) + log(*y*)
  - Better to sum logs of probabilities instead of multiplying probabilities.
- Class with highest un-normalized log probability score is still most probable.

$$c_{NB} = \underset{c_j \in C}{\operatorname{argmax}} \left(\log P(c_j) + \sum_{i \in positions} \log P(x_i \mid c_j) \right)$$

- Model is now just max of sum of weights

Summary: Naive Bayes is Not So Naive

- Robust to Irrelevant Features

  Irrelevant Features cancel each other without affecting results

- Very good in domains with many equally important features
- Optimal if the independence assumptions hold
- A good dependable baseline for text classification
  - **But we will see other classifiers that give better accuracy**

Sentiment Analysis

- Words can be quite informative

+ ...zany characters and richly applied satire, and some great plot twists
- It was pathetic. The worst part about it was the boxing scenes...
+ ...awesome caramel sauce and sweet toasty almonds. I love this place!
- ...awful pizza and ridiculously overpriced...

Frequency/Presence Features

- for sentiment classification and a number of other text classification tasks, whether a word occurs or not seems to matter more than its frequency

Four original documents:

- – it was pathetic the worst part was the boxing scenes
- – no plot twists or great scenes
- \+ and satire and great plot twists
- \+ great scenes great film

After per-document binarization:

- – it was pathetic the worst part boxing scenes
- – no plot twists or great scenes
- \+ and satire great plot twists
- \+ great scenes film

	NB Counts		Binary Counts	
	+	–	+	–
and	2	0	1	0
boxing	0	1	0	1
film	1	0	1	0
great	3	1	2	1
it	0	1	0	1
no	0	1	0	1
or	0	1	0	1
part	0	1	0	1
pathetic	0	1	0	1
plot	1	1	1	1
satire	1	0	1	0
scenes	1	2	1	2
the	0	2	0	1
twists	1	1	1	1
was	0	2	0	1
worst	0	1	0	1

Dealing with Negation

- A simple way: prepend the prefix NOT to every word after a token of logical negation *(n't, not, no, never)* until the next punctuation mark

didnt like this movie , but I

=> didnt NOT_like NOT_this NOT_movie , but I

NOT_like, *NOT_recommend* occur more often in negative document and act as cues for negative sentiment, while words like *NOT_bored*, *NOT_dismiss* will acquire positive associations.

- Parsing (covered later) will allow to identify more accurate scope of predicates that are modified by negation. But the simple approach works pretty well in practice.

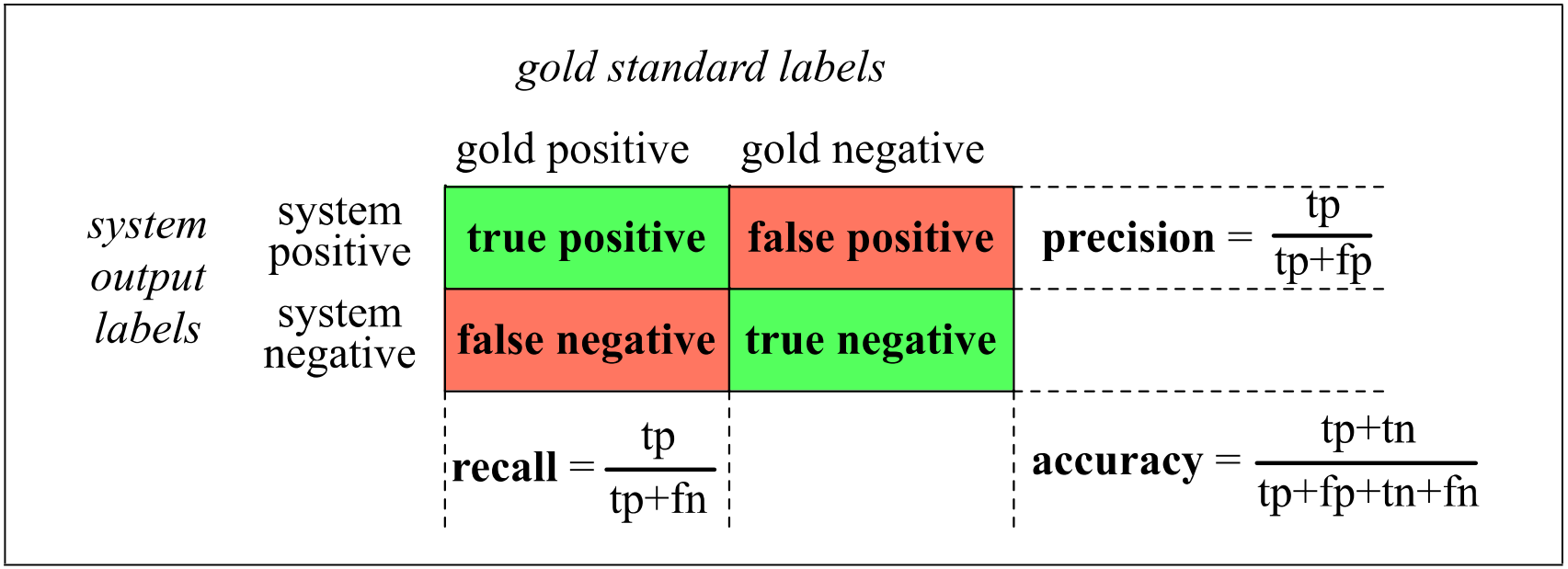
Insufficient Data?

- Instead of all words, use positive/negative word features from **sentiment lexicons** (e.g., General Inquirer, LIWC, MPQA subjectivity lexicon)
- The MPQA subjectivity lexicon has 6885 words, 2718 positive and 4912 negative, each marked for whether it is strongly or weakly biased.

positive : *admirable, beautiful, confident, dazzling, ecstatic, favor, glee, great*

negative: *awful, bad, bias, catastrophe, cheat, deny, envious, foul, harsh, hate*

Evaluation: Contingency Table



A combined measure: F

- A combined measure that assesses the P/R tradeoff is F measure (weighted harmonic mean):

$$F = \frac{1}{\alpha\frac{1}{P} + (1-\alpha)\frac{1}{R}} = \frac{(\beta^2+1)PR}{\beta^2 P + R} \quad \left(\text{with } \beta^2 = \frac{1-\alpha}{\alpha}\right)$$

- People usually use balanced F1 measure
  - i.e., with $\beta = 1$ (that is, $\alpha = ½$):

$$F_1 = \frac{2PR}{P+R}$$

Evaluation:
Classic Reuters-21578 Data Set

- Most (over)used data set, 21,578 docs (each 90 types, 200 toknens)
- 9603 training, 3299 test articles (ModApte/Lewis split)
- 118 categories
  - An article can be in more than one category
  - Learn 118 binary category distinctions
- Average document (with at least one category) has 1.24 classes
- Only about 10 out of 118 categories are large

Common categories
(#train, #test)

- Earn (2877, 1087)
- Acquisitions (1650, 179)
- Money-fx (538, 179)
- Grain (433, 149)
- Crude (389, 189)
- Trade (369,119)
- Interest (347, 131)
- Ship (197, 89)
- Wheat (212, 71)
- Corn (182, 56)

Reuters Text Categorization data set (**Reuters-21578)** document

Sec. 15.2.4

<REUTERS TOPICS="YES" LEWISSPLIT="TRAIN" CGISPLIT="TRAINING-SET" OLDID="12981" NEWID="798">

<DATE> 2-MAR-1987 16:51:43.42</DATE>

<TOPICS><D>livestock</D><D>hog</D></TOPICS>

<TITLE>AMERICAN PORK CONGRESS KICKS OFF TOMORROW</TITLE>

<DATELINE> CHICAGO, March 2 - </DATELINE><BODY>The American Pork Congress kicks off tomorrow, March 3, in Indianapolis with 160 of the nations pork producers from 44 member states determining industry positions on a number of issues, according to the National Pork Producers Council, NPPC.

Delegates to the three day Congress will be considering 26 resolutions concerning various issues, including the future direction of farm policy and the tax law as it applies to the agriculture sector. The delegates will also debate whether to endorse concepts of a national PRV (pseudorabies virus) control and eradication program, the NPPC said.

A large trade show, in conjunction with the congress, will feature the latest in technology in all areas of the industry, the NPPC added. Reuter

&#3;</BODY></TEXT></REUTERS>

Confusion matrix c

- For each pair of classes <c_1,c_2> how many documents from c_1 were incorrectly assigned to c_2?
  - $c_{3,2}$: 90 wheat documents incorrectly assigned to poultry

Docs in test set	Assigned UK	Assigned poultry	Assigned wheat	Assigned coffee	Assigned interest	Assigned trade
True UK	95	1	13	0	1	0
True poultry	0	1	0	0	0	0
True wheat	10	**90**	0	1	0	0
True coffee	0	0	0	34	3	7
True interest	-	1	2	13	26	5
True trade	0	0	2	14	5	10

Accuracy: (1 - error rate)

Fraction of docs classified correctly:

$$\frac{\sum_i c_{ii}}{\sum_j \sum_i c_{ij}}$$

Per class evaluation measures

Recall:

Fraction of docs in class *i* classified correctly: $\frac{c_{ii}}{\sum_j c_{ij}}$

Precision:

Fraction of docs assigned class *i* that are actually about class *i*: $\frac{c_{ii}}{\sum_j c_{ji}}$

Micro- vs. Macro-Averaging

If we have more than one class, how do we combine multiple performance measures into one quantity?

- **Macroaveraging**: Compute performance for each class, then average.
- **Microaveraging**: Collect decisions for all classes, compute contingency table, evaluate.

Micro- vs. Macro-Averaging: Example

Class 1

	Truth: yes	Truth: no
Classifier: yes	10	10
Classifier: no	10	970

Class 2

	Truth: yes	Truth: no
Classifier: yes	90	10
Classifier: no	10	890

Micro Ave. Table

	Truth: yes	Truth: no
Classifier: yes	100	20
Classifier: no	20	1860

- Macroaveraged precision: (0.5 + 0.9)/2 = 0.7
- Microaveraged precision: 100/120 = .83
- Microaveraged score is dominated by score on common classes

Micro- vs. Macro-Averaging: Example

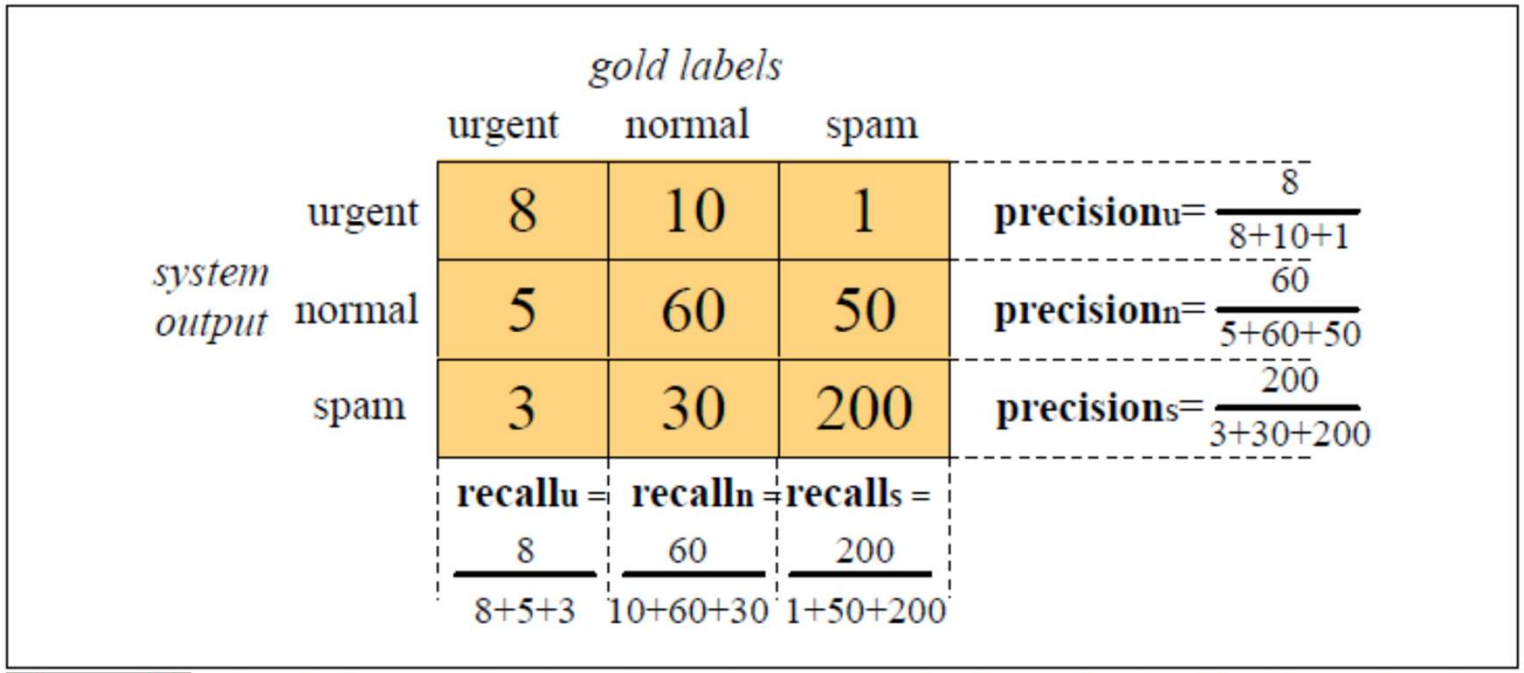


Figure 4.5 Confusion matrix for a three-class categorization task, showing for each pair of classes (c_1, c_2), how many documents from c_1 were (in)correctly assigned to c_2

Micro- vs. Macro-Averaging: Example

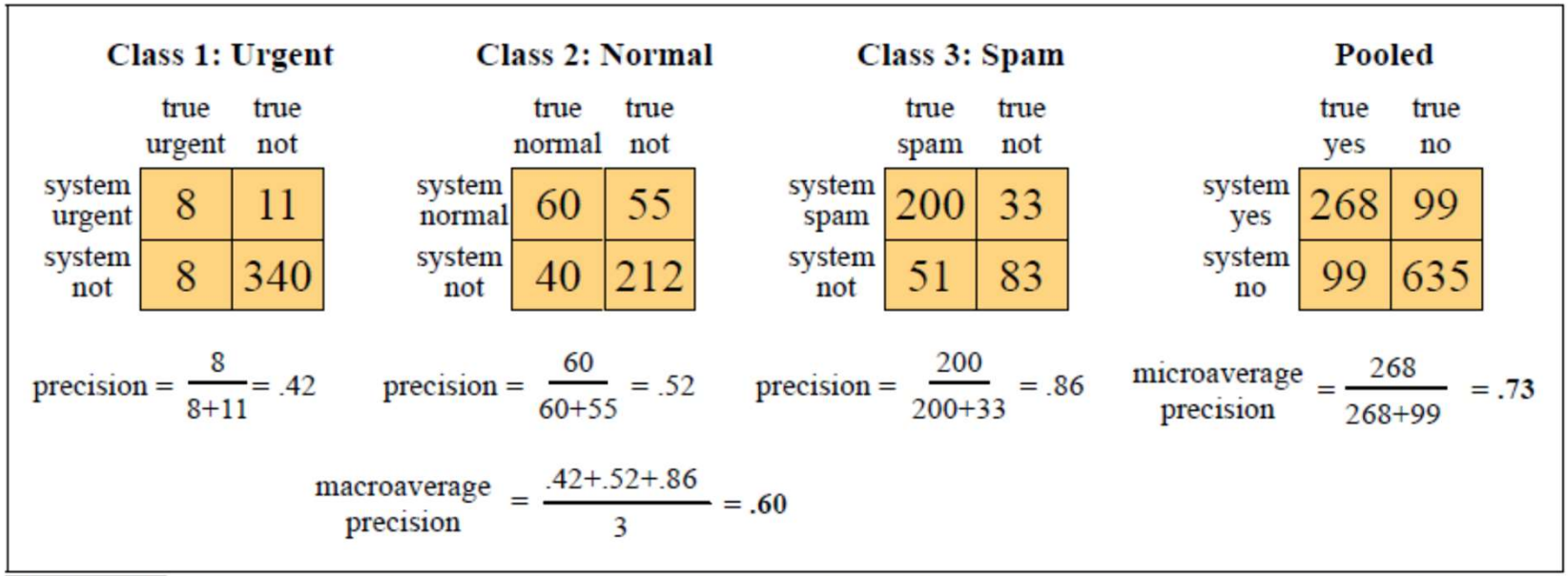


Figure 4.6 Separate contingency tables for the 3 classes from the previous figure, showing the pooled contingency table and the microaveraged and macroaveraged precision.

Evaluation

- Training data: train the model
- Validation (development) data: further tune some parameters
- Testing data: report the model performance

Cross-Validation

- Break up data into 10 folds
  - (Equal positive and negative inside each fold?)
- For each fold
  - Choose the fold as a temporary test set
  - Train on 9 folds, compute performance on the test fold
- Report average performance of the 10 runs

Cross-validation

- For language processing, we often need to look at the data to design features.
- So often have a fixed test set which has never been seen before