Language Models & Smoothing

$$
\begin{aligned}
p(\boldsymbol{W} = \boldsymbol{w}) &= p(W_1 = w_1, W_2 = w_2, \ldots, W_{L+1} = stop) \\
&= \left(\prod_{\ell=1}^{L} p(W_\ell = w_\ell \mid \boldsymbol{W}_{1:\ell-1} = \boldsymbol{w}_{1:\ell-1}) \right) p(W_{L+1} = stop \mid \boldsymbol{W}_{1:L} = \boldsymbol{w}_{1:L}) \\
&= \left(\prod_{\ell=1}^{L} p(w_\ell \mid history_\ell) \right) p(stop \mid history_L)
\end{aligned}
$$

$$
\begin{aligned}
\text{perplexity}(p(\cdot); \boldsymbol{w}) &= 2^{\left(-\frac{\log_2 p(\boldsymbol{w})}{|\boldsymbol{w}|}\right)} \\
&= p(\boldsymbol{w})^{-\frac{1}{|\boldsymbol{w}|}} \\
&= \sqrt[|\boldsymbol{w}|]{\frac{1}{p(\boldsymbol{w})}} \\
&= \sqrt[|\boldsymbol{w}|]{\frac{1}{\prod_{i=1}^{|\boldsymbol{w}|} p(w_i \mid w_{i-N+1}, \ldots, w_{i-1})}}
\end{aligned}
$$

Maximum likelihood ("relative frequency") estimation:

$$
\hat{p}_{\text{MLE}}(w_N \mid \langle w_1, \ldots, w_{N-1} \rangle) = \frac{\text{count}(\langle w_1, w_2, \ldots, w_{N-1}, w_N \rangle)}{\sum_{v \in \Sigma} \text{count}(\langle w_1, w_2, \ldots, w_{N-1}, v \rangle)}
$$

Add-λ smoothing (commonly, $\lambda = 1$, in which case this is also called Laplace smoothing):

$$
\hat{p}_{\text{add-}\lambda}(w_N \mid \langle w_1, \ldots, w_{N-1} \rangle) = \frac{\lambda + \text{count}(\langle w_1, w_2, \ldots, w_{N-1}, w_N \rangle)}{\lambda|\Sigma| + \sum_{v \in \Sigma} \text{count}(\langle w_1, w_2, \ldots, w_{N-1}, v \rangle)}
$$

Good–Turing discounted counts:

$$
c^* = \frac{(c+1) \times N_{c+1}}{N_c}
$$

Linear interpolation ("mixture model"):

$$
\begin{aligned}
\hat{p}_{\text{interp}}(w_N \mid \langle w_1, \ldots, w_{N-1} \rangle) =& \lambda_N \times \hat{p}_{\text{MLE}}(w_N \mid \langle w_1, \ldots, w_{N-1} \rangle) \\
&+ \lambda_{N-1} \times \hat{p}_{\text{MLE}}(w_N \mid \langle w_2, \ldots, w_{N-1} \rangle) \\
&\vdots \\
&+ \lambda_2 \times \hat{p}_{\text{MLE}}(w_N \mid \langle w_{N-1} \rangle) \\
&+ \lambda_1 \times \hat{p}_{\text{MLE}}(w_N) \\
&+ \lambda_0 \times \frac{1}{|\Sigma|}
\end{aligned}
$$

Noisy Channel & Edit Distance

$$D_{0,0} = 0$$

$$
D_{i,j} = \min \begin{cases} D_{i-1,j} + \text{inscost}(t_i) \\ D_{i,j-1} + \text{delcost}(s_j) \\ D_{i-1,j-1} + \text{substcost}(t_i, s_j) \end{cases}
$$

$$D_{0,0} = 0$$

$$
D_{i,j} = \min \begin{cases} D_{i-1,j} + \text{inscost}(t_i) \\ D_{i,j-1} + \text{delcost}(s_j) \\ D_{i-1,j-1} + \text{substcost}(t_i, s_j) \\ D_{i-2,j-2} + \text{transcost}(s_{j-1}, s_j) \text{if } s_{j-1} = t_i \text{ and } s_j = t_{i-1} \end{cases}
$$

+ Noisy Channel

The Noisy Channel Model assumes we want to recover the most likely system <u>output y</u> for an observed <u>input x</u> by modeling the probability distribution <u>p(y | x)</u>

$$
\begin{aligned}
y^* &= \arg\max_y p(y \mid x) \\
&= \arg\max_y \frac{p(x \mid y) \times p(y)}{p(x)} && \text{Bayes' rule} \\
&= \arg\max_y \frac{p(x \mid y) \times p(y)}{\sum_{y'} p(x \mid y') \times p(y')} \\
&= \arg\max_y p(x \mid y) \times p(y)
\end{aligned}
$$

Classification

+ Naïve Bayes Classifier

$\phi_j \leftarrow [\boldsymbol{\Phi}(x)]_j$

return

$\arg\max_{y'} p(y') \times \prod_j p(\phi_j \mid y')$

+ Naïve Bayes Learner

$$
\begin{array}{l|l|l}
\boldsymbol{x} \rightarrow & \forall y,\ p(y) \leftarrow \dfrac{\text{count}(y)}{N} & \\
\boldsymbol{y} \rightarrow & \forall y,\ \forall j,\ \forall f,\ p(\phi_j(x) = f \mid y) \leftarrow \dfrac{\text{count}(f, y)}{\text{count}(y)} & \rightarrow \boxed{p}
\end{array}
$$

Sentiment Analysis

Social Media (tweets), product reviews, discussion forum posts, and blog posts

Part of Speech Tags

open classes	closed classes
nouns	prepositions
verbs	particles
adjectives	determiners
adverbs	numerals
	pronouns
	conjunctions
	auxiliary verbs

CC	**Conj.**	**PRP**	**Pronoun, personal**
CD	**Numeral, Cardinal**	**RB**	**Adverb**
DT	**Determiner**	**RBR**	**Adverb, comparative**
IN	**Prep., Conj.**	**RP**	**Particle**
JJ	**Adj. or Numeral**	**VB**	**Verb, base form**
NN	**Noun, common, s.**	**VBD**	**Verb, past tense**
NNP	**Noun, proper, s.**	**VBG**	**V., present participle**
NNPS	**Noun, proper, pl.**	**VBN**	**V., past participle**
NNS	**Noun, common, pl.**	**VBP**	**V., present, not 3rd**

Two difficulties:

1) What tags? Unknown words?

2) Disambiguating words (more than one tag)

Hidden Markov Models

+ Markov Property: the state we move to at time t depends only on the state we were at time t-1

+ Symbols are observed, state sequence is hidden

- *q0*: start state ("silent")
- *qf*: final state ("silent")
- *Q*: set of "normal" states (excludes *q0* and final *qf*)
- Σ: vocabulary of observable symbols
- *γi,j*: probability of transitioning to *qj* given current state *qi*
- *ηi,w*: probability of emitting *w* ∈ Σ given current state *qi*

+ Viterbi Algorithm

$$y_n^* = \arg\max_{q_i \in Q} p(Y_1 = y_1^*, \ldots, Y_{n-1} = y_{n-1}^*, Y_n = q_i \mid \boldsymbol{x})$$
$$= \arg\max_{q_i \in Q} V[n-1, y_{n-1}^*] \cdot \gamma_{y_{n-1}^*, i} \cdot \eta_{i, x_n} \cdot \gamma_{i,f}$$
$$= \arg\max_{q_i \in Q} \gamma_{y_{n-1}^*, i} \cdot \eta_{i, x_n} \cdot \gamma_{i,f}$$

$$V[0, q_0] = 1$$

$$V[t, q_j] = \max_{q_i \in Q \cup \{q_0\}} V[t-1, q_i] \cdot \gamma_{i,j} \cdot \eta_{j, x_t}$$

$$\text{goal} = \max_{q_i \in Q} V[n, q_i] \cdot \gamma_{i,f}$$

V[*, *] ← 0
V[0, *q*0] ← 1
for *t* = 1 … *n*
  foreach *qj*
    foreach *qi*
      V[*t*, *qj*] ← max{*V*[*t*, *qj*] , *V*[*t* - 1, *qi*] × *γi,j* × *ηi,xt*}
foreach *qi*
  goal ← max{ goal, *V*[*n*, *qi*] × *γi,f* }
return goal

 # Syntactic Representation

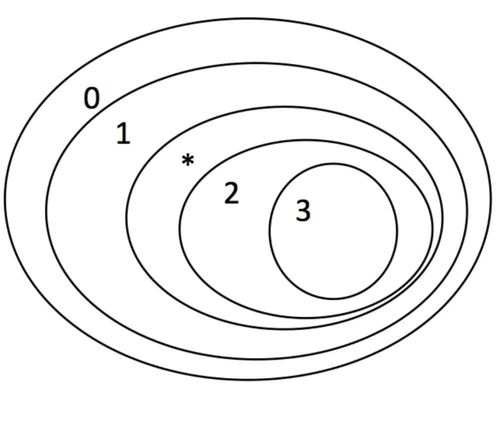
- Vocabulary of terminal symbols, Σ
- Set of nonterminal symbols (a.k.a. variables), N
- Special start symbol S ∈ N
- Production rules of the form X → α

where
  X ∈ N
  α ∈ (N ∪ Σ)*

Grammatical: said of a sentence in the language
Ungrammatical: said of a sentence **not** in the language
Derivation: sequence of top-down production steps
Parse tree: graphical representation of the derivation

A string is grammatical iff there exists a derivation for it.

 # Chomsky Hierarchy



	language class	automaton
0	recursively enumerable	Turing machine
1	context-sensitive	linear bounded automaton
*	mildly context-sensitive	thread automaton
2	context-free	pushdown automaton
3	regular	finite-state automaton

+ Pumping Lemma

If L is an infinite regular language, then there are strings x, y, and z such that y is not empty, xy^nz is in L, for all n >= 0

 # Context-free Recognition & CKY Algorithm

<u>Requires the grammar be in Chomsky normal form</u>

for *i* = 1 … *n*
  C[*i*-1, *i*] = { V | V → *wi* }
for ℓ = 2 … *n* // width
  for *i* = 0 … *n* - ℓ // left boundary
    k = *i* + ℓ // right boundary
      for *j* = *i* + 1 … *k* – 1 // midpoint
        C[*i*, *k*] = C[*i*, *k*] ∪
          { V | V → YZ, Y ∈ C[*i*, *j*], Z ∈ C[*j*, *k*] }
return true if S ∈ C[0, *n*]

$$C[i-1, i, w_i] = \text{TRUE}$$

$$C[i-1, i, V] = \begin{cases} \text{TRUE} & \text{if } V \to w_i \\ \text{FALSE} & \text{otherwise} \end{cases}$$

$$C[i, j, V] = \begin{cases} \text{TRUE} & \text{if } \exists j, Y, Z \text{ such that} \\ & V \to Y Z \\ & \text{ and } C[i, k, Y] \\ & \text{ and } C[k, j, Z] \\ & \text{ and } i < k < j \\ \text{FALSE} & \text{otherwise} \end{cases}$$

$$\text{goal} = C[0, n, S]$$

$\mathscr{L}_1$ Grammar	$\mathscr{L}_1$ in CNF
S → NP VP	*S → NP VP*
S → Aux NP VP	*S → X1 VP*
	X1 → Aux NP
S → VP	*S → book | include | prefer*
	S → Verb NP
	S → X2 PP
	S → Verb PP
	S → VP PP
NP → Pronoun	*NP → I | she | me*
NP → Proper-Noun	*NP → TWA | Houston*
NP → Det Nominal	*NP → Det Nominal*
Nominal → Noun	*Nominal → book | flight | meal | money*
Nominal → Nominal Noun	*Nominal → Nominal Noun*
Nominal → Nominal PP	*Nominal → Nominal PP*
VP → Verb	*VP → book | include | prefer*
VP → Verb NP	*VP → Verb NP*
VP → Verb NP PP	*VP → X2 PP*
	X2 → Verb NP
VP → Verb PP	*VP → Verb PP*
VP → VP PP	*VP → VP PP*
PP → Preposition NP	*PP → Preposition NP*

Figure 13.8 $\mathscr{L}_1$ Grammar and its conversion to CNF. Note that although they aren't shown here all the original lexical entries from $\mathscr{L}_1$ carry over unchanged as well.