MIT

Language Modelling for Speech Recognition

- Introduction
- n-gram language models
- Probability estimation
- Evaluation
- Beyond n-grams

Language Modelling for Speech Recognition

- Speech recognizers seek the word sequence $\hat{W}$ which is most likely to be produced from acoustic evidence A

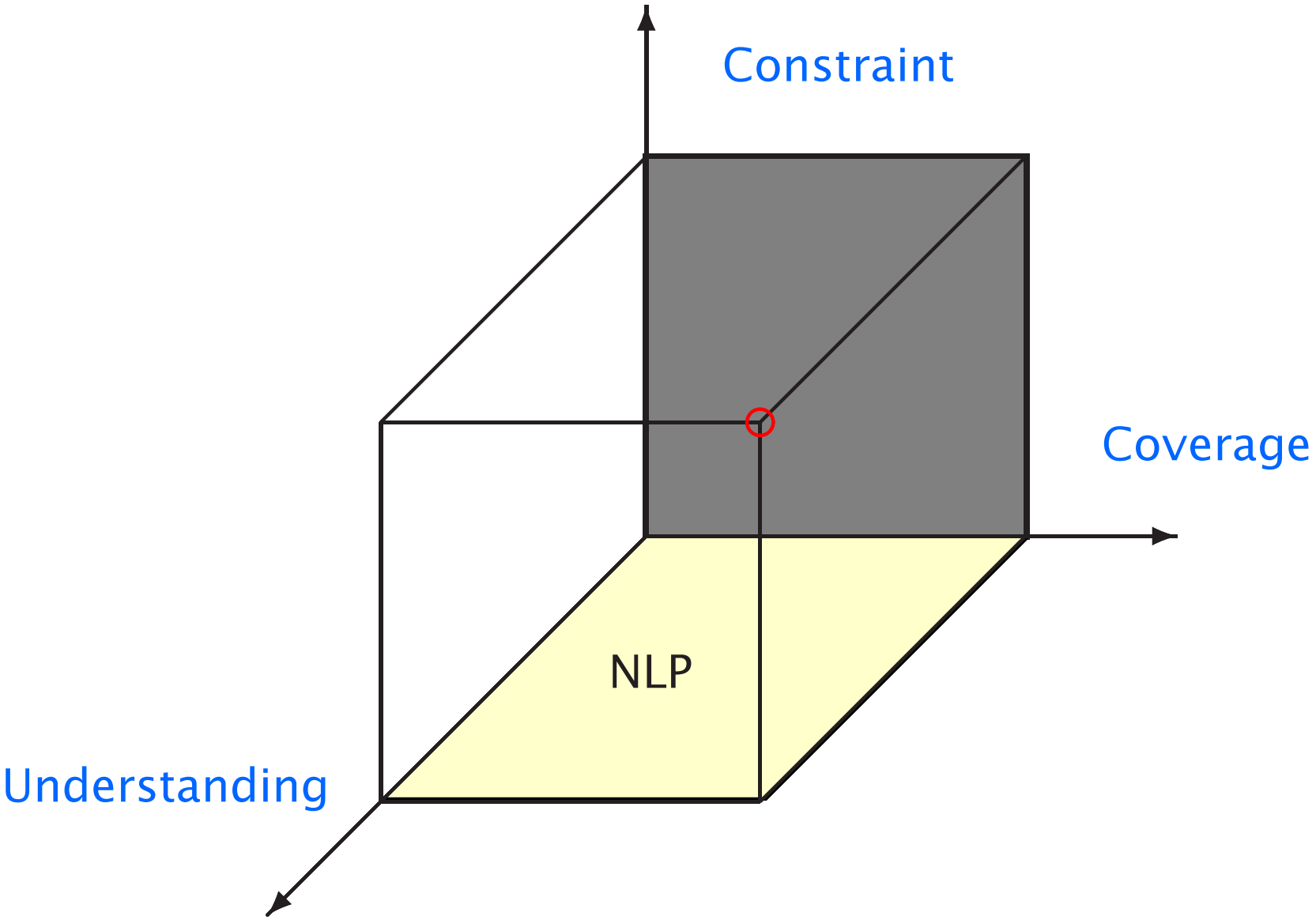
$$P(\hat{W}|A) = \max_W P(W|A) \propto \max_W P(A|W)P(W)$$

- Speech recognition involves acoustic processing, acoustic modelling, language modelling, and search
- Language models (LMs) assign a probability estimate $P(W)$ to word sequences $W = \{w_1, \ldots, w_n\}$ subject to

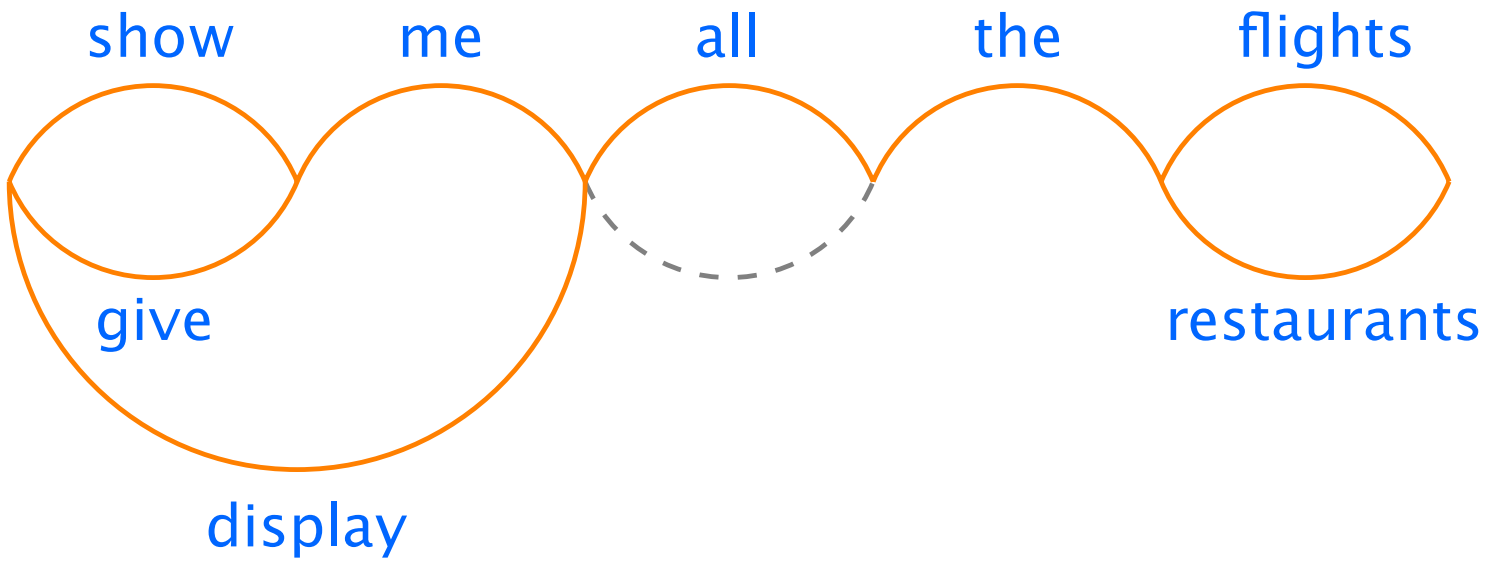
$$\sum_W P(W) = 1$$

- Language models help guide and constrain the search among alternative word hypotheses during recognition

Language Model Requirements



Finite-State Networks (FSN)

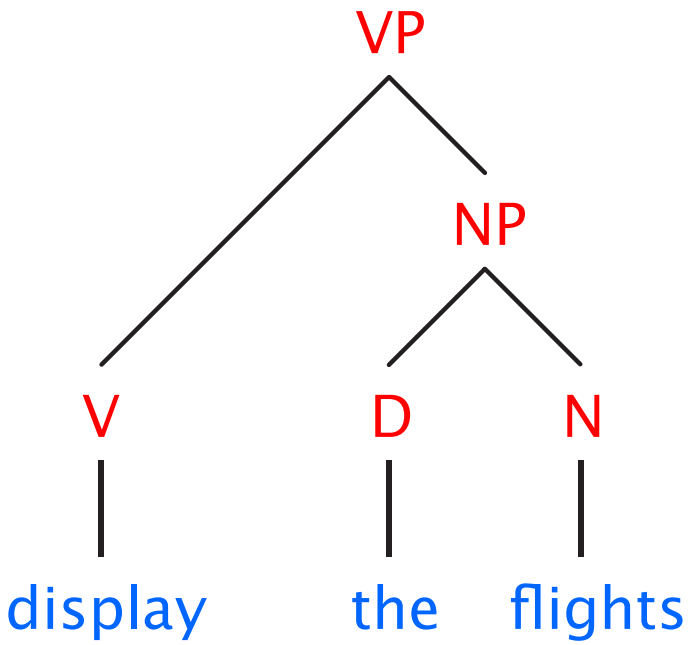


- Language space defined by a word network or graph
- Describable by a regular phrase structure grammar

$$A \Longrightarrow aB \mid a$$

- Finite coverage can present difficulties for ASR
- Graph arcs or rules can be augmented with probabilities

Context-Free Grammars (CFGs)



- Language space defined by context-free rewrite rules

  e.g., $A \Longrightarrow BC \mid a$

- More powerful representation than FSNs
- Stochastic CFG rules have associated probabilities which can be learned automatically from a corpus
- Finite coverage can present difficulties for ASR

Word-Pair Grammars

show → me	me → all → the	the → flights → restaurants

- Language space defined by lists of legal word-pairs
- Can be implemented efficiently within Viterbi search
- Finite coverage can present difficulties for ASR
- Bigrams define probabilities for all word-pairs and can produce a nonzero $P(W)$ for all possible sentences

Example of LM Impact (Lee, 1988)

- Resource Management domain
- Speaker-independent, continuous-speech corpus
- Sentences generated from a finite state network
- 997 word vocabulary
- Word-pair perplexity ~ 60, Bigram ~ 20
- Error includes substitutions, deletions, and insertions

	No LM	Word-Pair	Bigram
% Word Error Rate	29.4	6.3	4.2

LM Formulation for ASR

- Language model probabilities $P(W)$ are usually incorporated into the ASR search as early as possible
- Since most searches are performed unidirectionally, $P(W)$ is usually formulated as a chain rule

$$P(W) = \prod_{i=1}^{n} P(w_i | <>, \ldots, w_{i-1}) = \prod_{i=1}^{n} P(w_i | h_i)$$

  where $h_i = \{<>, \ldots, w_{i-1}\}$ is the word history for w_i

- h_i is often reduced to equivalence classes $\phi(h_i)$

$$P(w_i | h_i) \approx P(w_i | \phi(h_i))$$

  Good equivalence classes maximize the information about the next word w_i given its history $\phi(h_i)$

- Language models which require the full word sequence W are usually used as post-processing filters

n-gram Language Models

- n-gram models use the previous $n-1$ words to represent the history $\phi(h_i) = \{w_{i-1}, \ldots, w_{i-(n-1)}\}$
- Probabilities are based on frequencies and counts

$$\text{e.g.,} \quad f(w_3|w_1 w_2) = \frac{c(w_1 w_2 w_3)}{c(w_1 w_2)}$$

- Due to sparse data problems, n-grams are typically smoothed with lower order frequencies subject to

$$\sum_w P(w|\phi(h_i)) = 1$$

- Bigrams are easily incorporated in Viterbi search
- Trigrams used for large vocabulary recognition in mid-1970's and remain the dominant language model

IBM Trigram Example (Jelinek, 1997)

1	The	are	to	know	the	issues	necessary
2	This	will		have	this	problems	data
3	One	the		understand	these	the	information
4	Two	would		do	problems		above
5	A	also		get	any		other
6	Three	do		the	a		time
7	Please	need		use	problem		people
8	In			provide	them		operators
9	We			insert	all		tools
•				•			•
•				•			•
96				write			jobs
97				me			MVS
98				resolve			old
•							•
•							•
1639							reception
1640							shop
1641							important

IBM Trigram Example (con't)

1	role	and	the	next	be	metting	of
2	thing	from			two	months	<>
3	that	in				years	
4	to	to				meetings	
5	contact	are				to	
6	parts	with				weeks	
7	point	were				days	
8	for	requiring					
9	issues	still					
•		•					
•		•					
61		being					
62		during					
63		I					
64		involved					
65		would					
66		within					

n-gram Issues: Sparse Data (Jelinek, 1985)

- Text corpus of IBM patent descriptions
- 1.5 million words for training
- 300,000 words used to test models
- Vocabulary restricted to 1,000 most frequent words
- 23% of trigrams occurring in test corpus were absent from training corpus!
- In general, a vocabulary of size V will have V^n n-grams (e.g., 20,000 words will have 400 million bigrams, and 8 trillion trigrams!)

n-gram Interpolation

- Probabilities are a linear combination of frequencies

$$P(w_i|h_i) = \sum_j \lambda_j f(w_i|\phi_j(h_i)) \qquad \sum_j \lambda_j = 1$$

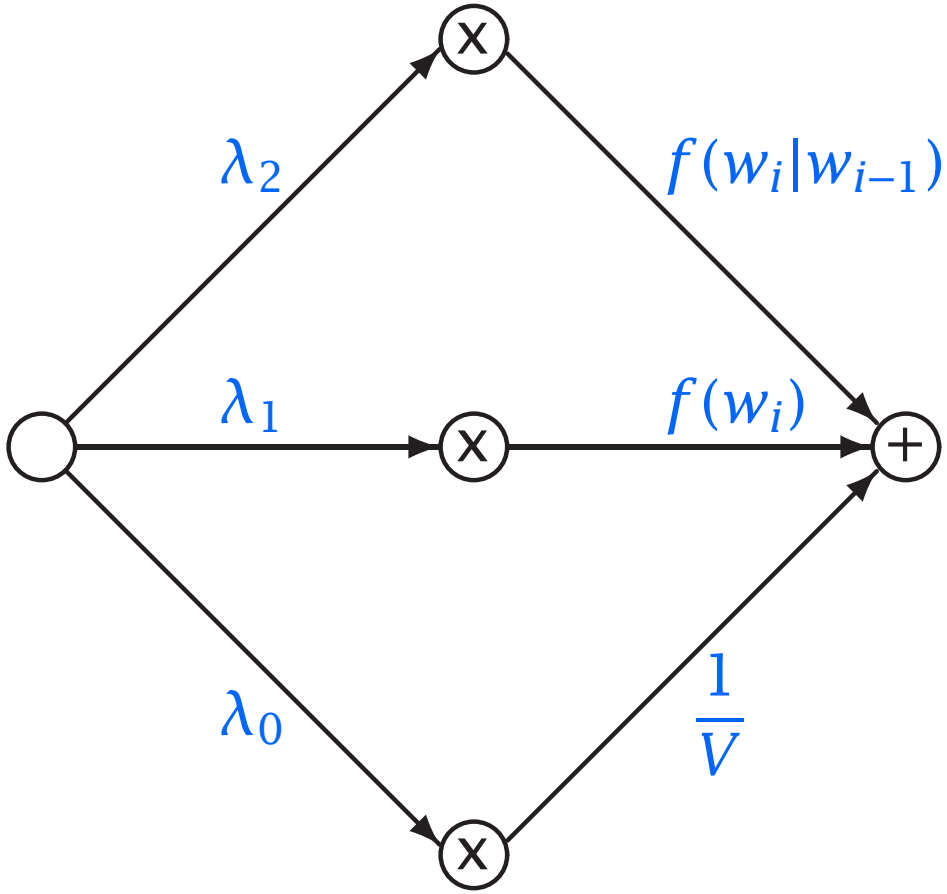
$$\text{e.g.,} \quad P(w_2|w_1) = \lambda_2 f(w_2|w_1) + \lambda_1 f(w_2) + \lambda_0 \frac{1}{V}$$

- λ's computed with EM algorithm on held-out data
- Different λ's can be used for different histories h_i
- Simplistic formulation of λ's can be used $\lambda = \dfrac{c(w_1)}{c(w_1)+k}$
- Estimates can be solved recursively:

$$P(w_3|w_1w_2) = \lambda_3 f(w_3|w_1w_2) + (1-\lambda_3)P(w_3|w_2)$$
$$P(w_3|w_2) = \lambda_2 f(w_3|w_2) + (1-\lambda_2)P(w_3)$$

Interpolation Example

$$P(w_i|w_{i-1}) = \lambda_2 f(w_i|w_{i-1}) + \lambda_1 f(w_i) + \lambda_0 \frac{1}{V}$$



Deleted Interpolation

1. Initialize λ's (e.g., uniform distribution)
2. Compute probability $P(j|w_i)$ that the j^{th} frequency estimate was used when word w_i was generated

$$P(j|w_i) = \frac{\lambda_j f(w_i|\phi_j(h_i))}{P(w_i|h_i)} \qquad P(w_i|h_i) = \sum_j \lambda_j f(w_i|\phi_j(h_i))$$

3. Recompute λ's for n_i words in held-out data

$$\lambda_j = \frac{1}{n_i} \sum_i P(j|w_i)$$

4. Iterate until convergence

Back-Off n-grams (Katz, 1987)

- ML estimates are used when counts are large
- Low count estimates are reduced (discounted) to provide probability mass for unseen sequences
- Zero count estimates based on weighted $(n-1)$-gram
- Discounting typically based on Good-Turing estimate

$$P(w_2|w_1) = \begin{cases} f(w_2|w_1) & c(w_1 w_2) \geq \alpha \\ f_d(w_2|w_1) & \alpha > c(w_1 w_2) > 0 \\ q(w_1)P(w_2) & c(w_1 w_2) = 0 \end{cases}$$

- Factor $q(w_1)$ chosen so that $\sum_{w_2} P(w_2|w_1) = 1$
- High order n-grams computed recursively

Good-Turing Estimate

- Probability a word will occur r times out of N, given θ

$$p_N(r|\theta) = \binom{N}{r} \theta^r (1-\theta)^{N-r}$$

- Probability a word will occur $r+1$ times out of $N+1$

$$p_{N+1}(r+1|\theta) = \frac{N+1}{r+1} \theta p_N(r|\theta)$$

- Assume n_r words occuring r times have same value of θ

$$p_N(r|\theta) \approx \frac{n_r}{N} \qquad p_{N+1}(r+1|\theta) \approx \frac{n_{r+1}}{N}$$

- Assuming large N, we can solve for θ or discounted r^*

$$\theta = P_r = \frac{r^*}{N} \qquad r^* = (r+1)\frac{n_{r+1}}{n_r}$$

Good-Turing Example (Church and Gale, 1991)

- GT estimate for an item occurring r times out of N is

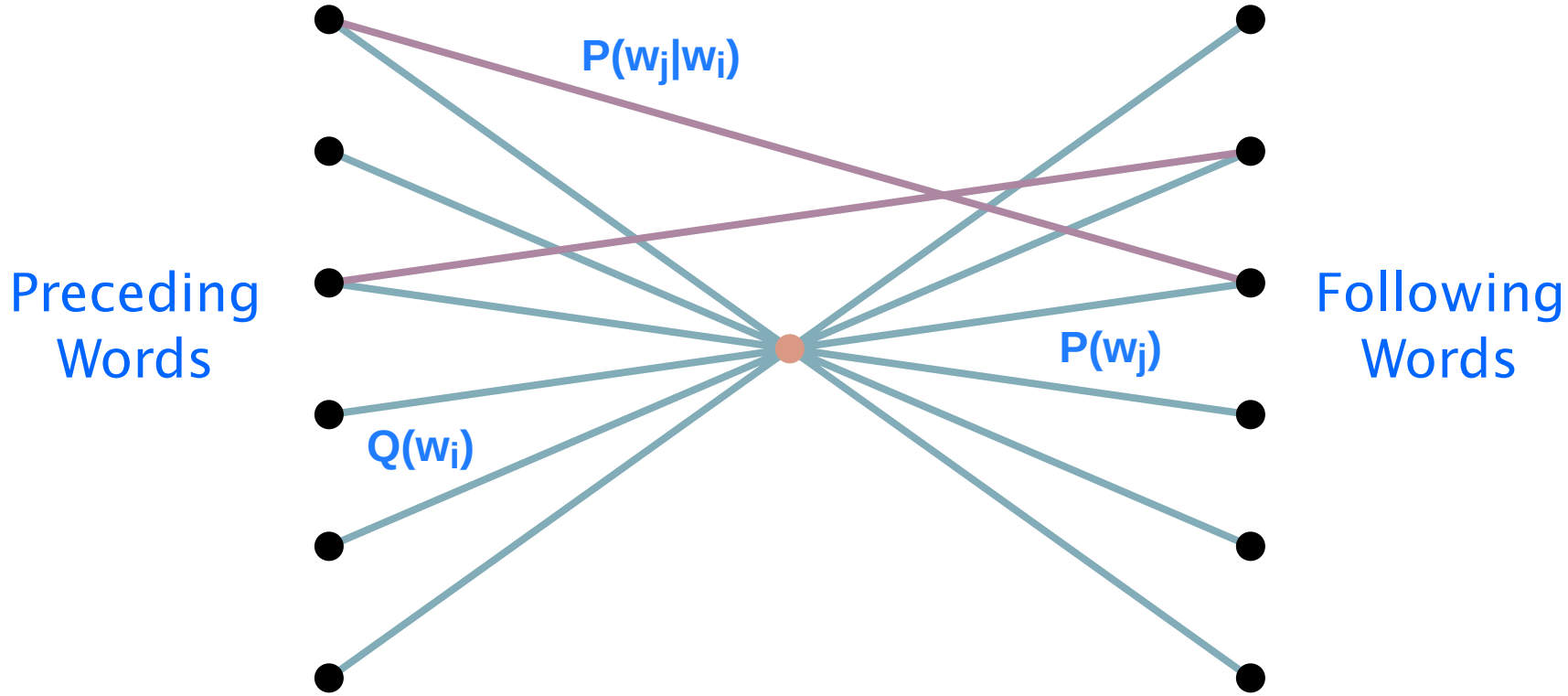
$$P_r = \frac{r^*}{N} \qquad r^* = (r+1)\frac{n_{r+1}}{n_r}$$

  where n_r is the number of items occurring r times

- Consider bigram counts from a 22 million word corpus of AP news articles (273,000 word vocabulary)

r	n_r	r^*
0	74,671,100,000	0.0000270
1	2,018,046	0.446
2	449,721	1.26
3	188,933	2.24
4	105,668	3.24
5	68,379	4.22

Integration into Viterbi Search



Bigrams can be efficiently incorporated into Viterbi search using an intermediate node between words

- Interpolated: $Q(w_i) = (1 - \lambda_i)$
- Back-off: $Q(w_i) = q(w_i)$

Evaluating Language Models

- Recognition accuracy
- Qualitative assessment
  - Random sentence generation
  - Sentence reordering
- Information-theoretic measures

Random Sentence Generation: Air Travel Domain Bigram

Show me the flight earliest flight from Denver
How many flights that flight leaves around is the Eastern Denver
I want a first class
Show me a reservation the last flight from Baltimore for the first
I would like to fly from Dallas
I get from Pittsburgh
Which just small
In Denver on October
I would like to San Francisco
Is flight flying
What flights from Boston to San Francisco
How long can you book a hundred dollars
I would like to Denver to Boston and Boston
Make ground transportation is the cheapest
Are the next week on AA eleven ten
First class
How many airlines from Boston on May thirtieth
What is the city of three PM
What about twelve and Baltimore

Random Sentence Generation: Air Travel Domain Trigram

What type of aircraft
What is the fare on flight two seventy two
Show me the flights I've Boston to San Francisco on Monday
What is the cheapest one way
Okay on flight number seven thirty six
What airline leaves earliest
Which airlines from Philadelphia to Dallas
I'd like to leave at nine eight
What airline
How much does it cost
How many stops does Delta flight five eleven o'clock PM that go from
What AM
Is Eastern from Denver before noon
Earliest flight from Dallas
I need to Philadelphia
Describe to Baltimore on Wednesday from Boston
I'd like to depart before five o'clock PM
Which flights do these flights leave after four PM and lunch and <unknown>

Sentence Reordering (Jelinek, 1991)

- Scramble words of a sentence
- Find most probable order with language model
- Results with trigram LM
  - Short sentences from spontaneous dictation
  - 63% of reordered sentences identical
  - 86% have same meaning

IBM Sentence Reordering

would I report directly to you
I would report directly to you

now let me mention some of the disadvantages
let me mention some of the disadvantages now

he did this several hours later
this he did several hours later

this is of course of interest to IBM
of course this is of interest to IBM

approximately seven years I have known John
I have known John approximately seven years

these people have a fairly large rate of turnover
of these people have a fairly large turnover rate

in our organization research has two missions
in our missions research organization has two

exactly how this might be done is not clear
clear is not exactly how this might be done

Quantifying LM Complexity

- One LM is better than another if it can predict an n word test corpus W with a higher probability $\hat{P}(W)$
- For LMs representable by the chain rule, comparisons are usually based on the average per word logprob, LP

$$LP = -\frac{1}{n}\log_2 \hat{P}(W) = -\frac{1}{n}\sum_i \log_2 \hat{P}(w_i|\phi(h_i))$$

- A more intuitive representation of LP is the perplexity

$$PP = 2^{LP}$$

  (a uniform LM will have PP equal to vocabulary size)

- PP is often interpreted as an average branching factor

Perplexity Examples

Domain	Size	Type	Perplexity
Digits	11	All word	11
Resource Management	1,000	Word-pair	60
		Bigram	20
Air Travel Understanding	2,500	Bigram	29
		4-gram	22
WSJ Dictation	5,000	Bigram	80
		Trigram	45
	20,000	Bigram	190
		Trigram	120
Switchboard Human-Human	23,000	Bigram	109
		Trigram	93
NYT Characters	63	Unigram	20
		Bigram	11
Shannon Letters	27	Human	~ 2

Language Entropy

- The average logprob LP is related to the overall uncertainty of the language, quantified by its entropy

$$H = -\lim_{n\to\infty} \frac{1}{n} \sum_{W} P(W) \log_2 P(W)$$

- If W is obtained from a well-behaved source (ergodic), $P(W)$ will converge to the expected value and H is

$$H = -\lim_{n\to\infty} \frac{1}{n} \log_2 P(W) \approx -\frac{1}{n} \log_2 P(W) \quad n >> 1$$

- The entropy H is a theoretical lower bound on LP

$$-\lim_{n\to\infty} \frac{1}{n} \sum_{W} P(W) \log_2 P(W) \leq -\lim_{n\to\infty} \frac{1}{n} \sum_{W} P(W) \log_2 \hat{P}(W)$$

Human Language Entropy (Shannon, 1951)

- An attempt to estimate language entropy of humans
- Involved guessing next words in order to measure subjects probability distribution
- Letters were used to simplify experiments

T	H	E	R	E		I	S		N	O		R	E	V	E	R	S	E
1	1	1	5	1	1	2	1	1	2	1	1	15	1	17	1	1	1	2
	O	N		A		M	O	T	O	R	C	Y	C	L	E		A	...
1	3	2	1	2	2	7	1	1	1	1	4	1	1	1	1	1	3	...

- $\hat{H} = -\sum \hat{P}(i) \log_2 \hat{P}(i)$ $\quad \hat{P}(1) = \frac{24}{37}$ $\quad \hat{P}(2) = \frac{6}{37}$ $\quad \hat{P}(3) = \frac{2}{37}$
- Shannon estimated $\hat{H} \approx 1$ bit/letter

Why do n-grams work so well?

- Probabilities are based on data (the more the better)
- Parameters determined automatically from corpora
- Incorporate local syntax, semantics, and pragmatics
- Many languages have a strong tendency toward standard word order and are thus substantially local
- Relatively easy to integrate into forward search methods such as Viterbi (bigram) or A^*

Problems with n-grams

- Unable to incorporate long-distance constraints
- Not well suited for flexible word order languages
- Cannot easily accommodate
  - New vocabulary items
  - Alternative domains
  - Dynamic changes (e.g., discourse)
- Not as good as humans at tasks of
  - Identifying and correcting recognizer errors
  - Predicting following words (or letters)
- Do not capture meaning for speech understanding

Clustering words

- Many words have similar statistical behavior
  - e.g., days of the week, months, cities, etc.
- n-gram performance can be improved by clustering words
  - Hard clustering puts a word into a single cluster
  - Soft clustering allows a word to belong to multiple clusters
- Clusters can be created manually, or automatically
  - Manually created clusters have worked well for small domains
  - Automatic clusters have been created bottom-up or top-down

Bottom-Up Word Clustering (Brown et al., 1992)

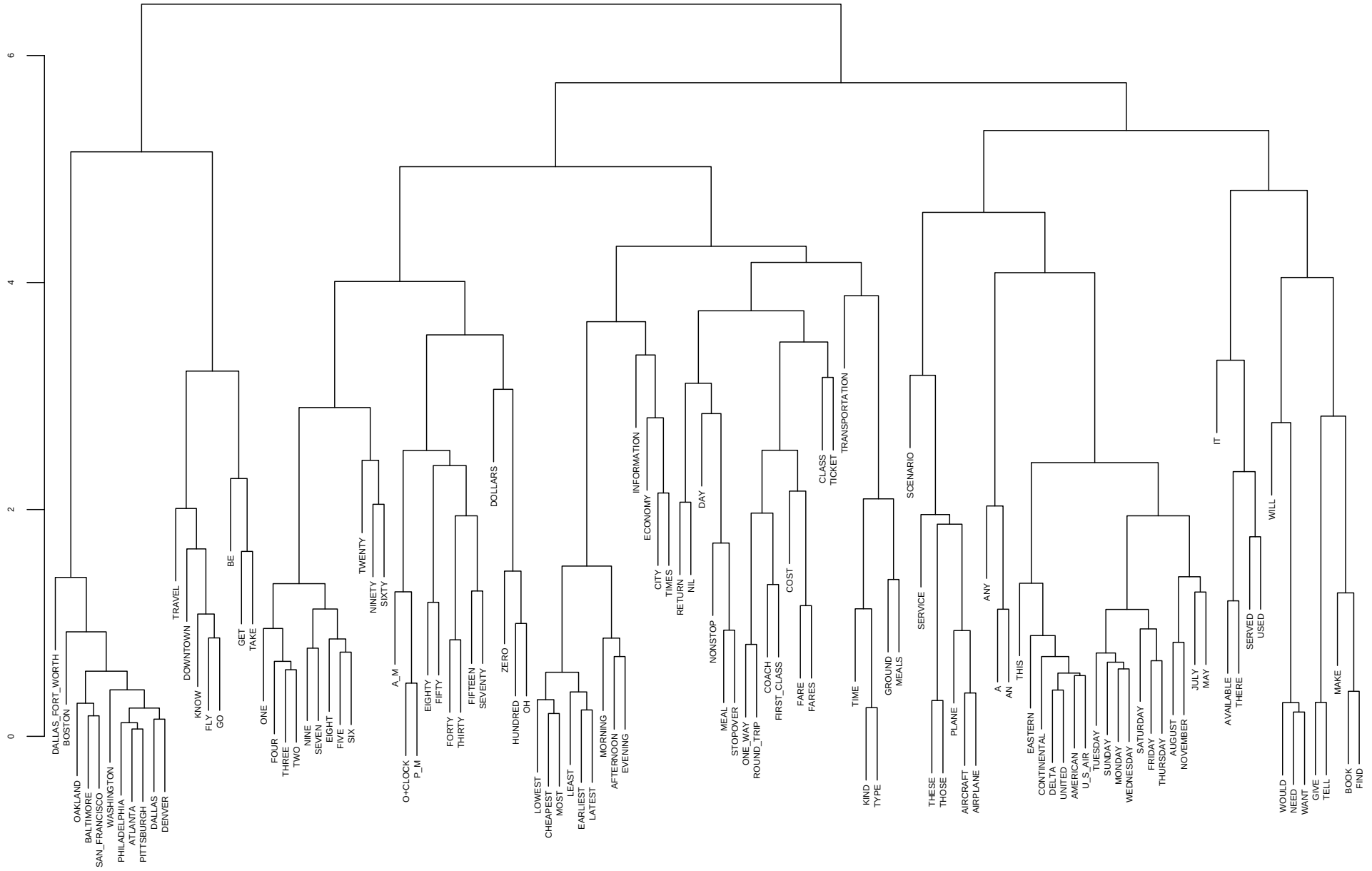
- Word clusters can be created automatically by forming clusters in a stepwise-optimal or greedy fashion
- Bottom-up clusters created by considering impact on metric of merging words w_a and w_b to form new cluster w_{ab}
- Example metrics for a bigram language model:
  - Minimum decrease in average mutual information

$$I = \sum_{i,j} P(w_i w_j) \log_2 \frac{P(w_j|w_i)}{P(w_j)}$$

  - Minimum increase in training set conditional entropy

$$H = -\sum_{i,j} P(w_i w_j) \log_2 P(w_j|w_i)$$

Example of Word Clustering



Word Class n-gram models

- Word class n-grams cluster words into equivalence classes

$$W = \{w_1, \ldots, w_n\} \rightarrow \{c_1, \ldots, c_n\}$$

- If clusters are non-overlapping, $P(W)$ is approximated by

$$P(W) \approx \prod_{i=1}^{n} P(w_i|c_i)P(c_i| <>, \ldots, c_{i-1})$$

- Fewer parameters than word n-grams
- Relatively easy to add new words to existing clusters
- Can be linearly combined with word n-grams if desired

Predictive Clustering (Goodman, 2000)

- For word class n-grams : $P(w_i|h_i) \approx P(w_i|c_i)P(c_i|c_{i-1}\ldots)$
- Predictive clustering is exact: $P(w_i|h_i) = P(w_i|h_i c_i)P(c_i|h_i)$
- History, h_i, can be clustered differently for the two terms
- This model can be larger than the n-gram , but has been shown to produce good results when combined with pruning

Phrase Class n-grams (PCNG) (McCandless, 1994)

- Probabilistic context-free rules parse phrases

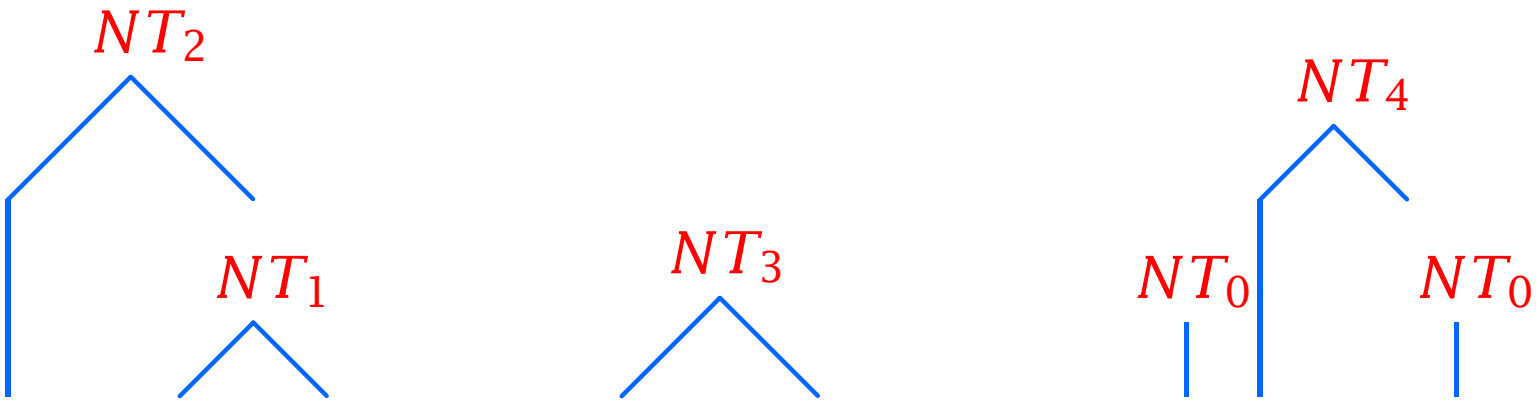
$$W = \{w_1, \ldots, w_n\} \rightarrow \{u_1, \ldots, u_m\}$$

- n-gram produces probability of resulting units
- $P(W)$ is product of parsing and n-gram probabilities

$$P(W) = P_r(W)P_n(U)$$

- Intermediate representation between word-based n-grams and stochastic context-free grammars
- Context-free rules can be learned automatically

PCNG Example



Please show me the cheapest flight from Boston to Denver

NT_2 the NT_3 from NT_0 NT_4

PCNG Experiments

- Air-Travel Information Service (ATIS) domain
- Spontaneous, spoken language understanding
- 21,000 train, 2,500 development, 2,500 test sentences
- 1,956 word vocabulary

Language Model	# Rules	# Params	Perplexity
Word Bigram	0	18430	21.87
+ Compound Words	654	20539	20.23
+ Word Classes	1440	16430	19.93
+ Phrases	2165	16739	15.87
PCNG Trigram	2165	38232	14.53
PCNG 4-gram	2165	51012	14.40

Decision Tree Language Models (Bahl et al., 1989)

- Equivalence classes represented in a decision tree
  - Branch nodes contain questions for history h_i
  - Leaf nodes contain equivalence classes
- Word n-gram formulation fits decision tree model
- Minimum entropy criterion used for construction
- Significant computation required to produce trees

Exponential Language Models

- $P(w_i|h_i)$ modelled as product of weighted features $f_j(w_i h_i)$

$$P(w_i|h_i) = \frac{1}{Z(h_i)} e^{\sum_j \lambda_j f_j(w_i h_i)}$$

  where λ's are parameters, and $Z(h_i)$ is a normalization factor

- Binary-valued features can express arbitrary relationships

$$\text{e.g., } f_j(w_i h_i) = \begin{cases} 1 & w_i = A \ \& \ w_{i-1} = B \\ 0 & else \end{cases}$$

- When $E(f(wh))$ corresponds to empirical expected value, ML estimates for λ's correspond to maximum entropy distribution
- ML solutions are iterative, and can be extremely slow
- Demonstrated perplexity and WER gains on large vocabulary tasks

Adaptive Language Models

- Cache-based language models incorporate statistics of recently used words with a static language model

$$P(w_i|h_i) = \lambda P_c(w_i|h_i) + (1-\lambda)P_s(w_i|h_i)$$

- Trigger-based language models increase word probabilities when key words observed in history h_i
  - Self triggers provide significant information
  - Information metrics used to find triggers
  - Incorporated into maximum entropy formulation

Trigger Examples (Lau, 1994)

- Triggers determined automatically from WSJ corpus (37 million words) using average mutual information
- Top seven triggers per word used in language model

Word	Triggers
stocks	stocks index investors market dow average industrial
political	political party presidential politics election president campaign
foreign	currency dollar japanese domestic exchange japan trade
bonds	bonds bond yield treasury municipal treasury's yields

Language Model Pruning

- n-gram language models can get very large (e.g., 6B/n-gram)
- Simple techniques can reduce parameter size
  - Prune n-grams with too few occurrences
  - Prune n-grams that have small impact on model entropy
- Trigram count-based pruning example:
  - Broadcast news transcription (e.g., TV, radio broadcasts)
  - 25K vocabulary; 166M training words ($\sim 1GB$), 25K test words

Count	Bigrams	Trigrams	States	Arcs	Size	Perplexity
0	$6.4M$	$35.1M$	$6.4M$	$48M$	$360MB$	157.4
1	$3.2M$	$11.4M$	$2.2M$	$17M$	$125MB$	169.4
2	$2.2M$	$6.3M$	$1.2M$	$10M$	$72MB$	178.1
3	$1.7M$	$4.4M$	$0.9M$	$7M$	$52MB$	185.1
4	$1.4M$	$3.4M$	$0.7M$	$5M$	$41MB$	191.9

Entropy-based Pruning (Stolcke, 1998)

- Uses KL distance to prune n-grams with low impact on entropy

$$D(P \parallel P') = \sum_{i,j} P(w_i|h_j) \log \frac{P(w_i|h_j)}{P'(w_i|h_j)} \qquad \frac{PP' - PP}{PP} = e^{D(P\|P')} - 1$$

1. Select pruning threshold θ
2. Compute perplexity increase from pruning each n-gram
3. Remove n-grams below θ, and recompute backoff weights

- Example: resorting Broadcast News N-best lists with 4-grams

θ	Bigrams	Trigrams	4-grams	Perplexity	% WER
0	11.1M	14.9M	0	172.5	32.9
0	11.1M	14.9M	3.3M	163.0	32.6
10^{-9}	7.8M	9.6M	1.9M	163.9	32.6
10^{-8}	3.2M	3.7M	0.7M	172.3	32.6
10^{-7}	0.8M	0.5M	0.1M	202.3	33.9

Perplexity vs. Error Rate (Rosenfeld et al., 1995)

- Switchboard human-human telephone conversations
- 2.1 million words for training, 10,000 words for testing
- 23,000 word vocabulary, bigram perplexity of 109
- Bigram-generated word-lattice search (10% word error)

Trigram Condition	Perplexity	% Word Error
Trained on Train Set	92.8	49.5
Trained on Train & Test Set	30.4	38.7
Trained on Test Set	17.9	32.9
No Parameter Smoothing	3.2	31.0
Perfect Lattice	3.2	6.3
Other Lattice	3.2	44.5

References

- X. Huang, A. Acero, and H. -W. Hon, *Spoken Language Processing*, Prentice-Hall, 2001.
- K. Church & W. Gale, A Comparison of the Enhanced Good-Turing and Deleted Estimation Methods for Estimating Probabilities of English Bigrams, *Computer Speech & Language*, 1991.
- F. Jelinek, *Statistical Methods for Speech Recognition,* MIT Press, 1997.
- S. Katz, Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer. *IEEE Trans. ASSP-35*, 1987.
- K. F. Lee, *The CMU SPHINX System,* Ph.D. Thesis, CMU, 1988.
- R. Rosenfeld, Two Decades of Statistical Language Modeling: Where Do We Go from Here?, *IEEE Proceedings,* 88(8), 2000.
- C. Shannon, Prediction and Entropy of Printed English, *BSTJ*, 1951.

More References

- L. Bahl et al., A Tree-Based Statistical Language Model for Natural Language Speech Recognition, *IEEE Trans. ASSP-37*, 1989.
- P. Brown et al., Class-based *n*-gram models of natural language, *Computational Linguistics*, 1992.
- R. Lau, Adaptive Statistical Language Modelling, S.M. Thesis, MIT, 1994.
- M. McCandless, Automatic Acquisition of Language Models for Speech Recognition, S.M. Thesis, MIT, 1994.
- R. Rosenfeld et al., Language Modelling for Spontaneous Speech, Johns Hopkins Workshop, 1995.
- A. Stolcke, Entropy-based Pruning of Backoff Language Models, http://www.nist.gov/speech/publications/darpa98/html/lm20/lm20.htm, 1998.