Key Points

- Maximum Entropy (MaxEnt) models provide flexible alternative to standard n-gram multinominal language models.
  - Feature-based model, brute force normalization over $\sum$
  - Feature definitions critical to approach
  - Want expected feature frequency = actual feature frequency
- Marginal distribution also matched in Kneser-Ney smoothing
  - Widely used method for n-gram language modeling.
- Instructive to compare features and optimization methods between MaxEnt and n-gram models.
• N-gram (Markov) language models and automata encoding

• Maximum Entropy (MaxEnt) language modeling
  – Features versus standard n-gram models
  – Parameter optimization

• “Backoff” features in MaxEnt language models
  – Biadsy et al., Interspeech 2014

• Kneser-Ney language models and marginalization constraints

• Imposing marginalization constraints on smoothed models
  – Roark et al., ACL 2013
Language models for Speech Recognition

• Assuming some familiarity with role of language models in automatic speech recognition (ASR)
  – Language model score combines with acoustic score as a prior

• Standard language modeling approach:
  – Closed vocabulary $\Sigma$, multinomial model over $w \in \Sigma$
  – Multinomial based on history of preceding words
  – Equivalence classes of histories for parameter estimation
  – Equivalence defined by history suffix (Markov assumption)
  – Regularization relies on link to history longest proper suffix
N-gram modeling

- Today, talking about backoff models, typically recursively defined:

\[
P(w \mid h) = \begin{cases} 
\tilde{P}(w \mid h) & \text{if } c(hw) > 0 \\
\alpha(h) \ P(w \mid h') & \text{otherwise}
\end{cases}
\]

where \( w = w_i, h = w_{i-k} \ldots w_{i-1}, h' = w_{i-k+1} \ldots w_{i-1} \) and s.t. \( 0 \leq \sum_{w : c(hw) > 0} \tilde{P}(w \mid h) \leq 1 \) and \( \alpha(h) \) ensures normalization

- Many ways to estimate \( \tilde{P} \): Katz, Absolute Discounting (AD), etc.

- Kneser-Ney is a variant of AD that modifies lower order parameters to preserve observed marginal distributions
Fragment of n-gram automata representation

(real semiring)
Log linear modeling

- A general stochastic modeling approach that does not assume a particular WFSA structure
  - WFSA structure can be imposed for a particular feature set, e.g., n-grams

- Flexibility of modeling allows for many overlapping features
  - e.g., use unigram, bigram and trigram simultaneously
  - or use POS-tags or morphologically-derived features

- Generative or discriminative training can be used

- Commonly used models for other sequence processing problems
MaxEnt language modeling

- Define a $d$-dimensional vector of features $\Phi$
  
  e.g. $\Phi_{1000}(h_iw_i) = 1$ if $w_{i-1}$ is to and $w_i$ is the, 0 otherwise

- Estimate a $d$-dimensional parameter vector $\Theta$

- Then

$$P_{\Theta}(w_i|h_i) = \frac{\exp(\Theta \cdot \Phi(h_iw_i))}{\sum_{w \in \Sigma} \exp(\Theta \cdot \Phi(h_iw))} = \frac{\exp \left( \sum_{s=1}^{d} \Theta_s \Phi_s(h_iw_i) \right)}{\sum_{w \in \Sigma} \exp \left( \sum_{s=1}^{d} \Theta_s \Phi_s(h_iw) \right)}$$

$$\log P_{\Theta}(w_i|h_i) = \Theta \cdot \Phi(h_iw_i) - \log \sum_{w \in \Sigma} \exp(\Theta \cdot \Phi(h_iw))$$
Finding optimal model parameters $\Theta$

- Train the model by maximizing the likelihood of training corpus
- Given a corpus $C$ of $N$ observations, find $\Theta$ that maximizes:

\[
LL(\Theta; C) = \sum_{w_i \in C} \log P_{\Theta}(w_i|h_i)
\]

\[
= \sum_{w_i \in C} \left( \Theta \cdot \Phi(h_iw_i) - \log \sum_{w \in \Sigma} \exp(\Theta \cdot \Phi(h_iw)) \right)
\]

\[
= \sum_{w_i \in C} \Theta \cdot \Phi(h_iw_i) - \sum_{w_i \in C} \log \sum_{w \in \Sigma} \exp(\Theta \cdot \Phi(h_iw))
\]

- Maximum likelihood $\Theta$ where the derivative of $LL(\Theta; C)$ is zero
Partial derivative wrt. a particular dimension

\[ LL(\Theta; C) = \sum_{w_i \in C} \Theta \cdot \Phi(h_i w_i) - \sum_{w_i \in C} \log \sum_{w \in \Sigma} \exp(\Theta \cdot \Phi(h_i w)) \]

- Partial derivative wrt. dimension \( j \): all other dimensions constant

- Useful facts about derivatives:
  \[ \frac{d}{dx} \log f(x) = \frac{1}{f(x)} \frac{d}{dx} f(x) \quad \text{and} \quad \frac{d}{dx} \exp f(x) = \exp f(x) \frac{d}{dx} f(x) \]

- Then:
  \[ \frac{\partial LL(\Theta; C)}{\partial \Theta_j} = \sum_{w_i \in C} \Phi_j(h_i w_i) - \sum_{w_i \in C} \sum_{w \in \Sigma} P_{\Theta}(w|h_i) \Phi_j(h_i w) \]
Expected frequency vs. actual frequency

- Maximum likelihood $\Theta$ has derivative zero, i.e., for all $j$

$$\sum_{w_i \in C} \Phi_j(h_i w_i) = \sum_{w_i \in C} \sum_{w \in \Sigma} P_{\Theta}(w|h_i) \Phi_j(h_i w)$$

Observed frequency of $\Phi_j$ = Expected frequency of $\Phi_j$

- Training algorithm: calculate gradients; update parameters; iterate
  - Objective function is convex, hence global maximum exists
  - Typical to regularize by penalizing parameter magnitude
- Summing over whole vocabulary at each point is expensive
  - e.g., many current ASR systems have 1M+ vocabularies
Why use MaxEnt models

- The approach has extreme flexibility for defining features
  - Multiple, overlapping, non-independent features
  - Features dealing with words, word classes, morphemes, etc.
  - For languages with highly productive morphology (Turkish, Russian, Korean, etc.), hard to capture constraints in ngram model.
    - “Feature engineering” becomes a key enterprise.

- No need for explicit smoothing
  - other than objective regularization mentioned above
Suppose you want to simply replace n-gram model with MaxEnt
  – Same feature space, different parameterizations of features
  – What is the feature space? n-grams... what else?
MaxEnt not required to explicitly smooth to backoff history
  – Has its own regularization, hence typically omit backoff
  – However, backoff style features can improve models
Paper also explores issues in scaling to large models
Backoff inspired features (Biadsy et al., 2014)

- Introduce some notation for n-gram features: Let
  \[ \text{Ngram}(w_1 \ldots w_n, i, k) = \langle w_{i-k}, \ldots w_{i-1}, w_i \rangle \]
  be a binary indicator feature for the \( k \)-order n-gram ending at position \( i \).

- A backoff feature ‘fires’ when the full ngram is not observed
  - In an n-gram model, each history has a backoff cost. Let
    \[ \text{SuffixBackoff}(w_1 \ldots w_n, i, k) = \langle w_{i-k}, \ldots w_{i-1}, \text{BO} \rangle \]
    be a backoff feature that fires iff \( w_{i-k} \ldots w_i \) is unobserved
  - What does ‘unobserved’ mean in this case?
    * Construct a feature ‘dictionary’ of n-grams from corpus
    * Only include in dictionary if frequency higher than threshold
Generalizing backoff features

• We can also define a ’prefix’ backoff feature analogously. Let
  \[ \text{PrefixBackoff}(w_1 \ldots w_n, i, k) = <\text{BO}, w_{i-k+1}, \ldots w_i> \]
  be a backoff feature that fires iff \( w_{i-k} \ldots w_i \) is unobserved

• Can also ‘forget’ more than just one word on either end:
  
  – Let \( \text{PrefixBackoff}_j(w_1 \ldots w_n, i, k) = <\text{BO}_k, w_{i-j}, \ldots w_i> \)
    indicate an unobserved \( k \)-order ngram with suffix \( w_{i-j} \ldots w_i \)
  
  – Similarly, let
    \[ \text{SuffixBackoff}_j(w_1 \ldots w_n, i, k) = <w_{i-k}, \ldots w_{i-k+j}, \text{BO}_k> \]
    indicate an unobserved \( k \)-order ngram with prefix \( w_{i-k} \ldots w_{i-k+j} \)
Example features

Let sentence $S = \text{“we will save the quail eggs”}$. Then:

- $\text{SuffixBackoff}(S, 5, 3) = <\text{will, save, the, BO}>$
- $\text{PrefixBackoff}(S, 5, 3) = <\text{BO, save, the, quail}>$
- $\text{SuffixBackoff}_0(S, 5, 3) = <\text{will, BO}_3>$
- $\text{SuffixBackoff}_1(S, 5, 3) = <\text{will, save, BO}_3>$
- $\text{PrefixBackoff}_0(S, 5, 3) = <\text{BO}_3, \text{quail}>$
- $\text{PrefixBackoff}_1(S, 5, 3) = <\text{BO}_3, \text{the, quail}>
Experimental setup

- Evaluation on American English speech recognition
  - See paper for full details on system and setup
- MaxEnt n-grams up to 5-grams and BO features up to 4 words
  - Baseline LM is also a 5-gram model (Katz backoff)
- N-best reranking experiments on large scale ASR system at Google
  - Mixture of MaxEnt model with baseline LM at fixed mixture
- Perplexity results (see paper) in a variety of conditions
- Word error rate on multiple anonymized voice-search data sets
  - including spoken search queries, questions and YouTube queries
Table 2 in Biadsy et al. (2014)

Table 2: WER results on 7 sub-corpora and overall, for the baseline recognizer (no reranking) versus reranking models trained with different feature sets.

<table>
<thead>
<tr>
<th>Test Set</th>
<th>Utts / Wds (×1000)</th>
<th>None</th>
<th>NG</th>
<th>NG+ Pk</th>
<th>NG+ Sk</th>
<th>NG+ Pk+Sk</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>22.5 / 98.0</td>
<td>12.7</td>
<td>12.6</td>
<td>12.4</td>
<td>12.4</td>
<td>12.4</td>
</tr>
<tr>
<td>2</td>
<td>17.8 / 74.0</td>
<td>12.7</td>
<td>12.5</td>
<td>12.4</td>
<td>12.4</td>
<td>12.3</td>
</tr>
<tr>
<td>3</td>
<td>16.2 / 61.1</td>
<td>17.3</td>
<td>17.1</td>
<td>16.7</td>
<td>16.8</td>
<td>16.7</td>
</tr>
<tr>
<td>4</td>
<td>18.0 / 64.0</td>
<td>12.8</td>
<td>12.7</td>
<td>12.6</td>
<td>12.6</td>
<td>12.5</td>
</tr>
<tr>
<td>5</td>
<td>7.4 / 50.7</td>
<td>16.8</td>
<td>16.6</td>
<td>16.2</td>
<td>16.2</td>
<td>16.2</td>
</tr>
<tr>
<td>6</td>
<td>7.3 / 31.9</td>
<td>15.1</td>
<td>15.0</td>
<td>14.8</td>
<td>14.8</td>
<td>14.9</td>
</tr>
<tr>
<td>7</td>
<td>19.6 / 69.1</td>
<td>16.5</td>
<td>16.2</td>
<td>15.9</td>
<td>15.9</td>
<td>15.9</td>
</tr>
<tr>
<td>all</td>
<td>108.9 / 448.8</td>
<td>14.6</td>
<td>14.4</td>
<td>14.2</td>
<td>14.2</td>
<td>14.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Active Features (M):</td>
<td>197.8</td>
<td>170.1</td>
<td>160.2</td>
<td>162.4</td>
<td></td>
</tr>
</tbody>
</table>
Transition back to n-gram models

- Recall backoff models, typically recursively defined:

\[
P(w \mid h) = \begin{cases} 
\tilde{P}(w \mid h) & \text{if } c(hw) > 0 \\
\alpha(h) P(w \mid h') & \text{otherwise}
\end{cases}
\]

where \( w = w_i \), \( h = w_{i-k} \ldots w_{i-1} \), \( h' = w_{i-k+1} \ldots w_{i-1} \) and

\[
s.t. \ 0 \leq \sum_{w: c(hw) > 0} \tilde{P}(w \mid h) \leq 1 \quad \text{and} \quad \alpha(h) \text{ ensures normalization}
\]

- Many ways to estimate \( \tilde{P} \): Katz, Absolute Discounting (AD), etc.

- Kneser-Ney is a variant of AD that modifies lower order parameters to preserve observed marginal distributions

- Goodman (2001) proved this improves language models in general
Kneser-Ney estimation

- This technique modifies only lower order distributions
  - Improve their utility within smoothing
- Intuition: **Royce** may occur frequently in a corpus but it is unlikely to follow anything but **Rolls**
  - Hence unigram probability should be small within bigram model
- A variant of absolute discounting
  - Highest order n-gram probs left the same
    \[
    \tilde{P}_{abs}(w_i|h) = \frac{\tilde{C}(hw_i)}{C(h)} + (1 - \lambda_h)\tilde{P}_{kn}(w_i|h')
    \]
Intuition of Kneser-Ney benefit

• Build a language model automaton, randomly generate text

• Suppose we have a bigram model
  – Want randomly generated unigrams to occur about as frequently as they were observed in the corpus we trained on

• If we use relative frequency for unigram distribution, Royce will be generated too frequently in random car corpus
  – Generated with high frequency following Rolls
  – Unigram prob. high, hence also generated in other contexts

• Kneser-Ney constrains lower order models so that lower-order n-grams are not overgenerated in random corpus
Marginal distribution constraints

- Such constraints widely used to improve models
  - To train MaxEnt models based on feature expected frequency
  - Kneser-Ney models match lower order n-gram expected freq.
- Standard n-gram models will over-generate certain n-grams
  - Because the same observations train each order of n-gram
  - Hence probability mass duplicated at each level of model
- Marginal distribution constraints remove n-gram probability mass already accounted for in higher-order distributions
  - Kneser-Ney has a convenient closed-form solution
Kneser-Ney approach

- Estimation method for lower orders of n-gram model
- Thus, if $h'$ is the backoff history for $h$

$$P(w \mid h') = \sum_{g : g' = h'} P(g, w \mid h')$$
Kneser-Ney approach

• Estimation method for lower orders of n-gram model

• Thus, if $h'$ is the backoff history for $h$

$$P(w | h') = \sum_{g:g'=h'} P(g, w | h')$$

$$= \sum_{g:g'=h', c(gw)>0} \tilde{P}(w | g) P(g | h') + \sum_{g:g'=h', c(gw)=0} \alpha(g) \tilde{P}(w | h') P(g | h')$$
Kneser-Ney approach

- Estimation method for lower orders of n-gram model

- Thus, if $h'$ is the backoff history for $h$

$$ P(w | h') = \sum_{g:g'=h'} P(g, w | h') $$

$$ = \sum_{g:g'=h', c(gw)>0} \tilde{P}(w | g) P(g | h') + \sum_{g:g'=h', c(gw)=0} \alpha(g) \tilde{P}(w | h') P(g | h') $$

- Work it out, and for AD under certain assumptions, we get:

$$ \tilde{P}(w | h') = \frac{|\{g : g' = h', c(gw) > 0\}|}{\sum_{v \in V} |\{g : g' = h', c(gv) > 0\}|} $$
Pruning Kneser-Ney

• Kneser-Ney generally achieves strong model improvements

• But, as noted in Siivola et al. (2007) and Chelba et al. (2010): KN doesn’t play well with pruning

• There are a couple of reasons put forward
  – First, calculation of $f(h)$ (state frequency) needed for pruning algorithms not correct when using KN parameters
  – Second, the marginal distribution constraints no longer hold (higher order arcs have been pruned away; model not the same)

• First problem can be fixed by carrying extra information around

• Or try imposing marginal distribution constraints on pruned model
Table 3 from Chelba et al. (2010)

- Broadcast News (128M words training; 692K test); 143k vocab
- 4-gram models estimated and aggressively pruned using SRILM
  - (why prune? what is perplexity?)

<table>
<thead>
<tr>
<th>4-gram models (31M ngrams in full model)</th>
<th>Backoff</th>
<th>Interpolated</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smoothing method</td>
<td>Perplexity</td>
<td>Perplexity</td>
</tr>
<tr>
<td></td>
<td>full</td>
<td>pruned</td>
</tr>
<tr>
<td>Absolute Discounting</td>
<td>120.5</td>
<td>197.3</td>
</tr>
<tr>
<td>Witten-Bell</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ristad</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Katz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kneser-Ney</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kneser-Ney (Chen-Goodman)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
• Broadcast News (128M words training; 692K test); 143k vocab
• 4-gram models estimated and aggressively pruned using SRILM
  – (why prune? what is perplexity?)

<table>
<thead>
<tr>
<th>4-gram models</th>
<th>Backoff</th>
<th>Interpolated</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Perplexity</td>
<td>Params</td>
</tr>
<tr>
<td></td>
<td>full</td>
<td>pruned</td>
</tr>
<tr>
<td>Absolute Discounting</td>
<td>120.5</td>
<td>197.3</td>
</tr>
<tr>
<td>Witten-Bell</td>
<td>118.8</td>
<td>196.3</td>
</tr>
<tr>
<td>Ristad</td>
<td>126.4</td>
<td>203.6</td>
</tr>
<tr>
<td>Katz</td>
<td>119.8</td>
<td>198.1</td>
</tr>
<tr>
<td>Kneser-Ney</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kneser-Ney (Chen-Goodman)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Table 3 from Chelba et al. (2010)**

- Broadcast News (128M words training; 692K test); 143k vocab
- 4-gram models estimated and aggressively pruned using SRILM
  – (why prune? what is perplexity?)

<table>
<thead>
<tr>
<th>4-gram models</th>
<th>Backoff</th>
<th>Interpolated</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Perplexity</td>
<td></td>
</tr>
<tr>
<td></td>
<td>full</td>
<td>pruned</td>
</tr>
<tr>
<td>Absolute Discounting</td>
<td>120.5</td>
<td>197.3</td>
</tr>
<tr>
<td>Witten-Bell</td>
<td>118.8</td>
<td>196.3</td>
</tr>
<tr>
<td>Ristad</td>
<td>126.4</td>
<td>203.6</td>
</tr>
<tr>
<td>Katz</td>
<td>119.8</td>
<td>198.1</td>
</tr>
<tr>
<td>Kneser-Ney</td>
<td>114.5</td>
<td><strong>285.1</strong></td>
</tr>
<tr>
<td>Kneser-Ney (Chen-Goodman)</td>
<td>116.3</td>
<td><strong>280.6</strong></td>
</tr>
</tbody>
</table>

28
Roark et al. (2013)


• Beyond problems described, some reasons to pursue our approach

• We base marginal distribution constraints on smoothed estimates
  – Kneser-Ney based on raw relative frequency estimates

• We base marginal distribution constraints on all orders of model
  – Kneser-Ney estimates order $k$ based on just order $k + 1$

• Add’l assumption in KN: denominator term assumed constant

• KN approach allows for easy count-based estimation
  – Our approach is more compute intensive

• Will now introduce some notation to make description easier.
String notation

- Alphabet of symbols $V$ (using $V$ instead of $\Sigma$ because lots of $\sum_{h \in V^*}$)
  - $V^*$, $V^+$ and $V^k$ as standardly defined:
    - strings of 0 or more; 1 or more; or exactly $k$ symbols from $V$
- A string $h \in V^*$ is an ordered list of zero or more symbols from an alphabet $V$
  - Let $h[i, j]$ be the substring beginning at the $i^{th}$ and ending at the $j^{th}$ symbol
  - $|h|$ denotes the length of $h$, i.e., $h \in V^{|h|}$
  - $h[1, k]$ is a prefix of $h$ (proper if $k < |h|$)
  - $h[j, |h|]$ is a suffix (proper if $j > 1$)
- For ease of notation, let $h^\triangleright = h[2, |h|]$, i.e., the longest suffix of $h$ that’s not $h$; and let $h^\triangleleft = h[1, |h| - 1]$, i.e., the longest prefix of $h$ that’s not $h$
  - Thus, the standardly defined backoff state of $h$ is denoted $h^\triangleright$
N-gram model notation

• For this talk, we assume that a backoff n-gram model \( G \):
  
  – Provides a multinomial probability \( P(w \mid h) \) for all \( w \in V, h \in V^* \)
    (we will call \( h \) the ‘history’; and \( w \) the ‘word’)
  
  – Explicitly represents only a subset of substrings \( hw \) in the model; say \( hw \in G \)
    * If \( hw \in G \), then assume all prefixes and suffixes of \( hw \) are also \( \in G \)
    * If \( hw \notin G \), then probability computed via longest suffix \( h[j, |h|]w \in G \)
      (Let \( \overline{hw} \) be the longest suffix of \( hw \) s.t. \( \overline{hw} \in G \))
  
  – Standard backoff model formulation, slight notational mod:

\[
P(w \mid h) = \begin{cases} 
\beta(hw) & \text{if } hw \in G \\
\alpha(h, h^\triangledown) P(w \mid h^\triangledown) & \text{otherwise}
\end{cases} \quad \text{(s.t. } 0 \leq \sum_{v:hv \in G} \beta(hv) < 1) \]

– Let \( \alpha(h, h) = 1 \), and \( \alpha(h, h[k, |h|]) = \prod_{i=2}^{k} \alpha(h[i-1, |h|], h[i, |h|]) \)
Fragment of n-gram automata representation (real semiring)
Marginal distribution constraint

- For notational simplicity, suppose we have an $n+1$-gram model
  - Largest history in model is of length $n$
- For state of interest, let $|h'| = k$
- Let $V^{n-k}h' = \{ h \in V^n : h' = h[n-k+1, n] \}$, i.e., strings of
  length $n$ with $h'$ suffix (not just those explicitly in the model)
- Recall the constraint in former notation:
  \[ P(w | h') = \sum_{g : g' = h'} P(g, w | h') \]
- in our current notation: \( P(w | h') = \sum_{h \in V^{n-k}h'} P(h, w | h') \)
• We can state our marginal distribution constraint as follows:

\[
P(w \mid h') = \sum_{h \in V^{n-k}h'} P(h, w \mid h')
\]

\[
= \sum_{h \in V^{n-k}h'} P(w \mid h) P(h \mid h')
\]

\[
= \sum_{h \in V^{n-k}h'} P(w \mid h) \frac{P(h)}{\sum_{g \in V^{n-k}h'} P(g)}
\]

\[
= \frac{1}{\sum_{g \in V^{n-k}h'} P(g)} \sum_{h \in V^{n-k}h'} P(w \mid h) P(h)
\]
Only interested in $h'w \in G$

\[ P(w \mid h') = \frac{1}{\sum_{g \in V^{n-k}h'} P(g) \sum_{h \in V^{n-k}h'} P(h)} \sum_{h \in V^{n-k}h'} P(w \mid h) P(h) \]

\[ = \frac{1}{\sum_{g \in V^{n-k}h'} P(g)} \left( \sum_{h \in V^{n-k}h'} \alpha(h, \overline{hw}) \beta(\overline{hw}) P(h) + \sum_{h \in V^{n-k}h'} \alpha(h, h') \beta(h'w) P(h) \right) \]

\[ P(w \mid h') \sum_{g \in V^{n-k}h'} P(g) = \sum_{h \in V^{n-k}h'} \alpha(h, \overline{hw}) \beta(\overline{hw}) P(h) + \beta(h'w) \sum_{h \in V^{n-k}h'} \alpha(h, h') P(h) \]
Solve for $\beta(h'w)$

\[
P(w | h') \sum_{g \in V^{n-k}h'} P(g) = \sum_{h \in V^{n-k}h'} \alpha(h, \bar{hw}^\perp) \beta(\bar{hw}) P(h) + \beta(h'w) \sum_{h \in V^{n-k}h'} \alpha(h, h') P(h)
\]

\[
P(w | h') \sum_{g \in V^{n-k}h'} P(g) - \sum_{h \in V^{n-k}h'} \alpha(h, \bar{hw}^\perp) \beta(\bar{hw}) P(h)
\]

\[
\beta(h'w) = \sum_{h \in V^{n-k}h'} \alpha(h, h') P(h)
\]

\[
\sum_{h \in V^{n-k}h'} \alpha(h, h') P(h)
\]
Numerator term #1

\[
\beta(h'w) = \frac{\sum_{g \in V^{n-k}h'} P(g) - \sum_{h \in V^{n-k}h'} \alpha(h, h\wedge) \beta(hw) P(h)}{\sum_{h \in V^{n-k}h'} \alpha(h, h') P(h)}
\]

- Product of (1) smoothed probability of \( w \) given \( h' \) and (2) sum of probability of all \( h \in V^n \) that backoff to \( h' \) (not just in model)
- Note, if \( h \not\in G \), still has probability (end up in lower-order state)
- Key difference with KN: smoothed not unsmoothed models
Numerator term #2

\[
\beta(h'w) = \frac{P(w \mid h') \sum_{g \in V^{n-k}h'} P(g) - \sum_{h \in V^{n-k}h'} \sum_{\bar{hw} > |h'w|} \alpha(h, \bar{hw}) \beta(\bar{hw}) P(h)}{\sum_{h \in V^{n-k}h'} \alpha(h, h') P(h)}
\]

- Subtract out the probability mass of \(hw\) n-grams associated with histories \(h\) for which \(h'w\) is not accessed in the model
- Implies algorithm to estimate \(\beta(hw)\) in descending n-gram order
- Key difference with KN: not just next highest order considered
Denominator term

\[
\beta(h'w) = \frac{P(w \mid h') \sum_{g \in V^{n-k}h'} P(g) - \sum_{h \in V^{n-k}h'} \alpha(h, \overline{hw}) \beta(\overline{hw}) P(h)}{\sum_{h \in V^{n-k}h'} \alpha(h, h') P(h)}
\]

- Divide by weighted sum of backoff cost over states that will access the \(h'w\) n-gram to assign probability (recall \(\alpha(h', h') = 1\))
- As with other values, can be calculated by recursive algorithm
- Key difference with KN: treated as a constant in KN estimation
Reduced equation complexity to give intuition

\[ \tilde{P}(w \mid h') = A - B + C \]

- Term A: total probability of emitting lower-order ngram
- Term B: probability already accounted for by higher-order ngrams
- Term C: backed-off probability mass from states using the lower-order parameter
Stationary history probabilities

• All three terms require $P(h)$ for all histories $h$

• For histories $h \not\in G$, their probability mass accumulates in their longest suffix $\overline{h} \in G$
  
  – Deterministic automaton; $\overline{h} \in G$ is an equivalence class

• Interested in the steady state probabilities of $h$ as $t \to \infty$

• Currently computed using the power method
  
  – Using backoff, every state has $|V|$ transitions to new states
  – Efficient: convert to real semiring representation with epsilons
  – Converges relatively quickly in practice
Key differences with Kneser-Ney

- Uses smoothed state and n-gram frequencies rather than observed
  - Regularized model presumably improvement vs. unregularized
  - Can be applied to any given backoff model, no extra info req’d
  - Numerator no longer guaranteed > 0?
- Considers full order of model, not just next-highest order
  - Though can memoize information at next-highest order
- Does not take denominator as a constant
  - Backoff weights $\alpha$ change with n-gram $\beta$ values
  - Requires iteration until $\alpha$ values converge
Implementation of algorithm

- Implemented within OpenGrm ngram library

- Worst case complexity $O(|V|^n)$ (model order $n$) typically far less
  
  For every lower-order state $h'$ worst case $O(|V|^{n-2})$
  
  For every state $h$ that backs-off to $h'$ worst case $O(|V|)$
  
  For every $w$ such that $h'w ∈ G$ worst case $O(|V|)$

- Unigram state does typically require $O(|V|^2)$ work (naive)
  
  – Can be dramatically reduced with optimization

- Process higher order states first, memoize data for next lower order

- For each ngram, accumulate prob mass for higher order states requiring and not-requiring backoff for that ngram
Experimental results

- All results for English Broadcast News task
  - LM Training/testing from LDC98T31 (1996 CSR Hub4 LM corpus)
  - AM Training from 1996/1997 Hub4 AM data set
- Automatic speech recognition experiments on 1997 Hub4 eval set
  - Triphone GMM acoustic model trained on approx. 150 hours
  - AM training methods incl.: semi-tied covariance, CMLLR, MMI
  - Multi-pass pass decoding strategy for speaker adaptation
  - Eval set consists of 32,689 words
Experimental results

- Received scripts for replicating 2010 results from Ciprian Chelba
  - Use SRILM for model shrinking and perplexity calculation
  - Convert to/from OpenFst to apply this algorithm
- Nearly identical results to Chelba et al. (2010)
  - Slightly fewer n-grams in full model, i.e., minor data diff
  - Perplexities all within a couple tenths of a point
- OpenFst/ARPA conversion adds a few extra arcs to the model
  - Prefix/suffix n-grams that are pruned by SRILM (“holes”)
  - Sanity check: roundtrip conversion doesn’t change perplexity
<table>
<thead>
<tr>
<th>Smoothing Method</th>
<th>Full Model</th>
<th>Pruned Model</th>
<th>Pruned $+_{MDC}$</th>
<th>$\Delta$</th>
<th>n-grams (×1000)</th>
<th>Model</th>
<th>WFST</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abs.Disc.</td>
<td>120.4</td>
<td>197.1</td>
<td></td>
<td></td>
<td>382.3</td>
<td>389.2</td>
<td></td>
</tr>
<tr>
<td>Witten-Bell</td>
<td>118.7</td>
<td>196.1</td>
<td></td>
<td></td>
<td>379.3</td>
<td>385.0</td>
<td></td>
</tr>
<tr>
<td>Ristad</td>
<td>126.2</td>
<td>203.4</td>
<td></td>
<td></td>
<td>394.6</td>
<td>395.9</td>
<td></td>
</tr>
<tr>
<td>Katz</td>
<td>119.7</td>
<td>197.9</td>
<td></td>
<td></td>
<td>385.1</td>
<td>390.8</td>
<td></td>
</tr>
<tr>
<td>Kneser-Ney†</td>
<td>114.4</td>
<td>234.1</td>
<td></td>
<td></td>
<td>375.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AD, WB, Katz Mixture</td>
<td>118.5</td>
<td>196.6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

†Kneser-Ney model pruned using -prune-history-lm switch in SRILM.
### Perplexity results

<table>
<thead>
<tr>
<th>Smoothing Method</th>
<th>Perplexity</th>
<th>n-grams</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Full Model</td>
<td>Pruned Model</td>
<td>+MDC</td>
</tr>
<tr>
<td>Abs.Disc.</td>
<td>120.4</td>
<td>197.1</td>
<td>187.4</td>
</tr>
<tr>
<td>Witten-Bell</td>
<td>118.7</td>
<td>196.1</td>
<td>185.7</td>
</tr>
<tr>
<td>Ristad</td>
<td>126.2</td>
<td>203.4</td>
<td>190.3</td>
</tr>
<tr>
<td>Katz</td>
<td>119.7</td>
<td>197.9</td>
<td>187.5</td>
</tr>
<tr>
<td>Kneser-Ney†</td>
<td>114.4</td>
<td>234.1</td>
<td></td>
</tr>
<tr>
<td>AD,WB,Katz Mixture</td>
<td>118.5</td>
<td>196.6</td>
<td>186.3</td>
</tr>
</tbody>
</table>

*Kneser-Ney model pruned using `-prune-history-lm` switch in SRILM.
Iterating estimation of steady state probabilities

![Graph showing Perplexity over Iterations of estimation (recalculating steady state probs) for different methods: Witten-Bell, Ristad, Katz, Absolute Discounting, and WB, AD, Katz mixture.](image)
## WER results

<table>
<thead>
<tr>
<th>Smoothing Method</th>
<th>First pass</th>
<th>Rescoring</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pruned Model</td>
<td>Pruned +MDC</td>
</tr>
<tr>
<td>Abs.Disc.</td>
<td>20.5</td>
<td>19.7</td>
</tr>
<tr>
<td>Witten-Bell</td>
<td>20.5</td>
<td>19.9</td>
</tr>
<tr>
<td>Katz</td>
<td>20.5</td>
<td>19.7</td>
</tr>
<tr>
<td>Mixture</td>
<td>20.5</td>
<td>19.6</td>
</tr>
<tr>
<td>Kneser-Ney&lt;sup&gt;a&lt;/sup&gt;</td>
<td>22.1</td>
<td>22.2</td>
</tr>
<tr>
<td>Kneser-Ney&lt;sup&gt;b&lt;/sup&gt;</td>
<td>20.5</td>
<td>20.6</td>
</tr>
</tbody>
</table>

KN model (a) original pruning; (b) pruned using `-prune-history-lm` switch in SRILM.
Discussion of results

• General method for re-estimating parameters in a backoff model
• Achieved perplexity reductions on all method tried so far
  – Perplexity of 185.7 best for models of this size (by 10 points)
• Initial somewhat clear best practices at this point
  – Important to iterate estimation of steady state probabilities
  – Not so important to iterate denominator calculation for new backoff $\alpha$ values
  – But shouldn’t treat denominator as a constant
• Remaining question: how does it perform for larger models?
Larger models

- See paper for explicit numbers on less dramatically pruned models
  - Smaller gains over baselines with less pruning (less smoothing)
  - Model degradation observed for lightly pruned Witten-Bell models, likely due to under-regularization
  - Performance similar to Kneser-Ney on unpruned model, when starting from Katz or Absolute Discounting
Summary of main points in lecture

- MaxEnt models rely on good feature engineering
  - Not only n-grams in n-gram models!
  - Tip of the iceberg in terms of new features

- Marginal distribution constraints important and widely used
  - For training MaxEnt models
  - Also basis of Kneser-Ney language modeling
  - Can be tricky to impose under some conditions