Language Model Rest Costs and Space-Efficient Storage

Kenneth Heafield    Philipp Koehn    Alon Lavie

Carnegie Mellon, University of Edinburgh

July 14, 2012
Complaint About Language Models

Make Search Expensive

\[
\frac{p_5(\text{is one of the})}{p_5(\text{is one})p_5(\text{of the})} \neq 1
\]

1. Better fragment scores
Complaints About Language Models

Make Search Expensive

\[
\frac{p_5(\text{is one of the})}{p_5(\text{is one})p_5(\text{of the})} \neq 1
\]

1. Better fragment scores

Use Too Much Memory

\[
\log p_5(\text{the | is one of}) = -0.5 \\
\log b_5(\text{is one of the}) = -1.2
\]

2. Collapse probability and backoff
Language Model Probability of Sentence Fragments

\[ \log p_5(\text{is one of the few}) = -6.62 \]

Why does it matter?
Decoders prune hypotheses based on score.
Baseline: How to Score a Fragment

\[
\begin{align*}
\log p_5(\text{is}) &= -2.63 \\
\log p_5(\text{one} \mid \text{is}) &= -2.03 \\
\log p_5(\text{of} \mid \text{is one}) &= -0.24 \\
\log p_5(\text{the} \mid \text{is one of}) &= -0.47 \\
+ \log p_5(\text{few} \mid \text{is one of the}) &= -1.26 \\
\hline
= \log p_5(\text{is one of the few}) &= -6.62
\end{align*}
\]
The Problem: Lower Order Entries

5-Gram Model: \( \log p_5(is) = -2.63 \)
Unigram Model: \( \log p_1(is) = -2.30 \)
Same training data.
$p_5(is)$ should be used when a bigram was not found.

**In the language model**

$$\log p_5(is \mid \text{australia}) = -2.21$$

**Not in the language model**

$$\log p_5(is \mid \text{periwinkle}) = \log b_5(\text{periwinkle}) + \log p_5(is) = -2.95$$
Backoff Smoothing

\( p_5(\text{is}) \) should be used when a bigram was not found.

In the language model

\[
\log p_5(\text{is} \mid \text{australia}) = -2.21
\]

Not in the language model

\[
\log p_5(\text{is} \mid \text{periwinkle}) = \log b_5(\text{periwinkle}) + \log p_5(\text{is}) = -2.95
\]

In Kneser-Ney smoothing, lower order probabilities assume backoff.
Use Lower Order Models for the First Few Words

<table>
<thead>
<tr>
<th>Baseline</th>
<th>Lower</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\log p_5(\text{is})$</td>
<td>$= -2.63$</td>
</tr>
<tr>
<td>$\log p_5(\text{one} \mid \text{is})$</td>
<td>$= -2.03$</td>
</tr>
<tr>
<td>$\log p_5(\text{of} \mid \text{is one})$</td>
<td>$= -0.24$</td>
</tr>
<tr>
<td>$\log p_5(\text{the} \mid \text{is one of})$</td>
<td>$= -0.47$</td>
</tr>
<tr>
<td>$\log p_5(\text{few} \mid \text{is one of the})$</td>
<td>$= -1.26$</td>
</tr>
<tr>
<td>$\log p_5(\text{is one of the few})$</td>
<td>$= -6.62$</td>
</tr>
<tr>
<td>$\log p_1$</td>
<td>$= -2.30$</td>
</tr>
<tr>
<td>$\log p_2$</td>
<td>$= -1.92$</td>
</tr>
<tr>
<td>$\log p_3$</td>
<td>$= -0.08$</td>
</tr>
<tr>
<td>$\log p_4$</td>
<td>$= -0.21$</td>
</tr>
<tr>
<td>$\log p_5$</td>
<td>$= -1.26$</td>
</tr>
<tr>
<td>$\log p_{\text{Low}}$</td>
<td>$= -5.77$</td>
</tr>
</tbody>
</table>
Which is Better?

Baseline: \( \log p_5(\text{is one of the few}) \) = \(-6.62\)

Lower Order: \( \log p_{\text{Low}}(\text{is one of the few}) \) = \(-5.77\)
Which is Better: Prediction Task

Baseline: \( \log p_5(\text{is one of the few}) = -6.62 \)

Lower Order: \( \log p_{\text{low}}(\text{is one of the few}) = -5.77 \)

Actual: \( \log p_5(\text{is one of the few } | \text{ <s> australia}) = -4.10 \)

Error

- Baseline: 2.52
- Lower Order: 1.67
- Actual: -4.10
The Lower Order Estimate is Better

Run the decoder and log error every time context is revealed.

<table>
<thead>
<tr>
<th>Length</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>.87</td>
<td>.24</td>
<td>.10</td>
<td>.09</td>
</tr>
<tr>
<td>Lower Order</td>
<td>.84</td>
<td>.18</td>
<td>.07</td>
<td>.04</td>
</tr>
</tbody>
</table>

**Table:** Mean squared error in predicting log probability.
**Storing Lower Order Models**

One extra float per entry, except for longest order.

**Unigrams**

<table>
<thead>
<tr>
<th>Words</th>
<th>$\log p_5$</th>
<th>$\log b_5$</th>
<th>$\log p_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>australia</td>
<td>-3.9</td>
<td>-0.6</td>
<td>-3.6</td>
</tr>
<tr>
<td>is</td>
<td>-2.6</td>
<td>-1.5</td>
<td>-2.3</td>
</tr>
<tr>
<td>one</td>
<td>-3.4</td>
<td>-1.0</td>
<td>-2.9</td>
</tr>
<tr>
<td>of</td>
<td>-2.5</td>
<td>-1.1</td>
<td>-1.7</td>
</tr>
</tbody>
</table>

No need for backoff $b_1$

If backoff occurs, the Kneser-Ney assumption holds and $p_5$ is used.
Fragment scores are more accurate, but require more memory.
Related Work

Score with and without sentence boundaries. [Sankaran et al, 2012]

Peek at future phrases. [Zens and Ney, 2008]

Coarse pass predicts scores for a finer pass. [Wuebker et al, Wed.]

[Vilar and Ney, 2011]
Score with and without sentence boundaries. [Sankaran et al, 2012]
Peek at future phrases. [Zens and Ney, 2008] [Wuebker et al, Wed.]
Coarse pass predicts scores for a finer pass. [Vilar and Ney, 2011]

All of these use fragment scores as a subroutine.
### Related Work II: Carter et al, Yesterday

**This Work**

\[ p(\text{is one of the}) \approx p(\text{is one})p(\text{of the}) \]

**Their Work**

\[ p(\text{is one of the}) \leq p(\text{is one})p(\text{of the}) \]

### Implementing Upper Bounds Within This Work

- Store upper bound probabilities instead of averages
- Account for positive backoff with the context

Three values per n-gram instead of their four.
Lower Order Summary

Previously
Fragment scores are more accurate, but require more memory.

Next
Save memory but make fragment scores less accurate.
## Saving Memory

<table>
<thead>
<tr>
<th>Words</th>
<th>log $p_5$</th>
<th>log $b_5$</th>
<th>log $q_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>australia</td>
<td>-3.9</td>
<td>-0.6</td>
<td>-4.5</td>
</tr>
<tr>
<td>is</td>
<td>-2.6</td>
<td>-1.5</td>
<td>-4.1</td>
</tr>
<tr>
<td>one</td>
<td>-3.4</td>
<td>-1.0</td>
<td>-4.4</td>
</tr>
<tr>
<td>of</td>
<td>-2.5</td>
<td>-1.1</td>
<td>-3.6</td>
</tr>
</tbody>
</table>

One less float per entry, except for longest order.
Related Work


This Work

- Memory comparable to storing counts.
- Higher quality Kneser-Ney smoothing.
How Backoff Works

\[ p(\text{periwinkle} \mid \text{is one of}) = p(\text{periwinkle} \mid \text{of}) \cdot b(\text{is one of}) \cdot b(\text{one of}) \]

because “of periwinkle” appears but “one of periwinkle” does not.
Assume backoff all the way to unigrams.

\[ q(\text{is one of}) = p(\text{is one of}) b(\text{is one of}) b(\text{one of}) b(\text{of}) \]
Pessimism

Assume backoff all the way to unigrams.

\[ q(\text{is one of}) = p(\text{is one of}) b(\text{is one of}) b(\text{one of}) b(\text{of}) \]

Sentence Scores Are Unchanged

\[ q(<s>\cdots</s>) = p(<s>\cdots</s>) \]

because \[ b(\cdots</s>) = 1 \]
Incremental Pessimism

\[ q(is) = p(is)b(is) \]

\[ q(one \mid is) = p(one \mid is) \frac{b(is \ one)b(one)}{b(is)} \]

These are terms in a telescoping series:

\[ q(is \ one) = q(is)q(one \mid is) \]
Using $q$

\[
\begin{align*}
\log q(\text{is}) &= -4.10 \\
\log q(\text{one} \mid \text{is}) &= -2.51 \\
\log q(\text{of} \mid \text{is one}) &= -0.94 \\
\log q(\text{the} \mid \text{is one of}) &= -1.61 \\
+ \log q(\text{few} \mid \text{is one of the}) &= 1.03 \\
\hline
\log q(\text{is one of the few}) &= -8.13
\end{align*}
\]

Store $q$, forget probability and backoff.
Using $q$

\[
\begin{align*}
\log q(\text{is}) &= -4.10 \\
\log q(\text{one} \mid \text{is}) &= -2.51 \\
\log q(\text{of} \mid \text{is one}) &= -0.94 \\
\log q(\text{the} \mid \text{is one of}) &= -1.61 \\
+ \log q(\text{few} \mid \text{is one of the}) &= 1.03 \\
\end{align*}
\]

\[
= \log q(\text{is one of the few}) = -8.13
\]

Store $q$, forget probability and backoff.

$q$ is not a proper probability distribution.
Pessimistic Backoff Summary

Collapse probability and backoff from two values to one value.
Stacking

**Lower Order and Pessimistic Combined**

- Same memory (one extra float, one less float).
- Better on the left, worse on the right.
Cube Pruning: Approximate Search

For each constituent, going bottom-up:

1. Make a priority queue over possible rule applications.
2. Pop a fixed number of hypotheses: the *pop limit*.

Larger pop limit $\implies$ more accurate search.
Cube Pruning: Approximate Search

For each constituent, going bottom-up:

1. Make a priority queue over possible rule applications.
2. Pop a fixed number of hypotheses: the pop limit.

Larger pop limit
Accurate fragment scores  $\Rightarrow$ more accurate search.
## Experiments

<table>
<thead>
<tr>
<th>Task</th>
<th>WMT 2011 German-English</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decoder</td>
<td>Moses</td>
</tr>
<tr>
<td>LM</td>
<td>5-gram from Europarl, news commentary, and news</td>
</tr>
<tr>
<td>Grammar</td>
<td>Hierarchical and target-syntax systems</td>
</tr>
<tr>
<td>Parser</td>
<td>Collins</td>
</tr>
</tbody>
</table>
Hierarchical Model Score and BLEU

![Graph showing hierarchical model score and BLEU](image)
Memory

Cost to add or savings from removing a float per entry.

<table>
<thead>
<tr>
<th>Structure</th>
<th>Baseline (MB)</th>
<th>Change (MB)</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Probing</td>
<td>4,072</td>
<td>517</td>
<td>13%</td>
</tr>
<tr>
<td>Trie</td>
<td>2,647</td>
<td>506</td>
<td>19%</td>
</tr>
<tr>
<td>8-bit quantized trie</td>
<td>1,236</td>
<td>140</td>
<td>11%</td>
</tr>
<tr>
<td>8-bit minimal perfect hash</td>
<td>540</td>
<td>140</td>
<td>26%</td>
</tr>
</tbody>
</table>
Summary

Lower Order Models
- 21-63% less CPU
- 13-26% more memory

Pessimistic Backoff
- 27% more CPU
- 13-26% less memory

Lower Order + Pessimistic
- 3% less CPU
- Same memory as baseline
Code

kheafield.com/code/kenlm
Also distributed with Moses and cdec.

**Lower Order**

`build_binary -r "1.arpa 2.arpa 3.arpa 4.arpa" 5.arpa 5.binary`

**Pessimistic Backoff**

Release planned