

Streaming for large scale NLP: Language Modeling

Amit Goyal, Hal Daumé III and Suresh Venkatasubramanian
School of Computing
University of Utah

email: {amitg,hal,suresh}@cs.utah.edu
web: <http://www.cs.utah.edu/~amitg/>

NAACL HLT 2009

3rd June 2009

Problem

- Large amounts of data in many NLP problems
- Many such problems require relative frequency estimation
- **Computationally expensive** on huge corpora

Problem

- Large amounts of data in many NLP problems
- Many such problems require relative frequency estimation
- **Computationally expensive** on huge corpora

Canonical Task

- Language Modeling: large-scale frequency estimation

Proposed Solution

- Trades off memory usage with accuracy of counts using **Streaming**
- Employs **small memory-footprint** to **approximate n -gram counts**

Problem

- Large amounts of data in many NLP problems
- Many such problems require relative frequency estimation
- **Computationally expensive** on huge corpora

Canonical Task

- Language Modeling: large-scale frequency estimation

Proposed Solution

- Trades off memory usage with accuracy of counts using **Streaming**
- Employs **small memory-footprint** to **approximate n -gram counts**

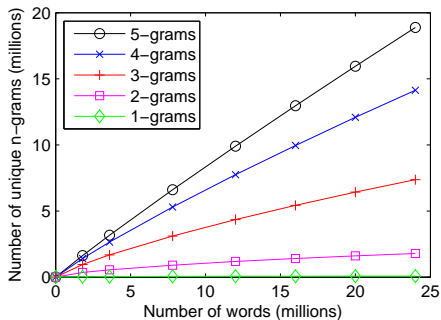
Findings

- Scales to billion-word corpora using conventional 8 GB machine
- SMT experiments show that these counts are effective

Large Scale Language Modeling

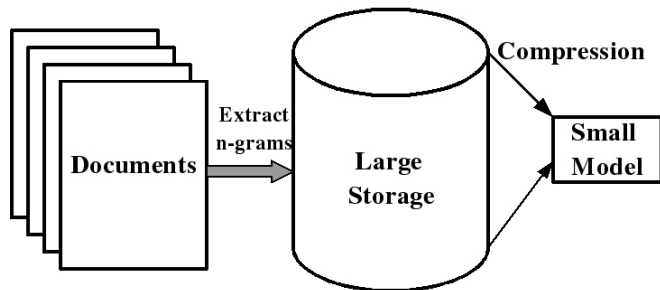
Goal: Building higher order language models (LMs) on huge data sets

Difficulties: Increase in $n \implies$ Increase in number of unique n -grams
 \implies Increase in memory usage



Example

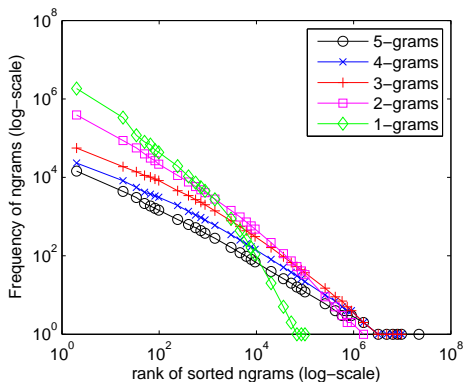
- 1500 machines got used for a day to compute 300 million unique n -grams from tera bytes of web data [Brants et al. (2007)]



- Prefix trees to store LM probabilities efficiently
[Federico and Bertoldi, SMT workshop at ACL 2006]
- Bloom and Bloomier filters: Compressed n -gram representation
[Talbot and Osborne; ACL 2007] [Talbot and Brants; 2008]
- Distributed word clustering for class-based LMs
[Uszkoreit and Brants; ACL 2008]

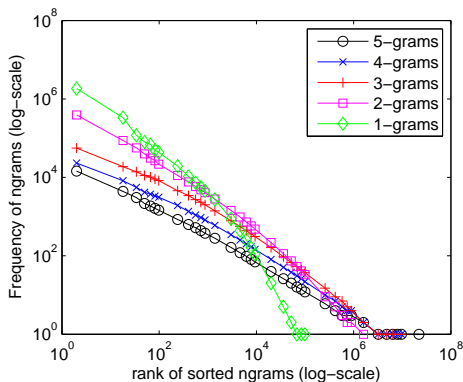
Zipf's law Phenomena

- Number of unique n -grams is large
- Low frequency count n -grams contribute most towards LM size



Zipf's law Phenomena

- Number of unique n -grams is large
- Low frequency count n -grams contribute most towards LM size



Key Idea: Throw away rare n -grams

n-gram Pruning Methods

Count pruning

- Discards all *n*-grams whose count $<$ pre-defined threshold

Entropy pruning

- Discards *n*-grams that change perplexity by less than a threshold

n -gram Pruning Methods

Count pruning

- Discards all n -grams whose count $<$ pre-defined threshold

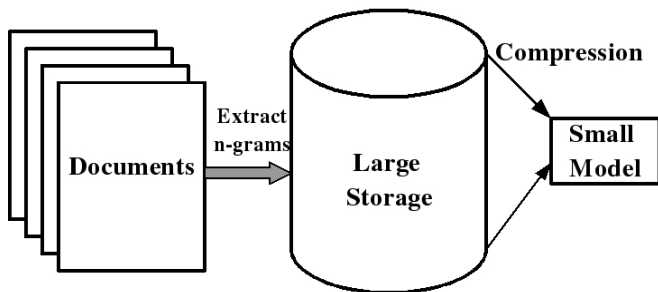
Entropy pruning

- Discards n -grams that change perplexity by less than a threshold

SMT experiments with 5-gram LM on large data:

Model	Size	BLEU
Exact	367.6m	28.7
100 count cutoff	1.1m	28.0
$5e-7$ ϵ entropy	28.5m	28.1

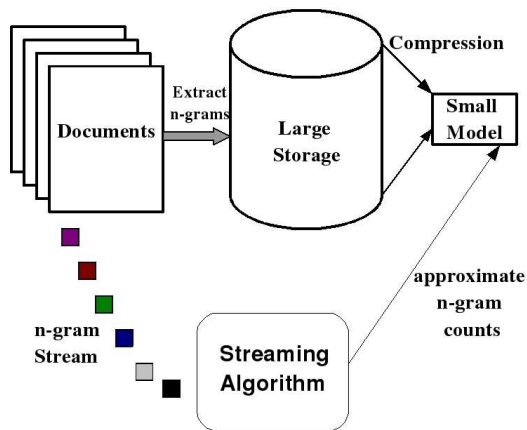
- Pruning method loses 0.7 BLEU points compared to exact model
- Decrease \implies 300 times smaller model



Difficulties with scaling pruning methods for large-scale LM:

- Computation time and memory usage to compute all counts is tremendous
- Requires enormous initial disk storage for n -grams

Proposed Solution

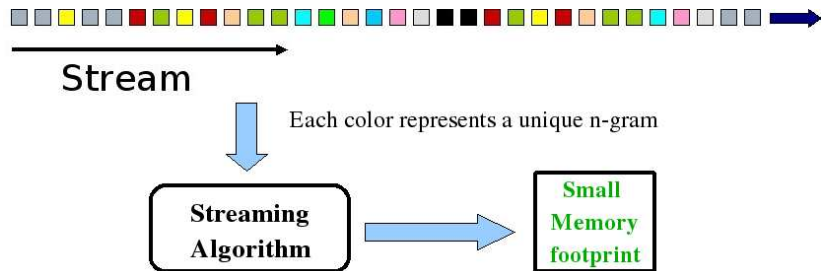


- Assume that multiple-GB models are infeasible
- **Goal:** Directly estimate a small model instead of first estimate a large model and then compress it
- Employ deterministic streaming algorithm [Manku and Motwani, 2002]

Streaming

Given: Stream of n -grams of length N .

Running Example: $n=5$ and $N=10^6$



- Algorithm can only read from left to right without going backwards
- Store only parts of input or other intermediate values
- Typical working storage space size $O(\log^k N)$

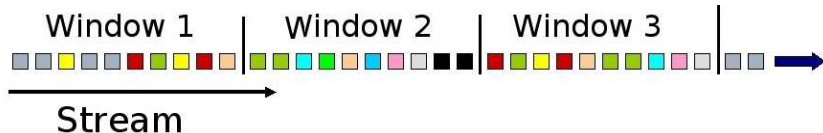
Algorithm: Lossy Counting [Manku and Motwani, 2002]

Step 1: Divide the stream into windows using $\epsilon \in (0, 1)$

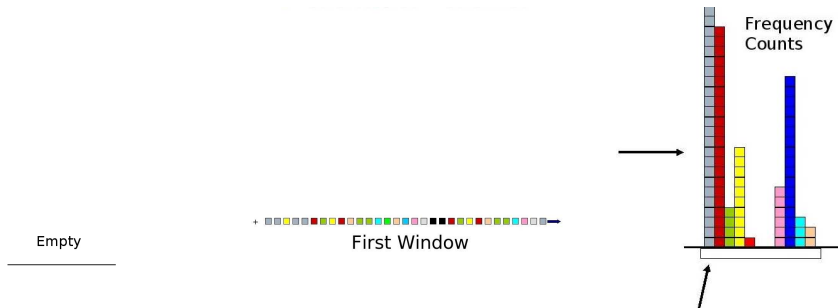
Window size = $\frac{1}{\epsilon}$; Total ϵN windows

Running Example: Set $\epsilon = 0.001$; $N = 10^6$

Window size = 10^3 ; Total 10^3 windows

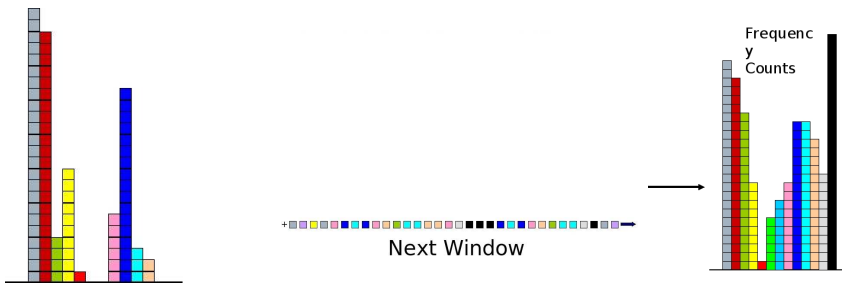


Algorithm in Action . . .



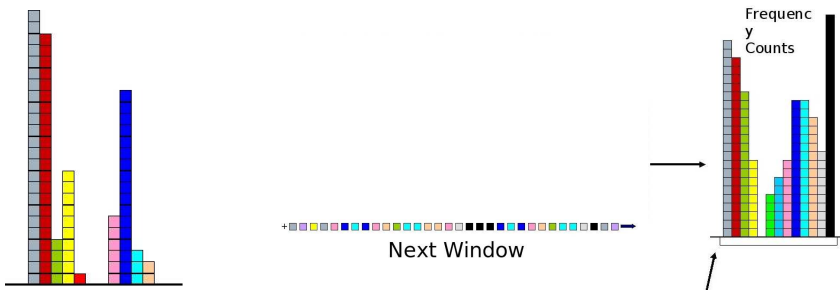
At window boundary, all counters are decremented by 1

Algorithm continued . . .



At window boundary, decrement all counters by 1

Algorithm continued . . .



At window boundary, all counters are decremented by 1

Algorithm Guarantees

$s \in (0, 1)$ is support. In practice, $s = 10\epsilon$

Running Example: $\epsilon=0.001$, $s = 0.01$

Algorithm Guarantees

$s \in (0, 1)$ is support. In practice, $s = 10\epsilon$

Running Example: $\epsilon=0.001$, $s = 0.01$

- All n -grams with actual counts $> sN$ (10^4) are output
- Returns no n -grams with actual counts $< (s\epsilon)N$ (9000)
- All reported counts \leq actual counts by at most ϵN (1000)
- Space used by the algorithm: $O(\frac{1}{\epsilon} \log(\epsilon N))$

Algorithm Guarantees

$s \in (0, 1)$ is support. In practice, $s = 10\epsilon$

Running Example: $\epsilon=0.001$, $s = 0.01$

- All n -grams with actual counts $> sN$ (10^4) are output
 - Returns no n -grams with actual counts $< (s\epsilon)N$ (9000)
 - All reported counts \leq actual counts by at most ϵN (1000)
 - Space used by the algorithm: $O(\frac{1}{\epsilon} \log(\epsilon N))$
-
- In practice, set $s = \epsilon$ to retain all generated counts
 - n -grams appearance more valuable than their counts

Evaluating stream n -gram counts

Data: English side of Europarl (EP): *38million* words

Portions of Gigaword i.e. afe and nyt + EP (EAN): *1.4billion* words

Accuracy: Ratio of # of sorted Top K stream n -grams found in # of Top K sorted true n -grams (Higher is better)

Evaluating stream n -gram counts

Data: English side of Europarl (EP): *38million* words

Portions of Gigaword i.e. afe and nyt + EP (EAN): *1.4billion* words

Accuracy: Ratio of # of sorted Top K stream n -grams found in # of Top K sorted true n -grams (Higher is better)

ϵ	5-gram produced	Acc
50e-8	245k	0.29
20e-8	726k	0.33
10e-8	1655k	0.35
5e-8	4018k	0.36

Table: Evaluating quality of 5-gram stream counts for different settings of ϵ on EAN corpus

Evaluating stream n -gram counts

Data: English side of Europarl (EP): *38million* words

Portions of Gigaword i.e. afe and nyt + EP (EAN): *1.4billion* words

Accuracy: Ratio of # of sorted Top K stream n -grams found in # of Top K sorted true n -grams (Higher is better)

ϵ	5-gram produced	Acc
50e-8	245k	0.29
20e-8	726k	0.33
10e-8	1655k	0.35
5e-8	4018k	0.36

Table: Evaluating quality of 5-gram stream counts for different settings of ϵ on EAN corpus

Top K	Accuracy
100k	0.99
500k	0.93
1000k	0.72
2000k	0.50
4018k	0.36

Table: Evaluating top K sorted 5-gram stream counts for $\epsilon=5e-8$ on EAN corpus

SMT Experimental Setup

- Training set: Europarl (EP) French-English parallel corpus: [Million](#) sentences
- Language Model data: EP and afe + nyt + EP (EAN)
- Development and Test set: News corpus of [1057](#) and [3071](#) sentences
- Evaluation on uncased test-set using BLEU metric (Higher is better)

SMT Experimental Setup

- Training set: Europarl (EP) French-English parallel corpus: Million sentences
- Language Model data: EP and afe + nyt + EP (EAN)
- Development and Test set: News corpus of 1057 and 3071 sentences
- Evaluation on uncased test-set using BLEU metric (Higher is better)

- Models Compared:
 - 4 baseline LMs (3, 5-gram on EP and EAN)
 - Count and Entropy pruning 5-gram LMs
 - Stream count LMs computed with two values of : $5e - 8$ and $10e - 8$ on EAN corpus

SMT Experiment Results

n -gram(ϵ)	BLEU	Mem GB
3 EP	25.6	2.7
5 EP	25.8	2.9
3 EAN	27.0	4.6
5 EAN	28.7	20.5

100 count cutoff	28.0	2.8
5e-7 ϵ entropy	28.1	3.0

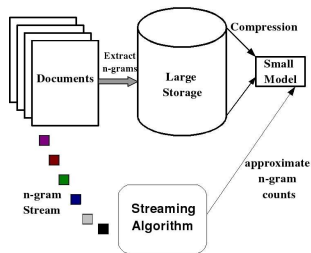
5(10e-8)	28.0	2.8
5(5e-8)	28.0	2.8
7(10e-8)	28.0	2.9
9(10e-8)	28.2	2.9

Baselines: Large LMs effective

Stream counts findings:

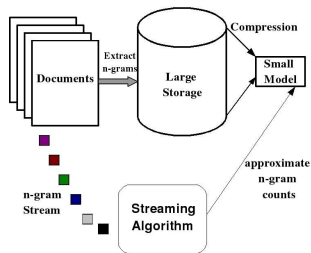
- Effective as pruning methods
- 0.7 Bleu worse to exact
- Memory Efficient
- 7 and 9-gram are also possible

Take Home Message:



- Directly estimate small model
- Memory efficient
- Counts are effective

Take Home Message:

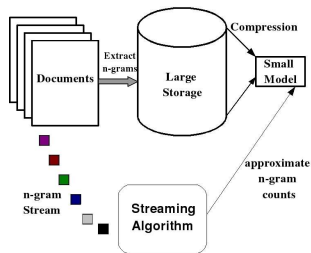


- Directly estimate small model
- Memory efficient
- Counts are effective

Future Directions:

- Use these LMs for speech recognition, information extraction etc.
- Streaming in other NLP applications
- Build streaming class-based and skip n -gram LMs

Take Home Message:



- Directly estimate small model
- Memory efficient
- Counts are effective

Future Directions:

- Use these LMs for speech recognition, information extraction etc.
- Streaming in other NLP applications
- Build streaming class-based and skip n -gram LMs

Thanks! Questions?