Efficient Language Modeling Algorithms with Applications to Statistical Machine Translation

Kenneth Heafield

September 20, 2013

CPU and RAM Costs Matter

“had to favor speed over performance” [Moreau et al, 2013]

“could not test whether this result holds in a large scale evaluation” [Durrani et al, 2013]

CPU and RAM Costs Matter

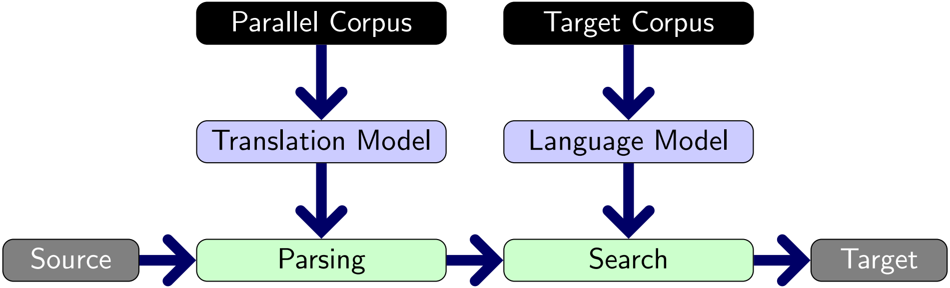
"had to favor speed over performance" [Moreau et al, 2013]

"could not test whether this result holds in a large scale evaluation" [Durrani et al, 2013]

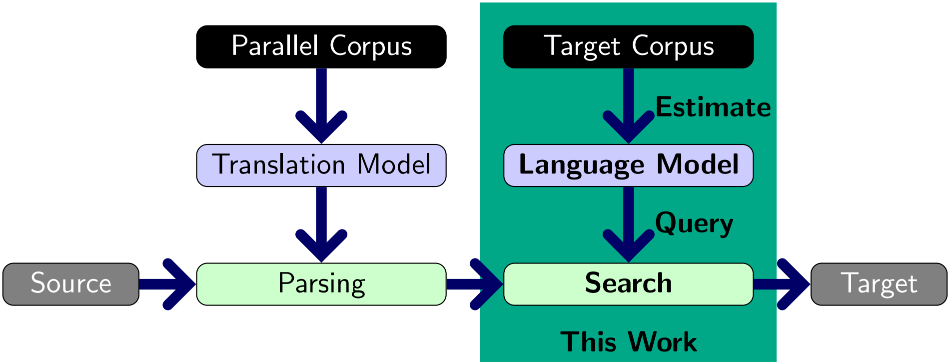
0.5–1.9% BLEU gain from English Gigaword

[Koehn et al, 2012]

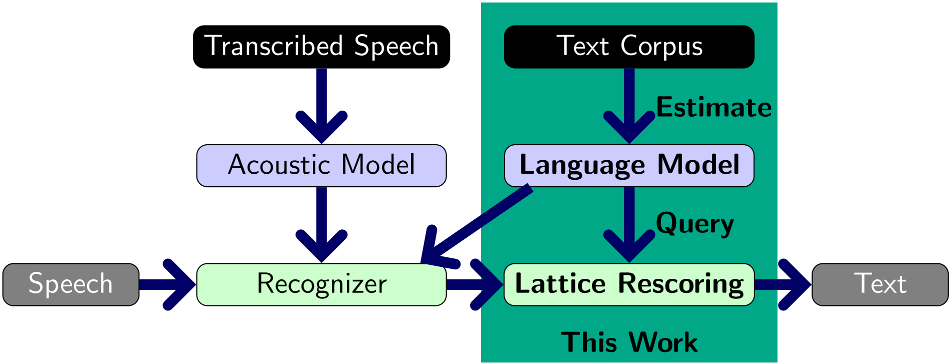
Application: Syntactic MT



Application: Syntactic MT

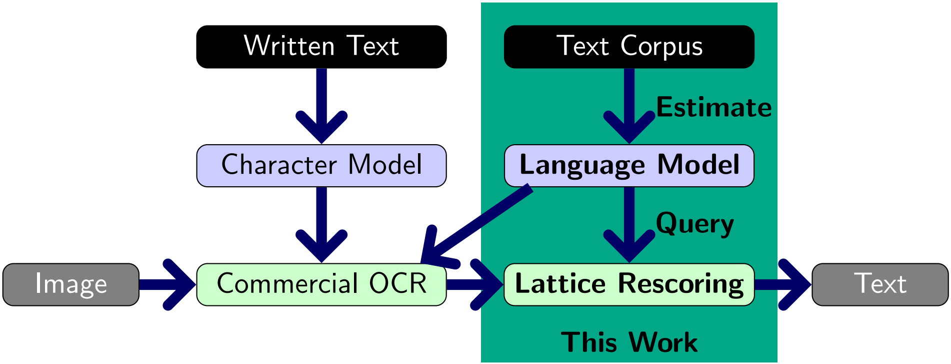


Speech Recognition



Some of the thesis is already used for speech [Kim et al, 2012; Si et al, 2013].

Optical Character Recognition



Numen (2013) is using most of this thesis for OCR.

Language Models Are Expensive

Store a sparse set of 121 billion *n*-grams $\Longrightarrow$ RAM

Language Models Are Expensive

Store a sparse set of 121 billion *n*-grams $\Longrightarrow$ RAM

Millions of probability queries per sentence $\Longrightarrow$ CPU

Language Models Are Expensive

Store a sparse set of 121 billion *n*-grams $\Longrightarrow$ RAM

Millions of probability queries per sentence $\Longrightarrow$ CPU

Probability does not multiply when strings are concatenated:

$$p(\text{saw the man}) \neq p(\text{saw})p(\text{the man})$$

$\Longrightarrow$ Search is hard $\Longrightarrow$ CPU

Thesis Problem

Much of the CPU and RAM cost is due to the language model.
Researchers routinely compromise quality due to these costs.

Costs Due To Language Models

Estimation from text

Probability queries

Search when the objective includes log probability

Results Preview

	Speed	**RAM**
Estimation from Text	7.1x	0.07x
Raw Queries	2.4x	0.57x
Decoding	3.2–10.0x	0.85x

Decoding performance includes $\approx$1.15x speedup from raw queries.
Baseline: SRILM and cube pruning (more later).

Outline

		Speed	RAM	Published
1	Estimation from Text	7.1x	0.07x	ACL 2013
2	Raw Queries	2.4x	0.57x	WMT 2011
3	Decoding	3.2–10.0x	0.85x	IWSLT 2011, EMNLP 2012, NAACL 2013

Estimating LMs is Costly

MIT	RAM
SRI	RAM, time
IRST	RAM, time, approximation
Berkeley	RAM, time, approximation

Estimating LMs is Costly

MIT	RAM
SRI	RAM, time
IRST	RAM, time, approximation
Berkeley	RAM, time, approximation
Microsoft	Delay some computation to query time
Google	100–1500 machines, optional stupid backoff

Estimating LMs is Costly

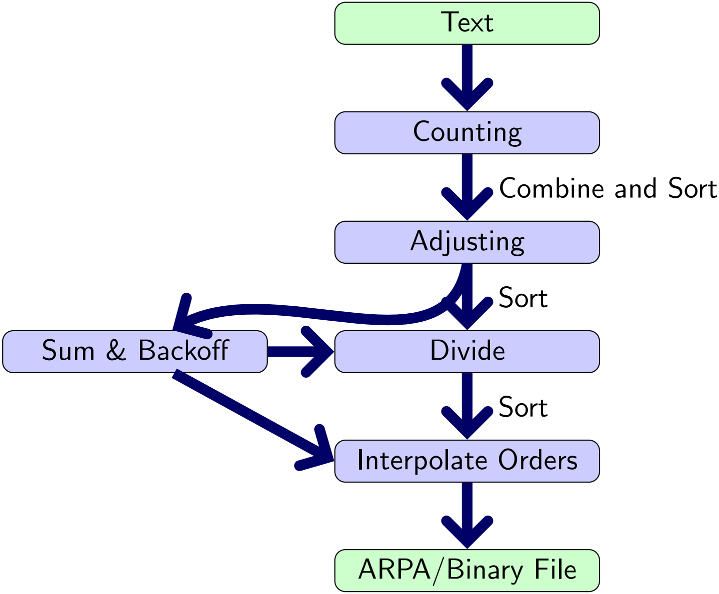
MIT	RAM
SRI	RAM, time
IRST	RAM, time, approximation
Berkeley	RAM, time, approximation
Microsoft	Delay some computation to query time
Google	100–1500 machines, optional stupid backoff

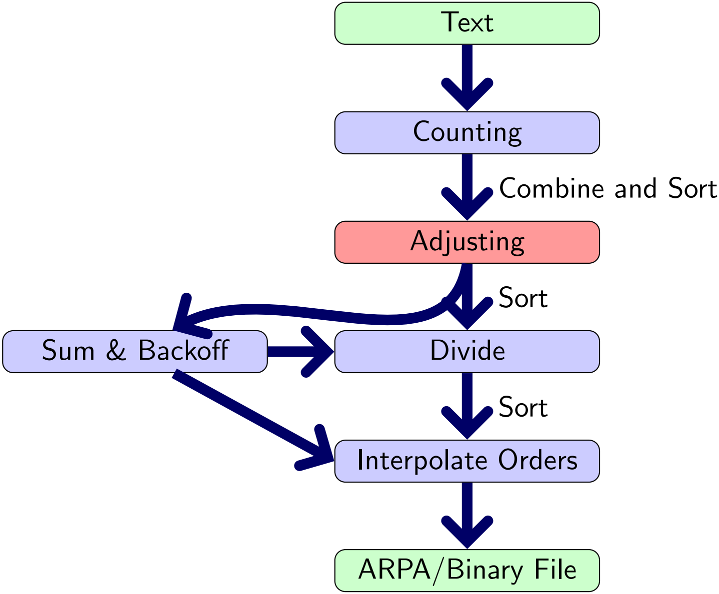
"When, oh when, will there be an alternative?"

lmplz Features

- Disk-based streaming and sorting
- User-specified RAM
- Fast
- Interpolated modified Kneser-Ney

7.7% of SRI's RAM, 14% of SRI's wall time


Text
Counting
Combine and Sort
Adjusting
Sort
Sum & Backoff
Divide
Sort
Interpolate Orders
ARPA/Binary File


Text
Counting
Combine and Sort
Adjusting
Sort
Sum & Backoff
Divide
Sort
Interpolate Orders
ARPA/Binary File

Adjusting

Adjusted counts are:

Trigrams Same as counts.

Others Number of unique words to the left.

Adjusting

Adjusted counts are:

Trigrams Same as counts.

Others Number of unique words to the left.

Suffix Sorted Input

3	2	1	Count
are	one	of	1
is	one	of	5
are	two	of	3

→

Output

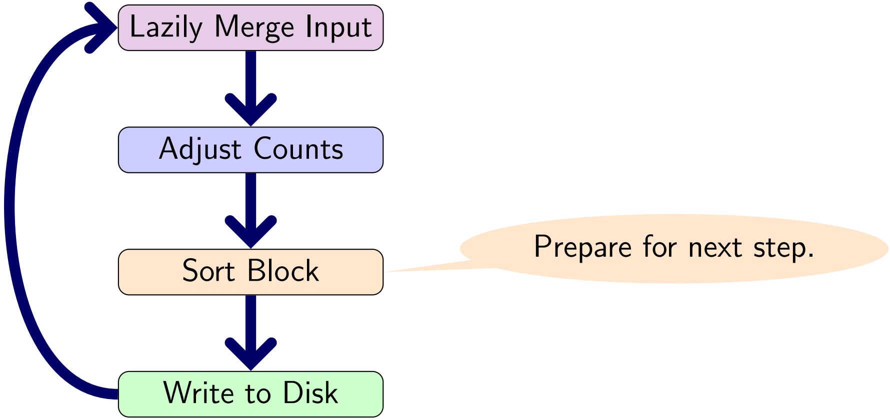
1-gram	Adjusted
of	2

Output

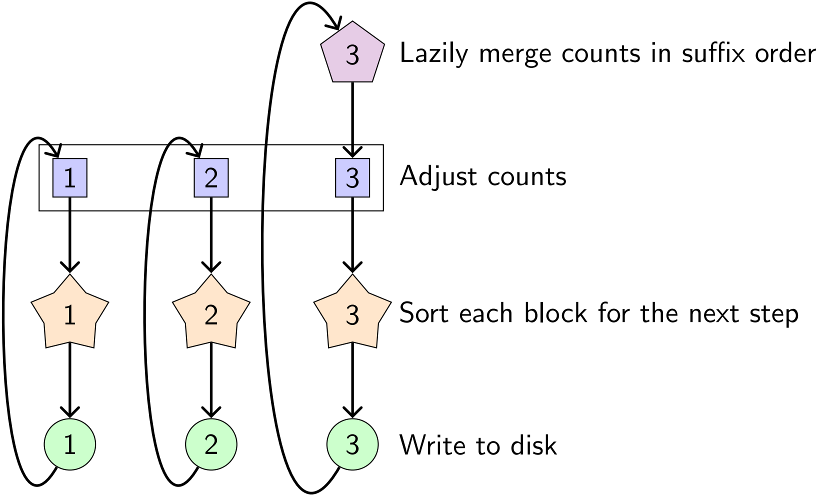
2-gram	Adjusted
one of	2
two of	1

Streaming Framework

Memory is divided into blocks. Blocks are recycled.



Adjusted Counts Detail

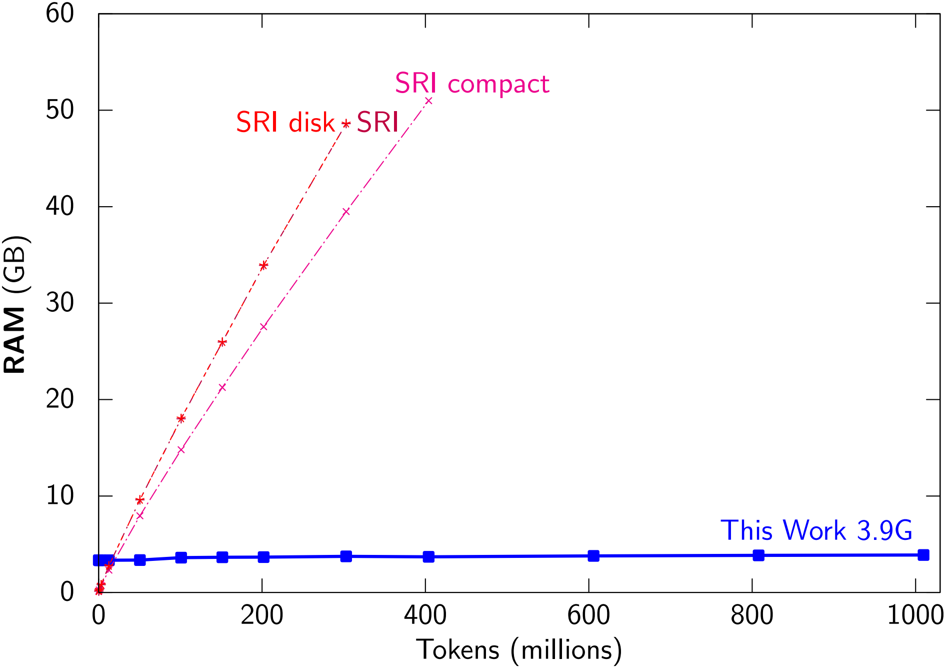


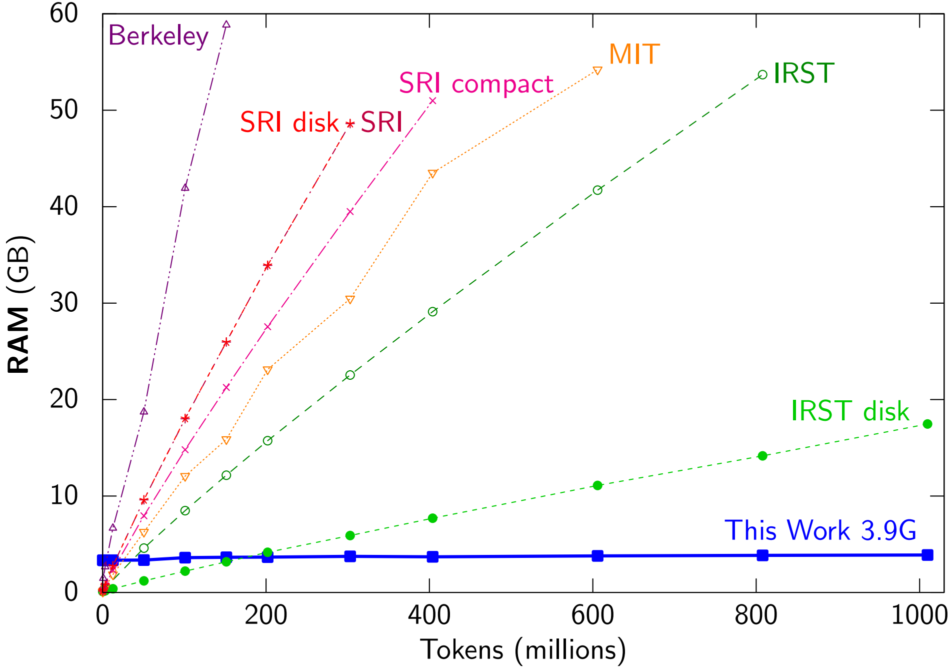
Each vertex is a thread $\Longrightarrow$ Simultaneous disk and CPU.

Experiment: Toolkit Comparison

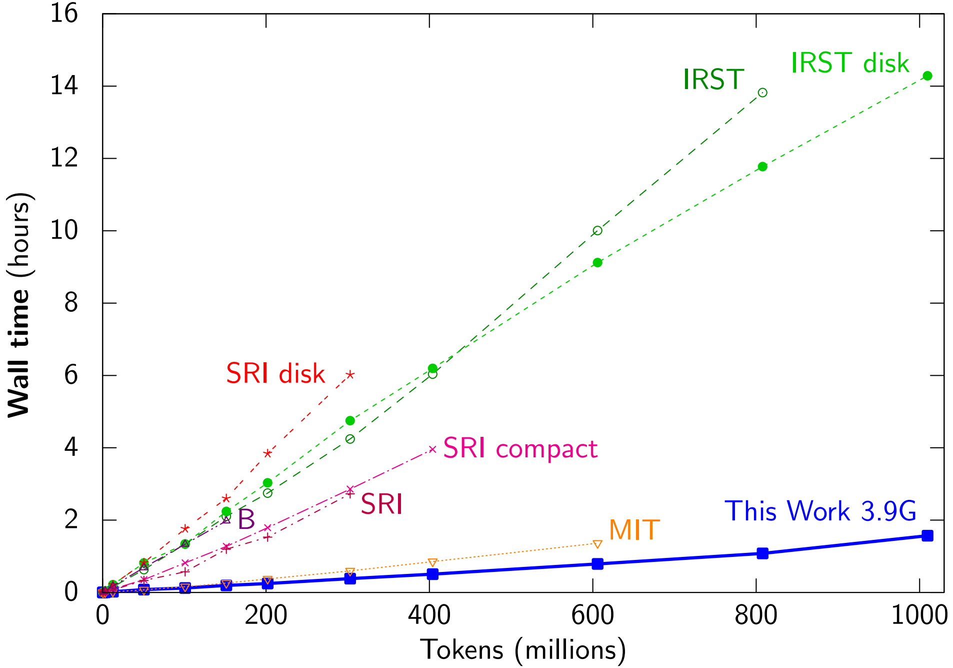
Task	Build an unpruned 5-gram LM
Data	Subset of English ClueWeb09 (webpages)
Machine	64 GB RAM
Output Format	Binary (or ARPA when faster)

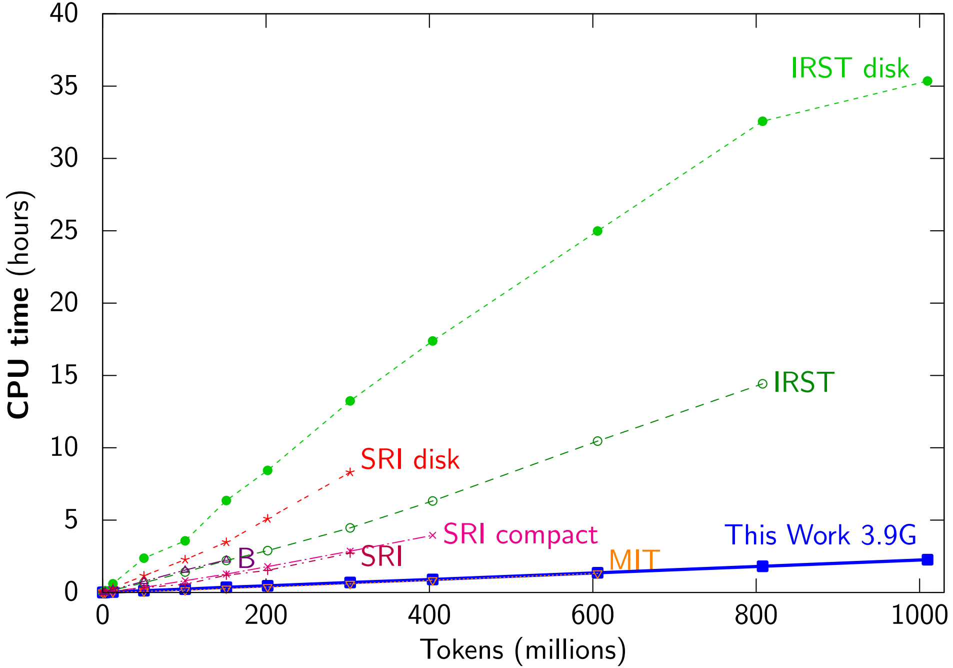
IRST disk: 3-way split. Peak RAM of any one process (as if run serially).
Berkeley: Binary search for minimum JVM memory.


SRI compact
SRI disk
SRI
This Work 3.9G
RAM (GB)
Tokens (millions)
0
10
20
30
40
50
60
0
200
400
600
800
1000



Berkeley
MIT
IRST
SRI compact
SRI disk
SRI
IRST disk
This Work 3.9G
RAM (GB)
Tokens (millions)
0
10
20
30
40
50
60
0
200
400
600
800
1000


Wall time (hours)
Tokens (millions)
0
2
4
6
8
10
12
14
16
0
200
400
600
800
1000
IRST
IRST disk
SRI disk
SRI compact
SRI
B
MIT
This Work 3.9G


IRST disk
IRST
SRI disk
SRI compact
This Work 3.9G
B
SRI
MIT
CPU time (hours)
Tokens (millions)
0
5
10
15
20
25
30
35
40
0
200
400
600
800
1000

Scaling

	Tokens	Smoothing	Machines	Days
This Work	126 billion	Kneser-Ney	**1**	2.8

Counts

	1	2	3	4	5
This Work 126B	393m	3,775m	17,629m	39,919m	59,794m
Pruned Google 1T	14m	315m	977m	1,313m	1,176m

(This work used a machine with 140 GB RAM and a RAID5 array.)

Scaling

	Tokens	Smoothing	Machines	Days	Year
This Work	126 billion	Kneser-Ney	**1**	2.8	2013
Google	31 billion	Kneser-Ney	400	2	2007
Google	230 billion	Kneser-Ney	?	?	2013
Google	1800 billion	Stupid	1500	1	2007

Counts

	1	2	3	4	5
This Work 126B	393m	3,775m	17,629m	39,919m	59,794m
Pruned Google 1T	14m	315m	977m	1,313m	1,176m

(This work used a machine with 140 GB RAM and a RAID5 array.)

Workshop on Statistical MT Results

1. Compress the big LM to 676 GB
2. Decode with 1 TB RAM
3. Make three WMT 2013 submissions

	Czech–English		French–English		Spanish–English	
	Rank	BLEU	Rank	BLEU	Rank	BLEU
This Work	1	28.16	1	33.37	1	32.55
Google	2–3	27.11	2–3	32.62	2	33.65
Baseline	3–5	27.38	2–3	32.57	3–5	31.76

Future Work on Estimation

- Pruning
- Linearly interpolate separately trained models
  $\rightarrow$ SRI's ARPA output is misleading.
- More smoothing methods
- Parallelization by data splitting

Outline

	Speed	RAM	Published
1 Estimation from Text	7.1x	0.07x	ACL 2013
2 Raw Queries	**2.4x**	**0.57x**	**WMT 2011**
3 Decoding	3.2–10.0x	0.85x	IWSLT 2011, EMNLP 2012, NAACL 2013

Raw Queries

Answer language model queries using less time and memory.

$$
\begin{aligned}
\log p(\text{iran} \mid\ & <\text{s}> &&) = -3.33437 \\
\log p(\text{is} \mid\ & <\text{s}>\ \text{iran} &&) = -1.05931 \\
\log p(\text{one} \mid\ & <\text{s}>\ \text{iran is} &&) = -1.80743 \\
\log p(\text{of} \mid\ & <\text{s}>\ \text{iran is one} &&) = -0.03705 \\
\log p(\text{the} \mid\ & \phantom{<\text{s}>}\ \text{iran is one of} &&) = -0.08317 \\
\log p(\text{few} \mid\ & \phantom{<\text{s}>\ \text{iran}}\ \text{is one of the} &&) = -1.20788
\end{aligned}
$$

Example Language Model

Unigrams

Words	log p	log b
<s>	$-\infty$	-2.0
iran	-4.1	-0.8
is	-2.5	-1.4
one	-3.3	-0.9
of	-2.5	-1.1

Bigrams

Words	log p	log b
<s> iran	-3.3	-1.2
iran is	-1.7	-0.4
is one	-2.0	-0.9
one of	-1.4	-0.6

Trigrams

Words	log p
<s> iran is	-1.1
iran is one	-2.0
is one of	-0.3

Example Queries

Unigrams

Words	log p	log b
<s>	$-\infty$	-2.0
iran	-4.1	-0.8
is	-2.5	-1.4
one	-3.3	-0.9
of	-2.5	-1.1

Bigrams

Words	log p	log b
<s> iran	-3.3	-1.2
iran is	-1.7	-0.4
is one	-2.0	-0.9
one of	-1.4	-0.6

Trigrams

Words	log p
<s> iran is	-1.1
iran is one	-2.0
is one of	-0.3

Query: <s> iran is

$$\log p(\text{is} \mid \text{<s> iran}) = -1.1$$

Query: iran is of

$$\begin{array}{lr} \log p(\text{of}) & -2.5 \\ \log b(\text{is}) & -1.4 \\ \log b(\text{iran is}) & + \ -0.4 \\ \hline \log p(\text{of} \mid \text{iran is}) & = -4.3 \end{array}$$

Trie Based

CMU-Cambridge	Early implementation
SRI	Popular, considered fast, high-memory
IRST	Smaller than SRI, single-threaded
MIT	Batch querying
TPT	Memory locality
Joshua	Java, “not as scalable as the SRILM” [Li et al]
Berkeley	Java

Trie Based

CMU-Cambridge	Early implementation
SRI	Popular, considered fast, high-memory
IRST	Smaller than SRI, single-threaded
MIT	Batch querying
TPT	Memory locality
Joshua	Java, “not as scalable as the SRILM” [Li et al]
Berkeley	Java

Lossy Low-Memory

Rand	Bloom maps
Shef	Minimal perfect hashing
Google	Minimal perfect hashing, larger than Shef

KenLM Features

- Faster than all baselines
- Lowest lossless memory
- Multithreaded
- Quick loading via memory mapping
- Easier to compile

Data Structures

Probing Fast.

Trie Small. But still fast.

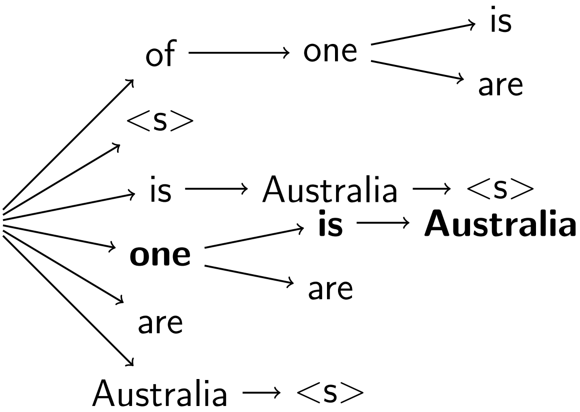
Probing

Hash every n-gram to a 64-bit integer. Ignore collisions.
Store n-grams in custom linear probing hash tables.

Fastest, 24 bytes/n-gram (still less than SRI).

Trie

Reverse *n*-grams, arrange in a trie.



Smaller than most, faster than all but probing.

Optimizing the Trie

CPU Interpolation search instead of binary search [Yehoshua et al, 1978]

RAM Pack at the bit level i.e. $\log p$ has no sign bit

Options to Save More Memory

Cluster floats into 2^f bins, store f bits/float.

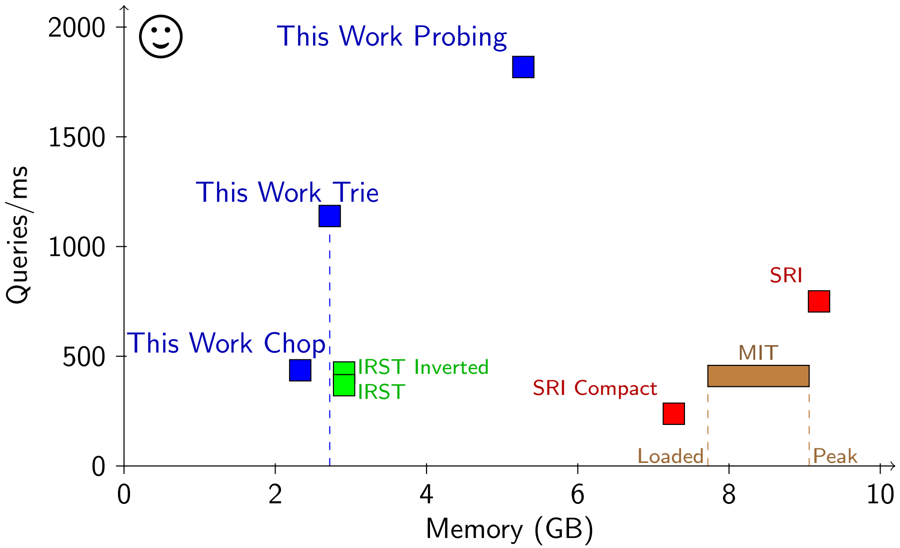
Chop bits from integer sequences, store offsets.

[Whittaker and Raj, 2001; Raj and Whittaker, 2003]

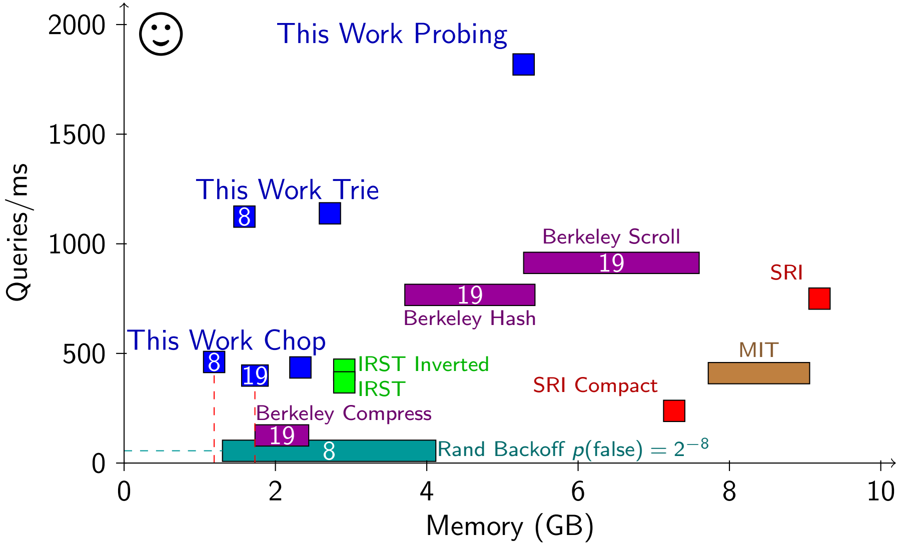
Experiment: Raw Queries

Task	Score the English Gigaword corpus
Model	5-gram Europarl + deduped news crawl 2011
Queries/ms	Excludes loading and file reading time
Loaded RAM	Resident after loading
Peak RAM	Peak virtual after scoring

Raw Queries: Exact Variants



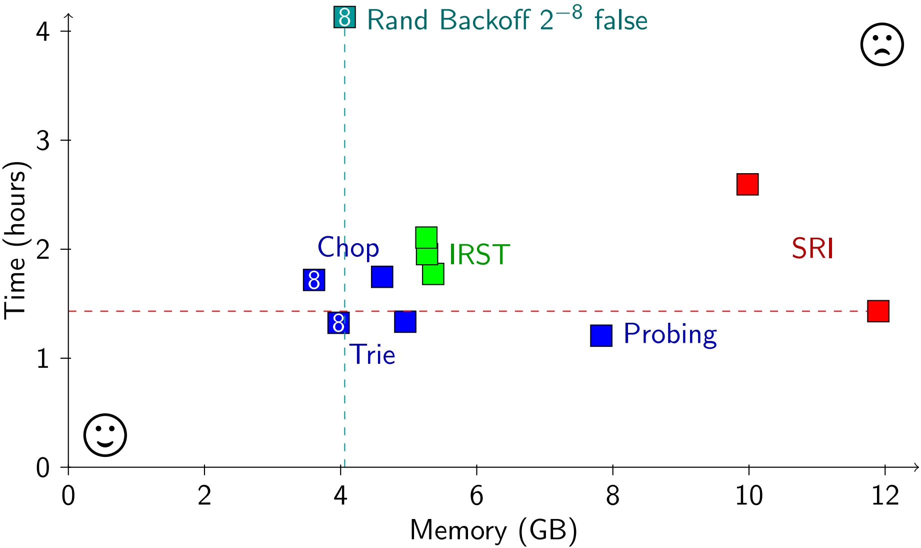
Raw Queries: All Tested Variants



Experiment: Translate 3003 Sentences

Task	WMT 2011 French–English baseline
Decoder	Moses
Model	5-gram Europarl+News LM (same as before)
Formalism	Phrase-based from Europarl
Time	Total wall time, including loading
Memory	Total resident memory after decoding

Moses Benchmarks: Single Threaded

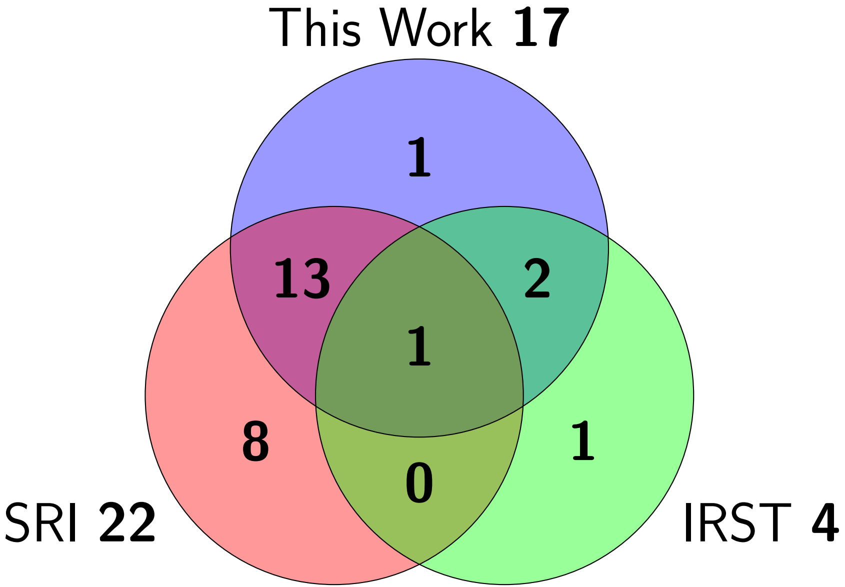


New: Yasuhara et al, EMNLP 2013

"An Efficient Language Model Using Double-Array Structures"

- 19% less RAM and 4-9% faster than this work's probing method.
- More RAM than the trie method.
- 4 days to build a data structure with 936 million *n*-grams.

WMT 2013 Adoption: Any Task



Outline

	Speed	RAM	Published
1 Estimation from Text	7.1x	0.07x	ACL 2013
2 Raw Queries	2.4x	0.57x	WMT 2011
3 Decoding	**3.2–10.0x**	**0.85x**	**IWSLT 2011, EMNLP 2012, NAACL 2013**

Decoding performance includes ≈1.15x speedup from raw queries.
Baseline: SRILM and cube pruning (more later).

Parsing-Based MT is Slow

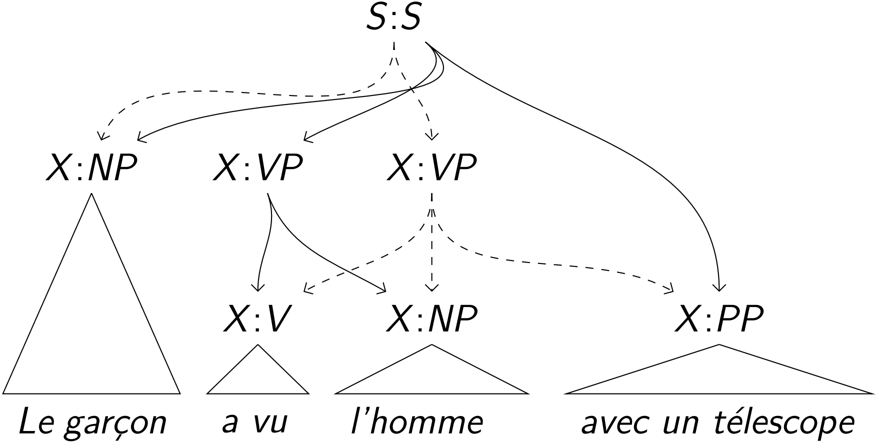
26 CPU hours to translate 3000 sentences

French–English system from Ammar et al [2013] using cdec, a 4-gram LM, and cube pruning with beam size 200.

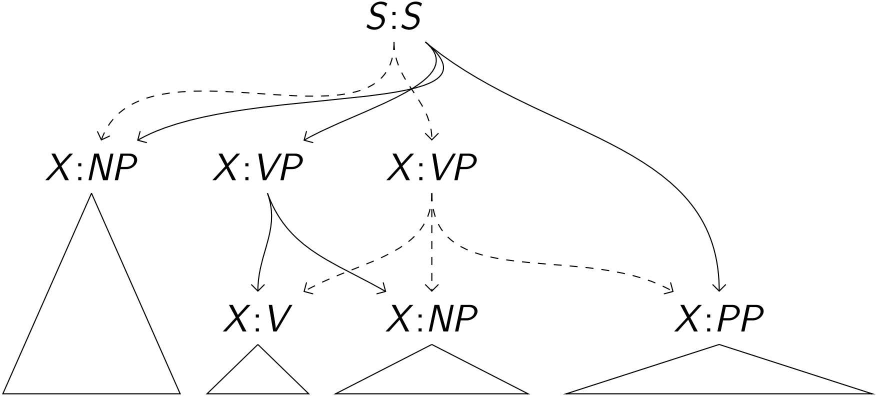
Decoding Example: Input

Le garçon *a vu* *l'homme* *avec un télescope*

Decoding Example: Parse with SCFG

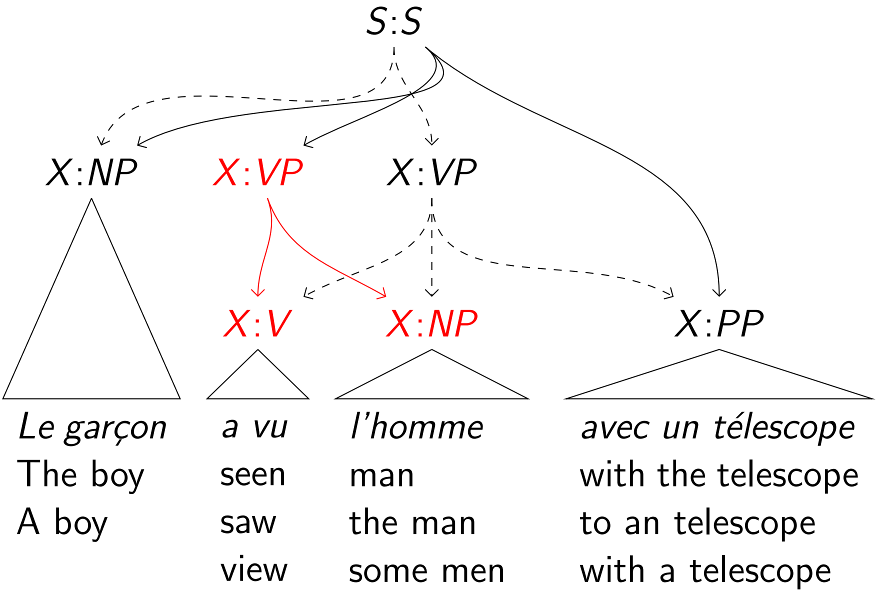


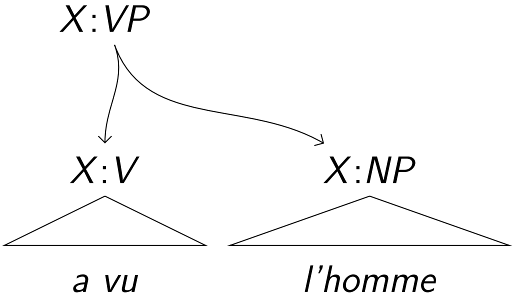
Decoding Example: Read Target Side



Le garçon	*a vu*	*l'homme*	*avec un télescope*
The boy	seen	man	with the telescope
A boy	saw	the man	to an telescope
	view	some men	with a telescope

Decoding Example: One Constituent





Hyp	**Hyp**
seen	man
saw	the man
view	some men

X:*VP*

X:*V*

a vu

Hyp

seen

saw

view

X:*NP*

l'homme

Hyp

man

the man

some men

→

X:*VP*

a vu l'homme

Hypothesis

seen man

seen the man

seen some men

saw man

saw the man

saw some men

view man

view the man

view some men

X:*VP*

X:*V*

a vu

Hyp	Score
seen	-3.8
saw	-4.0
view	-4.0

X:*NP*

l'homme

Hyp	Score
man	-3.6
the man	-4.3
some men	-6.3

X:*VP*

a vu l'homme

Hypothesis	Score
seen man	-8.8
seen the man	-7.6
seen some men	-9.5
saw man	-8.3
saw the man	-6.9
saw some men	-8.5
view man	-8.5
view the man	-8.9
view some men	-10.8

$X\!:\!VP$

$X\!:\!V$ $X\!:\!NP$

a vu

Hyp	Score
seen	-3.8
saw	-4.0
view	-4.0

l'homme

Hyp	Score
man	-3.6
the man	-4.3
some men	-6.3

$\rightarrow$

$X\!:\!VP$

a vu l'homme

Hypothesis	Score
saw the man	-6.9
seen the man	-7.6
saw man	-8.3
saw some men	-8.5
view man	-8.5
seen man	-8.8
view the man	-8.9
seen some men	-9.5
view some men	-10.8

$X{:}VP$

$X{:}V$ $X{:}NP$

a vu

Hyp	Score
seen	-3.8
saw	-4.0
view	-4.0

l'homme

Hyp	Score
man	-3.6
the man	-4.3
some men	-6.3

$X{:}VP$

a vu l'homme

Hypothesis	Score
saw the man	-6.9
seen the man	-7.6
saw man	-8.3
saw some men	-8.5
view man	-8.5
seen man	-8.8
view the man	-8.9
seen some men	-9.5
view some men	-10.8

Scores do not sum

X:*VP*

X:*V*　　*X*:*NP*

a vu

Hyp	Score
seen	-3.8
saw	-4.0
view	-4.0

l'homme

Hyp	Score
man	-3.6
the man	-4.3
some men	-6.3

→

X:*VP*

a vu l'homme

Hypothesis	Score
saw the man	-6.9
seen the man	-7.6
saw man	-8.3
~~saw some men~~	~~-8.5~~
~~view man~~	~~-8.5~~
~~seen man~~	~~-8.8~~
~~view the man~~	~~-8.9~~
~~seen some men~~	~~-9.5~~
~~view some men~~	~~-10.8~~

Pruning is Approximate

Appending Strings

Hypotheses are built by string concatenation.
Language model probability changes when this is done:

$$\frac{p(\text{saw the man})}{p(\text{saw})p(\text{the man})} = \frac{p(\text{the} \mid \text{saw})}{p(\text{the})}\frac{p(\text{man} \mid \text{saw the})}{p(\text{man} \mid \text{the})}$$

Appending Strings

Hypotheses are built by string concatenation.
Language model probability changes when this is done:

$$\frac{p(\text{saw the man})}{p(\text{saw})p(\text{the man})} = \frac{p(\text{the} \mid \text{saw})p(\text{man} \mid \text{saw the})}{p(\text{the}) \qquad p(\text{man} \mid \text{the})}$$

Log probability is part of the score
$\implies$ Scores do not sum
$\implies$ Local decisions may not be globally optimal
$\implies$ Search is hard.

Outline

	Speed	RAM	Published
1 Estimation from Text	7.1x	0.07x	ACL 2013
2 Raw Queries	2.4x	0.57x	WMT 2011
3 Decoding	3.2–10.0x	0.85x	
1 State and Recombination			IWSLT 2011
2 Score Estimates			EMNLP 2012
3 New Search Algorithm			NAACL 2013

Appending Strings

Hypotheses are built by string concatenation.
Language model probability changes when this is done:

$$c(\text{saw} \bullet \text{the man}) = \frac{p(\text{saw the man})}{p(\text{saw})p(\text{the man})} = \frac{p(\text{the} \mid \text{saw})}{p(\text{the})} \frac{p(\text{man} \mid \text{saw the})}{p(\text{man} \mid \text{the})}$$

What words does correction c examine?

Markov Assumption

A 5-gram language model uses up to 4 words of context:

$$p(\text{man} \mid \text{<s> the boy saw the}) = p(\text{man} \mid \text{the boy saw the})$$

$$\Longrightarrow$$

Correction c examines up to 4 words from each string:

$$\text{c}(\text{<s>}\ \underbrace{\boxed{\vdash \text{ the boy saw the}}}_{\text{Right State}} \bullet \underbrace{\boxed{\text{man with a telescope} \dashv}}_{\text{Left State}}\ .)$$

End of state

End of state

Hypotheses have State

X:X

Left State		**Right State**
countries that maintain diplomatic $\dashv$	relations	$\vdash$ with North Korea .
countries that maintain diplomatic $\dashv$	ties	$\vdash$ with North Korea .
countries that maintain some $\dashv$	diplomatic ties	$\vdash$ with North Korea .

State Controls Recombination

X:X

Left State		**Right State**
countries that maintain diplomatic ⊣	relations ties	⊢ with North Korea .
countries that maintain some ⊣	diplomatic ties	⊢ with North Korea .

The decoder may recombine hypotheses with equal state.

State Controls Recombination

X:X

Left State		**Right State**
countries that maintain diplomatic $\dashv$	relations ties	$\vdash$ with North Korea .
countries that maintain some $\dashv$	diplomatic ties	$\vdash$ with North Korea .

The decoder may recombine hypotheses with equal state.

Smaller state
$\Longrightarrow$ More recombination
$\Longrightarrow$ Reason over more hypotheses at once
$\Longrightarrow$ Improved time-accuracy tradeoff.

Efficiently Minimizing State

Li et al [2008]	Criterion for state minimization. "Inefficient implementation"
This Work (IWSLT 2011)	Repurpose log probability sign bit. Use existing lookups. Encode state to make queries faster.

Efficiently Minimizing State

Li et al [2008]	Criterion for state minimization. “Inefficient implementation”
This Work (IWSLT 2011)	Repurpose log probability sign bit. Use existing lookups. Encode state to make queries faster.

11% faster than right state minimization alone.

On hierarchical Chinese–English with beam size 1000.

Outline

	Speed	RAM	Published
1 Estimation from Text	7.1x	0.07x	ACL 2013
2 Raw Queries	2.4x	0.57x	WMT 2011
3 Decoding	3.2–10.0x	0.85x	
1 State and Recombination			IWSLT 2011
2 **Score Estimates**			**EMNLP 2012**
3 New Search Algorithm			NAACL 2013

Baseline: How to Score a Fragment

$$
\begin{array}{rll}
 & \log p_5(\text{is}) & = -2.63 \\
 & \log p_5(\text{one} \mid \text{is}) & = -2.03 \\
 & \log p_5(\text{of} \mid \text{is one}) & = -0.24 \\
 & \log p_5(\text{the} \mid \text{is one of}) & = -0.47 \\
+ & \log p_5(\text{few} \mid \text{is one of the}) & = -1.26 \\
\hline
= & \log p_5(\text{is one of the few}) & = -6.62
\end{array}
$$

The Problem: Lower Order Entries

5-Gram Model: $\log p_5(\text{is}) = -2.63$
Unigram Model: $\log p_1(\text{is}) = -2.30$
Same training data.

The Problem: Lower Order Entries

5-Gram Model: $\log p_5(\text{is}) = -2.63$
Unigram Model: $\log p_1(\text{is}) = -2.30$
Same training data.

In Kneser-Ney, lower orders have adjusted counts.

Build One Model For Each Order

		Baseline	Lower	
	$\log p_5(\text{is})$	$= -2.63$	-2.30	$= \log p_1$
	$\log p_5(\text{one} \mid \text{is})$	$= -2.03$	-1.92	$= \log p_2$
	$\log p_5(\text{of} \mid \text{is one})$	$= -0.24$	-0.08	$= \log p_3$
	$\log p_5(\text{the} \mid \text{is one of})$	$= -0.47$	-0.21	$= \log p_4$
$+$	$\log p_5(\text{few} \mid \text{is one of the})$	$= -1.26$	-1.26	$= \log p_5$
$=$	$\log p_5(\text{is one of the few})$	$= -6.62$	-5.77	$= \log p_{\text{Low}}$

Storing Lower Order Models

One extra float per entry, except for longest order.

Unigrams

Words	**log p_5**	**log b_5**	**log p_1**
australia	-3.9	-0.6	-3.6
is	-2.6	-1.5	-2.3
one	-3.4	-1.0	-2.9
of	-2.5	-1.1	-1.7

No need for backoff b_1

If backoff occurs, use of p_5 is appropriate.

Storing Lower Order Models

One extra float per entry, except for longest order.

Unigrams

Words	**log** p_5	**log** b_5	**log** p_1
australia	-3.9	-0.6	-3.6
is	-2.6	-1.5	-2.3
one	-3.4	-1.0	-2.9
of	-2.5	-1.1	-1.7

No need for backoff b_1

If backoff occurs, use of p_5 is appropriate.

Related: store upper bounds [Carter et al, also EMNLP 2012].

So far	Better estimates but more memory.
Next	Worse estimates with less memory.

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

$$q(<\mathrm{s}> \cdots </\mathrm{s}>) = p(<\mathrm{s}> \cdots </\mathrm{s}>)$$

$$\text{because } b(\cdots </\mathrm{s}>) = 1$$

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

$$q(<\text{s}> \cdots </\text{s}>) = p(<\text{s}> \cdots </\text{s}>)$$

$$\text{because } b(\cdots </\text{s}>) = 1$$

Telescoping

$$q(\text{is}) = p(\text{is})b(\text{is})$$

$$q(\text{one} \mid \text{is}) = p(\text{one} \mid \text{is})\frac{b(\text{is one})b(\text{one})}{b(\text{is})}$$

Saving Memory

Unigrams

Words	**log p_5**	**log b_5**
australia	-3.9	-0.6
is	-2.6	-1.5
one	-3.4	-1.0
of	-2.5	-1.1

$\xrightarrow{\text{Compress}}$

Unigrams

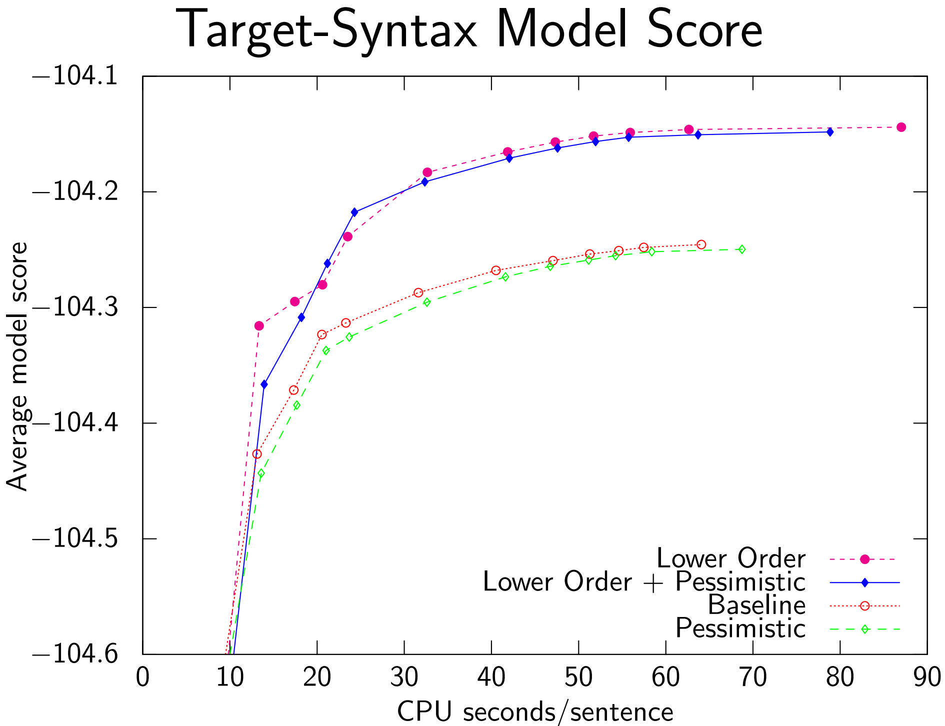
Words	**log q_5**
australia	-4.5
is	-4.1
one	-4.4
of	-3.6

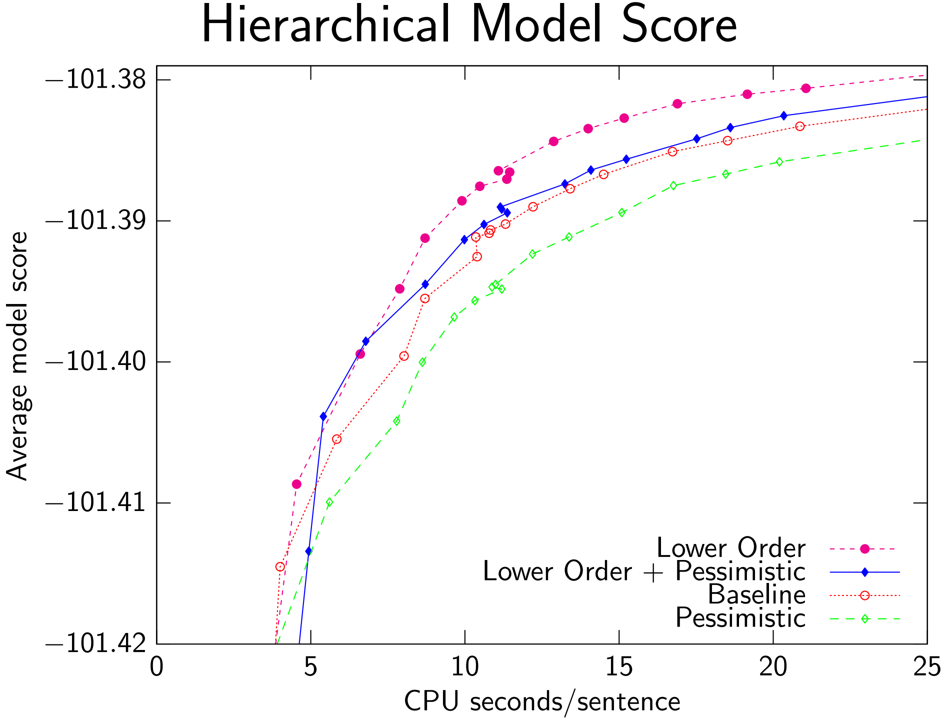
One less float per entry, except for longest order.

Backoff smoothing with RAM comparable to stupid backoff's counts.
Includes Kneser-Ney.

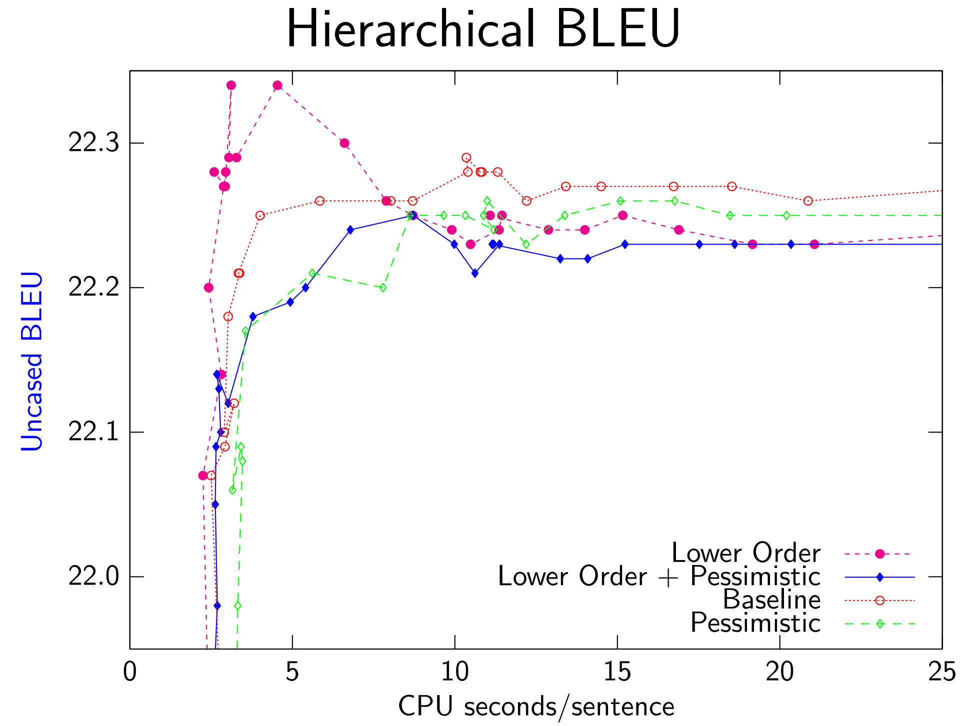
Experiments

Task	WMT 2011 German-English
Decoder	Moses with probing LM + state minimization
LM	5-gram from Europarl, news commentary, and news
Grammar	Target-syntax and hierarchical systems
Parser	Collins


Target-Syntax Model Score
Average model score
CPU seconds/sentence
−104.1
−104.2
−104.3
−104.4
−104.5
−104.6
0
10
20
30
40
50
60
70
80
90
Lower Order
Lower Order + Pessimistic
Baseline
Pessimistic



Hierarchical Model Score
Average model score
CPU seconds/sentence
−101.38
−101.39
−101.40
−101.41
−101.42
0
5
10
15
20
25
Lower Order
Lower Order + Pessimistic
Baseline
Pessimistic



Hierarchical BLEU
Uncased BLEU
22.3
22.2
22.1
22.0
0
5
10
15
20
25
CPU seconds/sentence
Lower Order
Lower Order + Pessimistic
Baseline
Pessimistic

Memory

Effect of adding or removing a float per entry.

Structure	**Baseline (MB)**	**Change (MB)**	**%**
Probing	4,072	517	13%
Trie	2,647	506	19%
8-bit quantized trie	1,236	140	11%
8-bit minimal perfect hash	540	140	26%

Outline

	Speed	RAM	Published
1 Estimation from Text	7.1x	0.07x	ACL 2013
2 Raw Queries	2.4x	0.57x	WMT 2011
3 Decoding	3.2–10.0x	0.85x	
1 State and Recombination			IWSLT 2011
2 Score Estimates			EMNLP 2012
3 **New Search Algorithm**			**NAACL 2013**

X:*VP*

X:*V*

a vu

Hyp	Score
seen	-3.8
saw	-4.0
view	-4.0

X:*NP*

l'homme

Hyp	Score
man	-3.6
the man	-4.3
some men	-6.3

→

X:*VP*

a vu l'homme

Hypothesis	Score
saw the man	-6.9
seen the man	-7.6
saw man	-8.3
~~saw some men~~	~~-8.5~~
~~view man~~	~~-8.5~~
~~seen man~~	~~-8.8~~
~~view the man~~	~~-8.9~~
~~seen some men~~	~~-9.5~~
~~view some men~~	~~-10.8~~

Pruning is Approximate

Beam Search [Lowerre, 1976; Chiang, 2005]

		man	**−3.6**	**the man**	**−4.3**	**some men**	**−6.3**
seen	**−3.8**	seen man	−8.8	seen the man	−7.6	seen some men	−9.5
saw	**−4.0**	saw man	−8.3	saw the man	−6.9	saw some men	−8.5
view	**−4.0**	view man	−8.5	view the man	−8.9	view some men	−10.8

Baseline: Cube Pruning [Chiang, 2007]

		man -3.6	the man -4.3	some men -6.3
seen	-3.8	Queue		
saw	-4.0			
view	-4.0			

Queue

Hypothesis	Sum
→seen man	$-3.8-3.6=-7.4$

Baseline: Cube Pruning [Chiang, 2007]

	man $\mathbf{-3.6}$	**the man** $\mathbf{-4.3}$	**some men** $\mathbf{-6.3}$
seen $\mathbf{-3.8}$	seen man -8.8	Queue	
saw $\mathbf{-4.0}$	Queue		
view $\mathbf{-4.0}$			

Queue

Hypothesis	**Sum**
→ saw man	$-4.0-3.6=-7.6$
seen the man	$-3.8-4.3=-8.1$

Baseline: Cube Pruning [Chiang, 2007]

	man -3.6	**the man** -4.3	**some men** -6.3
seen -3.8	seen man -8.8	Queue	
saw -4.0	saw man -8.3	Queue	
view -4.0	Queue		

Queue

Hypothesis	**Sum**
→ view man	$-4.0-3.6=-7.6$
seen the man	$-3.8-4.3=-8.1$
saw the man	$-4.0-4.3=-8.3$

Problem With Cube Pruning

Hyp
is a
are a

Hypothesis
countries that
countries which
country

Hypothesis
is a countries that
are a countries that
are a countries which
⋮

No notion that "a countries" is bad.

Problem With Cube Pruning

Hyp
is a
are a

Hypothesis
countries that
countries which
country

Hypothesis
is a countries that
are a countries that
are a countries which
⋮

No notion that “a countries” is bad.

Idea: group by outermost words.

Example Hypotheses

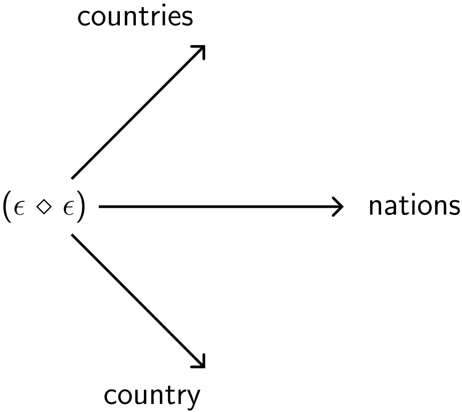
Left State		**Right State**
countries that ⊣	maintain diplomatic relations / ties	⊢ with North Korea .
countries that have ⊣	an embassy in	⊢ DPR Korea .
country ⊣	that maintains some diplomatic ties	⊢ in North Korea .
nations which has ⊣	some diplomatic ties	⊢ with DPR Korea .
country ⊣	that maintains some diplomatic ties	⊢ with DPR Korea .

Example Hypotheses

Left State		**Right State**
(countries that	$\dashv \diamond \vdash$	with North Korea .)
(nations which has	$\dashv \diamond \vdash$	with DPR Korea .)
(countries that have	$\dashv \diamond \vdash$	DPR Korea .)
(country	$\dashv \diamond \vdash$	in North Korea .)
(country	$\dashv \diamond \vdash$	with DPR Korea .)

$\dashv$ Left state is completely present.
$\diamond$ Stands for elided words
$\vdash$ Right state is completely present.

Group by Leftmost Word



Reveal Common Words in Each Group

$$
\begin{tikzcd}[column sep=small]
 & \text{(countries that } \diamond \text{ Korea .)} & \\
(\epsilon \diamond \epsilon) \arrow[ur] \arrow[rr] \arrow[dr] & & \text{(nations which has } \dashv \diamond \vdash \text{ with DPR Korea .)} \\
 & \text{(country } \dashv \diamond \text{ Korea .)} &
\end{tikzcd}
$$

Alternate Sides Until Tree is Full

($\epsilon \diamond \epsilon$)

- (countries that $\diamond$ Korea .)
  - (countries that $\dashv \diamond \vdash$ with North Korea .)
  - (countries that have $\dashv \diamond \vdash$ DPR Korea .)
- (nations which has $\dashv \diamond \vdash$ with DPR Korea .)
- (country $\dashv \diamond$ Korea .)
  - (country $\dashv \diamond \vdash$ in North Korea .)
  - (country $\dashv \diamond \vdash$ with DPR Korea .)

Using Rules

is a X:$NP1$ </s>

turns into

is a $(\epsilon \diamond \epsilon)$ </s>

X:$V1$ the X:$N2$

turns into

$\underbrace{(\epsilon \diamond \epsilon)}_{X:V1}$ the $\underbrace{(\epsilon \diamond \epsilon)}_{X:N2}$

Exploring and Backtracking

Does the LM like "is a (countries that $\diamond$ Korea .) </s>"?

Yes Try more detail.

No Consider alternatives.

Exploring and Backtracking

Does the LM like "is a (countries that $\diamond$ Korea .) </s>"?

Yes Try more detail.

No Consider alternatives.

Formally: priority queue containing breadcrumbs.

Split and Leave Breadcrumbs

(countries that $\dashv \diamond \vdash$ with North Korea .)

(countries that $\diamond$ Korea .)

(countries that have $\dashv \diamond \vdash$ DPR Korea .)

($\epsilon \diamond \epsilon$)

(nations which has $\dashv \diamond \vdash$ with DPR Korea .)

(country $\dashv \diamond \vdash$ in North Korea .)

(country $\dashv \diamond$ Korea .)

(country $\dashv \diamond \vdash$ with DPR Korea .)

Split and Leave Breadcrumbs

(countries that $\dashv \diamond \vdash$ with North Korea .)

(countries that $\diamond$ Korea .)

(countries that have $\dashv \diamond \vdash$ DPR Korea .)

($\epsilon \diamond \epsilon$) [1+]

(nations which has $\dashv \diamond \vdash$ with DPR Korea .)

(country $\dashv \diamond \vdash$ in North Korea .)

(country $\dashv \diamond$ Korea .)

(country $\dashv \diamond \vdash$ with DPR Korea .)

The queue entry

is a $(\epsilon \diamond \epsilon)$ </s>

splits into

Zeroth Child "is a (countries that $\diamond$ Korea .) </s>"
Other Children "is a $(\epsilon \diamond \epsilon)$[1+] </s>"

Children except the zeroth.

Summary So Far

A priority queue contains competing entries:

is a (countries that $\diamond$ Korea .) $</s>$
$(\epsilon \diamond \epsilon)$ the $(\epsilon \diamond \epsilon)$
is a $(\epsilon \diamond \epsilon)[1+]$ $</s>$

The algorithm pops the top entry,
splits a non-terminal, and pushes.

Summary So Far

A priority queue contains competing entries:

is a (countries that $\diamond$ Korea .) </s>
$(\epsilon \diamond \epsilon)$ the $(\epsilon \diamond \epsilon)$
is a $(\epsilon \diamond \epsilon)[1+]$ </s>

The algorithm pops the top entry,
splits a non-terminal, and pushes.

Next: Scoring queue entries

Scores come from the best descendant:

Score($\epsilon \diamond \epsilon$)=
Score(countries that $\dashv \diamond \vdash$ with North Korea .)

$$\geq$$

Score($\epsilon \diamond \epsilon$)[1+]=
Score(nations which has $\dashv \diamond \vdash$ with DPR Korea .)

Estimates Update as Words are Revealed

is a ($\epsilon \diamond \epsilon$) </s> → is a (countries that $\diamond$ Korea .) </s>

$p(\text{is})$	$p(\text{is})$
$p(\text{a} \mid \text{is})$	$p(\text{a} \mid \text{is})$
$p(\text{countries})$	$p(\text{countries} \mid \text{is a})$
$p(\text{that} \mid \text{countries})$	$p(\text{that} \mid \text{is a countries})$
$p(\text{</s>})$	$p(\text{</s>} \mid \text{Korea .})$

Summary: Processing a Constituent

1. **Initialize:** Push rules onto a priority queue.
2. **Best-First Loop:**
   1. Pop the top entry.
   2. If it's complete, add to the beam.
      Otherwise, split and push.
3. **Finalize:** Convert the beam to a tree (lazily).

Summary: Processing a Constituent

1. **Initialize:** Push rules onto a priority queue.
2. **Best-First Loop:**
   1. Pop the top entry.
   2. If it's complete, add to the beam.
      Otherwise, split and push.
3. **Finalize:** Convert the beam to a tree (lazily).

Process constituents in bottom-up order (like cube pruning).

Coarse-to-Fine [Zhang et al, 2008; Petrov et al, 2008]

Decode multiple times, each with more detail:

- LM order
- Word classes

Coarse-to-Fine [Zhang et al, 2008; Petrov et al, 2008]

Decode multiple times, each with more detail:

- LM order
- Word classes

Key Difference

Coarse-to-Fine	Lock-step refinement
This Work	Locally refine on demand

Future Work

- Use this work for each decoding pass
- Word classes for this work

Exact Algorithms

- Weighted finite state transducers [Iglesias et al, 2011]
- Integer linear programming [Rush et al, 2011]
- Later: upper bounds and LM refinement [Aziz et al, WMT 2013]

Currently intractable for large MT (7 hours for a 7-word sentence)
$\Longrightarrow$ Used as first pass of *approximate* coarse-to-fine.

Exact Algorithms

- Weighted finite state transducers [Iglesias et al, 2011]
- Integer linear programming [Rush et al, 2011]
- Later: upper bounds and LM refinement [Aziz et al, WMT 2013]

Currently intractable for large MT (7 hours for a 7-word sentence)
$\Longrightarrow$ Used as first pass of *approximate* coarse-to-fine.

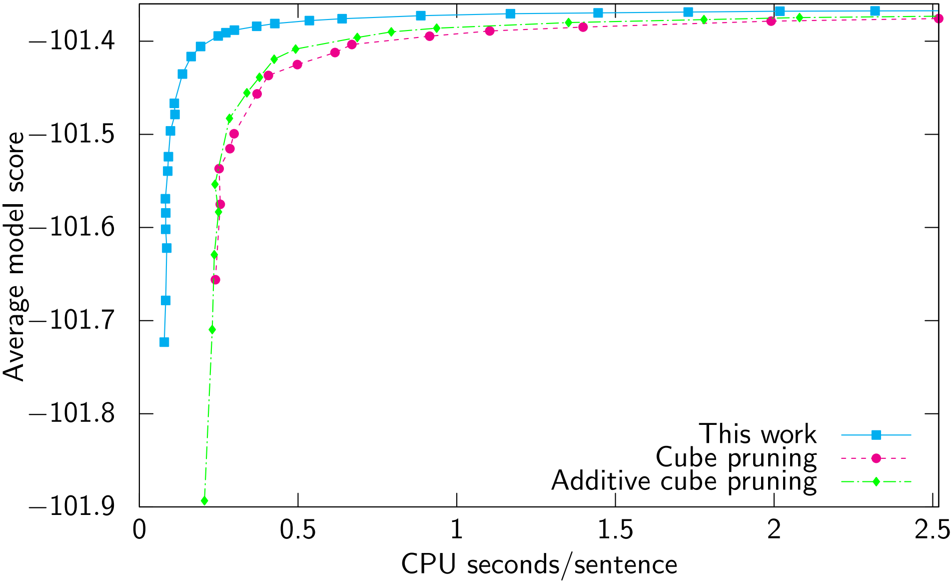
Key Difference

Approximation based on average-case scores *before* expanding hypotheses.

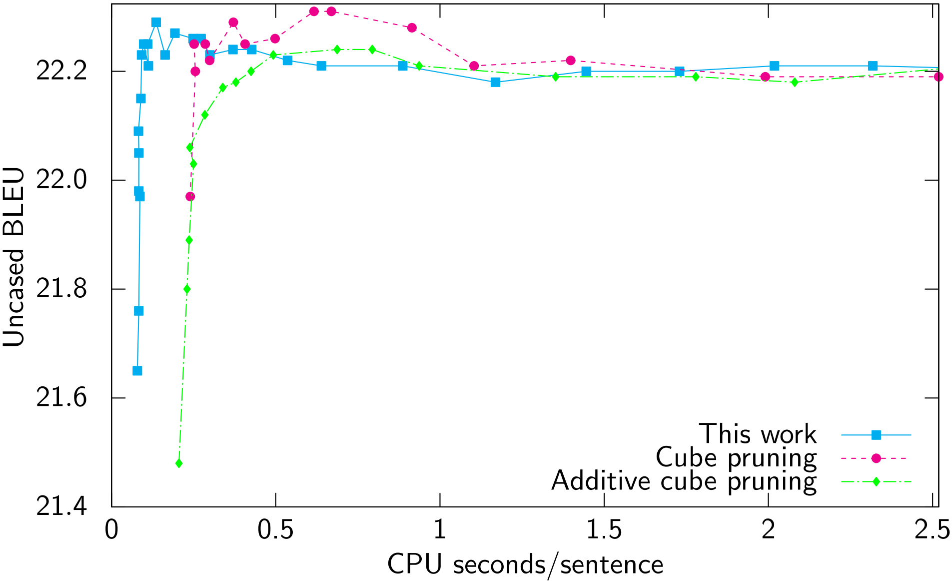
Experiment

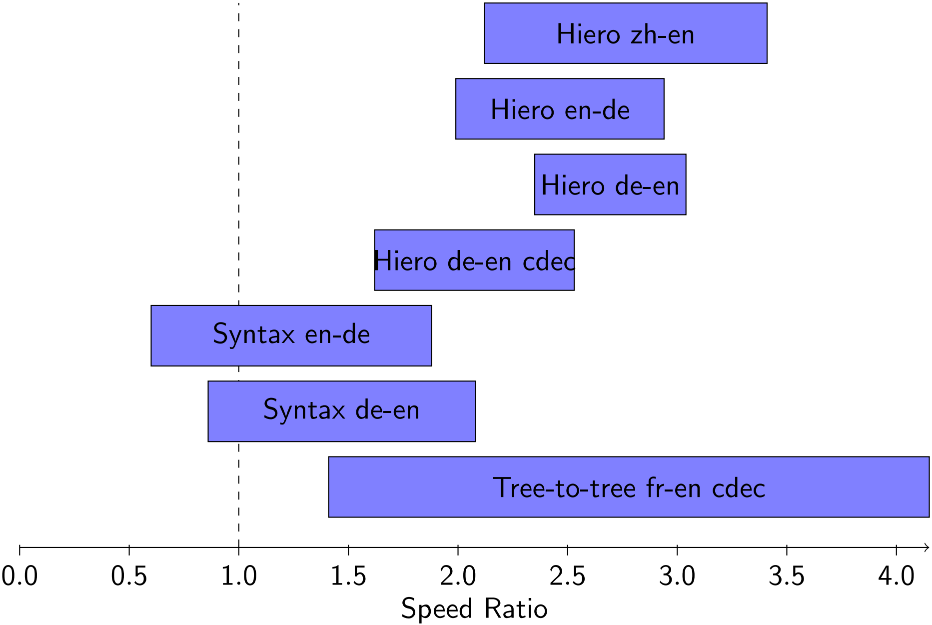
Task	WMT 2011 German-English
Built	[Koehn et al, 2011]
Model	Hierarchical
Decoder	Moses
Baseline	Queries + State + Rest Costs

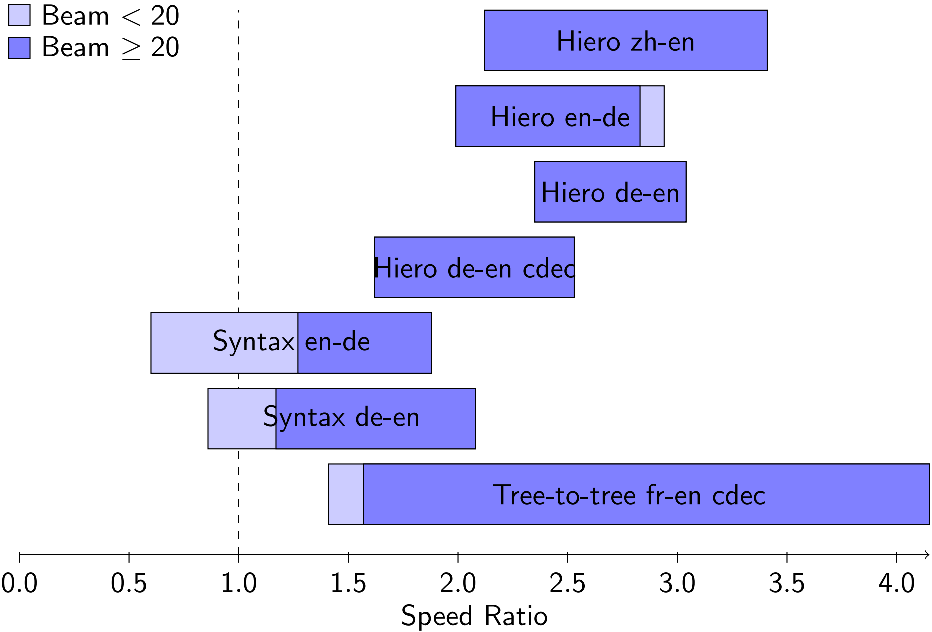
Moses Hierarchical



Moses Hierarchical




Hiero zh-en
Hiero en-de
Hiero de-en
Hiero de-en cdec
Syntax en-de
Syntax de-en
Tree-to-tree fr-en cdec
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Speed Ratio


Beam < 20
Beam ≥ 20
Hiero zh-en
Hiero en-de
Hiero de-en
Hiero de-en cdec
Syntax en-de
Syntax de-en
Tree-to-tree fr-en cdec
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Speed Ratio

Summary

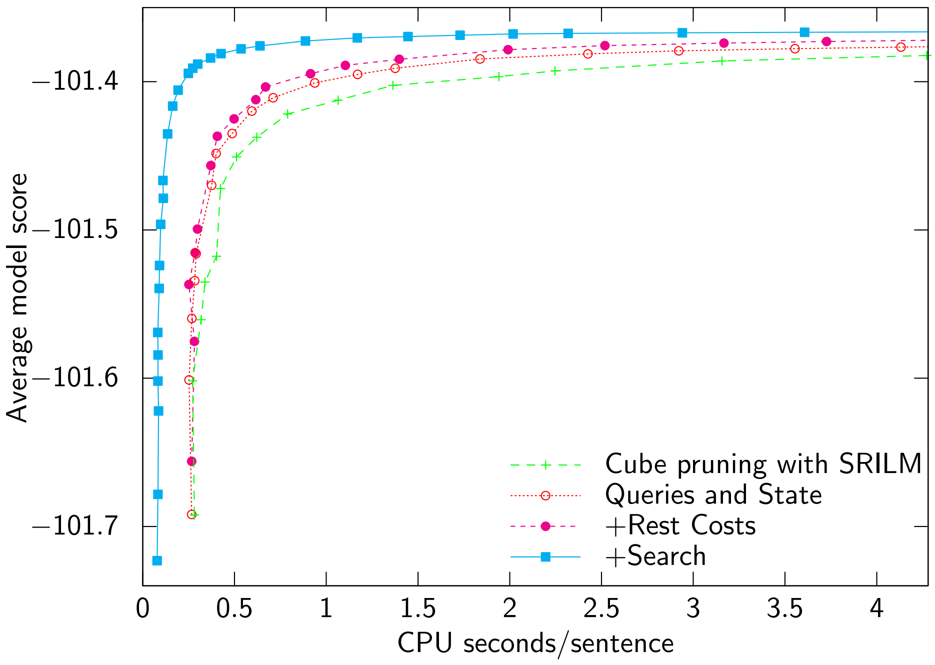
Optimized the entire LM pipeline from estimation to search.

Summary

Optimized the entire LM pipeline from estimation to search.

Comparison

Task	WMT 2011 German-English
Built	[Koehn et al, 2011]
Model	Hierarchical
Decoder	Moses



Average model score
−101.4
−101.5
−101.6
−101.7
0
0.5
1
1.5
2
2.5
3
3.5
4
CPU seconds/sentence
Cube pruning with SRILM
Queries and State
+Rest Costs
+Search

Decoder Support

	Moses	cdec	Joshua	Jane
Estimation	✓	✓	✓	
Queries	✓	✓	✓	✓
State	✓	✓	✓	✓
Rest Costs	✓	✓	✓	✓
Search	✓	✓		

`kheafield.com/code`

Questions?

Full State Minimization

Keep only words that might form cross-hypothesis *n*-grams.

Left State [Joshua]

For any word w, does the model contain:

w countries that maintain some	✗
w countries that maintain	✗
w countries that	✓

$\Longrightarrow$ Left state minimizes to "countries that" $\dashv$.

Full State Minimization

Keep only words that might form cross-hypothesis *n*-grams.

Left State [Joshua]

For any word w, does the model contain:

w countries that maintain some	✗
w countries that maintain	✗
w countries that	✓

$\Longrightarrow$ Left state minimizes to "countries that" $\dashv$.

Right State [SRI, Rand, Joshua]

For any word w, does the model contain:

with North Korea . w ✓

$\Longrightarrow$ Right state minimizes to $\vdash$ "with North Korea ."

Related Work on State

Joshua	Left and right but "inefficient implementation"
SRI	Right only, additional lookups
This Work	Repurpose memory, existing lookups Also: encode state to make queries faster

Experimental Setup

Task	NIST Chinese–English [Koehn, 2011]
LM	Xinhua and AFP from English Gigaword 4 + Parallel Data
Grammar	Hierarchical
Decoder	Moses with cube pruning and faster raw queries

Experimental Setup

Task	NIST Chinese–English [Koehn, 2011]
LM	Xinhua and AFP from English Gigaword 4 + Parallel Data
Grammar	Hierarchical
Decoder	Moses with cube pruning and faster raw queries

11% faster

Experiments: Systems

Hierarchical with Moses [Koehn, 2012]

- German–English also ported to cdec, Joshua, and Jane
- English–German
- Chinese–English

Target Syntax with Moses [Koehn, 2012]

- German–English
- English–German

Tree-to-Tree with cdec [Ammar et al, 2013]

- French–English

Experiments: Systems and Scenarios

Hierarchical with Moses [Koehn, 2012]

- German–English also ported to cdec, Joshua, and Jane
- English–German
- Chinese–English

Target Syntax with Moses [Koehn, 2012]

- German–English
- English–German

Tree-to-Tree with cdec [Ammar et al, 2013]

- French–English

Baseline and improved rest costs, 2–3 flavors of cube pruning.