Optimal Beam Search for Machine Translation

Alexander Rush, Yin-Wen Chang, and Michael Collins
EMNLP 2013

October 19, 2013
Beam Search

beam search is the *de facto* method for translation decoding

- very fast even in the worst-case
- accurate in practice
- implemented in many real-world systems
beam search is the *de facto* method for translation decoding

- very fast even in the worst-case
- accurate in practice
- implemented in many real-world systems

however it provides no formal guarantees about search error
**Goal**

**goal:** fast, optimal translation decoding
- better understand what makes translation hard
- quantify search error from beam search
- (at some point) improve translation accuracy
Overview

1. translation as constrained graph search
2. certificate properties of beam search
3. relaxation methods for bounding
4. experimental results
Tasks

- phrase-based translation

  wir müssen **diese kritik** ernst nehmen
  we must take **this criticism** seriously

- syntax-based translation
Overview

1. translation as constrained graph search

2. certificate properties of beam search

3. relaxation methods for bounding

4. experimental results
Phrase-Based Translation Score

- $\omega$; the translation model score
- $\sigma$; the language model score

example:

wir müssen diese kritik ernst nehmen

score =
Phrase-Based Translation Score

- $\omega$; the translation model score
- $\sigma$; the language model score

example:

\begin{align*}
\text{wir müssen} & \text{ diese kritik ernst nehmen} \\
\text{we must} & \text{ take this criticism seriously}
\end{align*}

score $= \omega(\text{wir müssen}, \text{we must}) + \sigma(\langle s \rangle, \text{we}) + \sigma(\text{we, must}) + \omega(\text{nehmen}, \text{take}) + \sigma(\text{must, take}) + \omega(\text{diese kritik}, \text{this criticism}) + \sigma(\text{take, this}) + \sigma(\text{these, criticism}) + \omega(\text{ernst, seriously}) + \sigma(\text{criticism, seriously}) + \sigma(\text{seriously, } \langle s \rangle)$
Phrase-Based Translation Score

- $\omega$; the translation model score
- $\sigma$; the language model score

**example:**

wir müssen diese kritik ernst **nehmen**
we must **take**

score = $\omega$(wir müssen, we must) + $\sigma$(<s>, we) + $\sigma$(we, must) + $\omega$(nehmen, take) + $\sigma$(must, take) +
Phrase-Based Translation Score

- $\omega$; the translation model score
- $\sigma$; the language model score

example:

wir müssen **diese kritik** ernst nehmen
we must take **this criticism**

score = $\omega$(wir müssen, we must) + $\sigma$(<s>, we) + $\sigma$(we, must) + $\omega$(nehmen, take) + $\sigma$(must, take) + $\omega$(diese kritik, this criticism) + $\sigma$(take, this) + $\sigma$(these, criticism) +
Phrase-Based Translation Score

- $\omega$; the translation model score
- $\sigma$; the language model score

**example:**

wir müssen diese kritik ernst nehmen
we must take this criticism seriously

score = $\omega$(wir müssen, we must) + $\sigma$($<$s$, we$) + $\sigma$(we, must) +
$\omega$(nehmen, take) + $\sigma$(must, take) +
$\omega$(diese kritik, this criticism) + $\sigma$(take, this) + $\sigma$(these, criticism) +
$\omega$(ernst, seriously) + $\sigma$(criticism, seriously) + $\sigma$(seriously, $</s>$)
Problem Representation

represent all possible translations in a weighted directed graph

- $\mathcal{V}, \mathcal{E}$; the vertices/edges in the graph.
- $\theta \in \mathbb{R}^{||\mathcal{E}||}$; the weights on edges.
- Vertices are # of source words used and last English word.

\[
\theta(e) = \omega(\text{diese kritik, this criticism}) + \\
\sigma(\text{must, this}) + \sigma(\text{this, criticism})
\]
take criticism must seriously
Best Path Problem

- $V, E$; the vertices/edges in the graph
- $\theta \in \mathbb{R}^{|E|}, \tau \in \mathbb{R}$; the weights on edges and an offset
- $X \subset \{0, 1\}^{|E|}$; the paths in the graph

\[
\max_{y \in X} \sum_{e \in E} \theta(e)y(e) + \tau = \max_{y \in X} \theta^\top y + \tau
\]

- there are an exponential number of paths $|X|$.

- can be solved in polynomial time, $O(|E|)$. 
wir müssen diese kritik ernst nehmen
wir müssen **diese kritik** ernst nehmen
**this criticism**
we must take criticism seriously
we must take this criticism seriously
wir müssen diese kritik ernst nehmen
this criticism we must this criticism
Constrained Paths

**problem:** constrain maximization to valid paths

- \( A \in \mathbb{R}^{|b| \times |\mathcal{E}|} \); a matrix of linear constraints
- \( b \in \mathbb{R}^{|b|} \); a constraint vector
- \( Ay \); a signature.

constrained paths

\[
\mathcal{X}' = \{ y \in \mathcal{X} : Ay = b \}
\]
Example: Source-Language Constraints

\[ A = \begin{pmatrix}
1 & 2 & 3 & 4 & \ldots \\
1 & 0 & 1 & 0 & \ldots \\
1 & 0 & 1 & 0 & \ldots \\
0 & 0 & 0 & 0 & \ldots \\
0 & 0 & 0 & 0 & \ldots \\
0 & 1 & 0 & 0 & \ldots \\
0 & 0 & 0 & 1 & \ldots
\end{pmatrix} \]

▶ each word must be used exactly once for valid \( y \in \mathcal{X}' \).

\[ Ay = b = \begin{pmatrix}
wir \\
müssen \\
diese \\
kritik \\
nehmen \\
ernst
\end{pmatrix} \begin{pmatrix} 1 \\
1 \\
1 \\
1 \\
1 \end{pmatrix} \]
wir müssen diese kritik ernst nehmen

<s>
</s>

take
criticism
must
seriously

Ay =

\[
\begin{pmatrix}
1 \\
1 \\
2 \\
2 \\
0 \\
0 \\
\end{pmatrix}
\]
Overview

1. translation as constrained graph search
2. certificate properties of beam search
3. relaxation methods for bounding
4. experimental results
**Beam Search**

**beam search:** explore hypotheses in order.

for each hypothesis check:

1. the hypothesis has a **valid** signature
2. the hypothesis is possibly still **optimal**
3. the hypothesis fits in the **beam**
Check 1: Signature Check

is this a valid bitstring?

\[ Ay = \begin{pmatrix}
   \text{wir} \\
   \text{müssen} \\
   \text{diese} \\
   \text{kritik} \\
   \text{nehmen} \\
   \text{ernst}
\end{pmatrix} \]

\[
\begin{pmatrix}
   1 \\
   1 \\
   1 \\
   1 \\
   0 \\
   0
\end{pmatrix}
\]
Check 2: Bounding

assume we can compute

- \( \text{lb} \leq \text{opt} \); a lower-bound on the optimal score
- \( \text{ubs} \in \mathbb{R}^{|
\n\n|} \); upper bounds on future scores

does the inequality \( \theta^\top y + \text{ubs}(v) \geq \text{lb} \) hold?
Check 3: Pruning

are there less than $\beta$ hypotheses better than $y$ in “beam”?

$\triangleright \beta$; size of a “hard” beam threshold.
Properties

1. **Feasible Path**
   the result returned by beam search is a lower bound on the optimal score.

2. **Certificate**
   if pruning (check 3) is never applied, the result returned by beam search is optimal.
Properties

1. **Feasible Path**
   the result returned by beam search is a lower bound on the optimal score.

2. **Certificate**
   if pruning (check 3) is never applied, the result returned by beam search is optimal.

*unfortunately* in practice certificates are rare.
Preview Results: Beam Search Optimality

![Bar chart showing the percentage of sentences for different beam search settings: baseline, Beam (100), Beam (1000), and Beam (100000). The chart compares 'certificate' and 'exact' tasks.]
Preview Results: Beam Search Certificates

![Graph showing the percentage of sentences for different beam sizes. The y-axis represents the percentage of sentences, and the x-axis represents different ranges of sentence lengths (10-20, 20-30, 30-40, 40-50, 50-). The graph compares baseline, Beam (100), Beam (1000), and Beam (100000).]
Overview

1. translation as constrained graph search
2. certificate properties of beam search
3. relaxation methods for bounding
4. experimental results
Lagrangian

recall:
\[
\max_{y \in \mathcal{X}'} \theta^T y + \tau
\]
\[
\mathcal{X}' = \{ y \in \mathcal{X} : Ay = b \}
\]

define the Lagrangian with multipliers \( \lambda \in \mathbb{R}^{|b|} \)
\[
L(\lambda) = \max_{y \in \mathcal{X}'} \theta^T y + \tau - \lambda^T (Ay - b)
\]
\[
= \max_{y \in \mathcal{X}'} (\theta^T - \lambda^T A)y + (\tau - \lambda^T b)
\]
\[
= \max_{y \in \mathcal{X}'} \theta'^T y + \tau'
\]
Properties

Lagrangian

\[ L(\lambda) = \max_{y \in \mathcal{X}} \theta^\top y + \tau - \lambda^T (Ay - b) = \max_{y \in \mathcal{X}} \theta'^\top y + \tau' \]

- **Preserves Constrained Scores**
  if \( y \in \mathcal{X}' \) then it has the same objective with modified weights

\[ \theta'^\top y + \tau' = \theta^\top y + \tau \]

- **Upper Bound**
  the best path is an upper bound on the optimal score

\[ L(\lambda) = \text{ub} \geq \text{opt} \]
Tighter Upper Bounds

**goal:** tightest upper bound

\[
\min_{\lambda \in \mathbb{R}^{\|b\|}} L(\lambda)
\]

**strategy:** minimize by subgradient descent ($\alpha_k \in \mathbb{R}$ is a rate)

\[
y = \arg \max_{y \in \mathcal{X}} \theta' \top y + \tau'
\]

\[
\lambda^{(k)} \leftarrow \lambda^{(k-1)} - \alpha_k (Ay - b)
\]
Preview Results: Lagrangian Relaxation and Bounds

Distribution of Sentences by Tightness of Bound

% of total sentences vs. binned gap $[L(\lambda)\text{-opt}]$

- Blue line: Original weights
Preview Results: Lagrangian Relaxation and Bounds

Distribution of Sentences by Tightness of Bound

% of total sentences

binned gap \([L(\lambda)-\text{opt}]\)

- Blue: Original weights
- Red: 5 LR rounds

-(0, 0.5]  (0.5, 1]  (1, 2]  (2, 4]  (4, 8]  (8, 100]

- 0  10  20  30  40  50
Preview Results: Lagrangian Relaxation and Bounds

Distribution of Sentences by Tightness of Bound

- **Original weights**
- **5 LR rounds**
- **10 LR rounds**

% of total sentences

binned gap \([L(\lambda)-\text{opt}]\)

- (0, 0.5]
- (0.5, 1]
- (1, 2]
- (2, 4]
- (4, 8]
- (8, 100]
Putting It All Together

Lagrangian Relaxation
- can decrease the upper bound score
- may not find optimal solution

Beam Search
- can increase the lower bound score
- tighter bounds increase chance of optimal solution

**strategy:** use modified weights from LR in Beam Search
procedure $\text{OptBeam}(\alpha, \beta)$

\begin{align*}
    &\lambda^{(0)} \leftarrow 0 \\
    &\text{lb}^{(0)} \leftarrow -\infty \\
    \text{for } k \text{ in } 1 \ldots K \text{ do} \\
    &\text{ub}^{(k)}, \lambda^{(k)}, \theta', \tau' \leftarrow \text{LRSubgradient}(\alpha_k, \lambda^{(k-1)}) \\
    &\text{lb}^{(k)}, \text{cert} \leftarrow \text{BeamSearch}(\theta', \tau', \text{lb}^{(k-1)}, \beta_k) \\
    &\text{if } \text{cert} \text{ then return } \text{lb}^{(k)}
\end{align*}
procedure OptBeam(\(\alpha, \beta\))
\[\lambda^{(0)} \leftarrow 0\]
\[\text{lb}^{(0)} \leftarrow -\infty\]
for \(k\) in 1 \ldots K do
\[\text{ub}^{(k)}, \lambda^{(k)}, \theta', \tau' \leftarrow \text{LRSubgradient}(\alpha_k, \lambda^{(k-1)})\]
\[\text{lb}^{(k)}, \text{cert} \leftarrow \text{BeamSearch}(\theta', \tau', \text{lb}^{(k-1)}, \beta_k)\]
if cert then return \(\text{lb}^{(k)}\)
procedure $\text{OptBeam}(\alpha, \beta)$

\[
\begin{align*}
\lambda^{(0)} & \leftarrow 0 \\
\text{lb}^{(0)} & \leftarrow -\infty \\
\text{for } k \text{ in } 1 \ldots K \text{ do} \\
\text{ub}^{(k)}, \lambda^{(k)}, \theta', \tau' & \leftarrow \text{LRSubgradient}(\alpha_k, \lambda^{(k-1)}) \\
\text{lb}^{(k)}, \text{cert} & \leftarrow \text{BeamSearch}(\theta', \tau', \text{lb}^{(k-1)}, \beta_k) \\
\text{if } \text{cert } \text{then return } \text{lb}^{(k)}
\end{align*}
\]
procedure \textsc{OptBeam}(\alpha, \beta)
\begin{align*}
&\lambda^{(0)} \leftarrow 0 \\
&\text{lb}^{(0)} \leftarrow -\infty \\
&\text{for } k \text{ in } 1 \ldots K \text{ do} \\
&\quad \text{ub}^{(k)}, \lambda^{(k)}, \theta', \tau' \leftarrow \text{LRSubgradient}(\alpha_k, \lambda^{(k-1)}) \\
&\quad \text{lb}^{(k)}, \text{cert} \leftarrow \text{BeamSearch}(\theta', \tau', \text{lb}^{(k-1)}, \beta_k) \\
&\quad \text{if cert then return } \text{lb}^{(k)}
\end{align*}
procedure OptBeam(α, β)
    \( \lambda^{(0)} \leftarrow 0 \)
    \( \text{lb}^{(0)} \leftarrow -\infty \)
    for \( k \) in 1 \ldots K \ do
        \( \text{ub}^{(k)}, \lambda^{(k)}, \theta', \tau' \leftarrow \text{LRSubgradient}(\alpha_k, \lambda^{(k-1)}) \)
        \( \text{lb}^{(k)}, \text{cert} \leftarrow \text{BeamSearch}(\theta', \tau', \text{lb}^{(k-1)}, \beta_k) \)
        if cert then return \( \text{lb}^{(k)} \)
procedure $\text{OptBeam}(\alpha, \beta)$
\[
\begin{align*}
\lambda^{(0)} & \leftarrow 0 \\
\text{lb}^{(0)} & \leftarrow -\infty
\end{align*}
\]
for $k$ in 1 \ldots K do
\[
\begin{align*}
\text{ub}^{(k)}, \lambda^{(k)}, \theta', \tau' & \leftarrow \text{LRSubgradient}(\alpha_k, \lambda^{(k-1)}) \\
\text{lb}^{(k)}, \text{cert} & \leftarrow \text{BeamSearch}(\theta', \tau', \text{lb}^{(k-1)}, \beta_k)
\end{align*}
\]
if cert then return $\text{lb}^{(k)}$
procedure \textsc{OptBeam}(\alpha, \beta) \\
\begin{align*}
\lambda^{(0)} &\leftarrow 0 \\
lb^{(0)} &\leftarrow -\infty \\
\text{for } k \text{ in } 1 \ldots K \text{ do} \\
&\quad ub^{(k)}, \lambda^{(k)}, \theta', \tau' \leftarrow \textsc{LRSubgradient}(\alpha_k, \lambda^{(k-1)}) \\
&\quad lb^{(k)}, \text{cert} \leftarrow \textsc{BeamSearch}(\theta', \tau', lb^{(k-1)}, \beta_k) \\
&\quad \text{if cert then return } lb^{(k)}
\end{align*}
procedure OptBeam(α, β)

\[\lambda^{(0)} \leftarrow 0\]
\[\text{lb}^{(0)} \leftarrow -\infty\]

for \( k \) in 1 ... \( K \) do

\[\text{ub}^{(k)}, \lambda^{(k)}, \theta', \tau' \leftarrow \text{LRSubgradient}(\alpha_k, \lambda^{(k-1)})\]
\[\text{lb}^{(k)}, \text{cert} \leftarrow \text{BeamSearch}(\theta', \tau', \text{lb}^{(k-1)}, \beta_k)\]

if cert then return \( \text{lb}^{(k)} \)
procedure OptBeam(α, β)
    \( \lambda^{(0)} \leftarrow 0 \)
    \( \text{lb}^{(0)} \leftarrow -\infty \)
    for \( k \) in 1 \ldots K \ do
        \( \text{ub}^{(k)}, \lambda^{(k)}, \theta', \tau' \leftarrow \text{LRSubgradient}(\alpha_k, \lambda^{(k-1)}) \)
        \( \text{lb}^{(k)}, \text{cert} \leftarrow \text{BeamSearch}(\theta', \tau', \text{lb}^{(k-1)}, \beta_k) \)
        if \( \text{cert} \) then return \( \text{lb}^{(k)} \)
procedure OptBeam($\alpha, \beta$)

$\lambda^{(0)} \leftarrow 0$

$lb^{(0)} \leftarrow -\infty$

for $k$ in 1...$K$ do

$ub^{(k)}, \lambda^{(k)}, \theta', \tau' \leftarrow \text{LRSubgradient}(\alpha_k, \lambda^{(k-1)})$

$lb^{(k)}, \text{cert} \leftarrow \text{BeamSearch}(\theta', \tau', lb^{(k-1)}, \beta_k)$

if cert then return $lb^{(k)}$
procedure $\text{OptBeam}(\alpha, \beta)$

$\lambda^{(0)} \leftarrow 0$

$\text{lb}^{(0)} \leftarrow -\infty$

for $k$ in 1...$K$ do

$\text{ub}^{(k)}, \lambda^{(k)}, \theta', \tau' \leftarrow \text{LRSubgradient}(\alpha_k, \lambda^{(k-1)})$

$\text{lb}^{(k)}, \text{cert} \leftarrow \text{BeamSearch}(\theta', \tau', \text{lb}^{(k-1)}, \beta_k)$

if cert then return $\text{lb}^{(k)}$
procedure $\text{OptBeam}(\alpha, \beta)$

$\lambda^{(0)} \leftarrow 0$

$\text{lb}^{(0)} \leftarrow -\infty$

\begin{algorithmic}
\For{$k$ in $1 \ldots K$}
    \State $\text{ub}^{(k)}, \lambda^{(k)}, \theta', \tau' \leftarrow \text{LRSubgradient}(\alpha_k, \lambda^{(k-1)})$
    \State $\text{lb}^{(k)}, \text{cert} \leftarrow \text{BeamSearch}(\theta', \tau', \text{lb}^{(k-1)}, \beta_k)$
    \If{cert} \textbf{return} $\text{lb}^{(k)}$
\EndIf
\EndFor
\end{algorithmic}
**procedure** OptBeam\((\alpha, \beta)\)

\[
\begin{aligned}
\lambda^{(0)} & \leftarrow 0 \\
lb^{(0)} & \leftarrow -\infty \\
\text{for } k \text{ in } 1 \ldots K \text{ do} & \\
ub^{(k)}, \lambda^{(k)}, \theta', \tau' & \leftarrow \text{LRSubgradient}(\alpha_k, \lambda^{(k-1)}) \\
lb^{(k)}, \text{cert} & \leftarrow \text{BeamSearch}(\theta', \tau', lb^{(k-1)}, \beta_k) \\
\text{if } \text{cert} \text{ then return } lb^{(k)}
\end{aligned}
\]
Preview Results: When is Beam Search Optimal

Beam Search Certificates by Tightness of Bound

% with certificate

binned gap \([L(\lambda)-\text{opt}]\)

\((0, 0.5]\) \quad (0.5, 1]\) \quad (1, 2]\) \quad (2, 4]\) \quad (4, 8]\) \quad (8, 100]\)

\(\beta = 10000\)

\(\beta = 1000\)
Overview

1. translation as constrained graph search
2. certificate properties of beam search
3. relaxation methods for bounding
4. experimental results
Data Sets

Phrase-Based Translation

- 1,824 sentences German-to-English sentences from Europarl (Koehn, 2005)
- trigram language model and distortion penalties
- full graph structure from Chang and Collins (2011)

Syntax-Based Translation

- 691 Chinese-to-English sentences from Huang and Mi (2010)
- tree-to-string translation model and trigram language model
- full hypergraph structure from Rush and Collins (2011)
Baselines: Provable Guarantees

- **Beam**: a beam search decoder based on original weights.
- **AStar**: A* search using original weights, $\theta$ and $\tau$.
- **ILP**: a general-purpose integer linear programming solver.
- **LR-Tight**: LR with incremental constraints.
- **OptBeam**: this work.
Syntax-Based Optimal Methods

- Beam (100)
- Beam (1000)
- ILP
- LR-Tight
- OptBeam

Bar chart showing:
- Sentences per minute
- Speed

Results for:
- Certificate
- Exact
Baselines: Approximate Methods

- **Moses-GC**: the standard Moses beam search decoder.
- **Moses**: Moses without gap constraints (see Chang and Collins (2011)).
- **CubePruning**: standard syntax-based decoding algorithm.
Phrase-Based Approximate Methods
Syntax-Based Approximate Methods

The graph shows the performance comparison of different methods in terms of sentences per minute. The methods compared are Cube (100), Cube (1000), and OptBeam.

The top graph demonstrates the speed performance, where Cube (100) is significantly faster than the other two methods.

The bottom graph illustrates the percentage of sentences in 'certificate' and 'exact' categories. Cube (100) shows a higher percentage in both categories compared to Cube (1000) and OptBeam.
Conclusion

summary:
- reviewed conditions for exact beam search for translation.
- used Lagrangian relaxation to improve upper bounds.
- empirically method is effective at finding optimal solutions.

future work:
- training full system using exact decoding.
- bounding technique applied to other approximate algorithms.
- building a toolkit for constrained dynamic programming.

thank you