Variational Decoding for Statistical Machine Translation

Zhifei Li, Jason Eisner, and Sanjeev Khudanpur
Center for Language and Speech Processing
Computer Science Department
Johns Hopkins University
Spurious Ambiguity

• Statistical models in MT exhibit spurious ambiguity
  • Many different derivations (e.g., trees or segmentations) generate the same translation string

• Regular phrase-based MT systems
  • phrase segmentation ambiguity

• Tree-based MT systems
  • derivation tree ambiguity
Spurious Ambiguity in Phrase Segmentations

• Same output: “machine translation software”
• Three different phrase segmentations
Spurious Ambiguity in Derivation Trees

- Same output: “machine translation software”
- Three different derivation trees
### Maximum A Posterior (MAP) Decoding

<table>
<thead>
<tr>
<th>Translation String</th>
<th>MAP</th>
<th>Derivation</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>red translation</td>
<td></td>
<td></td>
<td>0.16</td>
</tr>
<tr>
<td>blue translation</td>
<td></td>
<td></td>
<td>0.14</td>
</tr>
<tr>
<td>green translation</td>
<td></td>
<td></td>
<td>0.14</td>
</tr>
</tbody>
</table>

- **Exact MAP decoding**

\[
y^* = \arg \max_{y \in \text{Trans}(x)} p(y|x)
\]

\[
y^* = \arg \max_{y \in \text{Trans}(x)} \sum_{d \in D(x,y)} p(y, d|x)
\]

- **x**: Foreign sentence
- **y**: English sentence
- **d**: derivation

Monday, August 17, 2009
Maximum A Posterior (MAP) Decoding

<table>
<thead>
<tr>
<th>translation string</th>
<th>MAP</th>
<th>derivation</th>
<th>probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>red translation</td>
<td>0.28</td>
<td></td>
<td>0.16</td>
</tr>
<tr>
<td>blue translation</td>
<td></td>
<td></td>
<td>0.14</td>
</tr>
<tr>
<td>green translation</td>
<td></td>
<td></td>
<td>0.14</td>
</tr>
</tbody>
</table>

- Exact MAP decoding

\[ y^* = \arg \max_{y \in \text{Trans}(x)} p(y|x) \]
\[ = \arg \max_{y \in \text{Trans}(x)} \sum_{d \in D(x, y)} p(y, d|x) \]

- **x**: Foreign sentence
- **y**: English sentence
- **d**: derivation
# Maximum A Posterior (MAP) Decoding

<table>
<thead>
<tr>
<th>Translation String</th>
<th>MAP</th>
<th>Derivation</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>red translation</td>
<td>0.28</td>
<td></td>
<td>0.16</td>
</tr>
<tr>
<td>blue translation</td>
<td>0.28</td>
<td></td>
<td>0.14</td>
</tr>
<tr>
<td>green translation</td>
<td></td>
<td></td>
<td>0.13</td>
</tr>
</tbody>
</table>

- **Exact MAP decoding**

\[
y^* = \arg \max_{y \in \text{Trans}(x)} p(y|x)
\]

\[
y^* = \arg \max_{y \in \text{Trans}(x)} \sum_{d \in D(x, y)} p(y, d|x)
\]

- **x**: Foreign sentence
- **y**: English sentence
- **d**: derivation

Monday, August 17, 2009
Maximum A Posterior (MAP) Decoding

<table>
<thead>
<tr>
<th>translation string</th>
<th>MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>red translation</td>
<td>0.28</td>
</tr>
<tr>
<td>blue translation</td>
<td>0.28</td>
</tr>
<tr>
<td>green translation</td>
<td>0.44</td>
</tr>
</tbody>
</table>

• Exact MAP decoding

\[
y^* = \arg \max_{y \in \text{Trans}(x)} p(y|x)
\]

\[
= \arg \max_{y \in \text{Trans}(x)} \sum_{d \in D(x,y)} p(y, d|x)
\]

- \(x\): Foreign sentence
- \(y\): English sentence
- \(d\): derivation
Maximum A Posterior (MAP) Decoding

<table>
<thead>
<tr>
<th>Translation String</th>
<th>MAP</th>
<th>Derivation</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>red translation</td>
<td>0.28</td>
<td></td>
<td>0.16</td>
</tr>
<tr>
<td>blue translation</td>
<td>0.28</td>
<td></td>
<td>0.14</td>
</tr>
<tr>
<td>green translation</td>
<td>0.44</td>
<td></td>
<td>0.14</td>
</tr>
</tbody>
</table>

- Exact MAP decoding

\[ y^* = \arg \max_{y \in \text{Trans}(x)} p(y|x) \]

\[ = \arg \max_{y \in \text{Trans}(x)} \sum_{d \in D(x,y)} p(y, d|x) \]

- \( x \): Foreign sentence
- \( y \): English sentence
- \( d \): derivation
Hypergraph as a search space

A hypergraph is a compact structure to encode exponentially many trees.
The hypergraph defines a probability distribution over **derivation trees**, i.e. \( p(y, d \mid x) \), and also a distribution (implicit) over **strings**, i.e. \( p(y \mid x) \).

\[
X \rightarrow \langle X_0, \ X_0 \rangle
\]

\[
S \rightarrow \langle X_0, \ X_0 \rangle
\]

**Exact MAP decoding**

\[
y^* = \arg \max_{y \in \text{HG}(x)} p(y \mid x) = \arg \max_{y \in \text{HG}(x)} \sum_{d \in \mathcal{D}(x,y)} p(y, d \mid x)
\]

\( \text{exponential size} \)

**NP-hard (Sima’an 1996)**
Decoding with spurious ambiguity?

- Maximum a posterior (MAP) decoding
- Viterbi approximation
- N-best approximation (crunching) (May and Knight 2006)
## Viterbi Approximation

<table>
<thead>
<tr>
<th>Translation String</th>
<th>MAP</th>
<th>Viterbi</th>
<th>Derivation</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>red translation</td>
<td>0.28</td>
<td></td>
<td></td>
<td>0.16</td>
</tr>
<tr>
<td>blue translation</td>
<td>0.28</td>
<td></td>
<td></td>
<td>0.14</td>
</tr>
<tr>
<td>green translation</td>
<td>0.44</td>
<td></td>
<td></td>
<td>0.13</td>
</tr>
</tbody>
</table>

### Viterbi approximation

\[
y^* = \arg \max_{y \in \text{Trans}(x)} \max_{d \in D(x, y)} p(y, d|x) = Y(\arg \max_{d \in D(x)} p(y, d|x))
\]
### Viterbi Approximation

<table>
<thead>
<tr>
<th>Translation String</th>
<th>MAP</th>
<th>Viterbi</th>
<th>Derivation</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>red translation</td>
<td>0.28</td>
<td>0.16</td>
<td></td>
<td>0.16</td>
</tr>
<tr>
<td>blue translation</td>
<td>0.28</td>
<td>0.14</td>
<td></td>
<td>0.14</td>
</tr>
<tr>
<td>green translation</td>
<td>0.44</td>
<td>0.13</td>
<td></td>
<td>0.13</td>
</tr>
</tbody>
</table>

**Viterbi approximation**

\[
y^* = \arg \max_{y \in \text{Trans}(x)} \max_{d \in \mathcal{D}(x, y)} p(y, d|x)
\]

\[
= \mathcal{Y}(\arg \max_{d \in \mathcal{D}(x)} p(y, d|x))
\]
N-best Approximation

<table>
<thead>
<tr>
<th>translation string</th>
<th>MAP</th>
<th>Viterbi</th>
<th>4-best crunching</th>
<th>derivation</th>
<th>probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>red translation</td>
<td>0.28</td>
<td>0.16</td>
<td></td>
<td></td>
<td>0.16</td>
</tr>
<tr>
<td>blue translation</td>
<td>0.28</td>
<td>0.14</td>
<td></td>
<td></td>
<td>0.14</td>
</tr>
<tr>
<td>green translation</td>
<td>0.44</td>
<td>0.13</td>
<td></td>
<td></td>
<td>0.13</td>
</tr>
</tbody>
</table>

- **N-best approximation** (*crunching*) (May and Knight 2006)

\[
y^* = \arg \max_{y \in \text{Trans}(x)} \sum_{d \in D(x,y) \cap \text{ND}(x)} p(y, d|x)
\]
<table>
<thead>
<tr>
<th>translation string</th>
<th>MAP</th>
<th>Viterbi</th>
<th>4-best crunching</th>
<th>derivation</th>
<th>probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>red translation</td>
<td>0.28</td>
<td>0.16</td>
<td>0.16</td>
<td></td>
<td>0.16</td>
</tr>
<tr>
<td>blue translation</td>
<td>0.28</td>
<td>0.14</td>
<td>0.28</td>
<td></td>
<td>0.14</td>
</tr>
<tr>
<td>green translation</td>
<td>0.44</td>
<td>0.13</td>
<td>0.13</td>
<td></td>
<td>0.13</td>
</tr>
</tbody>
</table>

- N-best approximation (crunching) (May and Knight 2006)

\[ y^* = \arg \max_{y \in \text{Trans}(x)} \sum_{d \in D(x, y) \cap \text{ND}(x)} p(y, d|x) \]
<table>
<thead>
<tr>
<th>translation string</th>
<th>MAP</th>
<th>Viterbi</th>
<th>4-best crunching</th>
<th>derivation</th>
<th>probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>red translation</td>
<td>0.28</td>
<td>0.16</td>
<td>0.16</td>
<td></td>
<td>0.16</td>
</tr>
<tr>
<td>blue translation</td>
<td>0.28</td>
<td>0.14</td>
<td>0.28</td>
<td></td>
<td>0.14</td>
</tr>
<tr>
<td>green translation</td>
<td>0.44</td>
<td>0.13</td>
<td>0.13</td>
<td></td>
<td>0.13</td>
</tr>
</tbody>
</table>

- Exact MAP decoding under spurious ambiguity is **intractable**
- Viterbi and crunching are efficient, but ignore most derivations
- Our goal: develop an **approximation** that considers all the derivations **but** still allows **tractable** decoding
Variational Decoding

Decoding using Variational approximation

Decoding using a sentence-specific approximate distribution
Variational Decoding for MT: an Overview

Sentence-specific decoding

Three steps:

1. Generate a hypergraph

MAP decoding under P is intractable

Sentence-specific decoding

Foreign sentence x

SMT

Monday, August 17, 2009
Generate a hypergraph

Estimate a model from the hypergraph

Decide to use q* on the hypergraph

q* is an n-gram model over output strings.

\[
q^*(y | x) \approx \sum_{d \in D(x,y)} p(y, d | x)
\]
Variational Inference

• We want to do inference under $p$, but it is intractable

$$y^* = \arg \max_y p(y|x)$$

• Instead, we derive a simpler distribution $q^*$

$$q^* = \arg \min_{q \in Q} \text{KL}(p||q)$$

• Then, we will use $q^*$ as a surrogate for $p$ in inference

$$y^* = \arg \max_y q^*(y | x)$$
Variational Approximation

- $q^*$: an approximation having minimum distance to $p$
  
  $$q^* = \arg\min_{q \in Q} \text{KL}(p \| q)$$

- $q^*$ is obtained by minimizing the Kullback-Leibler divergence between $p$ and $q$ over the set of distributions $Q$.

- The expression for $q^*$ can be rewritten as:
  
  $$q^* = \arg\min_{q \in Q} \sum_{y \in \text{Trans}(x)} p \log \frac{p}{q}$$

- Further simplification gives:
  
  $$q^* = \arg\max_{q \in Q} \sum_{y \in \text{Trans}(x)} p \log q$$

- Three questions
  
  - How to parameterize $q$?
  - How to estimate $q^*$?
  - How to use $q^*$ for decoding?
Parameterization of $q \in \mathbb{Q}$

- Naturally, we parameterize $q$ as an $n$-gram model
- The probability of a string is a product of the probabilities of those $n$-grams appearing in that string

3-gram model

$y: a \ b \ c \ d \ e \ f$

$$q(y) = q(a) \cdot q(b|a) \cdot q(c|ab) \cdot q(d|bc) \cdot q(e|cd) \cdot q(f|de)$$

Other ways of parameterizations are possible!
Parameterization of $q \in Q$

- Naturally, we parameterize $q$ as an $n$-gram model
- The probability of a string is a product of the probabilities of those $n$-grams appearing in that string

**3-gram model**

$y: a \ b \ c \ d \ e \ f$

$$q(y) = q(a) \cdot q(b|a) \cdot q(c|ab) \cdot q(d|bc) \cdot q(e|cd) \cdot q(f|de)$$

how to estimate these $n$-gram probabilities?
Estimation of $q^* \in Q$

- Variational approximation
  
  $$q^* = \arg\max_{q \in Q} \sum_{y \in \text{Trans}(x)} p \log q$$

- $q^*$ is a maximum likelihood estimate (MLE) where $p$ is the empirical distribution

But in our case, $p$ is defined not by a corpus, but by a hypergraph for a given test sentence!

Estimate $\quad \rightarrow \quad$ bi-gram model

- brute force
- dynamic programming
Estimating $q^*$ from a hypergraph: brute force

Bi-gram estimation:

- unpack the hypergraph
Estimating $q^*$ from a hypergraph: brute force

### Bi-gram estimation:

1. **unpack the hypergraph**

   - **p=2/8**
     - $S \langle X_0, X_0 \rangle$
     - $X \langle X_0 \text{ de } X_1, X_0 X_1 \rangle$
     - $X \langle \text{dianzi shang, the mat} \rangle$
     - $X \langle \text{mao, a cat} \rangle$

   - **p=3/8**
     - $S \langle X_0, X_0 \rangle$
     - $X \langle X_0 \text{ de } X_1, X_0 \text{ 's } X_1 \rangle$
     - $X \langle \text{dianzi shang, the mat} \rangle$
     - $X \langle \text{mao, a cat} \rangle$

   - **p=1/8**
     - $S \langle X_0, X_0 \rangle$
     - $X \langle X_0 \text{ de } X_1, X_1 \text{ on } X_0 \rangle$
     - $X \langle \text{dianzi shang, the mat} \rangle$
     - $X \langle \text{mao, a cat} \rangle$

   - **p=2/8**
     - $S \langle X_0, X_0 \rangle$
     - $X \langle X_0 \text{ de } X_1, X_1 \text{ of } X_0 \rangle$
     - $X \langle \text{dianzi shang, the mat} \rangle$
     - $X \langle \text{mao, a cat} \rangle$

---

Monday, August 17, 2009
Estimating $q^*$ from a hypergraph: brute force

- **Bi-gram estimation:**
  - unpack the hypergraph
  - accumulate the soft-count of each bigram
  - normalize the counts

- **Pr(on | cat) = 1/8**
- **Pr(</s> | cat) = 5/8**
- **Pr(of | cat) = 2/8**
Estimating $q^*$ from a hypergraph: dynamic programming

**Bi-gram estimation:**

- run inside-outside on the hypergraph
- accumulate the **soft-count** of each bigram at each hyperedge
- normalize the counts
Decoding using $q^* \in \mathbb{Q}$

- Rescore the hypergraph $HG(x)$

$$y^* = \arg \max_{y \in HG(x)} q^*(y|x)$$

$q^*$ is an n-gram model.

- have efficient dynamic programming algorithms
- score the hypergraph using an n-gram model

John already told you how to do this 😊
KL divergences under different variational models

\[ q^* = \arg \min_{q \in Q} \text{KL}(p||q) = H(p, q) - H(p) \]

| Measure   | \( \overline{H}(p) \) | \( \overline{\text{KL}}(p||\cdot) \) |
|-----------|---------------------|-------------------------------|
| bits/word |                     | \( q_1^* \) | \( q_2^* \) | \( q_3^* \) | \( q_4^* \) |
| MT’04     | 1.36                | 0.97 | 0.32 | 0.21 | 0.17 |
| MT’05     | 1.37                | 0.94 | 0.32 | 0.21 | 0.17 |

- The larger the order \( n \) is, the smaller the KL divergence is!
- The reduction of KL divergence happens mostly when switching from unigram to bigram
KL divergences under different variational models

\[ q^* = \arg \min_{q \in Q} \text{KL}(p \| q) = H(p, q) - H(p) \]

<table>
<thead>
<tr>
<th>Measure</th>
<th>( H(p) )</th>
<th>( \text{KL}(p | \cdot) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>bits/word</td>
<td></td>
<td>( q_1^* ) ( q_2^* ) ( q_3^* ) ( q_4^* )</td>
</tr>
<tr>
<td>MT’04</td>
<td>1.36</td>
<td>0.97 0.32 0.21 0.17</td>
</tr>
<tr>
<td>MT’05</td>
<td>1.37</td>
<td>0.94 0.32 0.21 0.17</td>
</tr>
</tbody>
</table>

How to compute them on a hypergraph?

see (Li and Eisner, EMNLP’09)
### BLEU scores when using a single variational n-gram model

<table>
<thead>
<tr>
<th>Decoding scheme</th>
<th>MT’04</th>
<th>MT’05</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viterbi</td>
<td>35.4</td>
<td>32.6</td>
</tr>
<tr>
<td>1gram</td>
<td>25.9</td>
<td>24.5</td>
</tr>
<tr>
<td>2gram</td>
<td>36.1</td>
<td>33.4</td>
</tr>
<tr>
<td>3gram</td>
<td>36.0</td>
<td>33.1</td>
</tr>
<tr>
<td>4gram</td>
<td>35.8</td>
<td>32.9</td>
</tr>
</tbody>
</table>

- unigram performs very badly
- bigram achieves best BLEU scores

modeling error in p
BLEU cares about both low- and high-order n-gram matches

- Interpolating variational n-gram model for different n

\[
y^* = \arg \max_{y \in \text{HG}(x)} \sum_n \theta_n \cdot \log q^*_n(y \mid x)
\]

Viterbi and variational are different ways in approximating p

\[
y^* = \arg \max_{y \in \text{HG}(x)} \left( \sum_n \theta_n \cdot \log q^*_n(y \mid x) + \theta_v \cdot \log p_{\text{Viterbi}}(y \mid x) \right)
\]
Minimum Bayes Risk (MBR) decoding?

(Tromble et al. 2008)

(Denero et al. 2009)
Minimum Risk Decoding

- Maximum A Posteriori (MAP) decoding
  - find the most **probable** translation string
    
    $$y^* = \arg \max_{y \in HG(x)} p(y|x)$$

- Minimum risk decoding
  - find the **consensus** translation string
    
    $$y^* = \arg \min_{y \in HG(x)} \text{Risk}(y)$$
    
    $$\text{Risk}(y) = \sum_{y'} L(y, y') p(y'|x)$$
Variational Decoding (VD) vs. MBR (Tromble et al. 2008)

Both BLEU metric and our variational distributions happen to use n-gram dependencies.
• Variational decoding with interpolation

\[ y^* = \arg \max_{y \in \text{HG}(x)} \sum_n \theta_n \cdot \log q_n^*(y | x) \]

\[ q_n(y | x) = \prod_{w \in W_n} q(r(w) | h(w), x)^{c_w(y)} \]

\[ q(r(w) | h(w), x) = \frac{\sum_{y'} c_w(y') p(y' | x)}{\sum_{y'} c_{h(w)}(y') p(y' | x)} \]

• Minimum risk decoding (Tromble et al. 2008)

\[ y^* = \arg \max_{y \in \text{HG}(x)} \sum_n \theta_n \cdot g_n(y | x) \]

\[ g_n(y | x) = \sum_{w \in W_n} g(w | x) c_w(y) \]

\[ g(w | x) = \sum_{y'} \delta_w(y') p(y' | x) \]
• Variational decoding with interpolation

\[ y^* = \arg \max_{y \in \text{HG}(x)} \sum_n \theta_n \cdot \log q^*_n(y \mid x) \]

\[ q_n(y \mid x) = \prod_{w \in W_n} q(r(w) \mid h(w), x)^{c_w(y)} \]

\[ q(r(w) \mid h(w), x) = \frac{\sum_{y'} c_w(y')p(y' \mid x)}{\sum_{y'} c_{h(w)}(y')p(y' \mid x)} \]

• Minimum risk decoding (Tromble et al. 2008)

\[ y^* = \arg \max_{y \in \text{HG}(x)} \sum_n \theta_n \cdot g_n(y \mid x) \]

\[ g_n(y \mid x) = \sum_{w \in W_n} g(w \mid x)c_w(y) \]

\[ g(w \mid x) = \sum_{y'} \delta_w(y')p(y' \mid x) \]

non-probabilistic

very expensive to compute
BLEU Results on Chinese-English NIST MT Tasks

<table>
<thead>
<tr>
<th>Decoding scheme</th>
<th>MT’04</th>
<th>MT’05</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viterbi</td>
<td>35.4</td>
<td>32.6</td>
</tr>
<tr>
<td>MBR ($K=1000$)</td>
<td>35.8</td>
<td>32.7</td>
</tr>
<tr>
<td>Crunching ($N=10000$)</td>
<td>35.7</td>
<td>32.8</td>
</tr>
<tr>
<td>Crunching+MBR ($N=10000$)</td>
<td>35.8</td>
<td>32.7</td>
</tr>
<tr>
<td>Variational ($1to4gram+wp+vt$)</td>
<td><strong>36.6</strong></td>
<td><strong>33.5</strong></td>
</tr>
</tbody>
</table>

- variational decoding improves over Viterbi, MBR, and crunching
Conclusions

• Exact MAP decoding with spurious ambiguity is intractable

• Viterbi or N-best approximations are efficient, but ignore most derivations

• We developed a variational approximation, which considers all derivations but still allows tractable decoding

• Our variational decoding improves a state of the art baseline
Future directions

• The MT pipeline is full of intractable problems
  • variational approximation is a principled way to tackle these problems

• Decoding with spurious ambiguity is a common problem in many other NLP applications
  • Models with latent variables
  • Data oriented parsing (DOP)
  • Hidden Markov Models (HMM)
  • ......
Thank you!
谢谢！
Generate a hypergraph

Estimate a model from the hypergraph

Decode using \( q^* \) on the hypergraph

\[ q^*(y \mid x) \approx \sum_{d \in D(x, y)} p(y, d \mid x) \]