Variational Decoding for Statistical Machine Translation

Zhifei Li and Jason Eisner and Sanjeev Khudanpur

Department of Computer Science and Center for Language and Speech Processing
Johns Hopkins University

February 3, 2010

Outline

Ambiguity

- Inherent in natural language, central issue in NLP
- Useful ambiguity resolution: part-of-speech, sense, syntax tree
- Too much resolution: nuisance variable and spurious ambiguity
- MT systems: produce full derivation with each string
- Hidden variables and structure crucial for decoding, user only cares about output string

Translation Ambiguity

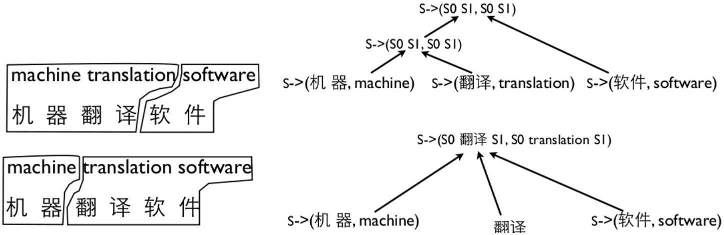

Figure: Multiple derivations for “machine translation software”

Choosing the Best Translation

- Goal: select string most likely over all possible derivations
- Ideal: measure goodness of string by summing over its derivations (marginalize out spurious ambiguity)
- Reality: computationally intractable (Sima'an, 1996; Casacuberta and Higuera, 2000)
- In practice: use Viterbi path, most likely derivation rather than string
- This work: use variational method to consider all derivations while remaining tractable

Background

Terminology:

- x: some input string
- $\mathrm{D}(x)$: set of derivations considered by MT system
- Each $d \in \mathrm{D}(x)$ yields some translation string $y = \mathrm{Y}(d)$

Translation:

- $\mathrm{D}(x, y) = \{d \in \mathrm{D}(x) : \mathrm{Y}(d) = y\}$: possible derivations for y
- $\mathrm{T}(y) = \{\mathrm{Y}(d) : d \in \mathrm{D}(x)\}$: possible translations

Decoding

Maximum A Posteriori (MAP):

- Choose the best output string y^* for input x:

$$y^* = \operatorname*{argmax}_{y \in T(x)} p(y|x)$$

- Requires marginalizing nuisance variable d:

$$y^* = \operatorname*{argmax}_{y \in T(x)} \sum_{d \in D(x,y)} p(y, d|x)$$

- Shown to be NP-hard (Sima'an, 1996)

Approximate Decoding

Viterbi:

- Change sum to max, output string for most likely path:

$$y^* = \underset{y \in T(x)}{\operatorname{argmax}} \max_{d \in D(x,y)} p(y, d|x)$$

- Simple and tractable, but ignores most derivations

N-best "crunching" May and Knight (2006):

- Sum over most likely derivations:

$$y^* = \underset{y \in T(x)}{\operatorname{argmax}} \sum_{d \in D(x,y) \cap ND(x)} p(y, d|x)$$

Variational Decoding

Variational approximate inference:

- Exact inference under complex model p is intractable
- Approximate posterior $p(y|x)$ using tractable model $q(y)$ where $q(y) \in \mathcal{Q}$ chosen to minimize information loss

Variational MT decoding:

- $\arg\max_y p(y|x)$ required for MAP decoding intractable
- Seek approximate distribution $q(y) \approx p(y|x)$ minimizing KL divergence:

$$q^* = \underset{q \in \mathcal{Q}}{\text{argmax}} \sum_{y \in T(x)} p \log q$$

Parametrization

Selecting a family of distributions $\mathcal{Q}$:

- Large family: complex q^* to better approximate p
- Smaller family: simple q^* with conditional independences, easier to compute
- Natural choice for strings: family of n-gram models
- As $n \to \infty$, $q^* \to p$ and computation becomes intractable

Parametrization

- Models $q \in \mathcal{Q}$ take the form:

$$q(y) = \prod_{w \in W} q(r(w)|h(w))^{c_w(y)}$$

- $w \in W$: n-gram which occurs $c_w(y)$ times in string y
- w may be divided into history $h(w)$ and current word $r(w)$
- Parameters: normalized conditional distributions $q(r(w)|h(w))$

Estimation

- If p is empirical distribution over training corpus, q^* is MLE n-gram model:

$$q^*(r(w)|h(w)) = \frac{c(w)}{c(h(w))}$$

- MT systems generate hypergraph $\mathrm{HG}(x)$ for input x
- If p is represented by $\mathrm{HG}(x)$, use expected counts:

$$q^*(r(w)|h(w)) = \frac{\bar{c}(w)}{\bar{c}(h(w))} = \frac{\sum_{y,d} c_w(y)p(y,d|x)}{\sum_{y,d} c_{h(w)}(y)p(y,d|x)}$$

Maximum Likelihood Estimation

Dynamic programming MLE(HG(x))

run inside-outside for hypergraph HG(x)
for v in HG(x) $\triangleright$ each node
    for $e \in B(v)$ $\triangleright$ each incoming hyperedge
        $c_e \leftarrow p_e \cdot \alpha(v)/Z(x)$ $\triangleright$ posterior weight
        for $u \in T(e)$ $\triangleright$ each antecedent node
            $c_e \leftarrow c_e \cdot \beta(u)$
        $\triangleright$ accumulate soft count
        for w in e $\triangleright$ each n-gram type
            $\bar{c}(w) += c_w(e) \cdot c_e$
            $\bar{c}(h(w)) += c_w(e) \cdot c_e$
$q^* \leftarrow$ MLE by formula
return q^*

- Inside-outside provides *inside* weight $\beta(v)$, *outside* weight $\alpha(v)$ for nodes v, and total weight of all derivations $Z(x)$
- Runtime linear in size of HG(x)

Decoding

Translating x:

- Construct q^* from HG(x) and use in place of p
- Crucial: restrict search space to original hypergraph

$$y^* = \operatorname*{argmax}_{y \in T(y)} q^*(y)$$

Choosing q^*: reality vs BLEU

- Best approximation of $p(y|x)$: single n-gram model q^* with n as large as posible
- BLEU metric gives partial credit over lower-order n-grams

Interpolation

Interpolate different orders of models to improve score:

$$y^* = \underset{y \in T(y)}{\operatorname{argmax}} \sum_n \theta_n \cdot \log q_n^*(y)$$

- Geometric interpolation weights θ_n MERT-tunable
- Choose n to optimize score for metric of choice

Variational Approximation vs Viterbi

- Viterbi and variational approximation both approximate $p(y|x)$, make different assumptions
- Viterbi: correct probability of *one* derivation, *ignores* most derivations
- Variational approximation: consider *all* derivations, uses only *aggregate statistics*

Desirable: interpolate further with Viterbi

$$y^* = \underset{y \in T(y)}{\operatorname{argmax}} \sum_n \theta_n \cdot \log q_n^*(y) + \theta_v \cdot \log p_{\text{Viterbi}}(y|x)$$

Similarity to Minimum-Risk Decoding

- Alternative to MAP: minimum Bayes risk

$$y^* = \arg\min_y \mathrm{R}(y) = \arg\min_y \sum_{y'} l(y, y') p(y'|x)$$

- Expected loss of y if true answer is y'
- Tromble et al. (2008) use n-gram based loss function, interpolate n-gram probabilities
- Similarity: both use interpolated n-gram probabilities to select best translation

Similarity to Minimum-Risk Decoding

MBR:

- Uses n-gram posterior probabilities, must be calculated over entire lattice
- Does not normalize over history
- Approximations of average n-gram precisions

Variational:

- Optimal n-gram probabilities calculated once using inside-outside
- Normalizes over history
- Proper probabilistic n-gram model

Main Results

Decoding scheme	MT'04	MT'05
Viterbi	35.4	32.6
MBR (K=1000)	35.8	32.7
Crunching (N =10000)	35.7	32.8
Crunching+MBR (N =10000)	35.8	32.7
Variational (1to4gram+wp+vt)	**36.6**	**33.5**

Table: BLEU scores for decoding schemes

- Chinese-to-English translation task using Joshua MT toolkit
- Training data: 1M sentence pairs sampled from NIST OpenMT corpora
- Tuning data: NIST MT03 set

KL Divergence

Measure	$\bar{\mathrm{H}}(p, \cdot)$				$\bar{\mathrm{H}}_d(p)$	$\bar{\mathrm{H}}(p)$
bits/word	q_1^*	q_2^*	q_3^*	q_4^*		$\approx$
MT'04	2.33	1.68	1.57	1.53	1.36	1.03
MT'05	2.31	1.69	1.58	1.54	1.37	1.04

Table: Cross-entropies for various q

- $\mathrm{KL}(p \| q) = \mathrm{H}(p, q) - \mathrm{H}(p)$
- Estimate of $\mathrm{H}(p)$ serves as bound for perfect approximation
- Higher order models better approximate p, best improvement from unigram to bigram

Variational Decoding for Statistical Machine Translation

Zhifei Li and Jason Eisner and Sanjeev Khudanpur

Department of Computer Science and Center for Language and Speech Processing
Johns Hopkins University

February 3, 2010