Decoding in Statistical Machine Translation

Christian Hardmeier

2016-05-04

Mid-course Evaluation

`http://stp.lingfil.uu.se/~sara/kurser/MT16/mid-course-eval.html`

Decoding

The *decoder* is the part of the SMT system that creates the translations.

Given a set of models, how can we translate *efficiently* and *accurately*?

UPPSALA UNIVERSITET

Decoding

Find the best translation among all possible translations.

$$t^* = \arg\max_t f(s,t) = \arg\max_t \sum_i \lambda_i h_i(s,t)$$

$f(s,t)$	Scoring function
$h_i(s,t)$	Feature functions
λ_i	Feature weights

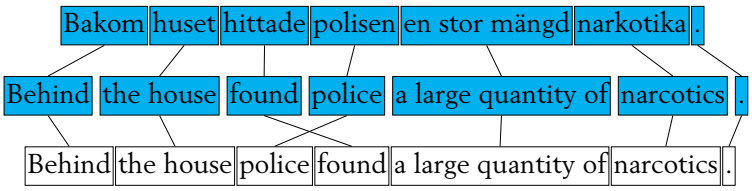
UPPSALA UNIVERSITET

Model error vs. search error

- **Model error:** The solution with the highest score under our models is not a good translation.
- **Search error:** The decoder cannot find the solution with the highest model score.

UPPSALA UNIVERSITET

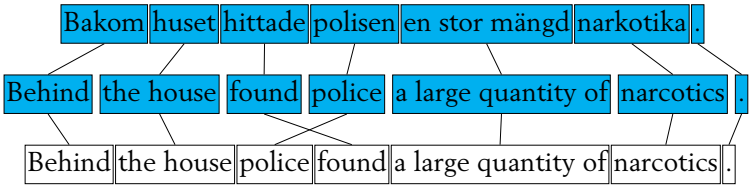
Phrase-based SMT: Generative Model



1. Phrase segmentation
2. Phrase translation
3. Output ordering

UPPSALA UNIVERSITET

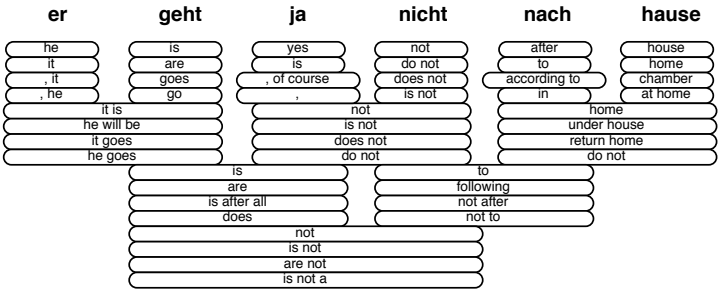
Phrase-based SMT: Generative Model



Behind the house
the house police
house police found
police found a
found a large

UPPSALA UNIVERSITET

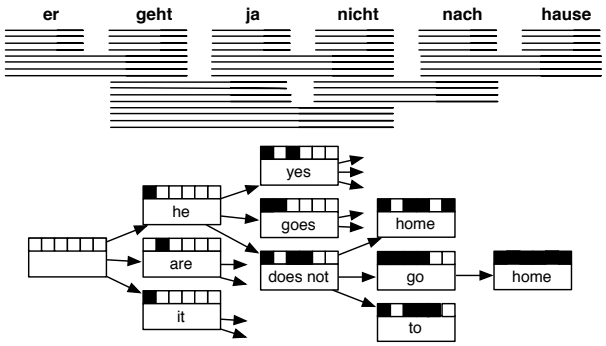
Translation Options



Illustrations by Philipp Koehn

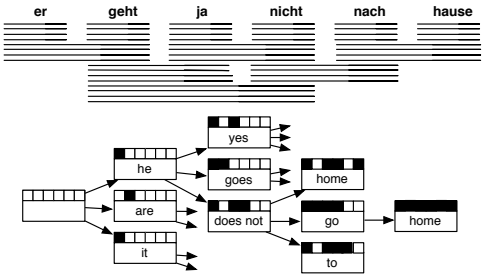
UPPSALA UNIVERSITET

Decoding by Hypothesis Expansion



Illustrations by Philipp Koehn

- Is it always possible to translate any sentence in this way?
- What would cause the process to break down
  so the decoder can't find a translation
  that covers the whole input sentence?
- How could you make sure that this never happens?



UPPSALA UNIVERSITET

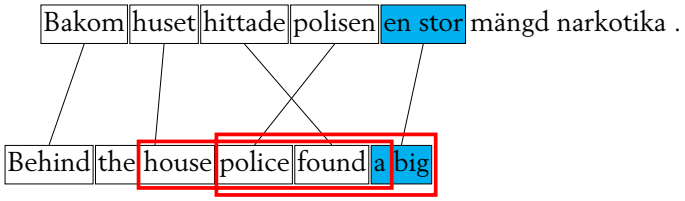
Decoding complexity

Naively, in a sentence of N words with T translation options for each phrase, we can have

- $O(2^N)$ phrase segmentations,
- $O(T^N)$ sets of phrase translations, and
- $O(N!)$ word reordering permutations.

UPPSALA UNIVERSITET

Exploiting Model Locality



To score a new hypothesis, we need:

- the score of the previous hypothesis
- the translation model score
- the new language model scores

UPPSALA UNIVERSITET

Hypothesis recombination

- The translation model only looks at the current phrase.
- The *n*-gram model only looks at a window of *n* words.
- The choices the decoder makes are independent of everything beyond this window!
- The decoder never reconsiders its choices once they've moved out of the *n*-gram history.

UPPSALA UNIVERSITET

Hypothesis recombination

Suppose we have these hypotheses with the same coverage, and we use a trigram language model:

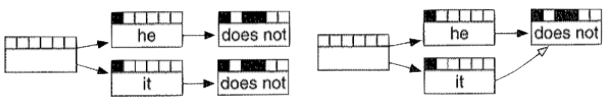
~~After~~ ~~the house~~ ~~police~~	~~Score = –12.5~~
Behind the house police	Score = –11.2
~~, the house police~~	~~Score = –22.0~~

- We already know the winner!
- We can discard the competing hypotheses.

UPPSALA UNIVERSITET

Hypothesis recombination

- Hypothesis recombination combines branches in the search graph:

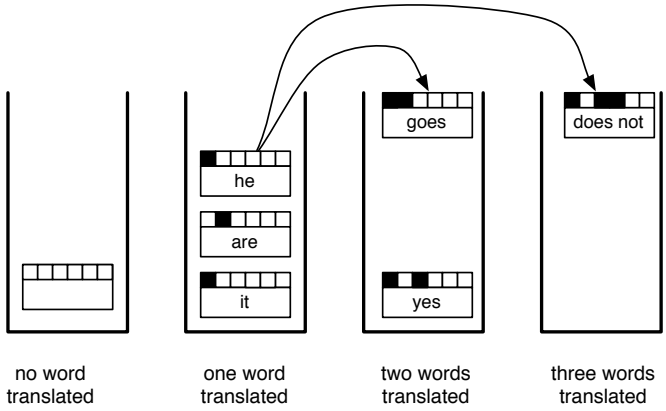


- It's a form of dynamic programming.
- Recombination reduces the search space substantially. . .
- . . . it preserves search optimality. . .
- . . . but decoding is still exponential!

Pruning

- To make decoding really efficient,
  we expand only hypotheses that look promising.
- Bad hypotheses should be *pruned* early
  to avoid wasting time on them.
- Pruning compromises search optimality!

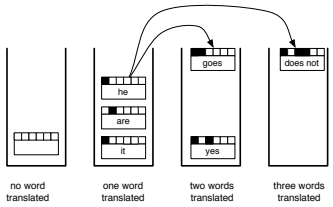
Stack decoding



Illustrations by Philipp Koehn

Stack decoding algorithm

```
1:  AddToStack(s_0, h_0)
2:  for i = 0 ... N − 1 do
3:      for all h ∈ s_i do
4:          for all t ∈ T do
5:              if Applicable(h, t) then
6:                  h' ← Expand(h, t)
7:                  j ← WordsCovered(h) + WordsCovered(t)
8:                  AddToStack(s_j, h') ← pruning magic goes here
9:              end if
10:         end for
11:     end for
12: end for
13: return best hypothesis on stack s_N
```



AddToStack(s, h)

```
1: for all h' ∈ s do
2:     if Recombinable(h, h') then
3:         add higher-scoring of h, h' to stack s, discard other
4:         return
5:     end if
6: end for
7: add h to stack s
8: if stack too large then
9:     prune stack
10: end if
```

How to prune

Histogram pruning
Keep no more than S hypotheses per stack.

Parameter: Stack size S

Threshold pruning
Discard hypotheses whose score is very low compared to that of the best hypothesis on the stack h^*:

$$\text{Score}(h) < \eta \cdot \text{Score}(h^*)$$

Parameter: Beam size η

Beam search: Complexity

- For each of the N words in the input sentence,
- expand S hypotheses
- by considering T translation options each:

$$O(S \cdot N \cdot T)$$

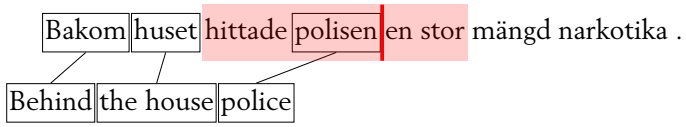
The number of translation options is linear in the sentence length:

$$O(S \cdot N^2)$$

UPPSALA UNIVERSITET

Distortion limit

- When translating between closely related languages, most reorderings are local...
- ...and anyhow, we haven't got any reasonable models for long-range reordering!
- If we impose a limit on reordering, the number of translation options to consider at each step is bounded by a constant.



UPPSALA UNIVERSITET

Distortion limit

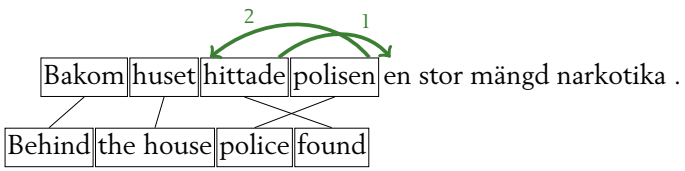
- When translating between closely related languages, most reorderings are local...
- ...and anyhow, we haven't got any reasonable models for long-range reordering!
- If we impose a limit on reordering, the number of translation options to consider at each step is bounded by a constant.

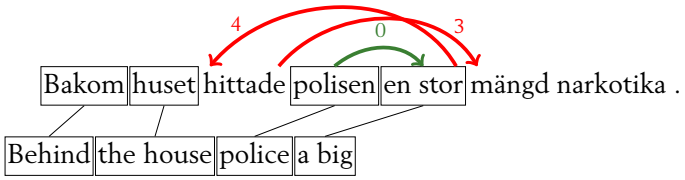
The number of hypotheses expanded by a beam search decoder with limited reordering is linear in the stack size and the input size:

$$O(S \cdot N)$$

UPPSALA UNIVERSITET

Incremental scoring and cherry picking





UPPSALA UNIVERSITET

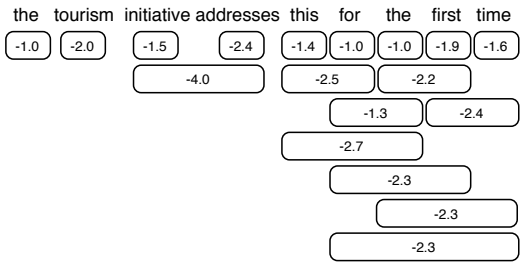
Incremental scoring and cherry picking

- The path that looks cheapest necessarily incurs a much higher cost later.
- Pruning may discard better options before this is recognised.
- To make scores more comparable, we should take into account unavoidable future costs.
- Compare hypotheses based on current score + future score.

UPPSALA UNIVERSITET

Future cost estimation

- Calculating the future cost exactly would amount to full decoding!
- Cheaper approximations can be computed by making additional independence assumptions.
  - Assume independence between models.
  - Ignore LM history across phrase boundaries.



Illustrations by Philipp Koehn

UPPSALA UNIVERSITET

Stack Decoding and A* Search

- Stack decoding is related to a standard search algorithm called A* search.
- In A* search, each partial hypothesis is evaluated with a *score* and a future cost estimate called *heuristic*.
- A heuristic is called *admissible* if it never underestimates the true future cost.
- A* search with an admissible heuristic is *optimal*.
- The future cost estimate of stack decoding is *not* admissible.

DP Beam Search Decoding: Evaluation

- DP beam search is by far the most popular search algorithm for phrase-based SMT.
- It combines high speed with reasonable accuracy by exploiting the constraints of the standard models.
- It works well with very local models.
  - Sentence-internal long-range dependencies increase search errors by inhibiting recombination.
  - No cross-sentence dependencies on the target side.
- Current state of the art: Almost perfect local fluency, but serious problems with long-range reordering and discourse-level phenomena.