Chapter 6
Decoding
Statistical Machine Translation
Decoding

- We have a mathematical model for translation

\[ p(e|f) \]

- Task of decoding: find the translation \( e_{\text{best}} \) with highest probability

\[ e_{\text{best}} = \arg\max_e p(e|f) \]

- Two types of error
  - the most probable translation is bad → fix the model
  - search does not find the most probably translation → fix the search

- Decoding is evaluated by search error, not quality of translations (although these are often correlated)
Translation Process

- Task: translate this sentence from German into English

\[ \text{er geht ja nicht nach hause} \]
Translation Process

• Task: translate this sentence from German into English

er geht ja nicht nach hause

• Pick phrase in input, translate
Translation Process

- Task: translate this sentence from German into English

\[ \text{er geht ja nicht nach hause} \]

- Pick phrase in input, translate
  - it is allowed to pick words out of sequence reordering
  - phrases may have multiple words: many-to-many translation

Chapter 6: Decoding
Translation Process

• Task: translate this sentence from German into English

er geht ja nicht nach hause

he does not go

• Pick phrase in input, translate
Translation Process

- Task: translate this sentence from German into English

> er geht ja nicht nach hause

he does not go home

- Pick phrase in input, translate
Computing Translation Probability

- Probabilistic model for phrase-based translation:

\[
e_{\text{best}} = \arg\max_e \prod_{i=1}^{I} \phi(\bar{f}_i|\bar{e}_i) \cdot d(\text{start}_i - \text{end}_{i-1} - 1) \cdot p_{LM}(e)
\]

- Score is computed incrementally for each partial hypothesis

- Components

  **Phrase translation**  Picking phrase \(\bar{f}_i\) to be translated as a phrase \(\bar{e}_i\)
  \(\rightarrow\) look up score \(\phi(\bar{f}_i|\bar{e}_i)\) from phrase translation table

  **Reordering**  Previous phrase ended in \(\text{end}_{i-1}\), current phrase starts at \(\text{start}_i\)
  \(\rightarrow\) compute \(d(\text{start}_i - \text{end}_{i-1} - 1)\)

  **Language model**  For \(n\)-gram model, need to keep track of last \(n-1\) words
  \(\rightarrow\) compute score \(p_{LM}(w_i|w_{i-(n-1)}, \ldots, w_{i-1})\) for added words \(w_i\)
<table>
<thead>
<tr>
<th>er</th>
<th>geht</th>
<th>ja</th>
<th>nicht</th>
<th>nach</th>
<th>hause</th>
</tr>
</thead>
<tbody>
<tr>
<td>he</td>
<td>is</td>
<td>yes</td>
<td>not</td>
<td>after</td>
<td>house</td>
</tr>
<tr>
<td>it</td>
<td>are</td>
<td>is</td>
<td>do not</td>
<td>to</td>
<td>home</td>
</tr>
<tr>
<td>, , he</td>
<td>goes</td>
<td>, of course</td>
<td>does not</td>
<td>according to</td>
<td>chamber</td>
</tr>
<tr>
<td>, he</td>
<td>go</td>
<td>,</td>
<td>is not</td>
<td>in</td>
<td>at home</td>
</tr>
<tr>
<td>it is</td>
<td>not</td>
<td>is not</td>
<td>not to</td>
<td>under house</td>
<td>return home</td>
</tr>
<tr>
<td>he will be</td>
<td>does not</td>
<td>do not</td>
<td>following</td>
<td>do not</td>
<td></td>
</tr>
<tr>
<td>it goes</td>
<td>not</td>
<td>is not</td>
<td>not after</td>
<td></td>
<td></td>
</tr>
<tr>
<td>he goes</td>
<td>is after all</td>
<td>are not</td>
<td>not to</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Many translation options to choose from
  - in Europarl phrase table: 2727 matching phrase pairs for this sentence
  - by pruning to the top 20 per phrase, 202 translation options remain
The machine translation decoder does not know the right answer
- picking the right translation options
- arranging them in the right order

→ Search problem solved by heuristic beam search
Decoding: Precompute Translation Options

consult phrase translation table for all input phrases

er geht ja nicht nach hause
Decoding: Start with Initial Hypothesis

initial hypothesis: no input words covered, no output produced
Decoding: Hypothesis Expansion

pick any translation option, create new hypothesis
Decoding: Hypothesis Expansion

create hypotheses for all other translation options
Decoding: Hypothesis Expansion

er geht ja nicht nach hause
are it he goes does not go to home
also create hypotheses from created partial hypothesis
Decoding: Find Best Path

er geht ja nicht nach hause
are it he goes does not yes go to home
backtrack from highest scoring complete hypothesis
Computational Complexity

- The suggested process creates exponential number of hypothesis

- Machine translation decoding is NP-complete

- Reduction of search space:
  - recombination (risk-free)
  - pruning (risky)
Recombination

• Two hypothesis paths lead to two matching hypotheses
  – same number of foreign words translated
  – same English words in the output
  – different scores

• Worse hypothesis is dropped
Recombination

- Two hypothesis paths lead to hypotheses indistinguishable in subsequent search
  - same number of foreign words translated
  - same last two English words in output (assuming trigram language model)
  - same last foreign word translated
  - different scores

- Worse hypothesis is dropped

Chapter 6: Decoding
Restrictions on Recombination

• **Translation model:** Phrase translation independent from each other
  → no restriction to hypothesis recombination

• **Language model:** Last $n - 1$ words used as history in $n$-gram language model
  → recombined hypotheses must match in their last $n - 1$ words

• **Reordering model:** Distance-based reordering model based on distance to end position of previous input phrase
  → recombined hypotheses must have that same end position

• Other feature function may introduce additional restrictions
Pruning

- Recombination reduces search space, but not enough
  (we still have a NP complete problem on our hands)

- Pruning: remove bad hypotheses early
  - put comparable hypothesis into stacks
    (hypotheses that have translated same number of input words)
  - limit number of hypotheses in each stack
• Hypothesis expansion in a stack decoder
  – translation option is applied to hypothesis
  – new hypothesis is dropped into a stack further down
Stack Decoding Algorithm

1: place empty hypothesis into stack 0
2: for all stacks 0...n − 1 do
3:   for all hypotheses in stack do
4:     for all translation options do
5:       if applicable then
6:         create new hypothesis
7:         place in stack
8:         recombine with existing hypothesis if possible
9:         prune stack if too big
10:     end if
11:   end for
12: end for
Pruning

- Pruning strategies
  - histogram pruning: keep at most $k$ hypotheses in each stack
  - stack pruning: keep hypothesis with score $\alpha \times$ best score ($\alpha < 1$)

- Computational time complexity of decoding with histogram pruning

  \[ O(\text{max stack size} \times \text{translation options} \times \text{sentence length}) \]

- Number of translation options is linear with sentence length, hence:

  \[ O(\text{max stack size} \times \text{sentence length}^2) \]

- Quadratic complexity
Reordering Limits

- Limiting reordering to maximum reordering distance

- Typical reordering distance 5–8 words
  - depending on language pair
  - larger reordering limit hurts translation quality

- Reduces complexity to linear

\[ O(\text{max stack size } \times \text{ sentence length}) \]

- Speed / quality trade-off by setting maximum stack size
Translating the Easy Part First?

the tourism initiative addresses this for the first time

both hypotheses translate 3 words
worse hypothesis has better score
Estimating Future Cost

• Future cost estimate: how expensive is translation of rest of sentence?

• Optimistic: choose cheapest translation options

• Cost for each translation option
  – **translation model**: cost known
  – **language model**: output words known, but not context
    → estimate without context
  – **reordering model**: unknown, ignored for future cost estimation
Cost Estimates from Translation Options

<table>
<thead>
<tr>
<th>the</th>
<th>tourism initiative addresses</th>
<th>this</th>
<th>for</th>
<th>the</th>
<th>first</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1.0</td>
<td>-2.0</td>
<td>-1.5</td>
<td>-2.4</td>
<td>-1.4</td>
<td>-1.0</td>
<td>-1.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-4.0</td>
<td></td>
<td>-2.5</td>
<td></td>
<td>-2.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-1.3</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-2.7</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-2.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Cost of cheapest translation options for each input span (log-probabilities)
Cost Estimates for all Spans

- Compute cost estimate for all contiguous spans by combining cheapest options

<table>
<thead>
<tr>
<th>first word</th>
<th>future cost estimate for $n$ words (from first)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>the</td>
<td>-1.0</td>
</tr>
<tr>
<td>tourism</td>
<td>-2.0</td>
</tr>
<tr>
<td>initiative</td>
<td>-1.5</td>
</tr>
<tr>
<td>addresses</td>
<td>-2.4</td>
</tr>
<tr>
<td>this</td>
<td>-1.4</td>
</tr>
<tr>
<td>for</td>
<td>-1.0</td>
</tr>
<tr>
<td>the</td>
<td>-1.0</td>
</tr>
<tr>
<td>first</td>
<td>-1.9</td>
</tr>
<tr>
<td>time</td>
<td>-1.6</td>
</tr>
</tbody>
</table>

- Function words cheaper (the: -1.0) than content words (tourism: -2.0)
- Common phrases cheaper (for the first time: -2.3) than unusual ones (tourism initiative addresses: -5.9)
Combining Score and Future Cost

• Hypothesis score and future cost estimate are combined for pruning
  
  – left hypothesis starts with hard part: the tourism initiative
    score: -5.88, future cost: -6.1 → total cost -11.98
  
  – middle hypothesis starts with easiest part: the first time
    score: -4.11, future cost: -9.3 → total cost -13.41
  
  – right hypothesis picks easy parts: this for ... time
    score: -4.86, future cost: -9.1 → total cost -13.96
Other Decoding Algorithms

- A* search
- Greedy hill-climbing
- Using finite state transducers (standard toolkits)
A* Search

- Uses *admissible* future cost heuristic: never overestimates cost
- Translation agenda: create hypothesis with lowest score + heuristic cost
- Done, when complete hypothesis created
Greedy Hill-Climbing

• Create one complete hypothesis with depth-first search (or other means)

• Search for better hypotheses by applying change operators
  – change the translation of a word or phrase
  – combine the translation of two words into a phrase
  – split up the translation of a phrase into two smaller phrase translations
  – move parts of the output into a different position
  – swap parts of the output with the output at a different part of the sentence

• Terminates if no operator application produces a better translation
Summary

• Translation process: produce output left to right

• Translation options

• Decoding by hypothesis expansion

• Reducing search space
  – recombination
  – pruning (requires future cost estimate)

• Other decoding algorithms