Introduction to Statistical Machine Translation

ESSLLI 2005

Chris Callison-Burch
Philipp Koehn

A long history

- Machine translation was one of the first applications envisioned for computers
- Warren Weaver (1949)
  "I have a text in front of me which is written in Russian but I am going to pretend that it is really written in English and that it has been coded in some strange symbols. All I need to do is strip off the code in order to retrieve the information contained in the text."
- First demonstrated by IBM in 1954 with a basic word-for-word translation system.

Commercially Interesting

- U.S. has invested in MT for intelligence purposes
- MT is popular on the web -- it is the most used of Google's special features
- EU spends more than €1,000,000,000 on translation costs each year. (Semi-) automating that could lead to huge savings

Academically Interesting

- Machine translation requires many other NLP technologies
- Potentially: parsing, generation, word sense disambiguation, named entity recognition, transliteration, pronoun resolution, natural language understanding, and real-world knowledge

What makes MT hard?

- Word order
- Word sense
- Pronouns
- Tense
- Idioms

Various approaches

- Word-for-word translation
- Syntactic transfer
- Interlingual approaches
- Controlled language
- Example-based translation
- Statistical translation

Statistical machine translation

- Find most probable English sentence given a foreign language sentence
- Automatically align words and phrases within sentence pairs in a parallel corpus
- Probabilities are determined automatically by training a statistical model using the parallel corpus

Probabilities

- Find most probable English sentence given a foreign language sentence

$$p(e|f)$$

$$\hat{e} = \arg\max_e \; p(e|f)$$

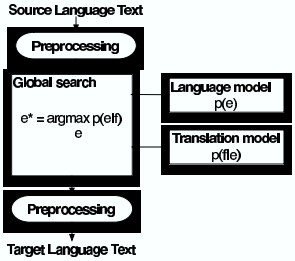
$$p(e|f) = \frac{p(e)p(f|e)}{p(f)}$$

$$\hat{e} = \arg\max_e \; p(e)p(f|e)$$

What the probabilities represent

- p(e) is the "Language model"
  - Assigns a higher probability to fluent / grammatical sentences
  - Estimated using monolingual corpora
- p(f|e) is the "Translation model"
  - Assigns higher probability to sentences that have corresponding meaning
  - Estimated using bilingual corpora

For people who don't like equations



Language Model

- Component that tries to ensure that words come in the right order
- Some notion of grammaticality
- Standardly calculated with a trigram language model, as in speech recognition
- Could be calculated with a statistical grammar such as a PCFG

Trigram language model

- p(I like bungee jumping off high bridges) =
  p(I | <s> <s>) *
  p(like | I <s>) *
  p(bungee | I like) *
  p(jumping | like bungee) *
  p(off | bungee jumping) *
  p(high | jumping off) *
  p(bridges | off high) *
  p(</s> | high bridges) *
  p(</s> | bridges </s>)

Calculating Language Model Probabilities

- Unigram probabilities

$$p(w_1) = \frac{count(w_1)}{total\ words\ observed}$$

Calculating Language Model Probabilities

- Bigram probabilities

$$p(w_2|w_1) = \frac{count(w_1 w_2)}{count(w_1)}$$

Calculating Language Model Probabilities

- Trigram probabilities

$$p(w_3|w_1 w_2) = \frac{count(w_1 w_2 w_3)}{count(w_1 w_2)}$$

Calculating Language Model Probabilities

- Can take this to increasingly long sequences of n-grams
- As we get longer sequences it's less likely that we'll have ever observed them

Backing off

- Sparse counts are a big problem
- If we haven't observed a sequence of words then the count = 0
- Because we're multiplying the n-gram probabilities to get the probability of a sentence the whole probability = 0

Backing off

- $.8 * p(w_3|w_1w_2) + .15 * p(w_3|w_2) + .049 * p(w_3) + .001$
- Avoids zero probs

Translation model

- p(f|e)... the probability of some foreign language string given a hypothesis English translation
- f = Ces gens ont grandi, vécu et oeuvré des dizaines d'années dans le domaine agricole.
- e = *Those people have grown up, lived and worked many years in a farming district.*
- e = *I like bungee jumping off high bridges.*

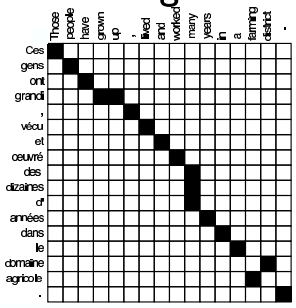
Translation model

- How do we assign values to p(f|e)?
- $$p(f|e) = \frac{count(f,e)}{count(e)}$$
- Impossible because sentences are novel, so we'd never have enough data to find values for all sentences.

Translation model

- Decompose the sentences into smaller chunks, like in language modeling
- $$p(f|e) = \sum_a p(a, f|e)$$
- Introduce another vairable *a* that represents alignments between the individual words in the sentence pair

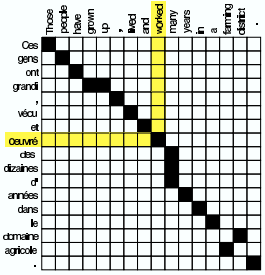
Word alignment



Alignment probabilities

- So we can calculate translation probabilities by way of these alignment probabilities
$$p(f|e) = \sum_a p(a, f|e)$$
- Now we need to define p(a, f | e)
$$p(a, f|e) = \prod_{j=1}^{m} t(f_j|e_i)$$

Calculating $t(f_j|e_i)$

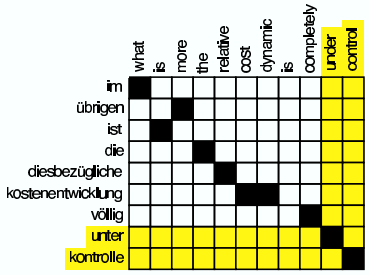


- Counting! I told you probabilities were easy!
- $= \frac{count(f_j, e_i)}{count(e_i)}$
- worked... fonctionné, travaillé, marché, oeuvré
- 100 times total 13 with this f. 13%

Calculating $t(f_j|e_i)$

- Unfortunately we don't have word aligned data, so we can't do this directly.
- OK, so it's not quite as easy as I said.
- Philipp will talk about how to do word alignments using EM on Wednesday.

Phrase Translation Probabilities



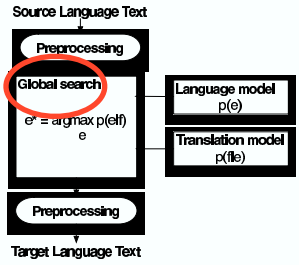
Phrase Translation Probabilities

Phrase Table

- Exhaustive table of source language phrases paired with their possible translations into the target language, along with probabilities

das thema	the issue	.51
	the point	.38
	the subject	.21

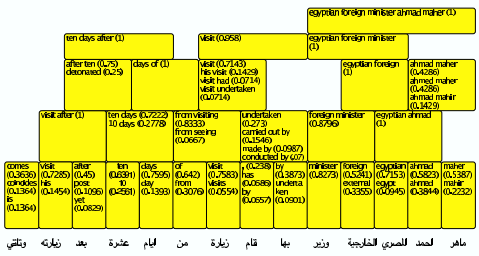
``Diagram Number 1"



The Search Process AKA ``Decoding"

- Look up all translations of every source phrase, using the phrase table
- Recombine the target language phrases that maximizes the translation model probability * the language model probability
- This search over all possible combinations can get very large so we need to find ways of limiting the search space

Looking up translations of source



The Search Space