CMPE 409 Machine Translation

IBM Models and Alignments

Murat ORHUN
(most of them adapted from the original slides of Prof. Philipp Koehn)

Istanbul Bilgi University

April 13, 2022

IBM Model 1

- Generative model: break up translation process into smaller steps
  - IBM Model 1 only uses lexical translation
- Translation probability
  - for a foreign sentence $\mathbf{f} = (f_1, ..., f_{l_f})$ of length l_f
  - to an English sentence $\mathbf{e} = (e_1, ..., e_{l_e})$ of length l_e
  - with an alignment of each English word e_j to a foreign word f_i according to the alignment function $a : j \to i$

$$p(\mathbf{e}, a|\mathbf{f}) = \frac{\epsilon}{(l_f + 1)^{l_e}} \prod_{j=1}^{l_e} t(e_j | f_{a(j)})$$

  - parameter ϵ is a normalization constant

Translation Probability

das

e	$t(e\|f)$
the	0.7
that	0.15
which	0.075
who	0.05
this	0.025

Haus

e	$t(e\|f)$
house	0.8
building	0.16
home	0.02
household	0.015
shell	0.005

ist

e	$t(e\|f)$
is	0.8
's	0.16
exists	0.02
has	0.015
are	0.005

kle

e
small
little
short
minor
petty

$$p(e, a|f) = \frac{\epsilon}{4^3} \times t(\text{the}|\text{das}) \times t(\text{house}|\text{Haus}) \times t(\text{is}|\text{ist}) \times t(\text{small}|\text{klein}$$

$$= \frac{\epsilon}{4^3} \times 0.7 \times 0.8 \times 0.8 \times 0.4$$

$$= 0.0028\epsilon$$

IBM Model 1 and EM

- EM Algorithm consists of two steps
- Expectation-Step: Apply model to the data
  - parts of the model are hidden (here: alignments)
  - using the model, assign probabilities to possible values
- Maximization-Step: Estimate model from data
  - take assign values as fact
  - collect counts (weighted by probabilities)
  - estimate model from counts
- Iterate these steps until convergence

IBM Model 1 and EM

- We need to be able to compute:
  - Expectation-Step: probability of alignments
  - Maximization-Step: count collection

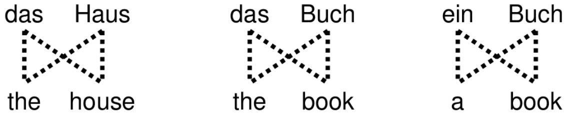
IBM Model 1 and EM: Pseudocode

Input: set of sentence pairs $(\mathbf{e}, \mathbf{f})$

Output: translation prob. $t(e|f)$

```
initialize t(e|f) uniformly
while not converged do
    // initialize
    count(e|f) = 0 for all e, f
    total(f) = 0 for all f
    for all sentence pairs (e,f) do
        // compute normalization
        for all words e in e do
            s-total(e) = 0
            for all words f in f do
                s-total(e) += t(e|f)
            end for
        end for
        // collect counts
        for all words e in e do
            for all words f in f do
                count(e|f) += t(e|f) / s-total(e)
                total(f) += t(e|f) / s-total(e)
            end for
        end for
    end for
    // estimate probabilities
    for all foreign words f do
        for all English words e do
            t(e|f) = count(e|f) / total(f)
        end for
    end for
end while
```

Conversion



e	f	initial	1st it.	2nd it.	3rd it.	...	final
the	das	0.25	0.5	0.6364	0.7479	...	1
book	das	0.25	0.25	0.1818	0.1208	...	0
house	das	0.25	0.25	0.1818	0.1313	...	0
the	buch	0.25	0.25	0.1818	0.1208	...	0
book	buch	0.25	0.5	0.6364	0.7479	...	1
a	buch	0.25	0.25	0.1818	0.1313	...	0
book	ein	0.25	0.5	0.4286	0.3466	...	0
a	ein	0.25	0.5	0.5714	0.6534	...	1
the	haus	0.25	0.5	0.4286	0.3466	...	0
house	haus	0.25	0.5	0.5714	0.6534	...	1

Perplexity

- Perplexity is a measurement of how well a probability distribution or probability model predicts a sample.
- A low perplexity indicates the probability distribution is good at predicting the sample.

$$\log_2 PP = -\sum_s \log_2 p(\mathbf{e}_s|\mathbf{f}_s)$$

Example

das

e	$t(e\|f)$
the	0.7
that	0.15
which	0.075
who	0.05
this	0.025

Haus

e	$t(e\|f)$
house	0.8
building	0.16
home	0.02
household	0.015
shell	0.005

ist

e	$t(e\|f)$
is	0.8
's	0.16
exists	0.02
has	0.015
are	0.005

klein

e	$t(e\|f)$
small	0.4
little	0.4
short	0.1
minor	0.06
petty	0.04

$$\begin{aligned} p(e,a|f) &= \frac{\epsilon}{4^3} \times t(\text{the}|\text{das}) \times t(\text{house}|\text{Haus}) \times t(\text{is}|\text{ist}) \times t(\text{small}|\text{klein}) \\ &= \frac{\epsilon}{4^3} \times 0.7 \times 0.8 \times 0.8 \times 0.4 \\ &= 0.0028\epsilon \end{aligned}$$

Perplexity

	Initial	1st it.	2nd it.	3rd it.	...	Final
$p\,(\mathit{the\ haus} \mid \mathit{das\ haus})$	0.0625	0.1875	0.1905	0.1913	...	0.1875
$p\,(\mathit{the\ book} \mid \mathit{das\ buch})$	0.0625	0.1406	0.1790	0.2075	...	0.25
$p\,(\mathit{a\ book} \mid \mathit{ein\ buch})$	0.0625	0.1875	0.1907	0.1913	...	0.1875
perplexity (ϵ=1)	4095	202.3	153.6	131.6	...	113.8

Fluent Output

- small step, little step
- which one is better?
- show google result
- language model

Language Model

- relying on Google counts is a neat trick
- longer and longer sentences
- generative modeling
- The most common method for language modeling is the use of **n-gram** language models.

Trigram language models

$$
\begin{aligned}
p(\mathbf{e}) &= p(e_1, e_2, \ldots, e_n) \\
&= p(e_1)p(e_2|e_1)\cdots p(e_n|e_1, e_2, \ldots, e_{n-1}) \\
&\simeq p(e_1)p(e_2|e_1)\cdots p(e_n|e_{n-2}, e_{n-1})
\end{aligned}
$$

Noisy-Channel Model

Combining a language model and translation model in this way is called the noisy-channel model.

$$\begin{aligned}\text{argmax}_{\mathbf{e}} p(\mathbf{e}|\mathbf{f}) &= \text{argmax}_{\mathbf{e}} \frac{p(\mathbf{f}|\mathbf{e})p(\mathbf{e})}{p(\mathbf{f})} \\ &= \text{argmax}_{\mathbf{e}} p(\mathbf{f}|\mathbf{e})p(\mathbf{e})\end{aligned}$$

Higher IBM Models

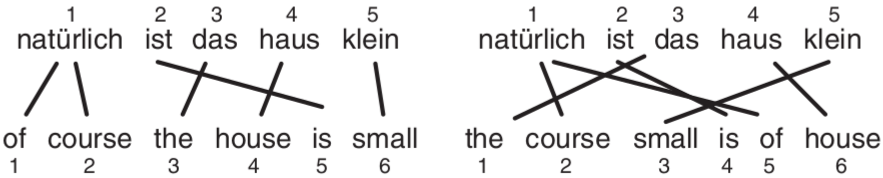
IBM Model 1	lexical translation
IBM Model 2	adds absolute reordering model
IBM Model 3	adds fertility model
IBM Model 4	relative reordering model
IBM Model 5	fixes deficiency

Fertility

- input words produce a specific number of output words in the output language.
- But some words produce multiple words or get dropped (producing zero words).
- A model for the fertility of words addresses this aspect of translation.

IBM Model 2

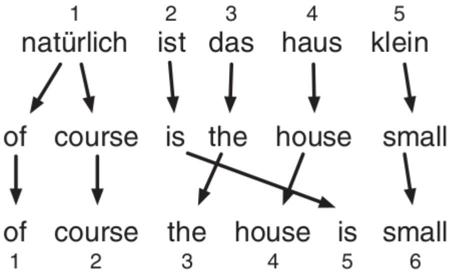
In IBM Model 1 the translation probabilities for the following two alternative translations are the same



IBM Model 2

- IBM Model 2 addresses the issue of alignment
- The translation of a foreign input word in position i to an English word in position j is modeled by an alignment probability distribution
- We can view translation under IBM Model 2 as a two-step process with a lexical translation step and an alignment step:

IBM Model 2



lexical translation step

alignment step

IBM Model 2

- Modeling alignment with an alignment probability distribution
- Translating foreign word at position i to English word at position j:

$$a(i|j, l_e, l_f)$$

- Putting everything together

$$p(\mathbf{e}, a|\mathbf{f}) = \epsilon \prod_{j=1}^{l_e} t(e_j|f_{a(j)})\ a(a(j)|j, l_e, l_f)$$

- EM training of this model works the same way as IBM Model 1

IBM Model 3

- how many words are generated from each input word.
- some German words like **zum** typically translate to two English words, i.e., **to the**.
- Others, such as the flavoring particle **ja**, get dropped.

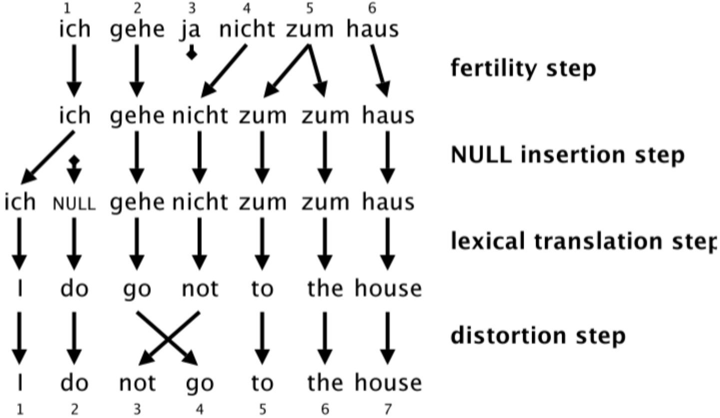
IBM Model 3

Model the fertility of input words directly with a probability distribution:

$$n(\phi|f)$$

$$n(1|\text{haus}) \simeq 1$$
$$n(2|\text{zum}) \simeq 1$$
$$n(0|\text{ja}) \simeq 1$$

Adding a model of fertilty



Sampling the Alignment Space

- raining IBM Model 3 with the EM algorithm
- Finding the most probable alignment by hillclimbing
- Sampling: collecting variations to collect statistics

Model 3 is already a powerful model

- Translation of words
- Reordering (distortion)
- Insertion of words (NULL insertion)
- Dropping of words (words with fertility 0)
- One-to-many translation (fertility)
- **IBM Model 4 further improves on Model 3**

Problem with IBM Model 3

- Distortion probability
- Large input and outputs
- Sparse data
- Relations sentences

IBM Model 4:Relative Distortion

In this model, the placement of the translation of an input word is typically based on the placement of the translation of the proceeding input word.

IBM Model 4

- Relative Distortion
- Word classes
- Not all words need to be reordered

IBM Model 5

- IBM Models 1–4 are *deficient*
  - some impossible translations have positive probability
  - multiple output words may be placed in the same position
  - $\rightarrow$ probability mass is wasted

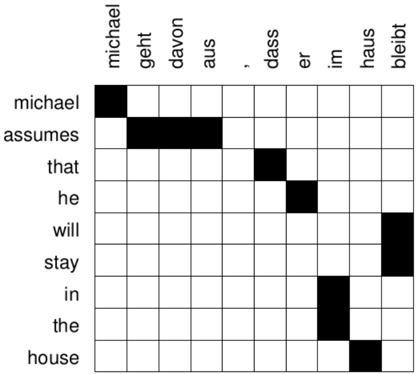
- IBM Model 5 fixes deficiency by keeping track of vacancies (available positions)

Conclusion

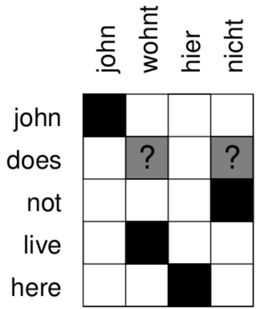
- IBM Models were the pioneering models in statistical machine translation
- Introduced important concepts
  - generative model
  - EM training
  - reordering models
- Only used for niche applications as translation model
- ... but still in common use for word alignment (e.g., GIZA++ toolkit)

Alignment Matrix

Given a sentence pair, which words correspond to each other?

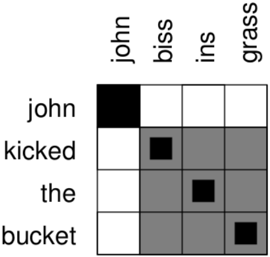


Word Alignment



Is the English word does aligned to
the German wohnt (verb) or nicht (negation) or neither?

Word Alignment



How do the idioms kicked the bucket and biss ins grass match up?
Outside this exceptional context, bucket is never a good translation for grass

Measuring Word Alignment Quality

- Manually align corpus with *sure* (S) and *possible* (P) alignment points ($S \subseteq P$)
- Common metric for evaluation word alignments: Alignment Error Rate (AER)

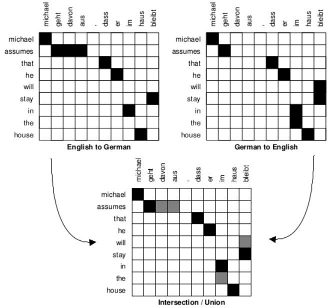
$$\text{AER}(S, P; A) = \frac{|A \cap S| + |A \cap P|}{|A| + |S|}$$

- AER $= 0$: alignment A matches all sure, any possible alignment points
- However: different applications require different precision/recall trade-offs

Word Alignment with IBM Models

- IBM Models create a **many-to-one** mapping
  - words are aligned using an alignment function
  - a function may return the same value for different input (one-to-many mapping)
  - a function can not return multiple values for one input (no many-to-one mapping)
- Real word alignments have **many-to-many** mappings

Symmetrizing Word Alignments



- Intersection of GIZA++ bidirectional alignments
- Grow additional alignment points [Och and Ney, CompLing2003]

Further Reading

- Chapter 4 of Koehn. P. (200p)

Recourse

- Jurafsky, D. and J. H. Martin. Speech and language processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, Second Edition, Upper Saddle River, NJ: Prentice-Hall, 2008.
- Koehn, P. (2009). Statistical Machine Translation. Cambridge: Cambridge University Press. doi:10.1017/CBO9780511815829