

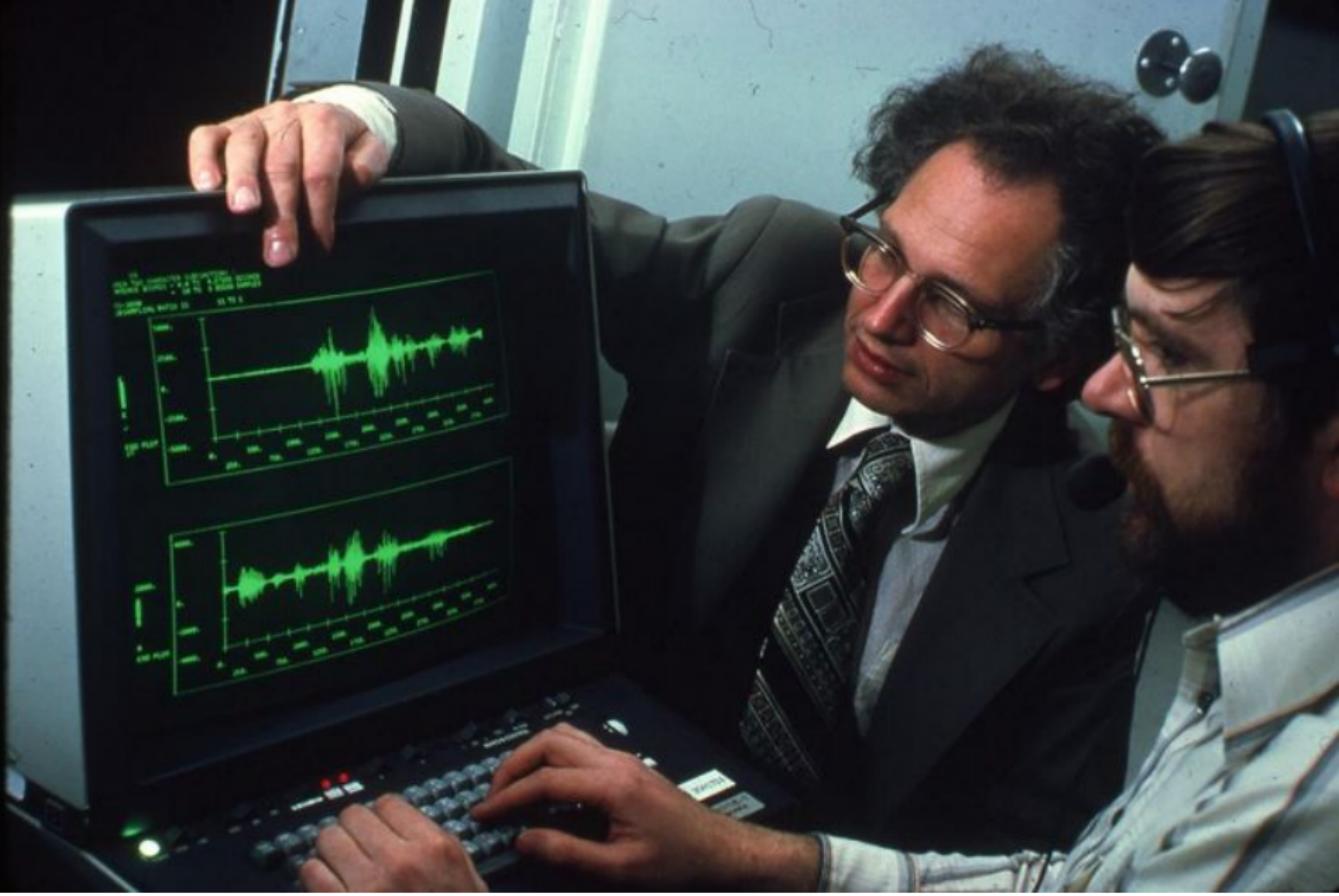
Machine Translation

Jordan Boyd-Graber

University of Maryland

Word-Based Models

Adapted from material by Philipp Koehn



Fred Jelinek showing off his ASR work at IBM (he later worked on MT)

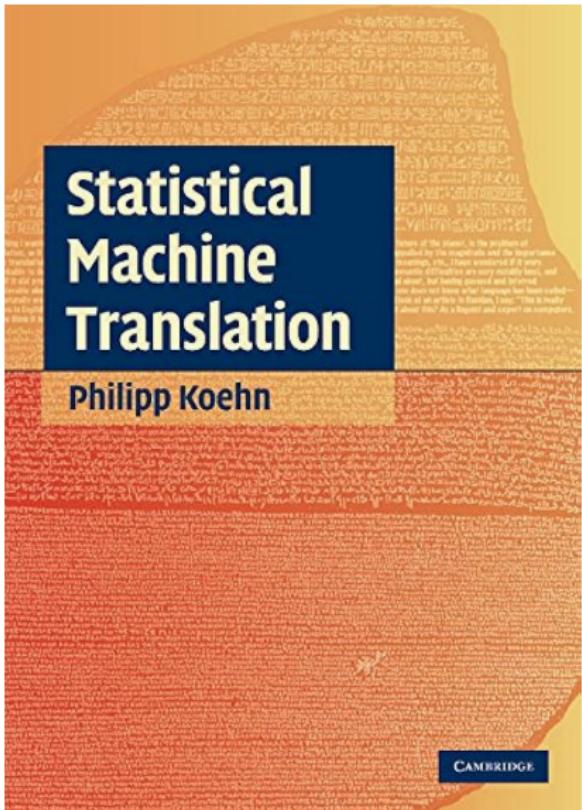
Roadmap

- Introduction to MT
- Components of MT system
- Word-based models
- Beyond word-based models

Roadmap

- Introduction to MT
- Components of MT system
- Word-based models
- Beyond word-based models: phrase-based and neural

Books by Philip Koehn



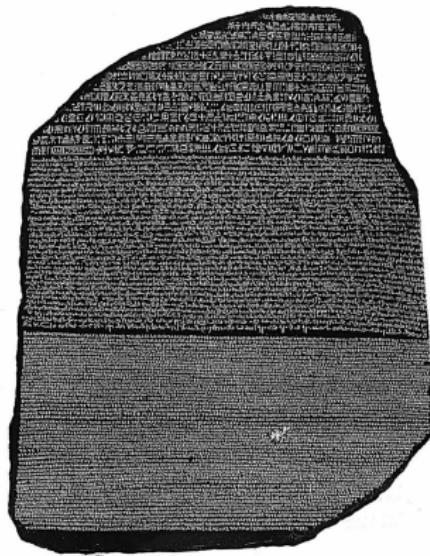
Philip Koehn est devenu l'un des leaders de la recherche en traduction automatique. Ses travaux ont contribué à la popularisation et au développement de la traduction automatique dans les années 1990 et 2000. Il a joué un rôle majeur dans le développement de la traduction statistique et a contribué à la mise en place de systèmes de traduction automatique capables de traiter des langues rares et peu documentées. Ses recherches ont également contribué à l'avancement des méthodes d'apprentissage automatique pour la traduction, et il a été l'un des premiers à utiliser des modèles de réseaux de neurones pour la traduction. Ses travaux ont également contribué à l'amélioration de la qualité de la traduction automatique dans divers domaines, tels que la traduction de documents officiels, la traduction de sites Web et la traduction de documents scientifiques. Ses contributions ont été reconnues par de nombreux prix et distinctions, dont le prix Turing de l'ACM et le prix de la meilleure publication dans le domaine de l'intelligence artificielle. Philip Koehn est actuellement professeur à l'université de Pennsylvanie et continue de faire des recherches dans le domaine de la traduction automatique.

Neural Machine Translation

Philip Koehn

CAMBRIDGE

What unlocks translations?

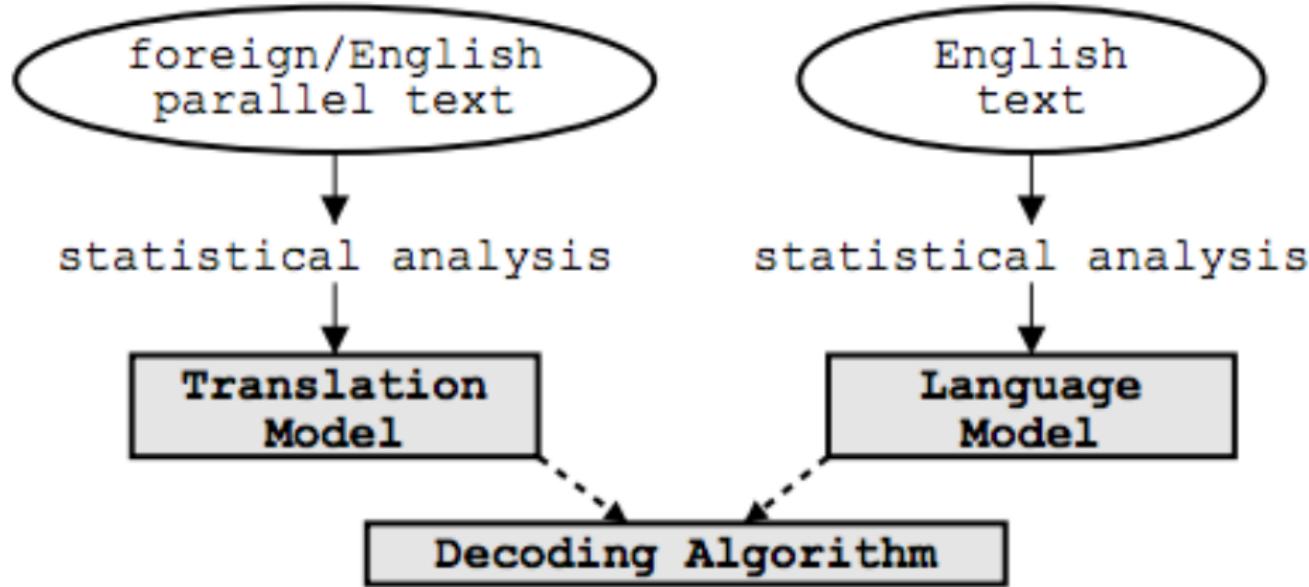


- Parallel data: Two languages, same meaning
- Rosetta stone: allowed us understand to Egyptian

What unlocks translations?



- Parallel data: Two languages, same meaning
- Rosetta stone: allowed us understand to Egyptian
- Where do we get them?
 - ▶ Some governments require translations (Canada, EU, Hong Kong)
 - ▶ Newspapers
 - ▶ Internet



Pieces of Machine Translation System

Terminology

- Source language: **f** (foreign)
- Target language: **e** (english)

Collect Statistics

Look at a parallel corpus (German text along with English translation)

Translation of Haus	Count
house	8,000
building	1,600
home	200
household	150
shell	50

Estimate Translation Probabilities

Maximum likelihood estimation

$$p_f(e) = \begin{cases} 0.8 & \text{if } e = \text{house}, \\ 0.16 & \text{if } e = \text{building}, \\ 0.02 & \text{if } e = \text{home}, \\ 0.015 & \text{if } e = \text{household}, \\ 0.005 & \text{if } e = \text{shell}. \end{cases}$$

1

2

3

4

das

Haus

ist

klein

|

|

|

|

the

house

is

small

1

2

3

4

Best case

1

2

3

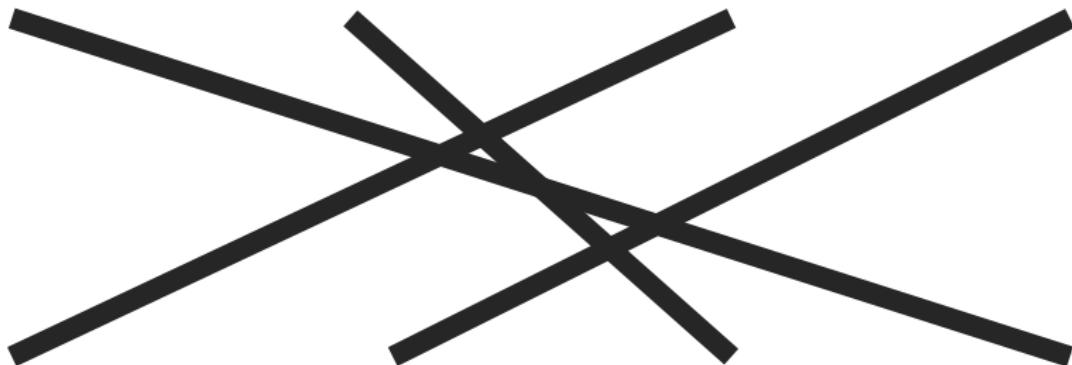
4

klein

ist

das

Haus



the

house

is

small

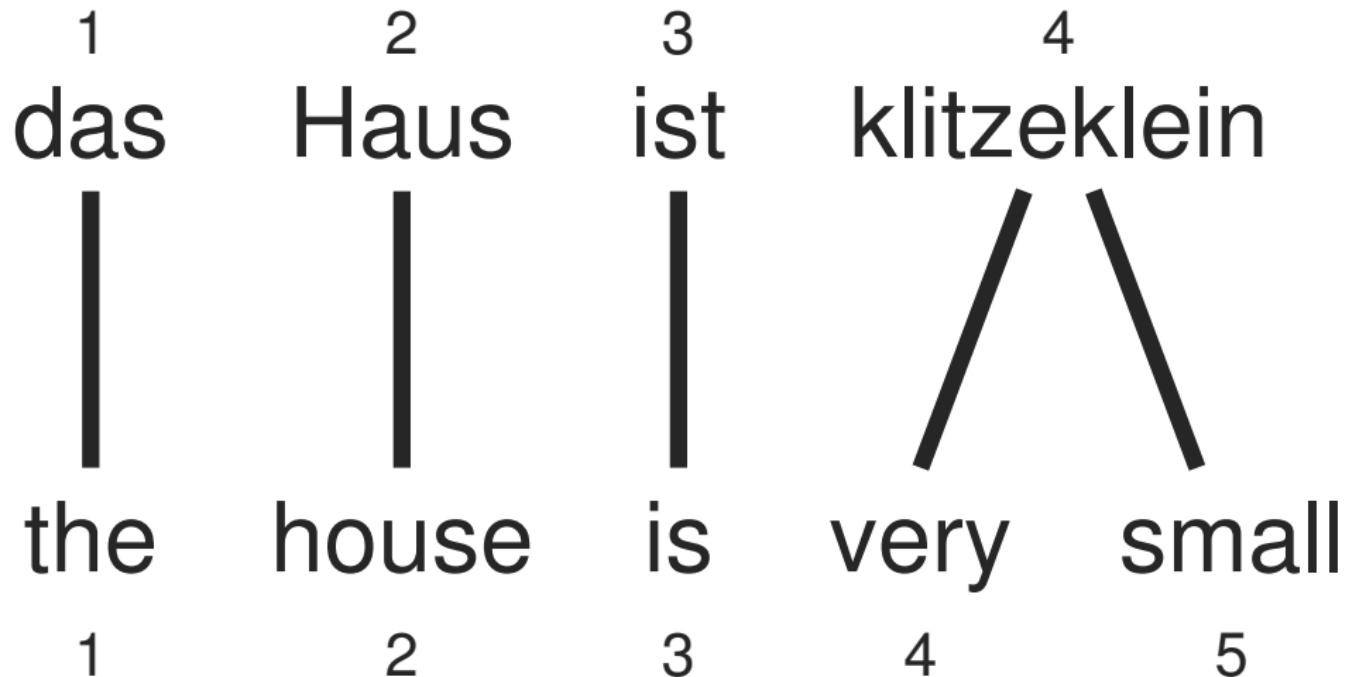
1

2

3

4

Words may be reordered during translation



A source word may translate into multiple target words

1

2

3

4

das

Haus

ist

klein



house

is

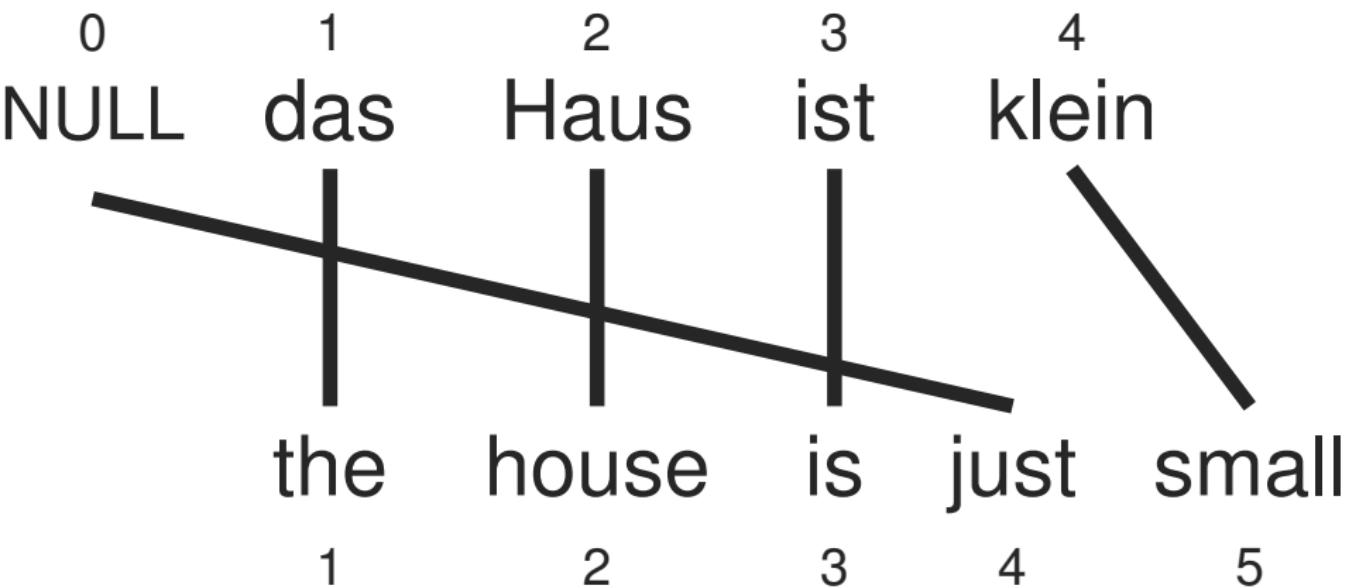
small

1

2

3

Words may be dropped when translated (**das**)



Words may be added during translation (**just**)

A family of lexical translation models

- A family translation models
- Uncreatively named: Model 1, Model 2, ...
- Foundation of all modern translation algorithms
- First up: Model 1

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 $\mathbf{f} = (f_1, \dots, f_{l_f})$ of length l_f
 - ▶ to an English sentence $\mathbf{e} = (e_1, \dots, e_{l_e})$ of length l_e
 - ▶ with an alignment of each English word e_j to a foreign word f_i according to the alignment function $a : j \rightarrow i$

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

- ▶ parameter ϵ is a normalization constant

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 $\mathbf{f} = (f_1, \dots, f_{l_f})$ of length l_f
 - ▶ to an English sentence $\mathbf{e} = (e_1, \dots, e_{l_e})$ of length l_e
 - ▶ with an alignment of each English word e_j to a foreign word f_i according to the alignment function $a : j \rightarrow i$

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

- ▶ parameter ϵ is a normalization constant

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 $\mathbf{f} = (f_1, \dots, f_{l_f})$ of length l_f
 - ▶ to an English sentence $\mathbf{e} = (e_1, \dots, e_{l_e})$ of length l_e
 - ▶ with an alignment of each English word e_j to a foreign word f_i according to the alignment function $a : j \rightarrow i$

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

- ▶ parameter ϵ is a normalization constant

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 $\mathbf{f} = (f_1, \dots, f_{l_f})$ of length l_f
 - ▶ to an English sentence $\mathbf{e} = (e_1, \dots, e_{l_e})$ of length l_e
 - ▶ with an alignment of each English word e_j to a foreign word f_i according to the alignment function $a : j \rightarrow i$

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

- ▶ parameter ϵ is a normalization constant

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 $\mathbf{f} = (f_1, \dots, f_{l_f})$ of length l_f
 - ▶ to an English sentence $\mathbf{e} = (e_1, \dots, e_{l_e})$ of length l_e
 - ▶ with an alignment of each English word e_j to a foreign word f_i according to the alignment function $a : j \rightarrow i$

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

- ▶ parameter ϵ is a normalization constant

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 $\mathbf{f} = (f_1, \dots, f_{l_f})$ of length l_f
 - ▶ to an English sentence $\mathbf{e} = (e_1, \dots, e_{l_e})$ of length l_e
 - ▶ with an alignment of each English word e_j to a foreign word f_i according to the alignment function $a : j \rightarrow i$

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

- ▶ parameter ϵ is a normalization constant

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 $\mathbf{f} = (f_1, \dots, f_{l_f})$ of length l_f
 - ▶ to an English sentence $\mathbf{e} = (e_1, \dots, e_{l_e})$ of length l_e
 - ▶ with an alignment of each English word e_j to a foreign word f_i according to the alignment function $a : j \rightarrow i$

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

- ▶ parameter ϵ is a normalization constant

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 $\mathbf{f} = (f_1, \dots, f_{l_f})$ of length l_f
 - ▶ to an English sentence $\mathbf{e} = (e_1, \dots, e_{l_e})$ of length l_e
 - ▶ with an alignment of each English word e_j to a foreign word f_i according to the alignment function $a : j \rightarrow i$

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

- ▶ parameter ϵ is a normalization constant

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
family	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

$$\begin{aligned}
 p(e, a | f) &= \frac{\epsilon}{5^4} \times t(\text{the} | \text{das}) \times t(\text{house} | \text{Haus}) \times t(\text{is} | \text{ist}) \times t(\text{small} | \text{klein}) \\
 &= \frac{\epsilon}{5^4} \times 0.7 \times 0.8 \times 0.8 \times 0.4 \\
 &= 0.00029\epsilon
 \end{aligned}$$

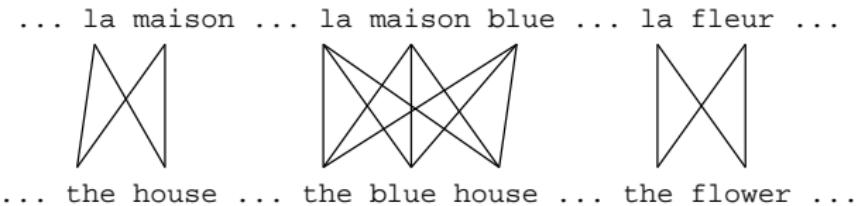
Learning Lexical Translation Models

- We would like to estimate the lexical translation probabilities $t(e|f)$ from a parallel corpus
- ... but we do not have the alignments
- Chicken and egg problem
 - ▶ if we had the alignments,
→ we could estimate the parameters of our generative model
 - ▶ if we had the parameters,
→ we could estimate the alignments

EM Algorithm

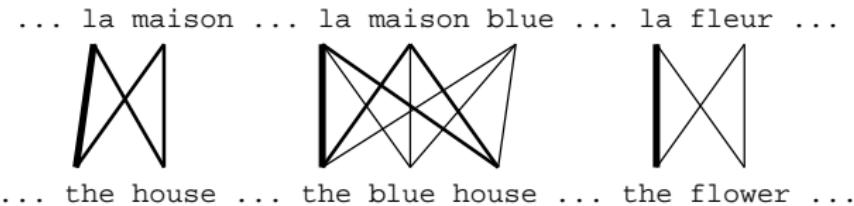
- Incomplete data
 - ▶ if we had complete data, would could estimate model
 - ▶ if we had model, we could fill in the gaps in the data
- Expectation Maximization (EM) in a nutshell
 1. initialize model parameters (e.g. uniform)
 2. assign probabilities to the missing data
 3. estimate model parameters from completed data
 4. iterate steps 2–3 until convergence

EM Algorithm



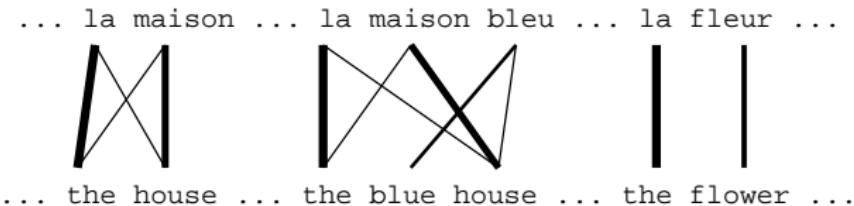
- Initial step: all alignments equally likely
- Model learns that, e.g., **la** is often aligned with **the**

EM Algorithm



- After one iteration
- Alignments, e.g., between **la** and **the** are more likely

EM Algorithm



- After another iteration
- It becomes apparent that alignments, e.g., between **fleur** and **flower** are more likely (pigeon hole principle)

EM Algorithm



- Convergence
- Inherent hidden structure revealed by EM

EM Algorithm

... la maison ... la maison bleu ... la fleur ...

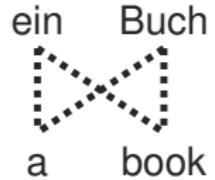
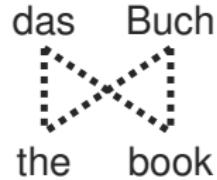
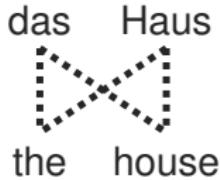
... the house ... the blue house ... the flower ...



$$\begin{aligned} p(\text{la}|\text{the}) &= 0.453 \\ p(\text{le}|\text{the}) &= 0.334 \\ p(\text{maison}|\text{house}) &= 0.876 \\ p(\text{bleu}|\text{blue}) &= 0.563 \\ \dots \end{aligned}$$

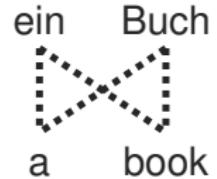
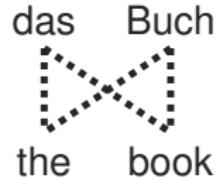
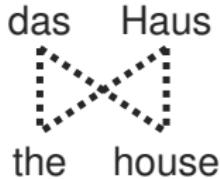
- Parameter estimation from the aligned corpus

Convergence



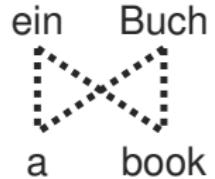
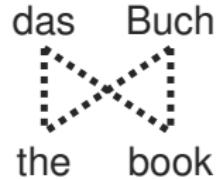
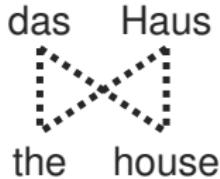
e	f	initial	1st it.	2nd it.	...	final
the	das	0.25	0.5	0.6364	...	1
book	das	0.25	0.25	0.1818	...	0
house	das	0.25	0.25	0.1818	...	0
the	buch	0.25	0.25	0.1818	...	0
book	buch	0.25	0.5	0.6364	...	1
a	buch	0.25	0.25	0.1818	...	0
book	ein	0.25	0.5	0.4286	...	0
a	ein	0.25	0.5	0.5714	...	1
the	haus	0.25	0.5	0.4286	...	0
house	haus	0.25	0.5	0.5714	...	1

Convergence



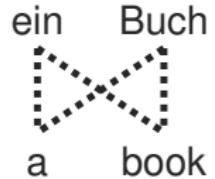
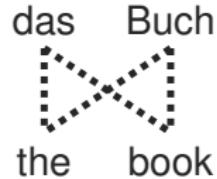
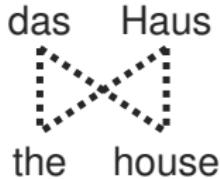
e	f	initial	1st it.	2nd it.	...	final
the	das	0.25	0.5	0.6364	...	1
book	das	0.25	0.25	0.1818	...	0
house	das	0.25	0.25	0.1818	...	0
the	buch	0.25	0.25	0.1818	...	0
book	buch	0.25	0.5	0.6364	...	1
a	buch	0.25	0.25	0.1818	...	0
book	ein	0.25	0.5	0.4286	...	0
a	ein	0.25	0.5	0.5714	...	1
the	haus	0.25	0.5	0.4286	...	0
house	haus	0.25	0.5	0.5714	...	1

Convergence



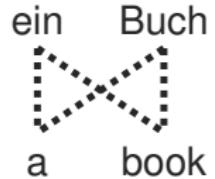
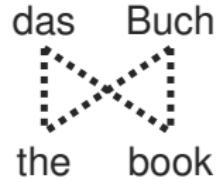
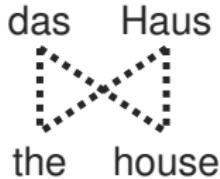
<i>e</i>	<i>f</i>	initial	1st it.	2nd it.	...	final
the	das	0.25	0.5	0.6364	...	1
book	das	0.25	0.25	0.1818	...	0
house	das	0.25	0.25	0.1818	...	0
the	buch	0.25	0.25	0.1818	...	0
book	buch	0.25	0.5	0.6364	...	1
a	buch	0.25	0.25	0.1818	...	0
book	ein	0.25	0.5	0.4286	...	0
a	ein	0.25	0.5	0.5714	...	1
the	haus	0.25	0.5	0.4286	...	0
house	haus	0.25	0.5	0.5714	...	1

Convergence



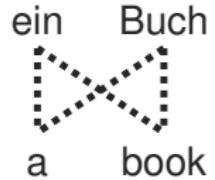
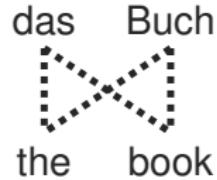
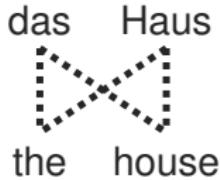
<i>e</i>	<i>f</i>	initial	1st it.	2nd it.	...	final
the	das	0.25	0.5	0.6364	...	1
book	das	0.25	0.25	0.1818	...	0
house	das	0.25	0.25	0.1818	...	0
the	buch	0.25	0.25	0.1818	...	0
book	buch	0.25	0.5	0.6364	...	1
a	buch	0.25	0.25	0.1818	...	0
book	ein	0.25	0.5	0.4286	...	0
a	ein	0.25	0.5	0.5714	...	1
the	haus	0.25	0.5	0.4286	...	0
house	haus	0.25	0.5	0.5714	...	1

Convergence



e	f	initial	1st it.	2nd it.	...	final
the	das	0.25	0.5	0.6364	...	1
book	das	0.25	0.25	0.1818	...	0
house	das	0.25	0.25	0.1818	...	0
the	buch	0.25	0.25	0.1818	...	0
book	buch	0.25	0.5	0.6364	...	1
a	buch	0.25	0.25	0.1818	...	0
book	ein	0.25	0.5	0.4286	...	0
a	ein	0.25	0.5	0.5714	...	1
the	haus	0.25	0.5	0.4286	...	0
house	haus	0.25	0.5	0.5714	...	1

Convergence



<i>e</i>	<i>f</i>	initial	1st it.	2nd it.	...	final
the	das	0.25	0.5	0.6364	...	1
book	das	0.25	0.25	0.1818	...	0
house	das	0.25	0.25	0.1818	...	0
the	buch	0.25	0.25	0.1818	...	0
book	buch	0.25	0.5	0.6364	...	1
a	buch	0.25	0.25	0.1818	...	0
book	ein	0.25	0.5	0.4286	...	0
a	ein	0.25	0.5	0.5714	...	1
the	haus	0.25	0.5	0.4286	...	0
house	haus	0.25	0.5	0.5714	...	1

Ensuring Fluent Output

- Our translation model cannot decide between **small** and **little**
- Sometime one is preferred over the other:
 - ▶ **small step**: 2,070,000 occurrences in the Google index
 - ▶ **little step**: 257,000 occurrences in the Google index
- Language model
 - ▶ estimate how likely a string is English
 - ▶ based on n-gram statistics

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

Write this in English



Die Kühe trinken Wasser.

yes

mouse

The

cat

water

drink

cows

pizza

Noisy Channel Model

- Bayes rule

$$p(a | b) = \frac{p(b | a)p(a)}{p(b)} \quad (1)$$

(2)

Noisy Channel Model

- Bayes rule

$$p(a | b) = \frac{p(b | a)p(a)}{p(b)} \quad (1)$$

- Turning English into Foreign

$$= \arg \max_{\mathbf{e}} p(\mathbf{e}) \quad (2)$$

Noisy Channel Model

- Bayes rule

$$p(a | b) = \frac{p(b | a)p(a)}{p(b)} \quad (1)$$

- Turning English into Foreign

$$= \arg \max_{\mathbf{e}} p(\mathbf{f} | \mathbf{e})p(\mathbf{e}) \quad (2)$$

Noisy Channel Model

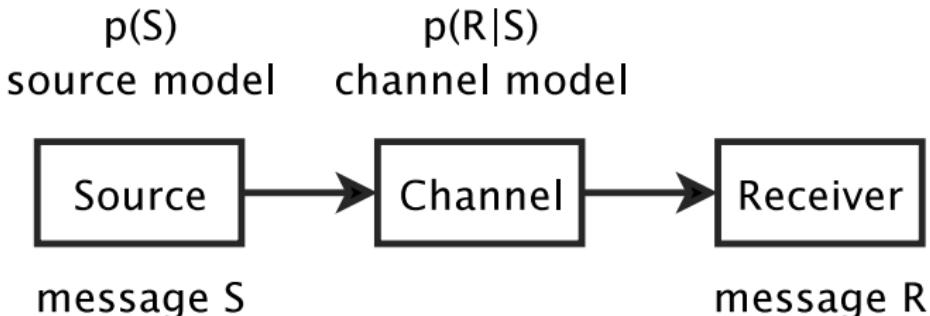
- Bayes rule

$$p(a | b) = \frac{p(b | a)p(a)}{p(b)} \quad (1)$$

- Turning English into Foreign

$$= \arg \max_{\mathbf{e}} \frac{p(\mathbf{f} | \mathbf{e})p(\mathbf{e})}{p(\mathbf{f})} \quad (2)$$

Noisy Channel Model



- Applying Bayes rule also called noisy channel model
 - ▶ we observe a distorted message R (here: a foreign string f)
 - ▶ we have a model on how the message is distorted
(here: translation model)
 - ▶ we have a model on what messages are probably
(here: language model)
 - ▶ we want to recover the original message S
(here: an English string e)

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

- Only IBM Model 1 has global maximum
 - ▶ training of a higher IBM model builds on previous model
- Computationally biggest change in Model 3
 - ▶ trick to simplify estimation does not work anymore
 - exhaustive count collection becomes computationally too expensive
 - ▶ sampling over high probability alignments is used instead

Legacy

- IBM Models were the pioneering models in statistical machine translation
- Introduced important concepts
 - ▶ generative model
 - ▶ EM training
 - ▶ reordering models

Attention vs. Alignment

What does Attention in Neural Machine Translation Pay Attention to?

Hamidreza Ghader and Christof Monz

Informatics Institute, University of Amsterdam, The Netherlands

h.ghader, c.monz@uva.nl

