UNIVERSITY OF AMSTERDAM

Introduction to Machine Translation

Reshmi G Pillai

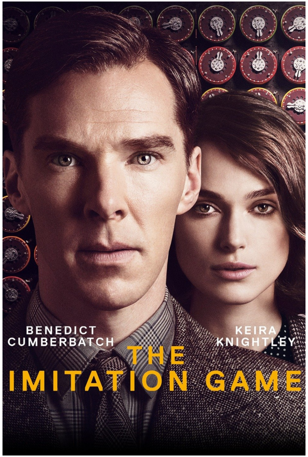
University of Amsterdam

Table of contents

A Brief History of MT

1940

Scientists at Bletchley park crack the **Enigma** using a proto-computer and can now decipher Nazi communication



1949

When I look at an article in Russian, I say: "This is really written in English, but it has been coded in some strange symbols. I will now proceed to decode"
- Warren Weaver

In the **Georgetown Experiment** IBM shows it can translate 60 simple sentences from Russian to English

```
IN:  Mi pyeryedayem mislyi
     posryedstvom ryechyi.

OUT: We transmit thoughts
     by means of speech.
```

Sentences in Russian are punched into standard cards for feeding into the electronic data processing machine for translation into English

1964

LANGUAGE
AND
MACHINES

COMPUTERS IN TRANSLATION AND LINGUISTICS

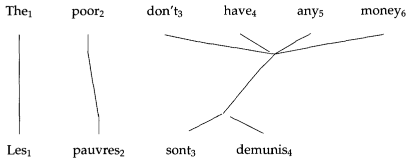
A Report by the
Automatic Language Processing Advisory Committee
Division of Behavioral Sciences
National Academy of Sciences
National Research Council

Publication 1416
National Academy of Sciences National Research Council
Washington, D. C. 1966

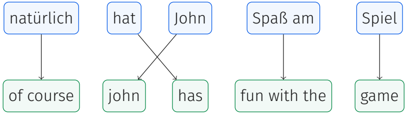
The ALPAC report in the US is highly skeptical of MT and funding is reduced dramatically

1993

IBM introduces a series of word-based statistical models, **IBM models 1-5**, that are induced from parallel data



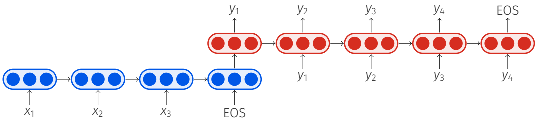
2003



Phrase-based SMT improves quality a lot over word-based models and becomes the basis for services like Google Translate

2013-2014

Neural Machine Translation is introduced and quickly becomes state-of-the-art



Alien Abduction

1. ok-voon ororok sprok .
   at-voon bichat dat .
2. ok-drubel ok-voon anok plok sprok .
   at-drubel at-voon pippat rrat dat .
3. erok sprok izok hihok ghirok .
   totat dat arrat vat hilat .
4. ok-voon anok drok brok jok .
   at-voon krat pippat sat lat .
5. wiwok farok izok stok .
   totat jjat quat cat .
6. lalok sprok izok jok stok .
   wat dat krat quat cat .
7. lalok farok ororok lalok sprok izok enemok .
   wat jjat bichat wat dat vat eneat .
8. lalok brok anok plok nok .
   iat lat pippat rrat nnat .
9. wiwok nok izok kantok ok-yurp .
   totat nnat quat oloat at-yurp .
10. lalok mok nok yorok ghirok clok .
    wat nnat gat mat bat hilat .
11. lalok nok crrrok hihok yorok zanzanok .
    wat nnat arrat mat zanzanat .
12. lalok rarok nok izok hihok mok .
    wat nnat forat arrat vat gat .

Dictionary

Arcturan	Centauri
arrat	hihok
at-drubel	ok-drubel
at-voon	ok-voon
at-yurp	ok-yurp
bat	clok
bichat	ororok
cat	stok
dat	sprok
eneat	enemok
forat	rarok
hilat	ghirok
jjat	farok

Arcturan	Centauri
krat	jok
lat	brok
mat	yorok
nnat	nok
oloat	kantok
pippat	anok
rrat	plok
totat	erok | wiwok
vat | quat	izok
wat | iat	lalok
zanzanat	zanzanok
???	crrrok

The aliens demand that you translate 3 new sentences!

13. ?
    iat lat pippat eneat hilat oloat at-yurp .
14. ?
    totat nnat forat arrat mat bat .
15. ?
    wat dat quat cat uskrat at-drubel .

Phew.. the aliens give you Centauri monolingual data!

ok-drubel anok ghirok farok . wiwok rarok nok zerok ghirok enemok . ok-drubel ziplok stok vok erok enemok kantok ok-yurp zinok jok yorok clok . lalok clok izok vok ok-drubel . ok-voon ororok sprok . ok-drubel ok-voon anok plok sprok . erok sprok izok hihok ghirok . ok-voon anok drok brok jok . wiwok farok izok stok . lalok sprok izok jok stok . lalok brok anok plok nok . lalok farok ororok lalok sprok izok enemok . wiwok nok izok kantok ok-yurp . lalok mok nok yorok ghirok clok . lalok nok crrrok hihok yorok zanzanok . lalok rarok nok izok hihok mok .

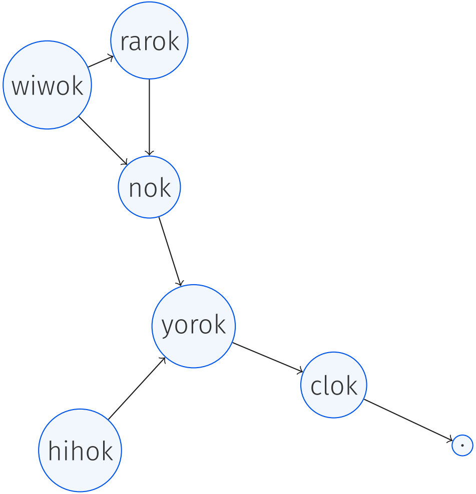
Bi-gram counts

1 . erok
7 . lalok
2 . ok-drubel
2 . ok-voon
3 . wiwok
1 anok drok
1 anok ghirok
2 anok plok
1 brok anok
1 brok jok
2 clok .
1 clok izok
1 crrrok hihok
1 drok brok
2 enemok .
1 enemok kantok
1 erok enemok
1 erok sprok
1 farok .
1 farok izok
1 farok ororok
1 ghirok .
1 ghirok clok
1 ghirok enemok
1 ghirok farok
1 hihok ghirok
1 hihok mok
1 hihok yorok
1 izok enemok
2 izok hihok
1 izok jok
1 izok kantok
1 izok stok
1 izok vok
1 jok .
1 jok stok
1 jok yorok
2 kantok ok-yurp
1 lalok brok
1 lalok clok
1 lalok farok
1 lalok mok
1 lalok nok
1 lalok rarok
2 lalok sprok
1 mok .
1 mok nok
1 nok .
1 nok crrrok
2 nok izok
1 nok yorok
1 nok zerok
1 ok-drubel .
1 ok-drubel anok
1 ok-drubel ok-voon
1 ok-drubel ziplok
2 ok-voon anok
1 ok-voon ororok
1 ok-yurp .
1 ok-yurp zinok
1 ororok lalok
1 ororok sprok
1 plok nok
1 plok sprok
2 rarok nok
2 sprok .
3 sprok izok
2 stok .
1 stok vok
1 vok erok
1 vok ok-drubel
1 wiwok farok
1 wiwok nok
1 wiwok rarok
1 yorok clok
1 yorok ghirok
1 yorok zanzanok
1 zanzanok .
1 zerok ghirok
1 zinok jok
1 ziplok stok

Sentence 1 done!

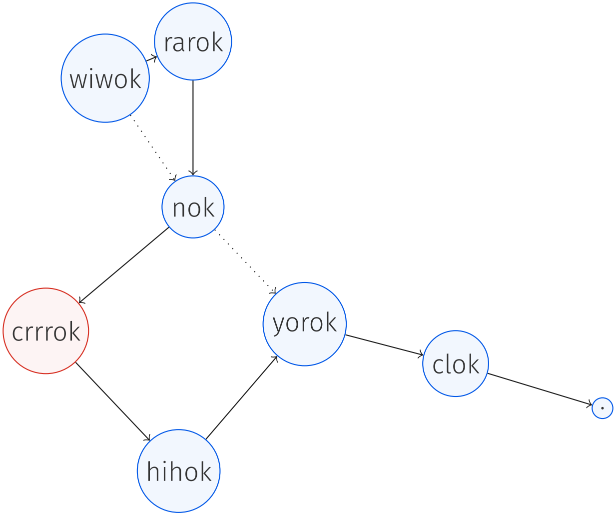
13. lalok brok anok ghirok enemok kantok ok-yurp .
    iat lat pippat eneat hilat oloat at-yurp .
14. ?
    totat nnat forat arrat mat bat .
15. ?
    wat dat quat cat uskrat at-drubel .

Putting a Centauri sentence in order



Problem: there is no path that connects all words!

Putting a Centauri sentence in order



Solution: add special word ‘crrrok’

Two down, one to go!

13. lalok brok anok ghirok enemok kantok ok-yurp .
    iat lat pippat eneat hilat oloat at-yurp .
14. wiwok rarok nok crrrok hihok yorok clok .
    totat nnat forat arrat mat bat .
15. ?
    wat dat quat cat uskrat at-drubel .

Translating sentence 3

13. lalok brok anok ghirok enemok kantok ok-yurp .
    iat lat pippat eneat hilat oloat at-yurp .
14. wiwok rarok nok crrrok hihok yorok clok .
    totat nnat forat arrat mat bat .
15. lalok sprok izok stok **????** ok-drubel .
    wat dat quat cat uskrat at-drubel .

We could guess the missing word by looking at the bi-gram counts

Congratulations!

The aliens hired you as their translator!

Was this realistic?

- Only 2 words were **ambiguous**
- **Sentence lengths** were very similar
- All sentences were very **short**
- We only used **bi-grams** for disambiguation
- Output order should depend on input order
  - John loves Mary
  - Mary loves John
- The data was **cooked** – without sentences (8) and (9) we would have difficulty to make the remaining alignments
- We did not use any **phrasal** dictionaries
- And: pronouns? inflectional morphology? structural ambiguity? domain knowledge? scope of negation?

Was this realistic?

- Only 2 words were **ambiguous**
- **Sentence lengths** were very similar
- All sentences were very **short**
- We only used **bi-grams** for disambiguation
- Output order should depend on input order
    - John loves Mary
    - Mary loves John
- The data was **cooked** – without sentences (8) and (9) we would have difficulty to make the remaining alignments
- We did not use any **phrasal** dictionaries
- And: pronouns? inflectional morphology? structural ambiguity? domain knowledge? scope of negation?
- **It was sort of real!** **You translated Spanish to English!**

You translated Spanish into English!

1. Garcia and associates.
   Garcia y asociados.
2. Carlos Garcia has three associates.
   Carlos Garcia tiene tres asociados.
3. his associates are not strong.
   sus asociados no son fuertes.
4. Garcia has a company also.
   Garcia tambien tiene una empresa.
5. its clients are angry.
   sus clientes están enfadados.
6. the associates are also angry.
   los asociados tambien están enfadados.
7. the clients and the associates are enemies.
   los clientes y los asociados son enemigos.
8. the company has three groups.
   la empresa tiene tres grupos.
9. its groups are in Europe.
   sus grupos están en Europa.
10. the modern groups sell strong pharmaceuticals.
    los grupos modernos venden medicinas fuertes.
11. the groups do not sell zanzanine.
    los grupos no venden zanzanina.
12. the small groups are not modern.
    los grupos pequeños no son modernos.

You also translated (13):
"la empresa tiene enemigos fuertes en Europa"
"the company has strong enemies in Europe"

If we hadn't flipped "ghirok" and "enemok", we would have gotten:
"the company has enemies strong in Europe"

And (14):
"sus grupos pequeños no venden medicinas"
"its small groups do not sell pharmaceuticals"

The word **'crrrok'** turns out to be the English word **'do'**!

Statistical Machine Translation

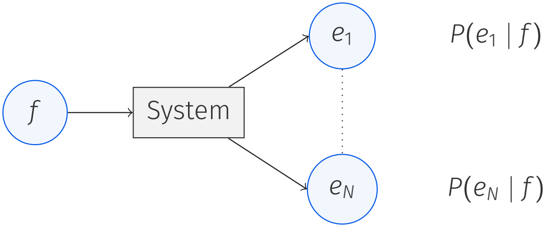
A Statistical Approach

Given a French sentence f, find English sentence $\hat{e}$ that maximizes $P(e \mid f)$

$$\hat{e} = \operatorname*{argmax}_{e} P(e \mid f)$$

"the most likely translation"

How not to do it



Bayes' Rule

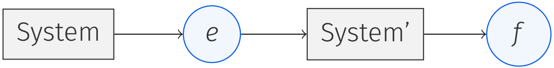
$$P(e \mid f) = \frac{P(f \mid e)\,P(e)}{P(f)}$$

The Noisy Channel

$$\operatorname*{argmax}_e P(e \mid f) = \operatorname*{argmax}_e \underbrace{P(f \mid e)}_{\text{channel}} \underbrace{P(e)}_{\text{source}}$$

- the source is the language model
- the channel is the translation model

Generative Story



- the story says French sentences come from English sentences
- we will use this model in the opposite direction

MT as Crime Scene Investigation

Sentence f is a "crime scene".

Our generative model might be something like: some person e decided to do the crime, and then that person actually did the crime. So we start reasoning about:

1. **who did it?** $P(e)$: motive, personality,...
2. **how did they do it?** $P(f \mid e)$: transportation, weapons, ...

These two things may conflict.
Someone with a good motive, but without the means - worries about good English
Someone who could easily have done the crime, but has no motive - worries about French that matches the given English

MT as Crime Scene Investigation

Word reordering

If we model $P(e \mid f)$ directly, there is not much margin for error.

We can use $P(f \mid e)$ to make sure that words in f are generally translations of words in e

$P(e)$ then ensures that the translation e is also grammatical

Word reordering

If we model $P(e \mid f)$ directly, there is not much margin for error.

We can use $P(f \mid e)$ to make sure that words in f are generally translations of words in e

$P(e)$ then ensures that the translation e is also grammatical

Would this work? Let's try it:

- have
- programming
- a
- seen
- never
- I
- language
- better

Word choice

The $P(e)$ model can also be useful for *selecting* English translations of French words.

We need this especially when the French word is **ambiguous**.

Word choice

The $P(e)$ model can also be useful for *selecting* English translations of French words.

We need this especially when the French word is **ambiguous**.

Example
A French word translates as either "**in**" or "**on**".

Now there may be two English strings with equally good $P(f \mid e)$ scores:

1. she is in the end zone
2. she is on the end zone

$P(e)$ selects the right one

IBM Model 3 [Brown et al., 1990, Brown et al., 1993]

TL;DR
Translate word by word, then scramble the words around into the right word order

First observations:

- English words may produce multiple French words
- English words may disappear

We need to account for this.

IBM Model 3 [Brown et al., 1990, Brown et al., 1993]

TL;DR
Translate word by word, then scramble the words around into the right word order

First observations:

- English words may produce multiple French words
- English words may disappear

We need to account for this.

The story of IBM Model 3

- For each English word e_i
  - choose a fertility ϕ_i
  - generate ϕ_i French words
  - generate spurious word
- Permute French words
  - assign an absolute position to each French word
  - … based on the absolute position of the English word that generates it

IBM3: Example

Mary **did** not **slap** the green witch

↓

Mary not **slap slap slap** the green witch

↓

Mary not slap slap slap **NULL** the green witch

↓

Mary no daba una botefada a la verde bruja

↓

Mary no daba una botefada a la **bruja verde**

IBM3: Parameters

1. Translation $t(\text{huis} \mid \text{house})$
2. Fertility $n(1 \mid \text{house})$
3. Spurious p
4. Position $d(1 \mid 2, |e|, |f|)$

How do we learn these parameters?

If we had **rewriting examples**, then we could estimate $n(0 \mid \text{`did'})$ by finding every ‘did’ and checking what happened to it

Example
If ‘did’ appeared 15,000 times and was deleted during the first rewriting step 13,000 times, then $n(0 \mid \text{`did'}) = \frac{13}{15}$

How do we learn these parameters?

If we had **rewriting examples**, then we could estimate $n(0 \mid \text{'did'})$ by finding every 'did' and checking what happened to it

Example
If 'did' appeared 15,000 times and was deleted during the first rewriting step 13,000 times, then $n(0 \mid \text{'did'}) = \frac{13}{15}$

Chicken-and-egg problem

- If we had **word alignments** instead of rewriting examples, we could also obtain the parameters. (But.. we don't!)
- If we had the **parameters** we could get the word alignments. (But.. we don't!)

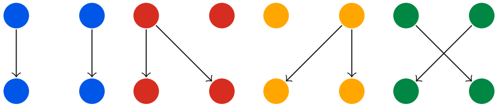
EM intuition

- Let's say we **do** have alignments, but for each sentence we have **multiple** ones
- Let's say we have **2** alignments for each sentence
- We don't know which one is best
- We could simply multiply the counts from both possible alignments by $\frac{1}{2}$
- We call these fractional counts

EM intuition

- Let's say we **do** have alignments, but for each sentence we have **multiple** ones
- Let's say we have **2** alignments for each sentence
- We don't know which one is best
- We could simply multiply the counts from both possible alignments by $\frac{1}{2}$
- We call these fractional counts

- We need to consider **all possible alignments**, not just 2
- No problem! We use **fractional counts**, and we just multiply with a smaller number.



EM

We start by assigning **uniform** parameter values to our $t(f \mid e)$

Example
Let's say we have 40000 French words in our vocabulary
Then each $t(f|e) = \frac{1}{40000}$

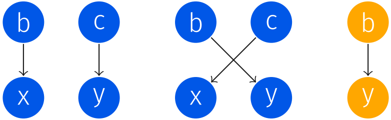
> *We can do the same for the other parameters, but for now let's focus on obtaining better $t(f \mid e)$ parameters*

EM: Example

Let's say we have a small **corpus** with only 2 sentences:

English	French
b c	x y
b	y

The first sentence has **two possibilities**, the second one has only **one**:



We have now **simplified** our model to be **IBM Model 1**:

$$P(a, f \mid e) = \prod_{j=1}^{M} t(f_j \mid e_{a_j})$$

i.e. multiply the probabilities of aligned words

EM: Initialization

Remember our corpus:

English	French
b c	x y
b	y

Start with uniform parameters:

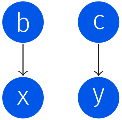
$$t(x \mid b) = \frac{1}{2}$$

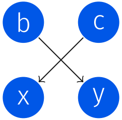
$$t(y \mid b) = \frac{1}{2}$$

$$t(x \mid c) = \frac{1}{2}$$

$$t(y \mid c) = \frac{1}{2}$$

Step 1
Compute $P(a, f|e)$ for each possible alignment





$$P(a, f|e) = \frac{1}{2} * \frac{1}{2} = \frac{1}{4}$$

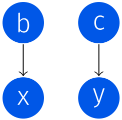
$$P(a, f|e) = \frac{1}{2} * \frac{1}{2} = \frac{1}{4}$$

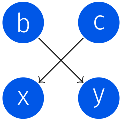
$$P(a, f|e) = \frac{1}{2}$$

EM: Step 2

Step 2

Normalize $P(a, f \mid e)$ to yield $P(a \mid e, f)$





$$P(a|e,f) = \frac{\frac{1}{4}}{\frac{1}{4} + \frac{1}{4}} = \frac{1}{2}$$

$$P(a|e,f) = \frac{\frac{1}{4}}{\frac{1}{4} + \frac{1}{4}} = \frac{1}{2}$$

$$P(a,f|e) = \frac{\frac{1}{2}}{\frac{1}{2}} = 1$$

EM: Step 3 and 4

Step 3
Collect fractional counts

$$tc(x \mid b) = \frac{1}{2}$$
$$tc(y \mid b) = \frac{1}{2} + 1 = 1\frac{1}{2}$$
$$tc(x \mid c) = \frac{1}{2}$$
$$tc(y \mid c) = \frac{1}{2}$$

EM: Step 3 and 4

Step 3
Collect fractional counts

$$tc(x \mid b) = \frac{1}{2}$$

$$tc(y \mid b) = \frac{1}{2} + 1 = 1\frac{1}{2}$$

$$tc(x \mid c) = \frac{1}{2}$$

$$tc(y \mid c) = \frac{1}{2}$$

Step 4

Normalize fractional counts

$$t(x \mid b) = \frac{\frac{1}{2}}{\frac{1}{2} + 1\frac{1}{2}} = \frac{1}{4}$$

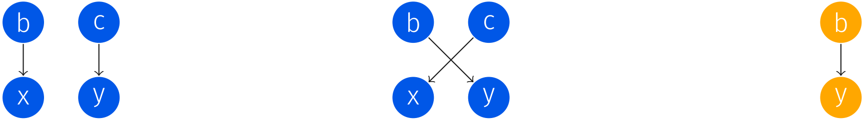
$$t(y \mid b) = \frac{1\frac{1}{2}}{\frac{1}{2} + 1\frac{1}{2}} = \frac{3}{4}$$

$$t(x \mid c) = \frac{\frac{1}{2}}{\frac{1}{2} + \frac{1}{2}} = \frac{1}{2}$$

$$t(y \mid c) = \frac{\frac{1}{2}}{\frac{1}{2} + \frac{1}{2}} = \frac{1}{2}$$

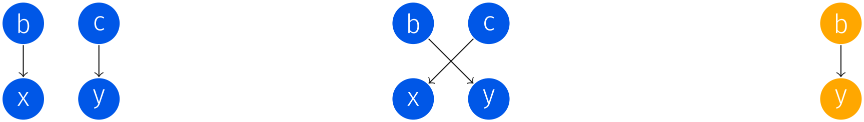
These are the revised parameters!

Step 1 (again, now using the new parameters)
Compute $P(a, f|e)$ for each possible alignment



EM: Repeat step 1

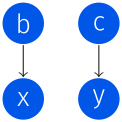
Step 1 (again, now using the new parameters)
Compute $P(a, f|e)$ for each possible alignment

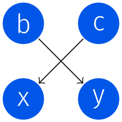


$$P(a, f|e) = \frac{1}{4} * \frac{1}{2} = \frac{1}{8}$$

EM: Repeat step 1

Step 1 (again, now using the new parameters)
Compute $P(a, f|e)$ for each possible alignment



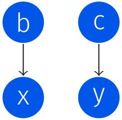


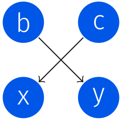
$$P(a, f|e) = \frac{1}{4} * \frac{1}{2} = \frac{1}{8}$$

$$P(a, f|e) = \frac{3}{4} * \frac{1}{2} = \frac{3}{8}$$

EM: Repeat step 1

Step 1 (again, now using the new parameters)
Compute $P(a, f|e)$ for each possible alignment





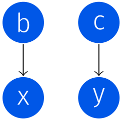
$$P(a, f|e) = \frac{1}{4} * \frac{1}{2} = \frac{1}{8}$$

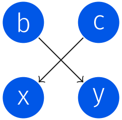
$$P(a, f|e) = \frac{3}{4} * \frac{1}{2} = \frac{3}{8}$$

$$P(a, f|e) = \frac{3}{4}$$

EM: Repeat step 2

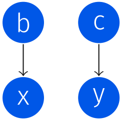
Step 2 (again)
Normalize $P(a, f \mid e)$ to yield $P(a \mid e, f)$

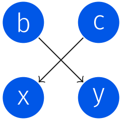




EM: Repeat step 2

Step 2 (again)
Normalize $P(a, f \mid e)$ to yield $P(a \mid e, f)$

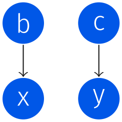


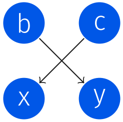


$$P(a|e,f) = \frac{\frac{1}{8}}{\frac{1}{8}+\frac{3}{8}} = \frac{1}{4}$$

EM: Repeat step 2

Step 2 (again)
Normalize $P(a, f \mid e)$ to yield $P(a \mid e, f)$



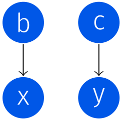


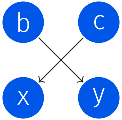
$$P(a|e,f) = \frac{\frac{1}{8}}{\frac{1}{8}+\frac{3}{8}} = \frac{1}{4}$$

$$P(a|e,f) = \frac{\frac{3}{8}}{\frac{1}{8}+\frac{3}{8}} = \frac{3}{4}$$

EM: Repeat step 2

Step 2 (again)
Normalize $P(a, f \mid e)$ to yield $P(a \mid e, f)$





$$P(a|e,f) = \frac{\frac{1}{8}}{\frac{1}{8}+\frac{3}{8}} = \frac{1}{4}$$

$$P(a|e,f) = \frac{\frac{3}{8}}{\frac{1}{8}+\frac{3}{8}} = \frac{3}{4}$$

$$P(a,f|e) = \frac{\frac{3}{4}}{\frac{3}{4}} = 1$$

EM: Repeat steps 3 and 4

Step 3 (again)
Collect fractional counts

$$tc(x \mid b) =$$

$$tc(y \mid b) =$$

$$tc(x \mid c) =$$

$$tc(y \mid c) =$$

EM: Repeat steps 3 and 4

Step 3 (again)
Collect fractional counts

$$tc(x \mid b) = \frac{1}{4}$$
$$tc(y \mid b) = \frac{3}{4} + 1 = 1\frac{3}{4}$$
$$tc(x \mid c) = \frac{3}{4}$$
$$tc(y \mid c) = \frac{1}{4}$$

Step 4 (again)
Normalize fractional counts

$$t(x \mid b) =$$
$$t(y \mid b) =$$
$$t(x \mid c) =$$
$$t(y \mid c) =$$

Even better parameters!

EM: Repeat steps 3 and 4

Step 3 (again)
Collect fractional counts

$$tc(x \mid b) = \frac{1}{4}$$

$$tc(y \mid b) = \frac{3}{4} + 1 = 1\frac{3}{4}$$

$$tc(x \mid c) = \frac{3}{4}$$

$$tc(y \mid c) = \frac{1}{4}$$

Step 4 (again)

Normalize fractional counts

$$t(x \mid b) = \frac{\frac{1}{4}}{\frac{1}{4} + 1\frac{3}{4}} = \frac{1}{8}$$

$$t(y \mid b) = \frac{1\frac{3}{4}}{\frac{1}{4} + 1\frac{3}{4}} = \frac{7}{8}$$

$$t(x \mid c) = \frac{\frac{3}{4}}{\frac{3}{4} + \frac{1}{4}} = \frac{3}{4}$$

$$t(y \mid c) = \frac{\frac{1}{4}}{\frac{3}{4} + \frac{1}{4}} = \frac{1}{4}$$

Even better parameters!

If we do this many many times..

$$t(x \mid b) = 0.0001$$
$$t(y \mid b) = 0.9999$$
$$t(x \mid c) = 0.9999$$
$$t(y \mid c) = 0.0001$$

Notes on EM

- Each iteration of the EM algorithm is guaranteed to improve $P(f \mid e)$
- EM is not guaranteed to find a global optimum, but rather only a local optimum
- Where EM ends up is therefore a function of where it starts

Notes on IBM Model 3

EM for Model 3 is just like this!

Except for:

- we use Model 3's formula for $P(a \mid f, e)$
- we also collect fractional counts for:
  - n (fertility)
  - p (spurious word insertion)
  - d (reordering)

Notes on IBM Model 3

EM for Model 3 is just like this!

Except for:

- we use Model 3's formula for $P(a \mid f, e)$
- we also collect fractional counts for:
  - n (fertility)
  - p (spurious word insertion)
  - d (reordering)

A few critical notes:

- The **distortion parameters** in Model 3 are a very weak description of word-order change in translation
- This model is **deficient**
  - The reordering step in the generative story allows words to pile up on top of each other!

Decoding

With a language model $p(e)$ and a translation model $p(f \mid e)$,
we want to find $\hat{e}$, the best translation:

$$\hat{e} = \arg\max_{e} P(f \mid e)\, P(e)$$

- This process of finding $\hat{e}$ is called decoding
- It is impossible to search through all possible sentences
- .. but we can inspect a highly relevant subset of such sentences

Phrase-based Statistical Machine Translation

Phrase-based SMT

Atomic units

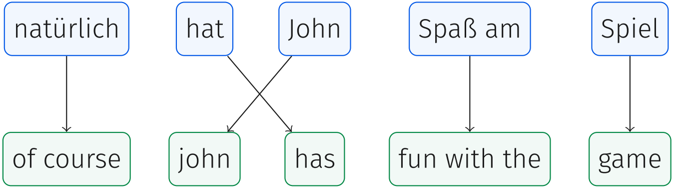
- In the IBM models, the atomic units of translation are **words**
- In phrase-based models, the atomic units are **phrases**, i.e. a few consecutive words

Advantages

- Handle **many-to-many** translation
- Capture **local context**
- **More data** gives us **more phrases**
- No more fertility, insertion, deletion

For a long time this was the main approach for Google Translate

Phrase alignment



segment the input, translate, reorder[1]

[1]Adapted from: Philipp Koehn. Statistical Machine Translation.

Phrase table for ‘natürlich’

Translation	Probability $\phi(\bar{e} \mid \bar{f})$
of course	0.5
naturally	0.3
of course ,	0.15
, of course ,	0.05

‘natürlich’ translates into two words, so we want a mapping to a phrase!

The Noisy Channel – same as before

$$\operatorname*{argmax}_{e} P(\mathrm{e} \mid \mathrm{f}) = \operatorname*{argmax}_{\mathrm{e}} \underbrace{P(\mathrm{f} \mid \mathrm{e})}_{\text{channel}} \underbrace{P(\mathrm{e})}_{\text{source}}$$

- the source is the language model
- the channel is the translation model (now using phrases!)

Decomposition of $P(\mathbf{f} \mid \mathbf{e})$

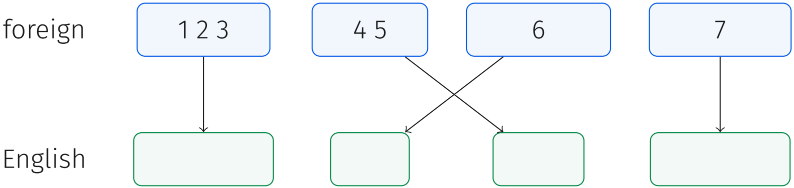
$$
\begin{aligned}
P(\mathbf{f} \mid \mathbf{e}) &= P(f_{1 \ldots M} \mid e_{1 \ldots N}) \\
&= \underbrace{\prod_i \phi(\bar{f}_i \mid \bar{e}_i)}_{\text{phrases}} \; \underbrace{d(start_i - end_{i-1} - 1)}_{\text{distance based reordering}}
\end{aligned}
$$

Decomposition of $P(\mathrm{f} \mid \mathrm{e})$

$$
\begin{aligned}
P(\mathrm{f} \mid \mathrm{e}) &= P(f_{1\ldots M} \mid e_{1\ldots N}) \\
&= \underbrace{\prod_i \phi(\bar{f}_i \mid \bar{e}_i)}_{\text{phrases}} \; \underbrace{d(start_i - end_{i-1} - 1)}_{\text{distance based reordering}}
\end{aligned}
$$

product of translating each English phrase into its foreign phrase & reordering

Distance based reordering

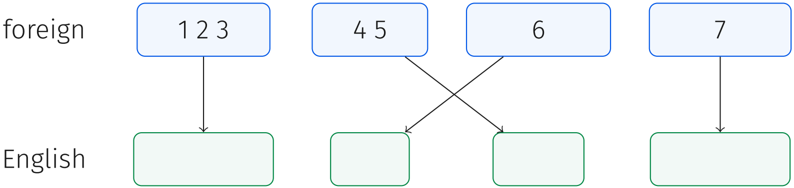


Q: What is the distance for the **second** English phrase?[2]

$$P(f_{1\ldots M} \mid e_{1\ldots N}) = \prod_i \phi(\bar{f}_i \mid \bar{e}_i)\ \underbrace{d(start_i - end_{i-1} - 1)}_{\text{distance based reordering}}$$

[2]Distance is measured on the foreign side!

Distance based reordering



Q: What is the distance for the **second** English phrase?[2]

$$P(f_{1\dots M} \mid e_{1\dots N}) = \prod_i \phi(\bar{f}_i \mid \bar{e}_i) \; \underbrace{d(start_i - end_{i-1} - 1)}_{\text{distance based reordering}}$$

Answer: $start_2$ - end_1 - 1 = 6 - 3 - 1 = 2

[2] Distance is measured on the foreign side!

Phrase extraction

How do we get phrases?
We extract all phrases that are consistent with a word alignment A

Phrase extraction

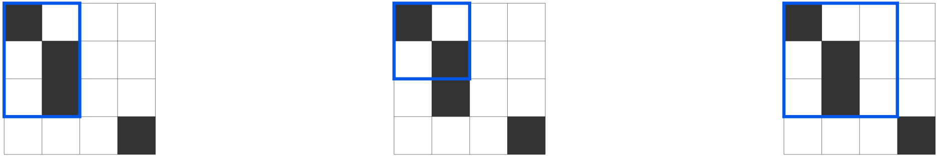
How do we get phrases?
We extract **all phrases** that are **consistent** with a word alignment A

Definition: Consistent phrase pair
A phrase pair $(\bar{f}, \bar{e})$ is consistent with A, if all words $f_1, \ldots, f_N$ in $\bar{f}$ that have alignment points in A, have these with words $e_1, \ldots, e_M$ in $\bar{e}$, and vice versa.

Phrase extraction

How do we get phrases?
We extract **all phrases** that are **consistent** with a word alignment A

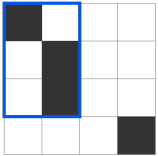
Definition: Consistent phrase pair
A phrase pair $(\bar{f}, \bar{e})$ is consistent with A, if all words $f_1, \ldots, f_N$ in $\bar{f}$ that have alignment points in A, have these with words $e_1, \ldots, e_M$ in $\bar{e}$, and vice versa.



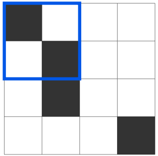
Phrase extraction

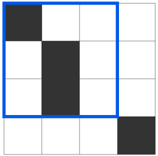
How do we get phrases?
We extract **all phrases** that are **consistent** with a word alignment A

Definition: Consistent phrase pair
A phrase pair $(\bar{f}, \bar{e})$ is consistent with A, if all words $f_1, \ldots, f_N$ in $\bar{f}$ that have alignment points in A, have these with words $e_1, \ldots, e_M$ in $\bar{e}$, and vice versa.



Consistent

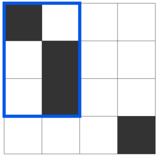




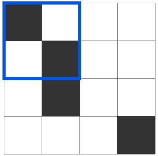
Phrase extraction

How do we get phrases?
We extract **all phrases** that are **consistent** with a word alignment A

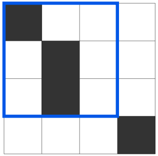
Definition: Consistent phrase pair
A phrase pair $(\bar{f}, \bar{e})$ is consistent with A, if all words $f_1, \ldots, f_N$ in $\bar{f}$ that have alignment points in A, have these with words $e_1, \ldots, e_M$ in $\bar{e}$, and vice versa.



Consistent



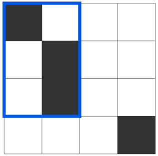
Inconsistent



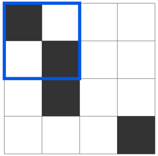
Phrase extraction

How do we get phrases?
We extract **all phrases** that are **consistent** with a word alignment A

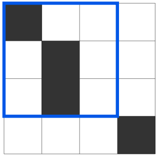
Definition: Consistent phrase pair
A phrase pair $(\bar{f}, \bar{e})$ is consistent with A, if all words $f_1, \ldots, f_N$ in $\bar{f}$ that have alignment points in A, have these with words $e_1, \ldots, e_M$ in $\bar{e}$, and vice versa.



Consistent



Inconsistent



Consistent

Phrase probabilities

- In the IBM models, there was a generative story about how all the English words turn into French words
- Here we do not choose among different phrase alignments
- We can choose to use many short phrases, or a few long ones, or anything in between
- We estimate the phrase translation probability $\phi(\bar{f}, \bar{e})$ by the relative frequency:

$$\phi(\bar{f}, \bar{e}) = \frac{\text{count}(\bar{e}, \bar{f})}{\sum_i \text{count}(\bar{e}, \bar{f}_i)}$$

Log-linear models

The phrase-based model so far already works well. So far we have:

- phrase translation probabilities
- reordering model d
- language model

Probabilities from each component are multiplied so that we can find best translation $\hat{e}$ with an **argmax**

We can put all of this in a general log-linear model:

$$p(x) = \exp \sum_{i=1}^{n} \lambda_i h_i(x)$$

which allows us to weight the components:

- $\lambda\phi$ for the translation model
- λd for the reordering model
- λLM for the language model

$$\begin{aligned}\hat{e} = \arg\max_e \quad & p_{\mathrm{LM}}(e)\, \lambda_{\mathrm{LM}} \\ & * \prod_i \phi(\bar{f}_i \mid \bar{e}_i)\, \lambda_\phi \\ & * d(\ldots)\, \lambda_d\end{aligned}$$

Log-linear models (2)

Since we have a log-linear model now, we can add all kinds of feature functions $h_i(x)$ together with a weight λ_i
Examples:

- Bi-directional translation probabilities
- Lexical weighting
- Word penalty (control output length)
- Phrase penalty

- Another improvement we can make is to obtain **lexicalized reordering probabilities**
- So far reordering is modelled just based on distance
- A popular way to do this is **MSD-reordering**: between 2 phrases, we want to predict:
  - (M) monotone order
  - (S) swap with previous phrase
  - (D) discontinuous

Decoding

- To find the best translation using our model, we need to perform **decoding**
- The search space is **huge**, so many **heuristics** are used in practice
- We can expand a translation hypothesis from **left-to-right**, one phrase at a time
- Every time we check the translation model, reordering model, and language model if this is a good idea
- We cannot keep all hypotheses in memory, so we put them in hypothesis stacks based on how many foreign words they cover
- When a stack gets too large, we prune it

Evaluation

Evaluation – How good are our translations?

```
Candidate: the the the the the the the
Ref 1: the cat is on the mat
Ref 2: there is a cat on the mat
```

Idea 1: Precision

$$P = \frac{\text{\# words in candidate that are in ref}}{\text{\# words in candidate}} = \frac{7}{7}$$

Evaluation – How good are our translations?

```
Candidate: the the the the the the the
Ref 1: the cat is on the mat
Ref 2: there is a cat on the mat
```

Idea 1: Precision

$$P = \frac{\text{\# words in candidate that are in ref}}{\text{\# words in candidate}} = \frac{7}{7}$$

Idea 2: Modified Precision
Clip the number of matching words (e.g. 7 for 'the') to their max. count in a ref. (e.g. only 2)

$$P = \frac{2}{7}$$

Evaluation – How good are our translations?

```
Candidate: the the the the the the the
Ref 1: the cat is on the mat
Ref 2: there is a cat on the mat
```

Idea 1: Precision

$$P = \frac{\text{\# words in candidate that are in ref}}{\text{\# words in candidate}} = \frac{7}{7}$$

Idea 2: Modified Precision
Clip the number of matching words (e.g. 7 for 'the') to their max. count in a ref. (e.g. only 2)

$$P = \frac{2}{7}$$

What is the modified precision for this?

```
Candidate: the cat
Ref 1: the cat is on the mat
Ref 2: there is a cat on the mat
```

$$P = \frac{2}{2} = 1$$

Evaluation – How good are our translations?

```
Candidate: the the the the the the the
Ref 1: the cat is on the mat
Ref 2: there is a cat on the mat
```

Idea 1: Precision

$$P = \frac{\text{\# words in candidate that are in ref}}{\text{\# words in candidate}} = \frac{7}{7}$$

Idea 2: Modified Precision
Clip the number of matching words (e.g. 7 for 'the') to their max. count in a ref. (e.g. only 2)

$$P = \frac{2}{7}$$

What is the modified precision for this?

```
Candidate: the cat
Ref 1: the cat is on the mat
Ref 2: there is a cat on the mat
```

$$P = \frac{2}{2} = 1$$

Can we use recall?
No, because there are multiple references.

Evaluation – How good are our translations?

```
Candidate: the the the the the the the
Ref 1: the cat is on the mat
Ref 2: there is a cat on the mat
```

Idea 1: Precision

$$P = \frac{\text{\# words in candidate that are in ref}}{\text{\# words in candidate}} = \frac{7}{7}$$

Idea 2: Modified Precision
Clip the number of matching words (e.g. 7 for 'the') to their max. count in a ref. (e.g. only 2)

$$P = \frac{2}{7}$$

What is the modified precision for this?

```
Candidate: the cat
Ref 1: the cat is on the mat
Ref 2: there is a cat on the mat
```

$$P = \frac{2}{2} = 1$$

Can we use recall?
No, because there are multiple references.

Solution: Brevity penalty
We multiply the score with $e^{1-\frac{r}{c}}$ if the total length of the candidates is shorter.

Evaluation – How good are our translations?

```
Candidate: the the the the the the the
Ref 1: the cat is on the mat
Ref 2: there is a cat on the mat
```

Idea 1: Precision

$$P = \frac{\text{\# words in candidate that are in ref}}{\text{\# words in candidate}} = \frac{7}{7}$$

Idea 2: Modified Precision
Clip the number of matching words (e.g. 7 for 'the') to their max. count in a ref. (e.g. only 2)

$$P = \frac{2}{7}$$

What is the modified precision for this?

```
Candidate: the cat
Ref 1: the cat is on the mat
Ref 2: there is a cat on the mat
```

$$P = \frac{2}{2} = 1$$

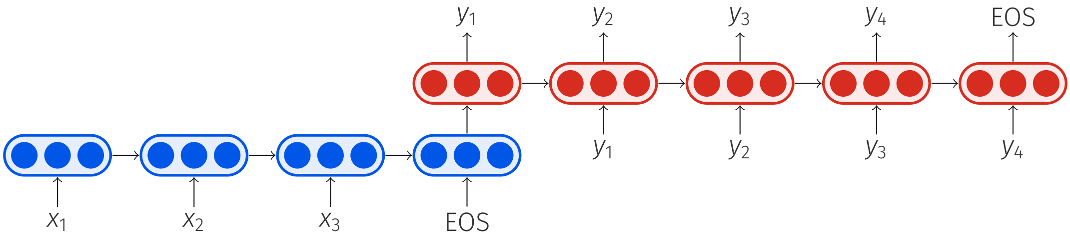
Can we use recall?
No, because there are multiple references.

Solution: Brevity penalty
We multiply the score with $e^{1-\frac{r}{c}}$ if the total length of the candidates is shorter.

BLEU
This is the basis for BLEU

Neural Machine Translation

Encoder-Decoder [Cho et al., 2014, Sutskever et al., 2014]



A blog post on how to implement an Encoder-Decoder *from scratch* in PyTorch:

`https://bastings.github.io/annotated_encoder_decoder/`

Google Translate Experiment

Try the following input:

```
iä
iä iä
iä iä iä
iä iä iä iä
iä iä iä iä iä
iä iä iä iä iä iä
iä iä iä iä iä iä iä
iä iä iä iä iä iä iä iä
iä iä iä iä iä iä iä iä iä
iä iä iä iä iä iä iä iä iä iä
iä iä iä iä iä iä iä iä iä iä iä
iä iä iä iä iä iä iä iä iä iä iä iä
etc..
```

What is going on here?

References i

Brown, P. F., Cocke, J., Pietra, S. A. D., Pietra, V. J. D., Jelinek, F., Lafferty, J. D., Mercer, R. L., and Roossin, P. S. (1990).
A statistical approach to machine translation.
Comput. Linguist., 16(2):79–85.

Brown, P. F., Pietra, V. J. D., Pietra, S. A. D., and Mercer, R. L. (1993).
The mathematics of statistical machine translation: Parameter estimation.
Computational linguistics, 19(2):263–311.

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014).
Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation.

In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002).
Bleu: a method for automatic evaluation of machine translation.
In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014).
Sequence to Sequence Learning with Neural Networks.
In *Neural Information Processing Systems (NIPS)*, pages 3104–3112.