6.891: Lecture 12 (October 20th, 2003)

Machine Translation Part III

Overview

- Recap: IBM Model 3
- IBM Model 4
- EM Training of Models 3 and 4
- Decoding

Recap: IBM Model 3

The generative process for $P(\mathbf{f}, \mathbf{a} \mid \mathbf{e})$:
(Example from Germann, NAACL 2003)

$\mathbf{e} =$	I	do	not	understand	the	logic	of	these	people
Pick fertilities	I		not not	understand	the	logic	of	these	people

$$P(\phi_1 \ldots \phi_l \mid \mathbf{e}) = \prod_{i=1}^{l} \mathbf{F}(\phi_i \mid e_i)$$

$$= \mathbf{F}(1 \mid I)\mathbf{F}(0 \mid do)\mathbf{F}(2 \mid not)\mathbf{F}(1 \mid understand)\mathbf{F}(1 \mid the) \times$$

$$\mathbf{F}(1 \mid logic)\mathbf{F}(1 \mid of)\mathbf{F}(1 \mid these)\mathbf{F}(1 \mid people)$$

Recap: IBM Model 3

The generative process for $P(\mathbf{f}, \mathbf{a} \mid \mathbf{e})$:
(Example from Germann, NAACL 2003)

$\mathbf{e} =$	I	do	not	understand	the	logic	of	these	people
Pick fertilities	I		not not	understand	the	logic	of	these	people
Replace words	Je		ne pas	comprends	la	logique	de	ces	gens

$$\prod_{i=1}^{l} \phi_i! \prod_{k=1}^{\phi_i} \mathbf{T}(\mathbf{f}_{i,k} \mid e_i) \quad = \quad 1! \times 0! \times 2! \times 1! \times 1! \times 1! \times 1! \times 1! \times 1! \times$$

$$\mathbf{T}(Je \mid I)\mathbf{T}(ne \mid not)\mathbf{T}(pas \mid not) \times$$

$$\mathbf{T}(comprends \mid understand)\mathbf{T}(la \mid the)\mathbf{T}(logique \mid logic) \times$$

$$\mathbf{T}(de \mid of)\mathbf{T}(ces \mid these)\mathbf{T}(gens \mid people)$$

Recap: IBM Model 3

The generative process for $P(\mathbf{f}, \mathbf{a} \mid \mathbf{e})$:
(Example from Germann, NAACL 2003)

$\mathbf{e} =$	I	do	not	understand	the	logic	of	these	people
Pick fertilities	I		not not	understand	the	logic	of	these	people
Replace words	Je		ne pas	comprends	la	logique	de	ces	gens

Reorder Je ne comprends pas la logique de ces gens

$$\prod_{i=1}^{l} \prod_{k=1}^{\phi_i} \mathbf{R}(\pi_{i,k} \mid i, l, m) = \mathbf{R}(j = 1 \mid i = 1, l = 9, m = 10)\mathbf{R}(2 \mid 3, 9, 10)\mathbf{R}(3 \mid 4, 9, 10) \times$$

$$\mathbf{R}(4 \mid 3, 9, 10)\mathbf{R}(5 \mid 5, 9, 10)\mathbf{R}(6 \mid 6, 9, 10) \times$$

$$\mathbf{R}(7 \mid 7, 9, 10)\mathbf{R}(8 \mid 8, 9, 10)\mathbf{R}(9 \mid 9, 9, 10)$$

The generative process for $P(\mathbf{f}, \mathbf{a} \mid \mathbf{e})$:

(Example from Germann, NAACL 2003)

$\mathbf{e} =$	I	do	not	understand	the	logic	of	these	people
Pick fertilities	I		not not	understand	the	logic	of	these	people
Replace words	Je		ne pas	comprends	la	logique	de	ces	gens

Reorder Je ne comprends pas la logique de ces gens

Spurious words Je ne comprends pas la logique de ces gens -la

$$P(\phi_0 \mid \phi_1 \ldots \phi_l) \prod_{k=1}^{\phi_0} \mathbf{T}(\mathbf{f}_{0,k} \mid NULL) = \frac{n!}{(n-\phi_0)!\phi_0!} p_1^{\phi_0} (1-p_1)^{n-\phi_0} \prod_{k=1}^{\phi_0} \mathbf{T}(\mathbf{f}_{0,k} \mid NULL)$$

$$= \frac{9!}{8!1!} p_1 (1-p_1)^8 \mathbf{T}(-la \mid NULL)$$

$$= 9 p_1 (1-p_1)^8 \mathbf{T}(-la \mid NULL)$$

Note: here $n = \sum_{i=1}^{l} \phi_l = m - \phi_0$

IBM Model 3: Summary

- Model 3 has the following parameter types

$\mathbf{T}(f \mid e)$	translation parameters
$\mathbf{R}(j \mid i, l, m)$	(reverse) alignment parameters
$\mathbf{F}(\phi \mid e)$	fertility parameters
p_1	parameter underlying ϕ_0

- Different stages in the generative model:

Stage	Description	Parameters
1	Pick fertilities $\phi_1 \ldots \phi_l$	$\mathbf{F}(\phi \mid e)$
2	Replace words	$\mathbf{T}(f \mid e)$
3	Reorder words	$\mathbf{R}(j \mid i, l, m)$
4	Spurious words	p_1 and $\mathbf{T}(f \mid NULL)$

- Can evaluate $P(\mathbf{f}, \mathbf{a} \mid \mathbf{e})$
  (**f** = French sentence of length m, $\mathbf{e}$ = English sentence of length l, $\mathbf{a}$ is an alignment) as

$$\frac{n!}{(n-\phi_0)!\phi_0!} p_1^{\phi_0} (1-p_1)^{n-\phi_0} \left(\prod_{i=1}^{l} \mathbf{F}(\phi_i \mid e_i) \phi_i! \right) \left(\prod_{i=1}^{l} \prod_{k=1}^{\phi_i} \mathbf{R}(\pi_{i,k} \mid i, l, m) \right) \left(\prod_{i=0}^{l} \prod_{k=1}^{\phi_i} \mathbf{T}(f_{i,k} \mid e_i) \right)$$

where $n = \sum_{i=1}^{l} \phi_i = m - \phi_0$ (error in last lecture's notes: I had $n = m$)

Overview

- Recap: IBM Model 3
- IBM Model 4
  - **Only difference from Model 3: different model of reordering stage**
- EM Training of Models 3 and 4
- Decoding

Recap: IBM Model 3

The generative process for $P(\mathbf{f}, \mathbf{a} \mid \mathbf{e})$:
(Example from Germann, NAACL 2003)

$\mathbf{e} =$	I	do	not	understand	the	logic	of	these	people
Pick fertilities	I		not not	understand	the	logic	of	these	people
Replace words	Je		ne pas	comprends	la	logique	de	ces	gens
Reorder	Je ne comprends pas la logique de ces gens								

$$\prod_{i=1}^{l}\prod_{k=1}^{\phi_i}\mathbf{R}(\pi_{i,k} \mid i, l, m) = \mathbf{R}(j = 1 \mid i = 1, l = 9, m = 10)\mathbf{R}(2 \mid 3, 9, 10)\mathbf{R}(3 \mid 4, 9, 10) \times$$

$$\mathbf{R}(4 \mid 3, 9, 10)\mathbf{R}(5 \mid 5, 9, 10)\mathbf{R}(6 \mid 6, 9, 10) \times$$

$$\mathbf{R}(7 \mid 7, 9, 10)\mathbf{R}(8 \mid 8, 9, 10)\mathbf{R}(9 \mid 9, 9, 10)$$

Another Example

e =	And	the	program	has	not	been	implemented
Pick fertilities		the	program	has	not not	been	implt'd implt'd implt'd
Replace words		le	programme	a	n' pas	ete	mis en application

Reorder le program a n'ete pas mis en application

(Note: I doubt this is correct French!)

The English words with fertility > 0

$\mathbf{e} =$	And	the_1	program_2	has_3	not_4	been_5	implemented_6
Pick fertilities		the	program	has	not not	been	implt'd implt'd implt'd
Replace words		le	programme	a	n' pas	ete	mis en application
Reorder	le program a n'ete pas mis en application						

The Heads in the French String

$e =$	And	the_1	$program_2$	has_3	not_4	$been_5$	$implemented_6$
Pick fertilities		the	program	has	not not	been	implt'd implt'd implt'd
Replace words		le	programme	a	n' pas	ete	mis en application

Reorder le_1 $program_2$ a_3 n'_4 ete_5 pas mis_6 en application

The head of each English word is the first French word aligned to it

The Centers in the French String

$\mathbf{e} =$	And	the_1	program_2	has_3	not_4	been_5	implemented_6
Pick fertilities		the	program	has	not not	been	implt'd implt'd implt'd
Replace words		le	programme	a	n' pas	ete	mis en application
Reorder	le program a n' ete pas mis en application						

The "center" of each English word is the ceiling of the average position of its translations in the French string

$$
\begin{aligned}
center(the) &= 1 \\
center(program) &= 2 \\
center(has) &= 3 \\
center(not) &= 5 \\
center(been) &= 5 \\
center(implemented) &= 8
\end{aligned}
$$

First Type of Alignment Parameter

$\mathbf{e} =$	And	the_1	program_2	has_3	not_4	been_5	implemented_6
Pick fertilities		the	program	has	not not	been	implt'd implt'd implt'd
Replace words		le	programme	a	n' pas	ete	mis en application

Reorder le_1 program_2 a_3 n'_4 ete_5 pas mis_6 en application

$\mathbf{R_1}(d \mid e, f)$ = probability of placing a head d positions to the right of the ceiling of the previous phrase, given that f is the French head word, and e is the English word for the previous phrase

$$\mathbf{R_1}(1 \mid NULL, le)$$
$$\mathbf{R_1}(1 \mid the, programme)$$
$$\mathbf{R_1}(1 \mid program, a)$$
$$\mathbf{R_1}(1 \mid has, n')$$
$$\mathbf{R_1}(1 \mid not, ete)$$
$$\mathbf{R_1}(2 \mid been, mis)$$

Second Type of Alignment Parameter

$\mathbf{e} =$	And	the$_1$	program$_2$	has$_3$	not$_4$	been$_5$	implemented$_6$
Pick fertilities		the	program	has	not not	been	implt'd implt'd implt'd
Replace words		le	programme	a	n' pas	ete	mis en application

Reorder le program a n' ete pas mis en application

$\mathbf{R}_{>1}(d \mid f)$ = probability of placing a non-head d positions to the right of the head of the phrase, given that f is the word being placed

$$\mathbf{R}_{>1}(2 \mid pas)$$
$$\mathbf{R}_{>1}(1 \mid en)$$
$$\mathbf{R}_{>1}(2 \mid application)$$

A Final Twist: Word Classes

$\mathcal{C}(w)$ is a function that maps each word w to one of ≈ 50 word classes

$$\mathbf{R_1}(1 \mid \mathcal{C}(NULL), \mathcal{C}(le))$$
$$\mathbf{R_1}(1 \mid \mathcal{C}(the), \mathcal{C}(programme))$$
$$\mathbf{R_1}(1 \mid \mathcal{C}(program), \mathcal{C}(a))$$
$$\mathbf{R_1}(1 \mid \mathcal{C}(has), \mathcal{C}(n'))$$
$$\mathbf{R_{>1}}(2 \mid \mathcal{C}(pas))$$
$$\mathbf{R_1}(1 \mid \mathcal{C}(not), \mathcal{C}(ete))$$
$$\mathbf{R_1}(2 \mid \mathcal{C}(been), \mathcal{C}(mis))$$
$$\mathbf{R_{>1}}(1 \mid \mathcal{C}(en))$$
$$\mathbf{R_{>1}}(2 \mid \mathcal{C}(application))$$

Model 4: Summary

- In reordering stage, place French phrases corresponding to English words in left-to-right order

- For each phrase, first pick position of the **head** of the phrase in relation to the **ceiling** of the previous phrase

$$\mathbf{R_1}(d \mid \mathcal{C}(e), \mathcal{C}(f))$$

where $d =$ position(head) - position(previous ceiling), and
e = English word for previous phrase, f = word being placed

- Next fill in remaining words' relative position to the head of the phrase

$$\mathbf{R}_{>\mathbf{1}}(d \mid \mathcal{C}(f))$$

where $d =$ position(non-head) - position(head), and
f = word being placed

Overview

- Recap: IBM Model 3
- IBM Model 4
- EM Training of Models 3 and 4
- Decoding

Recap

- We have various parameter types: $\mathbf{F}, \mathbf{T}, \mathbf{R}$ etc.

- For any (French, English, alignment) triple $(\mathbf{f}, \mathbf{e}, \mathbf{a})$ we can calculate

  $$P(\mathbf{f}, \mathbf{a} \mid \mathbf{e})$$

  as a function of the parameters

- Given a **training set** of $\mathbf{f}_k, \mathbf{e}_k$ pairs, our goal is to find parameters that maximize

  $$\sum_k \log P(f_k \mid e_k) = \sum_k \log \sum_{\mathbf{a} \in \mathcal{A}} P(f_k, \mathbf{a} \mid e_k)$$

  where $\mathcal{A}$ is the set of all possible alignments

EM, and Expected Counts

In training data, we have English/French sentence pairs:

$$\mathbf{e} = \text{I do not understand the logic of these people}$$
$$\mathbf{f} = \text{Je ne comprends pas la logique de ces gens -la}$$

- For EM training, we need to calculate **expected counts**
- E.g., given the current parameter values, what is the expected value for ϕ_3 (the fertility for "not") in this sentence?

$$\sum_{\mathbf{a} \in \mathcal{A}} P(\mathbf{a} \mid \mathbf{f}, \mathbf{e}) \phi_3(\mathbf{a}) = \sum_{\mathbf{a} \in \mathcal{A}} \frac{P(\mathbf{f}, \mathbf{a} \mid \mathbf{e})}{\sum_{\mathbf{a}' \in \mathcal{A}} P(\mathbf{f}, \mathbf{a}' \mid \mathbf{e})} \phi_3(\mathbf{a})$$

where $\phi_3(\mathbf{a})$ is the value for ϕ_3 in alignment $\mathbf{a}$

How do we do this for Models 3 and 4?

- Models 1 and 2 allowed efficient calculation of expected counts, in spite of exponential number of possible alignments

- Models 3 and 4 do not allow efficient exact calculations

- **An approximation**: use some heuristic to find a subset of **high probability alignments** $\bar{\mathcal{A}}$, then use brute-force to calculate

$$\sum_{\mathbf{a} \in \bar{\mathcal{A}}} \frac{P(\mathbf{f}, \mathbf{a} \mid \mathbf{e})}{\sum_{\mathbf{a}' \in \bar{\mathcal{A}}} P(\mathbf{f}, \mathbf{a}' \mid \mathbf{e})} \phi_3(\mathbf{a})$$

  We can afford to do this if $\bar{\mathcal{A}}$ is relatively small

Step 1: Calculate Most Likely Alignment under Model 2

- It's simple to calculate the single most likely alignment under model 2,

$$\mathbf{a}^{*,2} = \mathrm{argmax}_{\mathbf{a} \in \mathcal{A}} P_2(\mathbf{f}, \mathbf{a} \mid \mathbf{e})$$

  where $P_2(\mathbf{f}, \mathbf{a} \mid \mathbf{e})$ is defined by model 2

- Simply take

$$a_j^{*,2} = \mathrm{argmax}_j \left(\mathbf{T}(f_j \mid e_{a_j}) \mathbf{D}(j \mid i, l, m) \right)$$

Neighbourhoods of an Alignment

- Define the set of **neighbours** of $\mathbf{a}$

$$\mathcal{N}(\mathbf{a})$$

- An alignment $\mathbf{a}'$ is in $\mathcal{N}(\mathbf{a})$ if:
  - $\mathbf{a}' = \mathbf{a}$
  - $\mathbf{a}'$ can be constructed from $\mathbf{a}$ by changing one alignment variable $\mathbf{a}_j$
  - $\mathbf{a}'$ can be constructed from $\mathbf{a}$ by **swapping** the value for two alignment variables $\mathbf{a}_{j1}$ and $\mathbf{a}_{j2}$

Search for the Most Likely Alignment Under Models 3 or 4

- Say $\mathbf{a}^{*,2}$ is most likely alignment under Model 2

- $P_3(\mathbf{a}, \mathbf{f} \mid \mathbf{e})$ is probability under Model 3

- Calculate $\mathbf{a}^{*,3}$ as follows:

  Initialize: $\mathbf{a}^{*,3} = \mathbf{a}^{*,2}$

  Iterate until convergence:

$$\mathbf{a}^{*,3} = \mathrm{argmax}_{\mathbf{a} \in \mathcal{N}(\mathbf{a}^{*,3})} \quad P_3(\mathbf{a}, \mathbf{f} \mid \mathbf{e})$$

- Notes:
  - **Not** guaranteed to find highest prob. alignment under Model 3, i.e., $\mathrm{argmax}_{\mathbf{a} \in \mathcal{A}} \quad P_3(\mathbf{a}, \mathbf{f} \mid \mathbf{e})$
  - Same procedure used for model 4 to calculate $\mathbf{a}^{*,4}$

Search for the Most Likely Alignment Under Models 3 or 4

- Can use similar techniques to find

  $$\mathbf{a}^{*,3}_{i \to j}$$

  for all $i \in 1 \ldots l$ and $j \in 1 \ldots m$

- Here, $\mathbf{a}^{*,3}_{i \to j}$ is a high scoring alignment with constraint that $\mathbf{a}^{*,3}_{j} = i$.
  I.e., English word e_i must be linked to French word f_j

- For example, $\mathbf{a}^{*,3}_{1 \to 4}$ is an approximation of highest scoring alignment under Model 3 such that $a_4 = 1$.

Defining a Set of High Probability Alignments

- Define the set of high prob. alignments, $\bar{\mathcal{A}}$, as

$$\bar{\mathcal{A}} = \left\{ \mathbf{a} \;:\; \mathbf{a} \in \mathcal{N}(\mathbf{a}^{*,3}) \text{ or } \mathbf{a} \in \mathcal{N}(\mathbf{a}^{*,3}_{i \rightarrow j}) \text{ for some } i, j \right\}$$

- We can then calculate expected counts, for example

$$\sum_{\mathbf{a} \in \bar{\mathcal{A}}} \frac{P(\mathbf{f}, \mathbf{a} \mid \mathbf{e})}{\sum_{\mathbf{a}' \in \bar{\mathcal{A}}} P(\mathbf{f}, \mathbf{a}' \mid \mathbf{e})} \phi_3(\mathbf{a})$$

Overview

- Recap: IBM Model 3
- IBM Model 4
- EM Training of Models 3 and 4
- Decoding

Decoding

- Problem: for a given French sentence $\mathbf{f}$, find

$$\mathrm{argmax}_{\mathbf{e}} P(\mathbf{e}) P(\mathbf{f} \mid \mathbf{e})$$

or the "Viterbi approaximation"

$$\mathrm{argmax}_{\mathbf{e},\mathbf{a}} P(\mathbf{e}) P(\mathbf{f}, \mathbf{a} \mid \mathbf{e})$$

Decoding

- Decoding is NP-complete (see (Knight, 1999))
- IBM papers describe a *stack-decoding* or A^* *search* method
- A recent paper on decoding:

  Fast Decoding and Optimal Decoding for Machine Translation.
  Germann, Jahr, Knight, Marcu, Yamada. In ACL 2001.
- Introduces a *greedy* search method
- Compares the two methods to exact (integer-programming) solution

First Stage of the Greedy Method

- For each French word f_j, pick the English word e which maximizes

$$\mathbf{T}(e \mid f_j)$$

(an inverse translation table $\mathbf{T}(e \mid f)$ is required for this step)

- This gives us an initial alignment, e.g.,

Bien	intendu	,	il	parle	de	une	belle	victoire
Well	heard	,	it	talking	NULL	a	beautiful	victory

(Correct translation: *quite naturally, he talks about a great victory*)

Next Stage: Greedy Search

- First stage gives us an initial $(\mathbf{e}^0, \mathbf{a}^0)$ pair

- Basic idea: define a set of local transformations that map an $(\mathbf{e}, \mathbf{a})$ pair to a new $(\mathbf{e}', \mathbf{a}')$ pair

- Say $\Pi(\mathbf{e}, \mathbf{a})$ is the set of all $(\mathbf{e}', \mathbf{a}')$ reachable from $(\mathbf{e}, \mathbf{a})$ by some transformation, then at each iteration take

$$(\mathbf{e}^t, \mathbf{a}^t) = \text{argmax}_{(\mathbf{e},\mathbf{a}) \in \Pi(\mathbf{e}^{t-1}, \mathbf{a}^{t-1})} P(\mathbf{e}) P(\mathbf{f}, \mathbf{a} \mid \mathbf{e})$$

  i.e., take the highest probability output from results of all transformations

- Basic idea: iterate this process until convergence

The Space of Transforms

- CHANGE(j, e):
  Changes translation of f_j from e_{a_j} into e

- Two possible cases (take $e_{old} = e_{a_j}$):
  - Fertility of e_{old} is greater than 1, or $e_{old} = NULL$
    Place e at position in string that maximizes the alignment probability
  - Fertility of e_{old} is 1
    In this case, simply replace e_{old} with e

- Typically consider only (e, f) pairs such that e is in top 10 ranked translations for f under $\mathbf{T}(e \mid f)$
  (an inverse table of probabilities $\mathbf{T}(e \mid f)$ is required – this is described in Germann 2003)

The Space of Transforms

- CHANGE2$(j1, e1, j2, e2)$:
  Changes translation of f_{j1} from $e_{a_{j1}}$ into $e1$,
  and changes translation of f_{j2} from $e_{a_{j2}}$ into $e2$

- Just like performing CHANGE$(j1, e1)$ and CHANGE$(j2, e2)$ in sequence

The Space of Transforms

- TranslateAndInsert$(j, e1, e2)$:
  Implements CHANGE$(j, e1)$,
  (i.e. Changes translation of f_j from e_{a_j} into $e1$)
  and inserts e_2 at most likely point in the string

- Typically, e_2 is chosen from the 1024 words with highest probability of having fertility 0

The Space of Transforms

- RemoveFertilityZero(i):
  Removes e_i, providing that e_i has fertility 0 in the alignment

The Space of Transforms

- SwapSegments$(i1, i2, j1, j2)$:
  Swaps words $e_{i1} \ldots e_{i2}$ with words e_{j1} and e_{j2}

- Note: the two segments cannot overlap

The Space of Transforms

- JoinWords$(i1, i2)$:
  Deletes English word at position $i1$, and links all French words that were linked to e_{i1} to e_{i2}

An Example from Germann et. al 2001

Bien	intendu	,	il	parle	de	une	belle	victoire
Well	heard	,	it	talking	NULL	a	beautiful	victory

$$\Downarrow$$

Bien	intendu	,	il	parle	de	une	belle	victoire
Well	heard	,	it	talks	NULL	a	great	victory

$\text{CHANGE2}(5, talks, 8, great)$

An Example from Germann et. al 2001

Bien	intendu	,	il	parle	de	une	belle	victoire
Well	heard	,	it	talks	NULL	a	great	victory

$\Downarrow$

Bien	intendu	,	il	parle	de	une	belle	victoire
Well	understood	,	it	talks	about	a	great	victory

CHANGE2$(2, understood, 6, about)$

An Example from Germann et. al 2001

Bien	intendu	,	il	parle	de	une	belle	victoire
Well	understood	,	it	talks	about	a	great	victory

$\Downarrow$

Bien	intendu	,	il	parle	de	une	belle	victoire
Well	understood	,	he	talks	about	a	great	victory

CHANGE$(4, he)$

An Example from Germann et. al 2001

Bien	intendu	,	il	parle	de		une	belle	victoire
Well	understood	,	he	talks	about	a		great	victory

$\Downarrow$

Bien	intendu	,	il	parle	de		une	belle	victoire
quite	naturally	,	he	talks	about	a		great	victory

CHANGE2$(1, quite, 2, naturally)$

An Exact Method Based on Integer Programming

Method from Germann et. al 2001:

- Integer programming problems

$$3.2x_1 + 4.7x_2 - 2.1x_3$$ Minimize objective function

$$\begin{aligned} x_1 - 2.6x_3 &> 5 \\ 7.3x_2 &> 7 \end{aligned}$$ Subject to linear constraints

- Generalization of travelling salesman problem:
  Each town has a number of hotels; some hotels can be in multiple towns. Find the lowest cost tour of hotels such that each town is visited exactly once.

- In the MT problem:
  - Each city is a French word (all cities visited $\Rightarrow$ all French words must be accounted for)
  - Each hotel is an English word matched with one or more French words
  - The "cost" of moving from hotel i to hotel j is a sum of a number of terms. E.g., the cost of choosing "not" after "what", and aligning it with "ne" and "pas" is

$$\begin{aligned}
&\log(bigram(not \mid what) + \\
&\log(\mathbf{F}(2 \mid not) + \\
&\log(\mathbf{T}(ne \mid not) + \log(\mathbf{T}(pas \mid not)) \\
&\ldots
\end{aligned}$$

An Exact Method Based on Integer Programming

- Say distance between hotels i and j is d_{ij};
  Introduce x_{ij} variables where $x_{ij} = 1$ if path from hotel i to hotel j is taken, zero otherwise

- Objective function: maximize

$$\sum_{i,j} x_{ij} d_{ij}$$

- All cities must be visited once $\Rightarrow$ constraints

$$\forall \text{c} \in \text{cities} \quad \sum_{\text{i located in c}} \sum_{j} x_{ij} = 1$$

- Every hotel must have equal number of incoming and outgoing edges $\Rightarrow$

$$\forall i \sum_j x_{ij} = \sum_j x_{ji}$$

- Another constraint is added to ensure that the tour is fully connected