

CSE 842

Natural Language Processing

Lecture 21: Machine Translation

4/15/2009

CSE842, Spring 2009, MSU

1

Project Presentation Schedule

- April 27
 - Amaresh (13)
 - Yuanliang (13)
 - Tianbao and Jingshu (18)
 - Troy (13)
 - Sherine (13)
 - Mark (13)
- April 29
 - Matt Gerber and Matt Geimer (18)
 - Farhara and Jared (18)
 - Rui (13)
 - Kayra and Shawn (18)
 - Nagashri and Shweta (18)

4/15/2009

CSE842, Spring 2009, MSU

2

Topics to be covered

- Challenges in machine translation
- Evaluation of MT systems
- A brief introduction to statistical Machine Translation
 - Phrase-based Translation Model
- Alignment in MT:
 - IBM Model 1 and Model 2
- Decoding

4/15/2009

CSE842, Spring 2009, MSU

3

Why MT is difficult

- Lexical Ambiguity: multiple word senses
 - Book the flight
 - Read the book
- Different word orders
 - English word order is: subject –verb-object
 - Japanese word order is: subject-object-verb

English: IBM bought Lotus

Japanese: IBM Lotus bought

4/15/2009

CSE842, Spring 2009, MSU

4

Why MT is difficult

- Syntactic Structure is not Preserved Across Translation

The bottle floated into the cave



La botella entro a la cuerva flotando
(the bottle entered the cave floating)

Why MT is difficult

- Syntactic Ambiguity causes problems

John hit the dog with the stick



John golpeo el perro con el palo/que tenia el palo

Why MT is difficult

- Pronoun Resolution

The computer outputs the data, it is fast



La computadora imprime los datos; es rapida

The computer outputs the data; it is stored in ascii



La computador imprime los datos; **estan** almacenados en ascii

Evaluation of MT System

- Method 1: human evaluations, accurate, but expensive, slow
- “Cheap” and fast evaluation is essential
- BLEU metrics

Bleu MT Evaluation

- Candidate 1: It is a guide to action which ensures that the military always obeys the commands of the party
- Candidate 2: it is to insure the troops forever hearing the activity guidebook that party direct.

- Reference 1: It is a guide to action that ensures that the military will forever heed Party commands
- Reference 2: it is the guiding principle which guarantees the military forces always being under the command of the Party
- Reference 3: It is the practical guide for the army always to heed the directions of the party

Which Translation is better? What is the criteria you used.

Unigram Precision

- Unigram Precision of a candidate translation:
C/N
 - C: the number of words in the candidate which are in at least one reference translation.
 - N: the number of words in the candidate
- e.g., Candidate 1:
Precision = 17/18
(only *obeys* is missing from all reference translations)

Any Problem?

Modified Unigram Precision

- Problem with unigram precision
 - Candidate: the the the the the the the
 - Reference 1: the cat sat on the mat
 - Reference 2: there is a cat on the mat

Precision = 7/7 = 1???

Modified Unigram Precision

- Clipping
 - $Count_{clip} = \min(count, Max_Ref_Count)$
 - Max_Ref_Count: the maximum number of times a word occurs in any single reference translation
 - A candidate word can only be correct Max_Ref_Count times.
- Modified Unigram Precision:

$$\frac{\sum_i Count_{clip}}{N}$$

Previous example: modified precision = ?

Modified N-gram Precision

- Can generalize modified unigram precision to other n-grams
- Modified n-gram precision for a test corpus:

$$P_n = \frac{\sum_{C \in \{Candidates\}} \sum_{ngram \in C} Count_{clip}(ngram)}{\sum_{C \in \{Candidates\}} \sum_{ngram \in C} Count(ngram)}$$

i.e., number of correct ngrams in the candidates (after clipping) divided by total number of ngrams in the candidates

(So the everything is fine now? Is this enough?)

Precision Alone Isn't Enough

- Candidate 1: of the
- Reference 1: It is a guide to action that ensures that the military will forever heed Party commands
- Reference 2: it is the guiding principle which guarantees the military forces always being under the command of *the* Party
- Reference 3: It is the practical guide for the army always to heed the directions *of the* party

$$Precision (unigram) = 1$$

$$Precision (bigram) = 1$$

But Recall isn't Useful in this Case

- Standard measure used in addition to precision is **recall**:
 - Recall = C/N
 - C: the number of ngrams in candidate that are also in the reference
 - N: the number of words in the references

Candidate 1: I always invariably perpetually do

Candidate 2: I always do

Reference 1: I always do

Reference 2: I invariably do

Reference 3: I perpetually do

Sentence Brevity Penalty

- A high-scoring candidate translation must now match the reference translations in length, in word choice, and in word order.
- Candidate translations longer than their references are already penalized by the modified n-gram precision measurement

Sentence Brevity Penalty

- **Step 1:** for each candidate, compute closest matching reference (in terms of length)

e.g., our candidate is length 12, references are length 12, 15, 17. Best match is of length 12.

- **Step 2:** Say l_i is the length of the i th candidate, r_i is length of best match for the i th candidate, then compute

$$brevity = \frac{\sum_i r_i}{\sum_i l_i}$$

Sentence Brevity Penalty

- Step 3: compute brevity penalty

$$BP = \begin{cases} 1 & \text{if } brevity < 1 \\ e^{1-brevity} & \text{if } brevity \geq 1 \end{cases}$$

e.g., if $r_i = 1.1 \times l_i$ for all i (candidates are always 10% too short)
Then $BP = e^{-0.1} = 0.905$

Final BLEU Score

$$Bleu = BP \times (p_1 p_2 p_3 p_4)^{1/4}$$

i.e., BP multiplied by the geometric mean of the unigram, bigram, trigram, and four-gram precisions

A Brief Introduction to Statistical MT

- Parallel corpora are available in several language pairs
- Basic idea: use a parallel corpus as a training set of translation examples
- Classic example: IBM work on French-English translation, using the Canadian Hansards. (1.7 million sentences of 30 words or less in length)

The Noisy Channel Model

- Goal: translation system from French to English

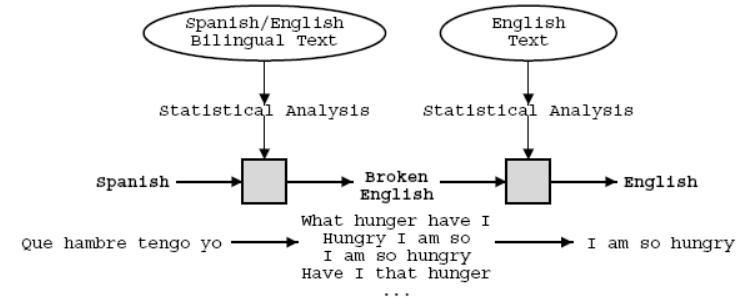
$$P(\mathbf{e}|\mathbf{f}) = \frac{P(\mathbf{e},\mathbf{f})}{P(\mathbf{f})} = \frac{P(\mathbf{e})P(\mathbf{f}|\mathbf{e})}{\sum_{\mathbf{e}'} P(\mathbf{e}')P(\mathbf{f}|\mathbf{e}')}$$

and

$$\arg \max_{\mathbf{e}} P(\mathbf{e}|\mathbf{f}) = \arg \max_{\mathbf{e}} P(\mathbf{e})P(\mathbf{f}|\mathbf{e})$$

- A Noisy Channel Model has two components:
 - $P(\mathbf{e})$ the language model: could be a trigram model, estimated from any data (parallel corpus not needed)
 - $P(\mathbf{f}|\mathbf{e})$ the translation model: trained from a parallel corpus of French/English pairs

From Koehn and Knight tutorial (2003)



From Koehn and Knight tutorial (2003)

- Translation from Spanish to English, candidate translations based on $P(\text{Spanish}|\text{English})$ alone:

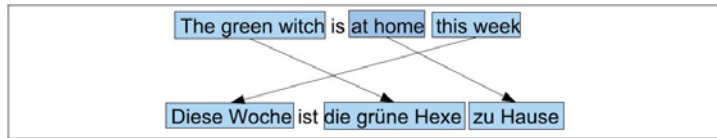
Que hambre tengo yo
 ↓
 What hunger have $P(S|E)=0.000014$
 Hungry I am so $P(S|E)=0.000001$
 I am so hungry $P(S|E) = 0.0000015$
 Have I that hunger $P(S|E) = 0.000020$

Example from Koehn and Knight tutorial

With $P(\text{Spanish}|\text{English}) * P(\text{English})$
 Que hambre tengo yo
 ↓
 What hunger have $P(S|E)P(E)=0.000014*0.000001$
 Hungry I am so $P(S|E)P(E)=0.000001*0.0000014$
 I am so hungry $P(S|E) P(E)= 0.0000015*0.0001$
 Have I that hunger $P(S|E) P(E)= 0.000020*0.00000098$

The Phrase-based Translation Model

- Modern SMT considers a better way to compute translation model $P(\mathbf{f}|\mathbf{e})$ is based on **phrases**.



- Generative Story:
 - Group English sentence \mathbf{e} into phrases, $\bar{e}_1, \bar{e}_2, \dots, \bar{e}_l$
 - Translate each English phrase \bar{e}_i into a French phrase \bar{f}_i
 - Optionally reorder each of the French phrases.

The Phrase-based Translation Model

$$P(\mathbf{f} | \mathbf{e}) = \prod_{i=1}^l \phi(\bar{f}_i, \bar{e}_i) d(a_i - b_{i-1})$$

$\phi(\bar{f}_i, \bar{e}_i)$: translation probability

$d(a_i - b_{i-1})$: distortion probability :

a_i : starting position of the foreign word (French) generated by \bar{e}_i

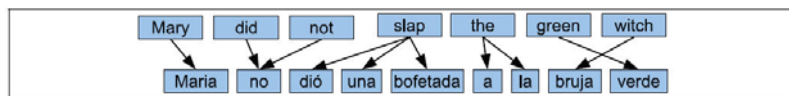
b_{i-1} : end position of the foreign word generated by \bar{e}_{i-1}

$$d(a_i - b_{i-1}) = \alpha^{|a_i - b_{i-1} - 1|}$$

Parameter estimation using parallel training data. But we don't have large, hand labeled phrase-aligned training sets.

We can extract phrases based on word alignment

Word Alignment



	Maria	no	dió	una	bofetada	a	la	bruja	verde
Mary	1								
did		1							
not			1						
slap				1	1	1			
the						1	1		
green								1	
witch									1

IBM Model 1: Alignment

- How do we model $P(\mathbf{f}|\mathbf{e})$
- English sentence \mathbf{e} has l words: $e_1 \dots e_l$,
French sentence \mathbf{f} has m words $f_1 \dots f_m$.
- An alignment \mathbf{a} identifies which English word each French word originated from
- Formally, an alignment \mathbf{a} is $\{a_1, \dots, a_m\}$, where each

$$a_j \in \{0..l\}$$
- There are $(l+1)^m$ possible alignments.

IBM Model 1: Alignment

- e.g., $l = 6, m = 7$
 \mathbf{e} = And the program has been implemented
 \mathbf{f} = Le programme a ete mis en application
- One alignment is
 $\{2, 3, 4, 5, 6, 6, 6, \}$

Another (bad) alignment is
 $\{1, 1, 1, 1, 1, 1, 1\}$

IBM Model 1: Alignment

- In IBM model 1, all alignment \mathbf{a} is equally likely

$$P(\mathbf{a} | \mathbf{e}) = C \times \frac{1}{(l+1)^m}, \quad \mathbf{a} \in A$$

where $C = \text{prob}(\text{length}(\mathbf{f}) = m)$ is a constant

- This is a **major** simplifying assumption, but it gets things start.

IBM Model 1: Translation Probabilities

- Next step: come up with an estimate for
 $P(\mathbf{f} | \mathbf{a}, \mathbf{e})$
- In model 1, this is:

$$P(\mathbf{f} | \mathbf{a}, \mathbf{e}) = \prod_{j=1}^m P(f_j | e_{a_j})$$

IBM Model 1: Translation Probabilities

e.g., $l = 6, m = 7$

\mathbf{e} = And the program has been implemented

\mathbf{f} = Le programme a ete mis en application

$\mathbf{a} = \{2,3,4,5,6,6,6\}$

$$\begin{aligned} P(\mathbf{f} | \mathbf{a}, \mathbf{e}) &= P(\text{Le} | \text{the}) \times P(\text{programme} | \text{program}) \times \\ &P(a | \text{has}) \times P(\text{ete} | \text{been}) \times \\ &P(\text{mis} | \text{implemented}) \times \\ &P(\text{en} | \text{implemented}) \times \\ &P(\text{application} | \text{implemented}) \end{aligned}$$

IBM Model 1: The Generative Process

To generate a French string \mathbf{f} from an English string \mathbf{e}

- Step 1: Pick the length of \mathbf{f} (all lengths equally probable, C)
- Step 2: Pick an alignment \mathbf{a} with probability $\frac{1}{(l+1)^m}$
- Step 3: Pick the French words with probability

$$P(\mathbf{f} | \mathbf{a}, \mathbf{e}) = \prod_{j=1}^m P(f_j | e_{a_j})$$

- The final result:

$$P(\mathbf{f}, \mathbf{a} | \mathbf{e}) = P(\mathbf{a} | \mathbf{e}) \times P(\mathbf{f} | \mathbf{a}, \mathbf{e}) = \frac{C}{(l+1)^m} \prod_{j=1}^m P(f_j | e_{a_j})$$

IBM Model 1:

- We have

$$P(\mathbf{f}, \mathbf{a} | \mathbf{e}) = \frac{C}{(l+1)^m} \prod_{j=1}^m P(f_j | e_{a_j})$$

- And:

$$P(\mathbf{f} | \mathbf{e}) = \sum_{\mathbf{a} \in A} \frac{C}{(l+1)^m} \prod_{j=1}^m P(f_j | e_{a_j})$$

Where A is the set of all possible alignments

IBM Model 2:

- Only difference: we now introduce alignment or distortion parameters

$D(i | j, l, m)$ = Probability that j 'th French word is connected to i 'th English word, given sentence lengths of \mathbf{e} and \mathbf{f} are l and m respectively

- Define $P(\mathbf{a} = \{a_1, \dots, a_m\} | \mathbf{e}, l, m) = \prod_{j=1}^m D(a_j | j, l, m)$

- Gives $P(\mathbf{f}, \mathbf{a} | \mathbf{e}, l, m) = \prod_{j=1}^m D(a_j | j, l, m) T(f_j | e_{a_j})$

Model 1 is a special case of Model 2, where

$$D(i | j, l, m) = \frac{1}{l+1} \text{ for all } i, j.$$

An Example

$$l = 6, m = 7$$

\mathbf{e} = And the Program has been implemented

\mathbf{f} = Le programme a ete mis en application

$$\mathbf{a} = \{2, 3, 4, 5, 6, 6, 6\} \quad P(\mathbf{a} | \mathbf{e}, l = 6, m = 7) = D(i = 2 | j = 1, l = 6, m = 7) \times$$

$$D(i = 3 | j = 2, l = 6, m = 7) \times$$

$$D(i = 4 | j = 3, l = 6, m = 7) \times$$

$$D(i = 5 | j = 4, l = 6, m = 7) \times$$

$$D(i = 6 | j = 5, l = 6, m = 7) \times$$

$$D(i = 6 | j = 6, l = 6, m = 7) \times$$

$$D(i = 6 | j = 7, l = 6, m = 7) \times$$

An Example

e.g., $l = 6, m = 7$

\mathbf{e} = And the program has been implemented

\mathbf{f} = Le programme a ete mis en application

$\mathbf{a} = \{2,3,4,5,6,6,6\}$

$$P(\mathbf{f} | \mathbf{a}, \mathbf{e}) = T(\text{Le} | \text{the}) \times T(\text{programme} | \text{program}) \times \\ T(a | \text{has}) \times T(\text{ete} | \text{been}) \times \\ T(\text{mis} | \text{implemented}) \times \\ T(\text{en} | \text{implemented}) \times \\ T(\text{application} | \text{implemented})$$

IBM Model 2: The Generative Process

To generate a French string \mathbf{f} from an English string \mathbf{e}

- Step 1: Pick the length of \mathbf{f} (all lengths equally probable, C)
- Step 2: Pick an alignment $\mathbf{a} = \{a_1, a_2, \dots, a_m\}$ with probability

$$\prod_{j=1}^m D(a_j | j, l, m)$$

- Step 3: Pick the French words with probability

$$P(\mathbf{f} | \mathbf{a}, \mathbf{e}) = \prod_{j=1}^m T(f_j | e_{a_j})$$

- The final result:

$$P(\mathbf{f}, \mathbf{a} | \mathbf{e}) = P(\mathbf{a} | \mathbf{e}) \times P(\mathbf{f} | \mathbf{a}, \mathbf{e}) = C \prod_{j=1}^m D(a_j | j, l, m) T(f_j | e_{a_j})$$

IBM Model 2:

- We have

$$P(\mathbf{f}, \mathbf{a} | \mathbf{e}) = C \prod_{j=1}^m D(a_j | j, l, m) T(f_j | e_{a_j})$$

- And:

$$P(\mathbf{f} | \mathbf{e}) = \sum_{\mathbf{a} \in A} C \prod_{j=1}^m D(a_j | j, l, m) T(f_j | e_{a_j})$$

Where A is the set of all possible alignments

A Hidden Variable Problem

- Training data is a set of $(\mathbf{f}_i, \mathbf{e}_i)$ pairs, the log likelihood of data is:

$$\sum_i \log P(\mathbf{f}_i | \mathbf{e}_i) = \sum_i \log \sum_{\mathbf{a} \in A} P(\mathbf{a} | \mathbf{e}_i) P(\mathbf{f}_i | \mathbf{a}, \mathbf{e}_i)$$

Where A is the set of all possible alignments

- We need to find model parameters (i.e., translation probabilities) to maximize the log likelihood function
- EM can be used for this problem: initialize translation probabilities randomly, and at each iteration choose

$$\Theta^t = \arg \max_{\Theta} \sum_i \sum_{\mathbf{a} \in A} P(\mathbf{a} | \mathbf{e}_i, \mathbf{f}_i, \Theta^{t-1}) \log(\mathbf{f}_i | \mathbf{a}, \mathbf{e}_i, \Theta)$$

where Θ^t are the parameter values at the t 'th iteration