CS388: Natural Language Processing
Lecture 17: Machine Translation I

Greg Durrett

Some slides adapted from Dan Klein, UC Berkeley

This Lecture

- MT and evaluation
- Word alignment
- Language models
- Phrase-based decoders
- Syntax-based decoders (probably next time)

MT Basics

MT Basics

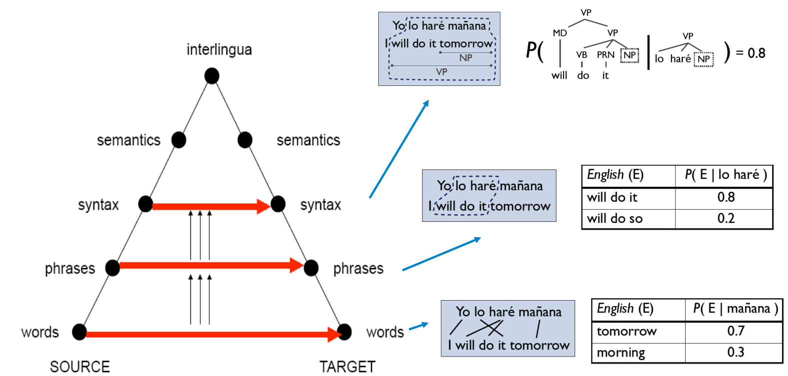
People's Daily, August 30, 2017

Trump Pope family watch a hundred years a year in the White House balcony

MT Ideally

- I have a friend => ∃x `friend(x,self)` => J'ai un ami
  J'ai une amie
  - May need information you didn't think about in your representation
  - Hard for semantic representations to cover everything
- Everyone has a friend => `∃x∀y friend(x,y)` / `∀x∃y friend(x,y)` => Tous a un ami
  - Can often get away without doing all disambiguation — same ambiguities may exist in both languages

Levels of Transfer: Vauquois Triangle



English (E)	P(E | lo haré)
will do it	0.8
will do so	0.2

English (E)	P(E | mañana)
tomorrow	0.7
morning	0.3

- Today: mostly phrase-based, some syntax

Slide credit: Dan Klein

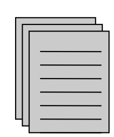
Phrase-Based MT

- Key idea: translation works better the bigger chunks you use
- Remember phrases from training data, translate piece-by-piece and stitch those pieces together to translate
  - How to identify phrases? Word alignment over source-target bitext
  - How to stitch together? Language model over target language
  - Decoder takes phrases and a language model and searches over possible translations
- NOT like standard discriminative models (take a bunch of translation pairs, learn a ton of parameters in an end-to-end way)

Phrase-Based MT

cat ||| chat ||| 0.9
the cat ||| le chat ||| 0.8
dog ||| chien ||| 0.8
house ||| maison ||| 0.6
my house ||| ma maison ||| 0.9
language ||| langue ||| 0.9
...

Phrase table P(f|e)



Unlabeled English data

Language model P(e)

$$P(e|f) \propto P(f|e)P(e)$$

Noisy channel model: combine scores from translation model + language model to translate foreign to English

"Translate faithfully but make fluent English"

Evaluating MT

- Fluency: does it sound good in the target language?
- Fidelity/adequacy: does it capture the meaning of the original?
- BLEU score: geometric mean of 1-, 2-, 3-, and 4-gram precision vs. a reference, multiplied by brevity penalty

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right).$$

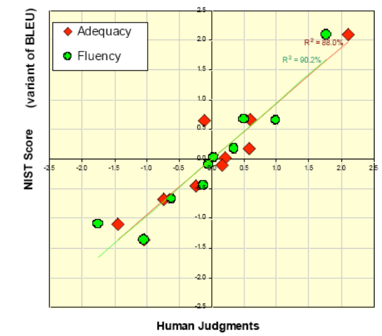
- Typically $n = 4$, $w_i = 1/4$

$$\text{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \le r \end{cases}.$$

- r = length of reference
  c = length of prediction
- Does this capture fluency and adequacy?

BLEU Score

- Better methods with human-in-the-loop
- HTER: human-assisted translation error rate
- If you're building real MT systems, you do user studies. In academia, you mostly use BLEU



slide from G. Doddington (NIST

Word Alignment

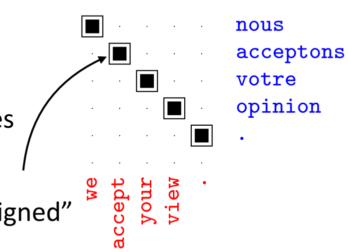
Word Alignment

- Input: a bitext, pairs of translated sentences

  nous acceptons votre opinion . ||| we accept your view

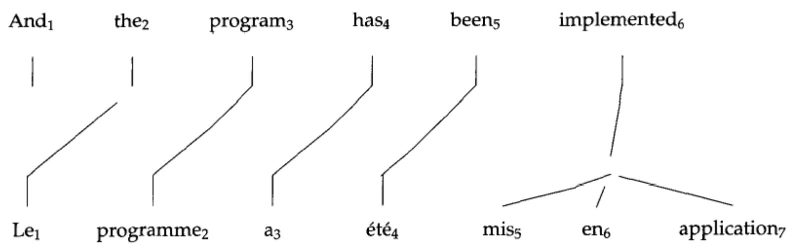
  nous allons changer d'avis ||| we are going to change our minds

- Output: alignments between words in each sentence
  - We will see how to turn these into phrases



"accept and acceptons are aligned"

1-to-Many Alignments



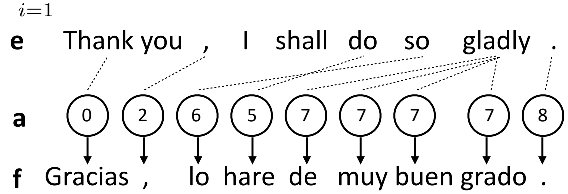
Word Alignment

- Models P(**f**|**e**): probability of "French" sentence being generated from "English" sentence according to a model
- Latent variable model: $P(\mathbf{f}|\mathbf{e}) = \sum_{\mathbf{a}} P(\mathbf{f}, \mathbf{a}|\mathbf{e}) = \sum_{\mathbf{a}} P(\mathbf{f}|\mathbf{a}, \mathbf{e})P(\mathbf{a})$
- Correct alignments should lead to higher-likelihood generations, so by optimizing this objective we will learn correct alignments

IBM Model 1

- Each French word is aligned to *at most* one English word

$$P(\mathbf{f}, \mathbf{a}|\mathbf{e}) = \prod_{i=1}^{n} P(f_i|e_{a_i})P(a_i)$$



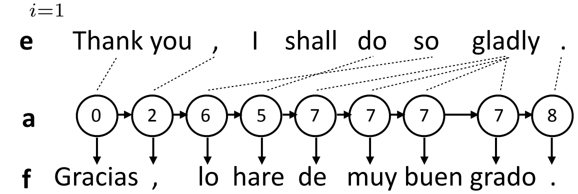
- Set P(a) uniformly (no prior over good alignments)
- $P(f_i|e_{a_i})$: word translation probability table

Brown et al. (1993)

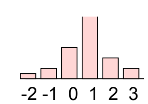
HMM for Alignment

- Sequential dependence between a's to capture monotonicity

$$P(\mathbf{f}, \mathbf{a}|\mathbf{e}) = \prod_{i=1}^{n} P(f_i|e_{a_i})P(a_i|a_{i-1})$$



- Alignment dist parameterized by jump size: $P(a_j - a_{j-1})$

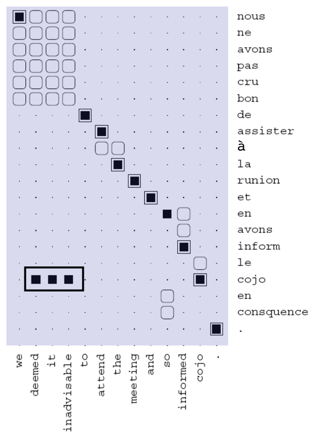


- $P(f_i|e_{a_i})$: same as before

Brown et al. (1993)

HMM Model

- Which direction is this?
- Alignments are generally monotonic (along diagonal)
- Some mistakes, especially when you have rare words (*garbage collection*)



Evaluating Word Alignment

- "Alignment error rate": use labeled alignments on small corpus

Model	AER
Model 1 INT	19.5
HMM E→F	11.4
HMM F→E	10.8
HMM AND	7.1
HMM INT	4.7
GIZA M4 AND	6.9

- Run Model 1 in both directions and intersect "intelligently"
- Run HMM model in both directions and intersect "intelligently"

Phrase Extraction

- Find contiguous sets of aligned words in the two languages that don't have alignments to other words

d'assister à la reunion et ||| to attend the meeting and

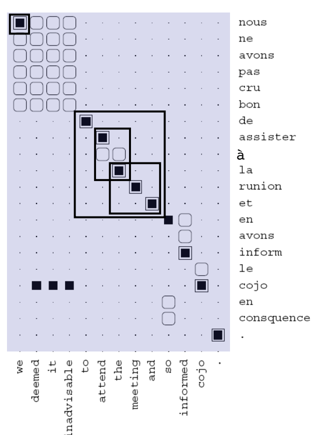
assister à la reunion ||| attend the meeting

la reunion and ||| the meeting and

nous ||| we

...

- Lots of phrases possible, count across all sentences and score by frequency

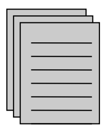


Language Modeling

Phrase-Based MT

```
cat ||| chat ||| 0.9
the cat ||| le chat ||| 0.8
dog ||| chien ||| 0.8
house ||| maison ||| 0.6
my house ||| ma maison ||| 0.9
language ||| langue ||| 0.9
...
```

Phrase table P(f|e)



Language model P(e)

Unlabeled English data

$$P(e|f) \propto P(f|e)P(e)$$

Noisy channel model: combine scores from translation model + language model to translate foreign to English

"Translate faithfully but make fluent English"

N-gram Language Models

I visited San _____ put a distribution over the next word

- Simple generative model: distribution of next word is a multinomial distribution conditioned on previous n-1 words

$$P(x|\text{visited San}) = \frac{\text{count}(\text{visited San}, x)}{\text{count}(\text{visited San})}$$

Maximum likelihood estimate of this probability from a corpus

- Just relies on counts, even in 2008 could scale up to 1.3M word types, 4B n-grams (all 5-grams occurring >40 times on the Web)

Smoothing N-gram Language Models

I visited San _____ put a distribution over the next word!

- Smoothing is very important, particularly when using 4+ gram models

$$P(x|\text{visited San}) = (1-\lambda)\frac{\text{count}(\text{visited San}, x)}{\text{count}(\text{visited San})} + \lambda\frac{\text{count}(\text{San}, x)}{\text{count}(\text{San})}$$

smooth this too!

- One technique is "absolute discounting:" subtract off constant *k* from numerator, set lambda to make this normalize (*k*=1 is like leave-one-out)

$$P(x|\text{visited San}) = \frac{\text{count}(\text{visited San}, x) - k}{\text{count}(\text{visited San})} + \lambda\frac{\text{count}(\text{San}, x)}{\text{count}(\text{San})}$$

- Kneser-Ney smoothing: this trick, plus low-order distributions modified to capture fertilities (how many distinct words appear in a context)

Engineering N-gram Models

- For 5+-gram models, need to store between 100M and 10B context-word-count triples

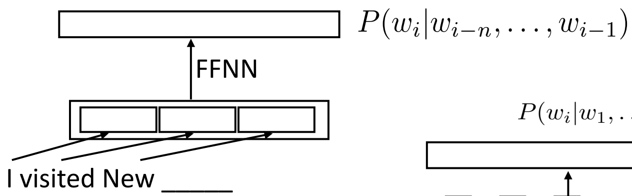
(a) Context-Encoding			(b) Context Deltas			(c) Bits Required		
w	c	val	Δw	Δc	val	$\|\Delta w\|$	$\|\Delta c\|$	$\|val\|$
1933	15176585	3	1933	15176585	3	24	40	3
1933	15176587	2	+0	+2	1	2	3	3
1933	15176593	1	+0	+5	1	2	3	3
1933	15176613	8	+0	+40	8	2	9	6
1933	15179801	1	+0	+188	1	2	12	3
1935	15176585	298	+2	15176585	298	4	36	15
1935	15176589	1	+0	+4	1	2	6	3

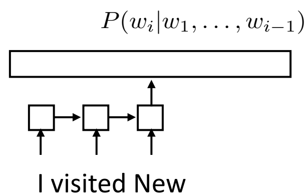
- Make it fit in memory by *delta encoding* scheme: store deltas instead of values and use variable-length encoding

Pauls and Klein (2011), Heafield (2011)

Neural Language Models

- Early work: feedforward neural networks looking at context





- Variable length context with RNNs:
  - Works like a decoder with no encoder
- Slow to train over lots of data!

Mnih and Hinton (2003)

Evaluation

- (One sentence) negative log likelihood: $\sum_{i=1}^{n} \log p(x_i|x_1, \ldots, x_{i-1})$
- Perplexity: $2^{-\frac{1}{n}\sum_{i=1}^{n} \log_2 p(x_i|x_1,\ldots,x_{i-1})}$
  - NLL (base 2) averaged over the sentence, exponentiated
  - NLL = -2 -> on average, correct thing has prob 1/4 -> PPL = 4. PPL is sort of like branching factor

Results

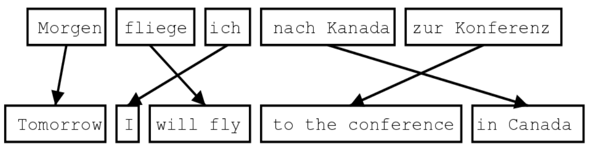
- Evaluate on Penn Treebank: small dataset (1M words) compared to what's used in MT, but common benchmark
- Kneser-Ney 5-gram model with cache: PPL = 125.7
- LSTM: PPL ~ 60-80 (depending on how much you optimize it)
- Melis et al.: many neural LM improvements from 2014-2017 are subsumed by just using the right regularization (right dropout settings). So LSTMs are pretty good

Merity et al. (2017), Melis et al. (2017)

Decoding

Phrase-Based Decoding

- Inputs:
  - Language model that scores $P(e_i|e_1,\ldots,e_{i-1}) \approx P(e_i|e_{i-n-1},\ldots,e_{i-1})$
  - Phrase table: set of phrase pairs (**e**, **f**) with probabilities P(**f**|**e**)
- What we want to find: **e** produced by a series of phrase-by-phrase translations from an input **f**, possibly with reordering:



Phrase lattices are big!

这	7人	中包括	来自	法国	和	俄罗斯	的	宇航	员	.
the	7 people	including	by some		**and**	the russian	**the**	the astronauts		,
it	7 people included		by france		and the	the russian		international astronautical	of rapporteur .	
this	7 out	including the	**from**	the french	and the russian		the fifth		.	
these	7 among	including from		the french and		of the russian	of	space	members	.
that	7 persons	including from the		of france	and to	russian	of the	aerospace	members .	
	7 include		from the	of france and		russian		**astronauts**		. the
	7 numbers include		**from france**		and russian		of astronauts who			. ”
	7 populations include		those from france		and russian			astronauts .		
	7 deportees included		come from	**france**	**and russia**		in	astronautical	personnel	;
	7 philtrum	including those from		**france and**		**russia**	a space		**member**	
		including representatives from		france and the		**russia**		astronaut		
		include	came from	**france and russia**			by cosmonauts			
		include representatives from		french	**and russia**			cosmonauts		
		include	came from france		and russia 's			cosmonauts .		
		includes	coming from	french and		russia 's		cosmonaut		
				french and russian			's	astronavigation	member .	
				french	**and russia**		**astronauts**			
					and russia 's				special rapporteur	
					, and	**russia**			rapporteur	
					, and russia				rapporteur .	
					, and russia					
					or	russia 's				

Slide credit: Dan Klein

Phrase-Based Decoding

Input	lo haré | rápidamente | .	*The decoder... tries different segmentations,*
Translations	I'll do it | quickly | .	*translates phrase by phrase,*
	quickly | I'll do it | .	*and considers reorderings.*

$$\arg\max_{\mathbf{e}} [P(\mathbf{f}|\mathbf{e}) \cdot P(\mathbf{e})]$$

- Decoding objective (for 3-gram LM)

$$\arg\max_{\mathbf{e}} \left[\prod_{\langle \bar{e},\bar{f}\rangle} P(\bar{f}|\bar{e}) \cdot \prod_{i=1}^{|\mathbf{e}|} P(e_i|e_{i-1},e_{i-2}) \right]$$

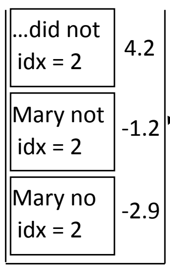
Slide credit: Dan Klein

Monotonic Translation

- If we translate with beam search, what state do we need to keep in the beam?
  - What have we translated so far?
  - What words have we produced so far?
  - When using a 3-gram LM, only need to remember the last 2 words!

$$\arg\max_{\mathbf{e}} \left[\prod_{\langle \bar{e},\bar{f}\rangle} P(\bar{f}|\bar{e}) \cdot \prod_{i=1}^{|\mathbf{e}|} P(e_i|e_{i-1},e_{i-2}) \right]$$

Monotonic Translation

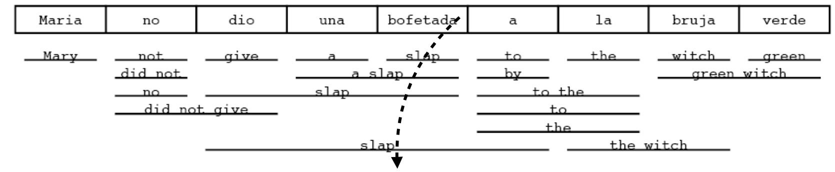


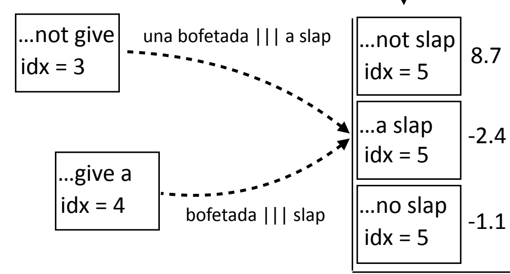
$$\text{score} = \log [\underbrace{P(\text{Mary})\ P(\text{not} \mid \text{Mary})}_{\text{LM}}\ \underbrace{P(\text{Mary} \mid \text{Maria})\ P(\text{not} \mid \text{no})}_{\text{TM}}]$$

In reality: score = α log P(LM) + β log P(TM)

...and TM is broken down into several features

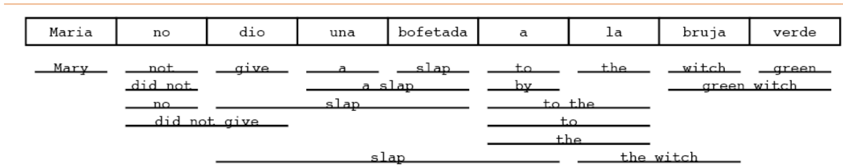
Monotonic Translation



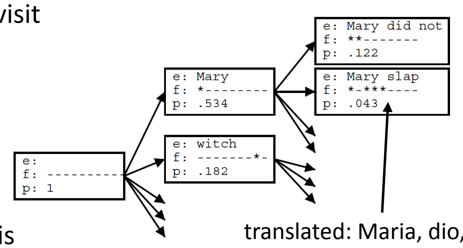


- Several paths can get us to this state, max over them (like Viterbi)
- Variable-length translation pieces = semi-HMM

Non-Monotonic Translation



- Non-monotonic translation: can visit source sentence "out of order"
- State needs to describe which words have been translated and which haven't
- Big enough phrases already capture lots of reorderings, so this isn't as important as you think



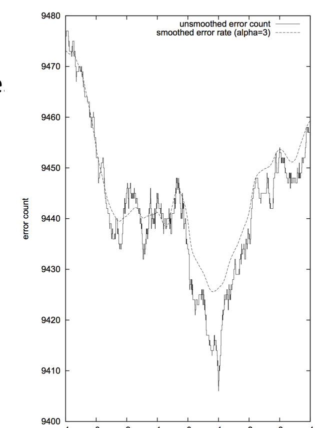
translated: Maria, dio, una, bofetada

Training Decoders

score = α log P(LM) + β log P(TM)

...and TM is broken down into several feature

- Usually 5-20 feature weights to set, want to optimize for BLEU score which is not differentiable
- MERT (Och 2003): decode to get 1000-best translations for each sentence in a small training set (<1000 sentences), do line search on parameters to directly optimize for BLEU



Moses

- Toolkit for machine translation due to Philipp Koehn + Hieu Hoang
  - Pharaoh (Koehn, 2004) is the decoder from Koehn's thesis
- Moses implements word alignment, language models, and this decoder, plus *a ton* more stuff
  - Highly optimized and heavily engineered, could more or less build SOTA translation systems with this from 2007-2013
- Next time: results on these and comparisons to neural methods

Syntax

Syntactic MT

- Rather than use phrases, use a *synchronous context-free grammar*

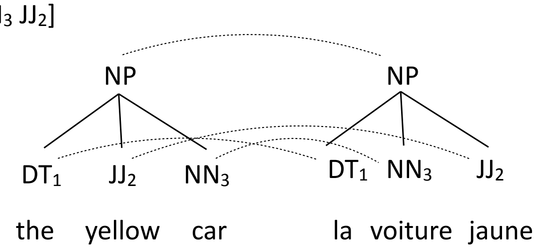
$NP \rightarrow [DT_1\ JJ_2\ NN_3;\ DT_1\ NN_3\ JJ_2]$

$DT \rightarrow$ [the, la]

$DT \rightarrow$ [the, le]

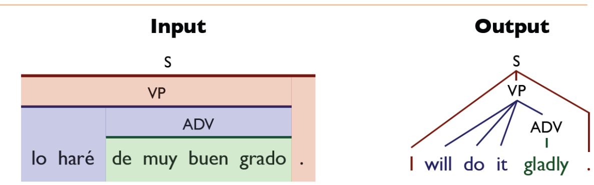
$NN \rightarrow$ [car, voiture]

$JJ \rightarrow$ [yellow, jaune]



- Translation = parse the input with "half" of the grammar, read off the other half
- Assumes parallel syntax up to reordering

Syntactic MT



- Use lexicalized rules, look like "syntactic phrases"
- Leads to HUGE grammars, parsing is slow

Grammar

S → ⟨ VP . ; I VP . ⟩ **OR** S → ⟨ VP . ; you VP . ⟩

VP → ⟨ lo haré ADV ; will do it ADV ⟩

S → ⟨ lo haré ADV . ; I will do it ADV . ⟩

ADV → ⟨ de muy buen grado ; gladly ⟩

Slide credit: Dan Klein

Takeaways

- Phrase-based systems consist of 3 pieces: aligner, language model, decoder
  - HMMs work well for alignment
  - N-gram language models are scalable and historically worked well
  - Decoder requires searching through a complex state space
- Lots of system variants incorporating syntax
- Next time: neural MT