Word Based Model

Example contd…

<table>
<tr><td rowspan="2">I will go home</td><td>我 会 回 家</td></tr>
<tr><td>1 2 3 4</td></tr>
<tr><td rowspan="2"></td><td></td></tr>
<tr><td></td></tr>
<tr><td rowspan="2"></td><td></td></tr>
<tr><td></td></tr>
</table>

Possible Translations

Token	**Tgt Idx**
I	
Will	
go	
home	

-Taken From Google translation English to Simplified Chinese

Example contd…

I will go home	我 会 回 家
	1 2 3 4
I will eat	我 会 吃
	5 6 7
Chen will eat	陈 会 吃
	8 9 10
I will eat Chen	

Possible Translations

Token	**Tgt Idx**
I	
Will	
go	
home	
eat	
Chen	

Example contd…

I will go home	我 会 回 家
	1 2 3 4
I will eat	我 会 吃
	5 6 7
Chen will eat	陈 会 吃
	8 9 10
I will eat Chen	我 会 吃 陈
	1 2 7 8

Possible Translations →

Token	**Tgt Idx**
I	1,5
Will	2,6,9
go	3 or 4
home	3 or 4
eat	7,10
Chen	8

Example contd…

- Intuition behind word based model.
- Motivation behind this: sentence too long for translation
- This generative model isn’t state of the arts. But ideas ⇒ PBSMT
- IBM Model stems from original work on SMT by IBM in late 80’s and early 90’s.

- Lexical Translation:
    - In bilingual dictionary (or [google translate](#)) for english hindi ,
        - Home - घर, मकान, निवास,जन्मभूमि, कुटुंब
    - Statistics : frequency of home and घर, मकान, निवास,जन्मभूमि, कुटुंब
    - Lexical translation probability: #(home,घर)/#(home) [MLE]
    - Alignment function
        - ***a***: j→ i (maps from target to source)
        - Ex.
            - This house is small
            - I will go home
        - Words can be dropped → NULL Token
    - Translation probability distribution for full sentence

IBM Model 1

- Terms
  - Translation probability: $P(e_j|f_i)$ [$f \Rightarrow e$]
  - Alignment function ***a***(j) → i [i^{th} word is aligned with j^{th}]
  - Calculate using Expectation Maximization Algorithm on sentence aligned parallel text.
- Problem Setting
  - Let $f=(f_1,f_2,\ldots f_{lf})$ with length l_f be source sentence and $e=(e_1,e_2,\ldots e_{le})$ be target sentence of length l_e.
  - Translation probability for target word e_j given source word f_i:
    - $p(\mathbf{e}|\mathbf{f}) = \frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})$
    - Fraction is normalizing fraction
      - $(l_f+1)^{le}$: possible alignments between these two sentences.
      - l_f+1: one is added for NULL token on target side.

- Incomplete data
  a. Sentence aligned, not word aligned (***a*** is not known)
  b. Chicken and egg
  c. EM
- Expectation Maximization Algorithm
  a. Initialize model parameters (e.g. uniform)
  b. Assign probabilities to the missing data
  c. Estimate model parameters from completed data
  d. Iterate steps b–c until convergence
- t(e|f) : lexical translation probability that f → e
- Count (e|f) : evidence (for every co occurrence in a pair add probability t(e|f)) that particular word f translates to e.
- S-total (e) : sum of probability that * → e [For normalization]

IBM Model 1 contd…

```
Input: set of sentence pairs (e,f)
Output: translation prob. t(e|f)
initialize t(e|f) uniformly
while not converged do
  // initialize
  count(e|f) = 0 for all e,f
  total(f) = 0 for all f
  for all sentence pairs (e,f) do
    // compute normalization
    for all words e in e do
      s-total(e) = 0
      for all words f in f do
        s-total(e) += t(e|f)
      end for
    end for
    // collect counts
    for all words e in e do
      for all words f in f do
        count(e|f) += t(e|f) / s-total(e)
        total(f) += t(e|f) / s-total(e)
      end for
    end for
  end for
  // estimate probabilities
  for all foreign words f do
    for all English words e do
      t(e|f) = count(e|f) / total(f)
    end for
  end for
end while
```

Example for IBM Model 1 EM

Image Taken from SMT by Philipp Koehn

Ipython Notebook

IBM Model 1 contd…

- Now given translation probability there is one more feature/factor aiding in choosing translation output.
- E.g. for say “small house” we can have following options (for wbm):
  - छोटा घर
  - लघु घर
  - लघु सदन
  - छोटा सदन
- Intuitively we would choose छोटा घर since it appear more fluent and common.
- This can be taken care by language model which looks at fluency on target side. And aid in choosing group of words in sequence (again generative model) which have higher probability. (similar to POS Tagging using HMM)

Noisy Channel Model

- Combining translation model and language model [fluent and faithfulness]
- Best translation **e** for input sentence **f**
    - Bayes rule
        - $\text{argmax}_{\mathbf{e}} p(\mathbf{e}|\mathbf{f}) = \text{argmax}_{\mathbf{e}} p(\mathbf{f}|\mathbf{e}) p(\mathbf{e}) / p(\mathbf{f})$
        - $\Rightarrow \text{argmax}_{\mathbf{e}} p(\mathbf{e}|\mathbf{f}) = \text{argmax}_{\mathbf{e}} p(\mathbf{f}|\mathbf{e}) p(\mathbf{e})$
    - Translation direction changed i.e. maximizing $p(\mathbf{f}|\mathbf{e})$ instead of $p(\mathbf{e}|\mathbf{f})$

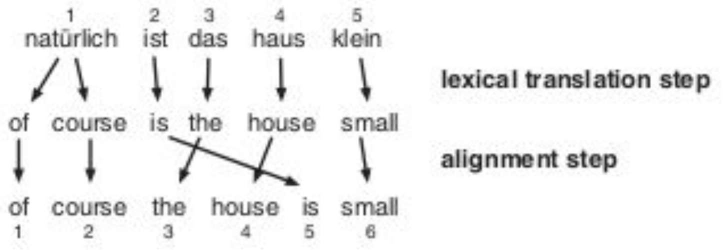
Higher IBM Models

- IBM Model 1: lexical translations
- IBM Model 2: adds absolute alignment model
- IBM Model 3: adds fertility model
- IBM Model 4: adds relative alignment model
- IBM Model 5: fixes deficiency

IBM Model 2

- Now IBM Model 1 gave idea of alignment which was still **implicit** in translation probability of the model.
- In this model we get **explicit alignment model** based on the positions of the input and output words. Translation of a source input word in position *i* to an source word in position *j* is modeled by an alignment probability distribution
  - $\mathbf{a}(i|j,l_e,l_f)$ where l_e and l_f are length of source and target sentences respectively.
- Translation in this model can be viewed as a two-step process with lexical translation step and an alignment step:

1.



2. Two steps are combined to form this model

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

Image Taken from SMT by Philipp Koehn

IBM Model 2

- Carry t from IBM model 1 [as initial values]
- Uniform probability initialization of alignment function [everything aligns with everything else]
- Count_a and total_a for alignment model

IBM Model 3

- Fertility of input words (i.e. how many words in target side are translated from a source word) is taken into consideration in this model.
  - *n(ϕ|f)*, this probability distribution function indicates how many words does source word *f* translates to.
  - So for source words to be dropped (having no translation candidate) will have this value as 0.
  - Words may be formed from NULL token, whose fertility is model in same way as that of any source word.
  - Distortion predicts output word positions based on input word.
  - Figure given below will give clear idea of how this model is built on top of previous two models and how it will work.

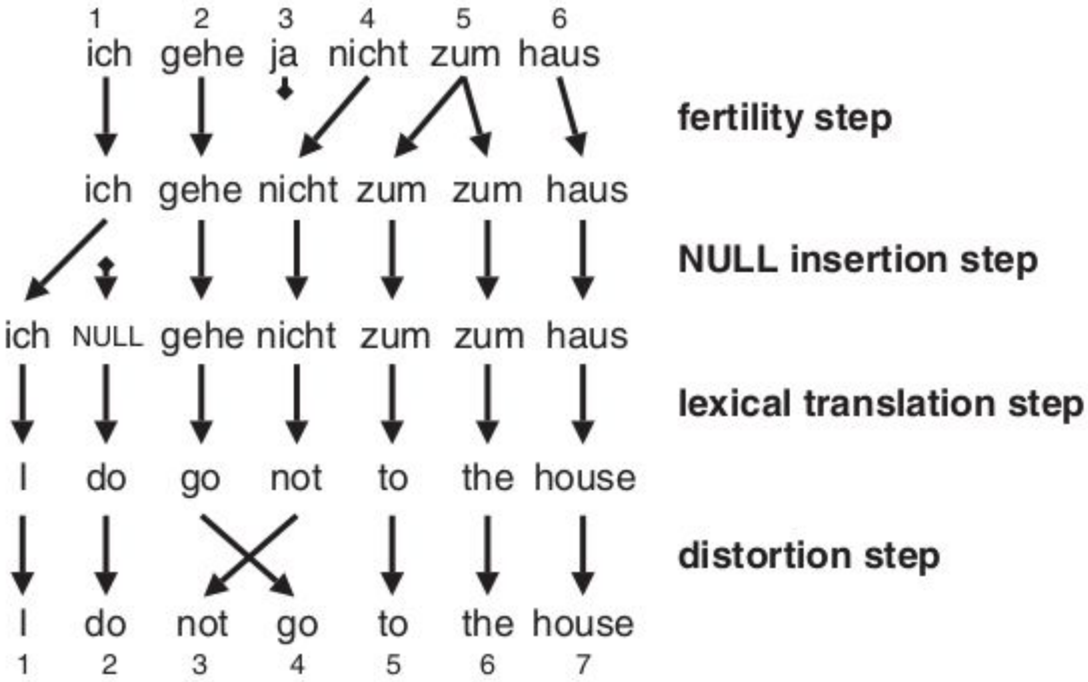


Image Taken from SMT by Philipp Koehn

IBM Model 4 & 5

- Model 4
  - Model 3 is a pretty decent model which takes into account major transformation in word based translation process namely translation of words, reordering, insertion of words, dropping of words, and one-to-many translation.
  - There is one problem with model 3: formulation of distortion probability distribution which might be sparse for large input and output sentences.
  - In translation, large phrases tend to move together so in model 4 relative distortion model is introduced.
- Model 5
  - Now according to previous models nothing prohibits the placement of an output word into a position that has already been filled.
  - IBM Model 5 introduces notion of keeping track of vacant word positions and allow placements only into these vacant positions.

Outline…

1. Problem Statement
2. Historical Perspective
3. Statistical Machine Translation Model
    a. Word Based Model
    b. Phrase Based Model
    c. Language Model
    d. Decoding
4. Evaluation
5. Neural Net Based Machine Translation Model
    a. Sequence to Sequence Translation Model
    b. Jointly learning alignment and Translation Model (attention model)
6. References
7. Questions

Phrase based model - Motivation

- Word based model may not be the best candidate for smallest unit of translation as they will often break down in frequent one-to-many mappings (and vice versa).
- Translating group of words instead of single word helps in resolving ambiguities. Ex1, Ex2.
- If we have large parallel corpora then we can learn more variation and more useful phrase translations.
- Conceptually this model is simpler to understand since we do away with notion of fertility, insertion, deletion of word based models.
- Phrases here is just group of words in sequence and not linguistic phrases.

Mathematical definition of PBM

$$\mathbf{e}_{\text{best}} = \text{argmax}_{\mathbf{e}}\, p(\mathbf{e}|\mathbf{f})$$
$$= \text{argmax}_{\mathbf{e}}\, p(\mathbf{f}|\mathbf{e})\, p_{\text{LM}}(\mathbf{e})$$

- Bayes rules to invert translation direction and integrate LM.
- Translation probability p(f|e) is decomposed into

$$p(\bar{f}_1^I|\bar{e}_1^I) = \prod_{i=1}^{I} \phi(\bar{f}_i|\bar{e}_i)\, d(\text{start}_i - \text{end}_{i-1} - 1)$$

- The source sentence f is broken up into I phrases.
- Reordering is handled by distance-based reordering model. start_i is the position of the first word of the source phrase that translates to i^{th} target phrase and end_i as position of the last word of that source phrase.
- *d* is an exponentially decaying cost function where $d(x)=\alpha^{|x|}$ with an appropriate parameter *α* so that *d* is a probability distribution. Hence movement of phrases over large distance during translation are more expensive than shorter movement.

Phrase table learning

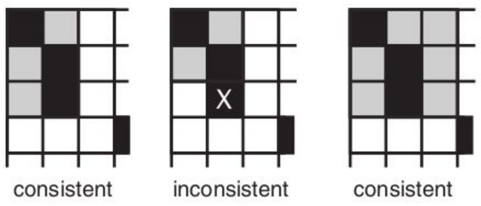
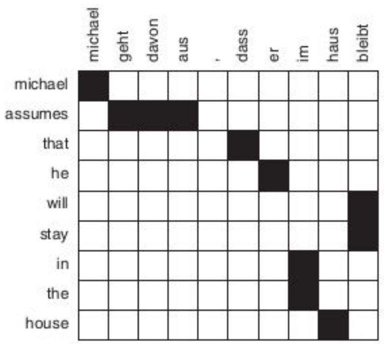
- Phrase table is a strength of PBSMT and stores entries like these:
  - europas ||| in europe ||| 0.0251019 0.066211 0.0342506 0.0079563
- To get such table we first find word alignment between each sentence pair of the parallel corpus. And then extract phrase pairs that are consistent with this word alignment.
- Definition of consistency:
  - We call a phrase pair (***f***,***e***) consistent with an alignment **A**, if all words in ***f*** that have alignment points in **A** have these with words in ***e*** and vice versa.
  - 

Image Taken from SMT by Philipp Koehn

Phrase extraction algorithm

```
Input: word alignment A for sentence pair (e,f)
Output: set of phrase pairs BP
for e_start = 1 ... length(e) do
  for e_end = e_start ... length(e) do
    // find the minimally matching foreign phrase
    (f_start,f_end) = ( length(f), 0 )
    for all (e,f) ∈ A do
      if e_start ≤ e ≤ e_end then
        f_start = min( f, f_start )
        f_end = max( f, f_end )
      end if
    end for
    add extract(f_start,f_end,e_start,e_end) to set BP
  end for
end for
function extract(f_start,f_end,e_start,e_end)
return {} if f_end == 0 // check if at least one alignment point
// check if alignment points violate consistency
for all (e,f) ∈ A do
  return {} if e < e_start or e > e_end
end for
// add pharse pairs (incl. additional unaligned f)
E = {}
f_s = f_start
repeat
  f_e = f_end
  repeat
    add phrase pair (e_start .. e_end, f_s .. f_e) to set E
    f_e ++
  until f_e aligned
  f_s --
until f_s aligned
return E
```



michael – michael
michael assumes – michael geht davon aus ; michael geht davon aus ,
michael assumes that – michael geht davon aus , dass
michael assumes that he – michael geht davon aus , dass er
michael assumes that he will stay in the house
– michael geht davon aus , dass er im haus bleibt
assumes – geht davon aus ; geht davon aus ,
assumes that – geht davon aus , dass
assumes that he – geht davon aus , dass er
assumes that he will stay in the house
– geht davon aus , dass er im haus bleibt
that – dass ; , dass
that he – dass er ; , dass er
that he will stay in the house
– dass er im haus bleibt ; , dass er im haus bleibt
he – er
he will stay in the house – er im haus bleibt
will stay – bleibt
will stay in the house – im haus bleibt
in the – im
in the house – im haus
house – haus

Image Taken from SMT by Philipp Koehn

Phrase table learning

- Now since we have extracted phrase pairs we can count how many times particular phrase was aligned with some particular phrase.

Log-linear model

- Phrase-based SMT model described so far is:

$$e_{\text{best}} = \text{argmax}_e \prod_{i=1}^{I} \phi(\bar{f}_i|\bar{e}_i)\, d(\text{start}_i - \text{end}_{i-1} - 1) \prod_{i=1}^{|\mathbf{e}|} p_{\text{LM}}(e_i|e_1 \ldots e_{i-1})$$

- Say there is a language pair (say hindi - punjabi) in which structure is similar. But model will be better if more importance is given to translation model part of above equation. Hence hinting us to give more weight to phrase translation part.
- Formally this is done by introducing weights λ_{Φ}, λ_d, λ_{LM} aiding in scaling contribution of each component.

$$e_{\text{best}} = \text{argmax}_e \prod_{i=1}^{I} \phi(\bar{f}_i|\bar{e}_i)^{\lambda_\phi}\, d(\text{start}_i - \text{end}_{i-1} - 1)^{\lambda_d} \prod_{i=1}^{|\mathbf{e}|} p_{\text{LM}}(e_i|e_1 \ldots e_{i-1})^{\lambda_{\text{LM}}}$$

Log-linear model

$$
\begin{aligned}
p(e,a|f) = \exp\Bigg[& \lambda_\phi \sum_{i=1}^{I} \log \phi(\bar{f}_i|\bar{e}_i) \\
& + \lambda_d \sum_{i=1}^{I} \log d(a_i - b_{i-1} - 1) \\
& + \lambda_{LM} \sum_{i=1}^{|\mathbf{e}|} \log p_{LM}(e_i|e_1 \ldots e_{i-1}) \Bigg]
\end{aligned}
$$

Benefits of using this model:

- Weighing f different model components may lead to improvement in translation quality.
- This model allow us to add more model components as feature functions.

References

- http://www.aclweb.org/anthology/J93-2003
- http://mt-class.org/jhu/slides/lecture-ibm-model1.pdf
- SMT By Philip Kohen