The Translation Model

IBM Model 1

$$P(F|E) = \sum_{A} P(F,A|E)$$

We start with IBM Model 1, so called because it is the first and simplest of five models proposed by IBM researchers in a seminal paper (Brown et al., 1993).

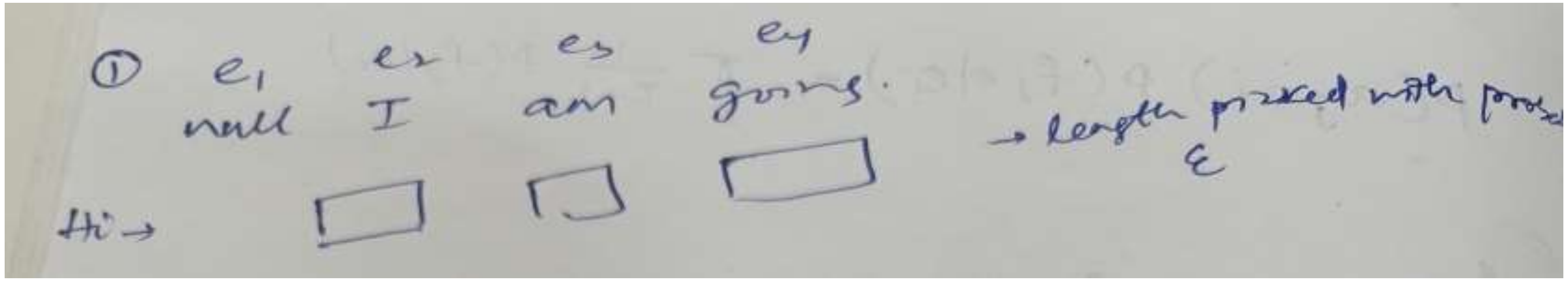
Here's the general IBM Model 1 generative story for how we generate a Spanish sentence from an English sentence $E = e_1, e_2, ..., e_I$ of length I:

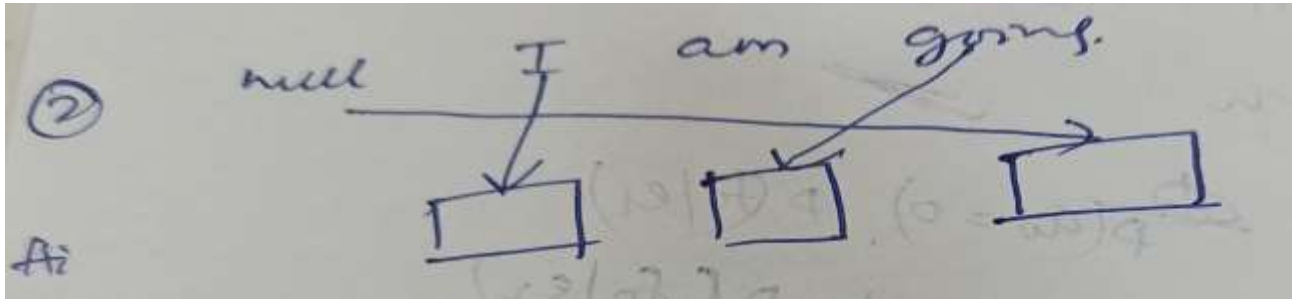
1. Choose a length J for the Spanish sentence, henceforth $F = f_1, f_2, ..., f_J$.
2. Now choose an alignment $A = a_1, a_2, ..., a_J$ between the English and Spanish sentences

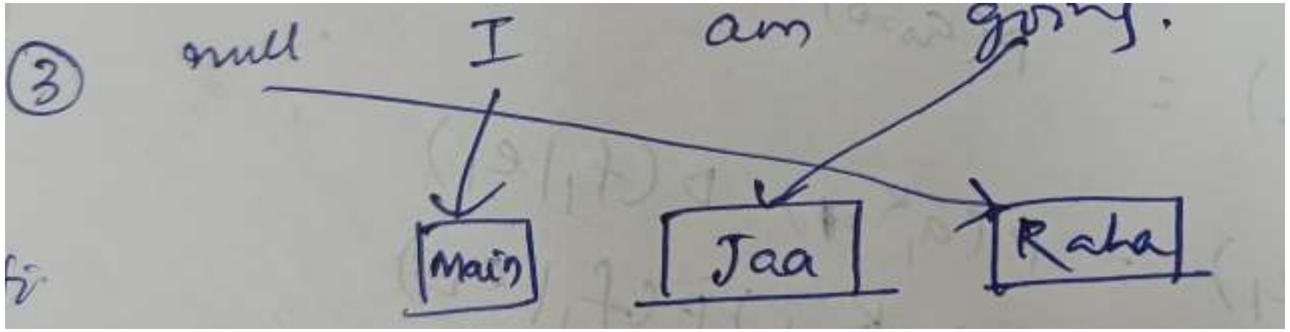
models proposed by IBM researchers in a seminal paper

Here's the general IBM Model 1 generative story for how we generate a Spanish sentence from an English sentence $E = e_1, e_2, \ldots, e_I$ of length I:

1. Choose a length J for the Spanish sentence, henceforth $F = f_1, f_2, \ldots, f_J$.
2. Now choose an alignment $A = a_1, a_2, \ldots, a_J$ between the English and Spanish sentences.
3. Now for each position j in the Spanish sentence, choose a Spanish word f_j by translating the English word that is aligned to it.







A = {main → I, Jaa → going, Raha → null}

- e_{a_j} is the English word that is aligned to the Spanish word f_j

$e_{a_1} \rightarrow I$, $e_{a_2} \rightarrow$ going $e_{a_3} \rightarrow$ null

- $t(f_x|e_y)$ is the probability of translating e_y by f_x (i.e., $P(f_x|e_y)$)

t(main | null) = 0.1 t(main | I) = 0.3 t(main | am) = 0.1

t(main | going) = 0.2

t(Jaa | null) = 0.1 t(Jaa | I) = 0.1 t(Jaa | am) = 0.1

t(Jaa | going) = 0.5

t(Raha | null) = 0.1 t(Raha | I) = 0.1 t(Raha | am) = 0.1

t(Raha | going) = 0.3

We'll work our way backwards from step 3. So suppose we already knew the length J and the alignment A, as well as the English source E. The probability of the Spanish sentence would be

$$P(F|E,A) = \prod_{j=1}^{J} t(f_j|e_{a_j}) \tag{25.17}$$

$$P(\text{main Jaa Raha} \mid \text{null I am going}, A) = t(\text{main}|\text{I}) * t(\text{Jaa}|\text{going})\ t(\text{Raha}|\text{null})$$

$$= 0.1 * 0.5 * 0.1$$

$$=$$

Now let's formalize steps 1 and 2 of the generative story. This is the probability $P(A|E)$ of an alignment A (of length J) given the English sentence E. IBM Model 1 makes the (very) simplifying assumption that each alignment is equally likely. How many possible alignments are there between an English sentence of length I and a Spanish sentence of length J? Again assuming that each Spanish word must come from one of the I English words (or the 1 NULL word), there are $(I+1)^J$ possible alignments. Model 1 also assumes that the probability of choosing length J is some small constant ϵ. The combined probability of choosing a length J and then choosing any particular one of the $(I+1)^J$ possible alignments is

$$P(A|E) = \frac{\epsilon}{(I+1)^J} \tag{25.18}$$

$$P(A|\text{null I am going}) = \frac{\varepsilon}{4^3}$$

$$(I+1)^J \tag{25.18}$$

We can combine these probabilities as follows:

$$
\begin{aligned}
P(F,A|E) &= P(F|E,A) \times P(A|E) \\
&= \frac{\epsilon}{(I+1)^J} \prod_{j=1}^{J} t(f_j|e_{a_j})
\end{aligned}
\tag{25.19}
$$

P(main Jaa Raha, A | null I am going) =

$\frac{\varepsilon}{4^3} * 0.005$

$$(25.19)$$

This probability, $P(F,A|E)$, is the probability of generating a Spanish sentence F through a particular alignment. To compute the total probability $P(F|E)$ of generating F, we just sum over all possible alignments:

$$\begin{aligned} P(F|E) &= \sum_A P(F,A|E) \\ &= \sum_A \frac{\epsilon}{(I+1)^J} \prod_{j=1}^{J} t(f_j|e_{a_j}) \end{aligned} \tag{25.20}$$

Recap

Statistical MT :- 27.03.2021

Given a foriegn sentance F $\longrightarrow$ we want to predict the most likely sentance E with the same meaning.

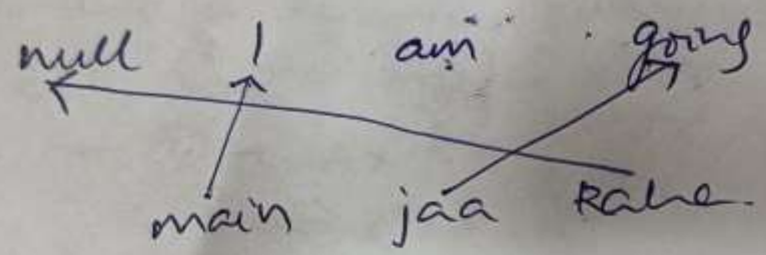
$$\hat{E} = \arg\max_E P(E|F)$$

translation model.

$$= \arg\max_E P(F|E) * P(E) \longrightarrow \text{language model}$$

$$E \xrightarrow{\text{noisy channel}} F \longrightarrow \hat{E}$$

Alignment :- is a function from output to input of the translation model.



$$P(F\cancel{,A}|E) = \sum_{A} P(F,A|E)$$

$$= \sum_{J=1}^{I} \frac{\epsilon}{(I+1)^J} \prod_{j=1}^{J} t(f_j|e_{a_j})$$

$$\boxed{\hat{A} = \underset{A}{\operatorname{argmax}}\, P(F,A|E)}$$

$$\cancel{= \underset{A}{\operatorname{argmax}} \prod t(f_j|e_{a_j})}$$

Training alignment models :-

sentence level alignment

word level alignment -

for each sentence pair (F_s, e_s) we need to learn an alignment $A = a_1^J$ and the corresponding translation probabilities.

$$P(A|E,F) = \frac{P(A,F|E)}{\sum_A P(A,F|E)}$$ → if we knew translation probabilities

$$t(\text{main}/I) = \frac{C(\text{main}, I)}{C(I)}$$ → if we knew the perfect alignment

$$P(F,A|E) \simeq \prod^J t(f_j|e_{a_j})$$ → simplification.

green house   the house

mera ghar   EK ghar

$V_F = \{$ mera, ghar, EK $\}$ $V_E = \{$ green, house, the $\}$

$t(\text{mera}|\text{green}) = \frac{1}{3}$ $t(\text{ghar}|\text{green}) = \frac{1}{3}$ $t(\text{EK}|\text{green}) = 1/3$

$t(\text{mera}|\text{house}) = \frac{1}{3}$ $t(\text{ghar}|\text{house}) = \frac{1}{3}$ $t(\text{EK}|\text{house}) = 1/3$

$t(\text{mera}|\text{the}) = \frac{1}{3}$ $t(\text{ghar}|\text{the}) = \frac{1}{3}$ $t(\text{EK}|\text{the}) = 1/3$.

ExStep 1 :-

(a) -

green house / mera ghar   green house / mera ghar (crossed)   the house / EK ghar   the house / EK ghar (crossed)

$\frac{1}{9}$ $\frac{1}{9}$ $\frac{1}{9}$

$P(a,f|e) = \frac{1}{3} \times \frac{1}{3} = \frac{1}{9}$ $\frac{1}{9}$

(b) normalize $P(a|e,f) = \dfrac{P(a,e|f)}{\sum_a P(a,e|f)}$

~~P(a|e,f)~~ $\dfrac{1/9}{2/9} = \frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$

(c) $t\text{count}(\text{mera}|\text{green}) = \frac{1}{2}$ $t(\text{ghar}|\text{green}) = \frac{1}{2}$ $t(\text{EK}|\text{green}) = 0$

$t\text{count}(\text{mera}|\text{house}) = \frac{1}{2}$ $t(\text{ghar}|\text{house}) = \frac{1}{2} + \frac{1}{2}$ $t(\text{EK}|\text{house}) = \frac{1}{2}$

$t\text{count}(\text{mera}|\text{the}) = 0$ $t(\text{ghar}|\text{the}) = \frac{1}{2}$ $t(\text{EK}|\text{the}) = \frac{1}{2}$

total green = 1
house 2
the = 1

$t(\text{mera}|\text{green}) = \frac{1/2}{1} = \frac{1}{2}$ $t(\text{ghar}|\text{green}) = 1/2 / 1 = \frac{1}{2}$ $t(\text{EK}|\text{green}) = 0$

$t(\text{mera}|\text{house}) = \frac{1/2}{2} = \frac{1}{4}$ $t(\text{ghar}|\text{house}) = \frac{1}{2}$ $t(\text{EK}|\text{house}) = \frac{1}{2}/2 = \frac{1}{4}$

$t(\text{mera}|\text{the}) = 0$ $t(\text{ghar}|\text{the}) = \frac{1}{2}$ $t(\text{EK}|\text{the}) = \frac{1}{2}$

Phrase-based Translation models:

- Sequence of words are the units.
  - organize english source words into phrases $E = \bar{e}_1, \bar{e}_2 \ldots \bar{e}_I$
  - translate each english phrase to corresponding french phrase $\bar{f}_i$ i.e. $F = \bar{f}_1 \bar{f}_2 \ldots \bar{f}_I$
  - reorder $\bar{f}_i$'s if required.

$\phi(\bar{f}_i|\bar{e}_i)$: translation probability of generating $\bar{f}_i$ from $\bar{e}_i$

a_i : start of foriegn phrase generated by i^{th} english phrase $\bar{e}_i$

b_{i-1} : end posn of foreign phrase generated by $i-1^{th}$ english phrase $\bar{e}_{i-1}$

$d(a_i, b_{i-1})$: distortion probability $- \alpha^{|a_i - b_{i-1} - 1|}$

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

e Many didnot slap the green witch

f Manyne hari bhutni ko nahi mara

$P(f|e) = P(\text{manyne} \mid \text{many})\, d(1-0)$

$P(\text{nahi} \mid \text{did not})\, d(3-1)$

$P(\text{mara} \mid \text{slap})\, d(4-3)$

$P(\text{hari bhutni ko} \mid \text{the green witch})\, d(2-4)$

— paramaters $\phi(\bar{f_i}, \bar{e_i})$

- to ~~learn~~ find ϕ as well as α, we need a parallel corpus where in each pair, each english phrase is mapped to ~~one~~ foriegn language phrase.

$$-\ \phi(\bar{f_i}, \bar{e_i}) = \frac{\text{count}(\bar{f_i}, \bar{e_i})}{\sum_{\bar{f_i}} \text{count}(\bar{f_i}, \bar{e_i})}$$

Many did not slap the green witch.

Many ne hari bhutni ko nahi mara.

Many ne hari bhutni ko nahi mara

Many ne

Many

did

not

slap

the

green

witch.

Extracting aligned pair of phrases

→ use two different aligners - one from F to E, the other from E to F

→ take intersection of the two alignments to get high precision aligned words

→ take the union of the alignments to get low precision aligned words

→ use a classifier to incrementally ~~update~~ select words from union and add back to minimal intersective alignment. (Koehn 2003

Decoding for phrase-based MT

$$\hat{E} = \underset{E}{\operatorname{argmax}} \; P(F/E) \; P(E)$$

- finding the english sentence that maximizes the translation & lang. model probabilities is a search problem — Decoding.

MT decoders are a special case of A* search.

— stack decoding. implemented using a priority Queue.

Function StackDecoding (source sentence) returns (target sentence)

init - stack contains null hypothesis

do.

pop the best hypothesis off the stack

if h is complete, return h

for each possible expansion h' of h

allign a score to h'

push h' onto the stack

Decoding for phrase-based MT

$$\hat{E} = \underset{E}{\operatorname{argmax}} \; P(F/E) \; P(E)$$

- finding the english sentence that maximizes the translation & lang. model probabilities is a search problem — Decoding.
- MT decoders are a special case of A* search.
  - stack decoding. implemented using a priority Queue.

Function StackDecoding (source sentence) returns (target sentence)

init - stack contains null hypothesis

do.
  pop the best hypothesis off the stack
  if h is complete, return h
  for each possible expansion h' of h
    assign a score to h'
    push h' onto the stack

E:

F: ********

Cost: 100

E: Mary

F: M******

cost: 200

E: mary did not

F: MN***N*

cost: 202

E: watch

F: ***B***

cost: 300

E: NO

F: ****~~M~~N*

cost 100

E: NO slap

F: *****NM

cost: 500

E: NO slap

F: ****KNM

cos: 550.

For a set of partially translated phrases

$S = (F, E)$,

$$\text{cost}(E, F) = \prod_{i \in S} \phi(\bar{f}_i, \bar{e}_i)\, d(a_i - b_{i-1})\, P(E)$$

By combining the current cost with future cost,
for current F — for remaining phrases F

the state cost gives an estimate of total probability of search path for eventual complete translation sentence passing through the current node.

- for future cost, distortion prob. is ignored for simplicity.

beam search pruning - keep only the K promising states at each level

Evaluation Metrics

Accuracy

Actual \ Predicted	T	F
T	TP	FN
F	FP	TN

$$Acc = \frac{1}{N} \sum_{i}^{N} \delta\left(y^{(i)} = \hat{y}\right)$$

class imbalance

Precision, Recall, F1 measure

Actual \ Predicted	T	F
T	TP	FN
F	FP	TN

Precision $\hat{p} \rightarrow \dfrac{TP}{TP+FP}$

Recall $\hat{r} = \dfrac{TP}{TP+FN}$

F_1 measure $\rightarrow \dfrac{2\hat{r}\hat{p}}{\hat{r}+\hat{p}}$

Receiver operating characteristic (ROC) curve

Actual \ Predicted	T	F
T	TP	FN
F	FP	TN

x axis False positive rate $\frac{FP}{FP+TN}$

y axis True positive rate $\frac{TP}{TP+FN}$

ROC can be summarized using AUC.

Receiver operating characteristic (ROC) curve

Predicted

Actual	T	F
T	TP	FN
F	FP	TN

x axis False positive rate $\frac{FP}{FP+TN}$

y axis True positive rate $\frac{TP}{TP+FN}$

ROC can be summarized using AUC:

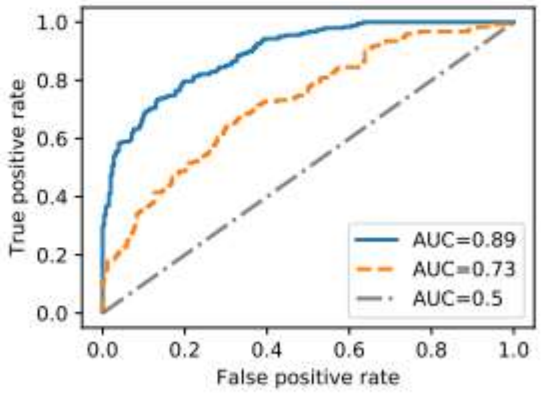


Figure 4.4: ROC curves for three classifiers of varying discriminative power, measured by AUC (area under the curve)

Evaluation Metrics for MT: BLEU

- P_n (modified n-gram precision) $= \dfrac{\text{\# n-grams in both reference \& hypothesis translation}}{\text{\# ngrams in the hypothesis translation}}$

$$BLUE = \exp\left(\frac{1}{N}\sum_{n=1}^{N} \log P_n\right) * BP \quad , \quad BP = \begin{cases} 1 & \text{if } c > r \\ e^{1-\frac{r}{c}} & \text{if } c \leq r \end{cases}$$

c = total length of ~~reference~~ hypothesis

r = effective reference length.

Evaluation Metrics for MT: BLEU

- P_n (modified n-gram precision) $= \dfrac{\text{\# n-grams in both reference \& hypothesis translation}}{\text{\# ngrams in the hypothesis translation}}$

$$BLUE = \exp\left(\frac{1}{N}\sum_{n=1}^{N} \log P_n\right) * BP \quad , \quad BP = \begin{cases} 1 & \text{if } c > r \\ e^{1-\frac{r}{c}} & \text{if } c \le r \end{cases}$$

c = total length of ~~reference~~ hypothesis

r = effective reference length.

$$e^{1-\frac{5}{3}} = 0.51$$

Reference · Vinay likes programming in python

		P_1	P_2	P_3	P_4	BP	Blue
hypothesis 1	TO vinay it like to program python	$\frac{2}{7}$	0	0	0	1	.21
2	Vinay likes python	$\frac{3}{3}$	$\frac{1}{2}$	0	0	.51	.33
3	vinay likes programming in his pyjamas	$\frac{4}{6}$	$\frac{3}{5}$	$\frac{2}{4}$	$\frac{1}{3}$	1	.76

Evaluation Metrics for MT: chrF

- character F-score

chrP percentage of character 1-grams, 2-grams, ..., k-grams in the hypothesis that occur in the reference, averaged.

chrR of character 1-grams, 2-grams,..., k-grams in the reference that occur in the hypothesis, averaged.

The metric then computes an F-score by combining chrP and chrR using a weighting parameter β. It is common to set $\beta = 2$, thus weighing recall twice as much as precision:

$$\text{chrF}\beta = (1+\beta^2)\frac{\text{chrP}\cdot\text{chrR}}{\beta^2\cdot\text{chrP}+\text{chrR}} \tag{10.24}$$

For $\beta = 2$, that would be:

$$\text{chrF2} = \frac{5\cdot\text{chrP}\cdot\text{chrR}}{4\cdot\text{chrP}+\text{chrR}}$$

Evaluation Metrics for MT: chrF

- character F-score

For example, consider two hypotheses that we'd like to score against the reference translation *witness for the past*. Here are the hypotheses along with chrF values computed using parameters $k = \beta = 2$ (in real examples, k would be a higher number like 6):

REF:	`witness for the past,`	
HYP1:	`witness of the past,`	chrF2,2 = .86
HYP2:	`past witness`	chrF2,2 = .62

Let's see how we computed that chrF value for HYP1 (we'll leave the computation of the chrF value for HYP2 as an exercise for the reader). First, chrF ignores spaces, so we'll remove them from both the reference and hypothesis:

REF: `witnessforthepast,` (18 unigrams, 17 bigrams)
HYP1: `witnessofthepast,` (17 unigrams, 16 bigrams)

Evaluation Metrics for MT: chrF

- character F-score

Next let's see how many unigrams and bigrams match between the reference and hypothesis:

unigrams that match: `w i t n e s s f o t h e p a s t ,` (17 unigrams)
bigrams that match: `wi it tn ne es ss th he ep pa as st t,` (13 bigrams)

We use that to compute the unigram and bigram precisions and recalls:

unigram P:	17/17 = 1	unigram R:	17/18 = .944
bigram P:	13/16 = .813	bigram R:	13/17 = .765

Finally we average to get chrP and chrR, and compute the F-score:

$$\begin{aligned}
\text{chrP} &= (17/17+13/16)/2 = .906 \\
\text{chrR} &= (17/18+13/17)/2 = .855 \\
\text{chrF2,2} &= 5\frac{\text{chrP} * \text{chrR}}{4\text{chrP}+\text{chrR}} = .86
\end{aligned}$$

Evaluation Metrics for MT

Translations are evaluated along two dimensions:

1. **adequacy:** how well the translation captures the exact meaning of the source sentence. Sometimes called **faithfulness** or **fidelity**.
2. **fluency:** how fluent the translation is in the target language (is it grammatical, clear, readable, natural).