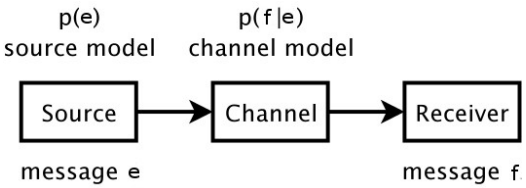
3 Noisy Channel model

We observe a distorted message R (foreign string f). We have a model on how the message is distorted (translation model $t(f|e)$) and also a model on which original messages are probable (language model $p(e)$).
Our object is to recover the original message S (English string e).



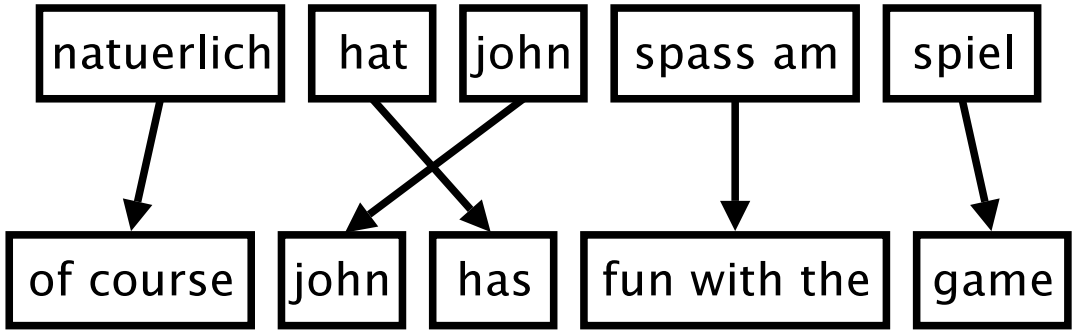
Derivation of "noisy channel model" in a probabilistic framework using the Bayes rule

$$\begin{aligned}\hat{e} &= \arg\max_e p(e|f) \\ &= \arg\max_e \frac{p(f|e)p(e)}{p(f)} \\ &= \arg\max_e \underbrace{p(f|e)}_{\text{translation model}} \cdot \underbrace{p(e)}_{\text{language model}}\end{aligned} \tag{1}$$

Advantages of generative SMT models:
The translation problem can be broken up into simpler problems: translation model and language model. Like that, simpler problems can be solved separately and estimation and model definitions are independent.

Note: "Source" in the noise channel model should be read as "original". In terms of generative SMT we called the "original" the "target".

4 Phrase-based SMT



- Foreign input is segmented into phrases.
- Phrases are translated into English.
- Phrases are reordered.

This works already better then word-based translation models (which are still used in alignment).

Phrase-based SMT as generative model:

$$\begin{aligned}\hat{e} &= \arg\max_e p(e|f) \\ &= \arg\max_e \underbrace{p(f|e)}_{\text{translation model}} \cdot \underbrace{p(e)}_{\text{language model}}\end{aligned} \tag{2}$$

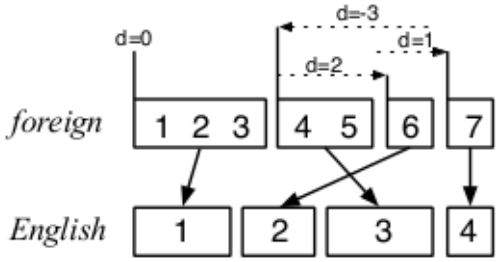
4.1 Reordering model in Phrase-based SMT

$$p(\bar{f}_1^I|\bar{e}_1^I) = \prod_{i=1}^{I} \underbrace{\phi(\bar{f}_i|\bar{e}_i)}_{\text{phrase translation probability}} \cdot \underbrace{d(\text{start}_i - \text{end}_{i-1} - 1)}_{\text{(phrase) distortion probability}} \tag{3}$$

d is called "distortion probability", "reordering probability" or rather **"distortion cost"**. It describes the number of words skipped to the right (+) or left (-) when taking foreign words out of sequence.

start_i is the position of the **first** word of a **foreign** phrase corresponding to the **i$^{\text{th}}$ English** phrase (in the figure above, $\text{start}_4 = 4$).

end_{i-1} is the position of the **last** word of a **foreign** phrase corresponding to the **previous English** phrase (in the figure above, $\text{end}_3 = 2$).



phrase	translates	movement	distance	calculation
1	1–3	start at beginning	0	1-0-1=0
2	6	skip over 4–5	+2	6-3-1=2
3	4–5	move back over 4–6	-3	4-6-1=-3
4	7	skip over 6	+1	7-5-1=+1

Example (phrase 2):
$d(\text{start}_i - \text{end}_{i-1} - 1) = d(\text{start}_2 - \text{end}_1 - 1) = d(6 - 3 - 1) = 2$

Example (phrase 3):
$d(\text{start}_i - \text{end}_{i-1} - 1) = d(\text{start}_3 - \text{end}_2 - 1) = d(4 - 6 - 1) = -3$

Scoring function: $d(x) = \alpha^{|x|}$

4.2 Weighted models in SMT

With th knowledge we have we can build the "standard" generative model consisting of 3 submodels / modules:

- phrase translation model $\phi(\bar{f}|\bar{e})$
- distortion model d
- language model $p_{\text{LM}}(e)$

The generative model defines joint probability by assuming the independence of modules $\Rightarrow$ **product model**.

The standard model was derived (inspired) more or less from a well-defined optimization task (max. likelihood). Quite expected, likelihood is not what people report when asked to evaluate a translation – we have a model-task discrepancy. More complex objectives usually require more flexible models.

From an engineering perspective, the developer may notice that the translations need to be more fluent and that increasing a language model's importance is necessary. This is not possible within the noisy channel model (violates the Bayes rule). Moreover, he may argue that combining several language models might be beneficial.

To summarize, reasons for a move from the product model to more general log-linear models:

- including more features,
- tuning the relative importance of features.

While we violate the assumptions of the initial model we still want to land on some known model

4.2.1 Weighted models

Add weight to modules:

$$\begin{aligned}\hat{e} = \arg\max_e \prod_i^I \phi(\bar{f}_i|\bar{e}_i)^{\lambda\phi} \\ \cdot d(\text{start}_i - \text{end}_{i-1} - 1)^{\lambda d} \\ \cdot \prod_{j=1}^{|e|} p_{\text{LM}}(e_j|e_1,\ldots,e_{j-1})^{\lambda_{\text{LM}}}\end{aligned} \tag{4}$$

Now we work with different, more expressive probability models
$\Rightarrow$ **log-linear model**.

4.2.2 log-linear model

$$\begin{aligned}p(x) = \exp\sum_{i=1}^n \lambda_i h_i(x) \\ \lambda_i = \text{parameter} \\ h_i = \text{features}\end{aligned} \tag{5}$$

SMT as log-linear model:

- feature-function $h_1 = \log \phi \Rightarrow \phi = e^{h_1}$
- feature-function $h_2 = \log d \Rightarrow d = e^{h_2}$
- feature-function $h_3 = \log p_{\text{LM}} \Rightarrow p_{\text{LM}} = e^{h_3}$

$$\begin{aligned} \phi^{\lambda\phi} d^{\lambda d} p_{\text{LM}}^{\lambda \text{LM}} &= e^{h_1\lambda_1} e^{h_2\lambda_2} e^{h_3\lambda_3} \\ &= \prod_{i=1}^{3} e^{h_i\lambda_i} \\ &= e^{\sum_{i=1}^{3} h_i\lambda_i} \end{aligned} \tag{6}$$

$$\begin{aligned} \hat{e} = \arg\max_e \ \exp(&\lambda_\phi \sum_i^I \log \phi(\bar{f}_i|\bar{e}_i) \\ &+ \lambda_d \sum_i^I \log d(\text{start}_i - \text{end}_{i-1} - 1) \\ &+ \lambda_{\text{LM}} \sum_{j=1}^{|e|} \log p_{\text{LM}}(e_j|e_1, \ldots, e_{j-1})) \end{aligned} \tag{7}$$

Although we obtained log-linear models as a generalization of a product model, it turns out that the following formulations are dual:

- MaxEnt + moment conservation
- ML + Gibbs

4.2.3 Advantages of log-linear model

- Standard modules can be weighted.
- Additional feature functions can be added easily.
- 3 standard features: ϕ, d, p_{LM}

Additional features: With the help of feature we can now easily inject domain knowledge into the model. The final answer whether these new feature are useful will be always given by experiment, but having some intuition is necessary to design them.

- Word count:

  $\text{wc}(e) = \log |e|^{\omega}$, $\begin{smallmatrix}\omega<1 \text{ prefers fewer words}\\ \omega>1 \text{ prefers more words}\end{smallmatrix}$

  The wc feature corrects the bias of the language model towards short translations.

- Phrase count:

  $\text{pc}(e) = \log |I|^{\rho}$, $\begin{smallmatrix}I=\text{number of phrases}\\ \rho<1 \text{ prefers fewer phrases, i.e., longer phrase}\\ \rho>1 \text{ prefers shorter phrases, i.e., more phrases}\end{smallmatrix}$

  The pc feature fine-tunes fine or coarse phrase segmentation. Trade-off: longer phrases are more grammatical but less statistically reliable; combinations of shorter phrases are more often disfluent but their statistics can be estimated on smaller corpora. The task of this feature is to automatically resolve this trade-off for your particular case (data).

  In contrast, choosing more linguistically motivated boundaries was not shown to be especially helpful.

- Multiple language models

- Multiple translation models:
  e.g., src-trg and trg-src translation models

- Bidirectional alignment probabilities

  Example: very long English phrase $\bar{e}$ is extracted along with a foreign phrase $\bar{f}$. In results, $\phi(\bar{f}|\bar{e})$ is high, LM will like it as well – will be often used. Add the reverse direction $\phi(\bar{e}|\bar{f})$ (small) to prevent this from happening, and weight the relative importance of both model.

- Lexically weighted phrase translation probabilities:

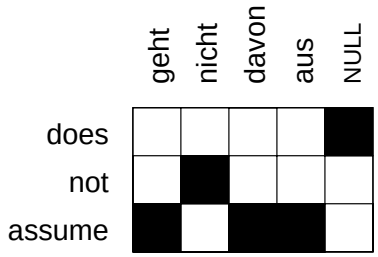
  Lexical weighting of phrases with word translation probabilities for rare phrases with unreliable phrase translation probabilities. We would like

to back-off to the word level to compensate for the missing information:

$$\text{lex}(\bar{e}|\bar{f},a) = \prod_{i=1}^{|\bar{e}|} \frac{1}{|\{j|(i,j)\in a\}|} \sum_{\forall(i,j)\in a} w(e_i|f_j)$$

Each English word e_i in a phrase $\bar{e}$ is generated independently by an aligned foreign word f_j in $\bar{f}$ with the word translation probability $w(e_i|f_j)$ or average if multiple alignment is possible.

Again, we can use both $\text{lex}(\bar{e}|\bar{f})$ and $\text{lex}(\bar{f}|\bar{e})$.



Example: $\text{lex}(\bar{e}|\bar{f},a) = w(\text{does}|\text{NULL})$
$\cdot\, w(\text{not}|\text{nicht})$
$\cdot\, \frac{1}{3}(w(\text{assume}|\text{geht}) + w(\text{assume}|\text{davon})$
$+\, w(\text{assume}|\text{aus}))$

4.3 Lexicalized Reordering

Reordering based on distance in *words* is not expressive enough. We want to be able to distinguish the reordering cost for different lexical phrases.
Lexical reordering learns 3 types of reordering for each lexical *phrase*:

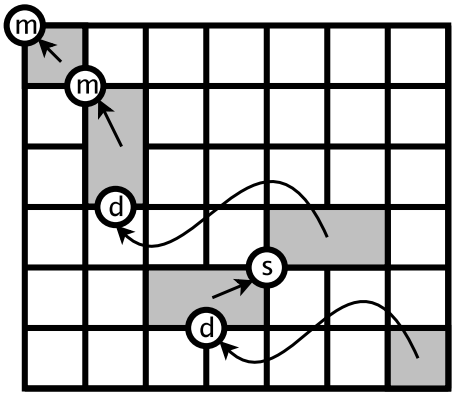
$$\text{orientation} \in \{\text{monotone}, \text{swap}, \text{discontinuous}\}$$

Learning orientation preference

$p(\text{orientation}|\bar{f},\bar{e})$ is the lexical reordering probability distribution.

During phrase extraction from alignment, check:

- if a word alignment point to the top left exists
  ⇒ monotone (m)
- elsif a word alignment point to the top right exists
  ⇒ swap (s)
- else
  ⇒ discontinuous (d)



Estimation of lexical reordering probability:

unsmoothed estimate:

$$\hat{p}(\text{orientation}|\bar{e},\bar{f}) = \frac{\text{count}(\text{orientation},\bar{e},\bar{f})}{\sum_o \text{count}(\text{orientation},\bar{e},\bar{f})}$$

smoothed estimate:

$$\hat{p}(\text{orientation}|\bar{e},\bar{f}) = \frac{\lambda p(\text{orientation}) + \text{count}(\text{orientation},\bar{e},\bar{f})}{\lambda \cdot 1 + \sum_o \text{count}(\text{orientation},\bar{e},\bar{f})}$$

$$\text{where } p(\text{orientation}) = \frac{\sum_{\bar{f}} \sum_{\bar{e}} \text{count}(\text{orientation},\bar{e},\bar{f})}{\sum_o \sum_{\bar{f}} \sum_{\bar{e}} \text{count}(\text{orientation},\bar{e},\bar{f})}$$

⇒ linear interpolation with unlexicalized orientation model.

4.4 Direct training of phrases

Why not try to directly learn **phrase-alignment** with EM?

Goal: Directly learn phrases using EM without heuristic extraction from word alignments.

1. Initialize: Start with uniform $\phi(\bar{e}, \bar{f})$ phrase pair probabilities.
2. E-Step: Assign phrase alignment probabilities to all phrases in all sentence pairs.
3. M-Step: Collect counts for phrase pairs $(\bar{e}, \bar{f})$, weighted by the phrase alignment probability.
4. Update phrase translation probabilities $\phi(\bar{e}, \bar{f})$.

Problems:

- Method overfits easily: Long phrase pairs, often spanning entire sentences, are preferred.
- Inefficient, because of large space of alignments.

Solutions:

- Restrict the phrase length; disallow phrases or phrase pairs that occur only once.
- Use old-style heuristics: Word alignments restricts possible phrases.
- New-style Bayesian approach: Define prior that imposes a bias towards shorter phrases.