CA4012

Statistical Machine Translation

Week 9: Decoding

Lecturer: Mohammed Hasanuzzaman

Lab Tutor: Longyue Wang, Meghan Dowling

29.03.2018

Recap

- What is the difference between a phrase-based translation model and a word-based translation model?
- Why do we still care about word-based models?
- What are pros and cons of shorter phrases and longer phrases?
- What does consistency mean when extracting bilingual phrases?

Recap

- What is the difference between a phrase-based translation model and a word-based translation model?
- Phrase-based model:
  - translate phrases as atomic units
  - Phrases can capture more local context
  - Phrase extraction is based on word alignments generated from word-based translation model
  - Longer phrases will face a data sparseness problem
- Word-based translation model:
  - Translate words as atomic units
  - Lack of local context
  - Higher IBM models introduce more computational complexity

Recap

- Why do we still care about word-based models?
- From word-based models,
  - we can obtain word alignment links so that we can extract phrases.
  - we can obtain word translation probabilities so that we can calculate lexical translation probabilities for phrases

Recap

- What are pros and cons of shorter phrases and longer phrases?
- Shorter phrases:
  - occur more frequently, so they will more often be applicable to previously unseen sentences.
  - lack of local context compared to longer phrases.
- Longer phrases:
  - capture more local context and help us to translate larger chunks of text at one time, maybe even occasionally an entire sentence.
  - potential data sparseness problem if the phrase is too long.

Recap

- What does consistency mean when extracting bilingual phrases?
- A phrase pair (e, f) is consistent with a bidirectional word alignment A if and only if
  - For all words e_i in e, if e_i is aligned to a word f_j in A, then f_j is in f.
  - For all words f_j in f, if f_j is aligned to a word e_i in A, then e_i is in e.
  - There exists e_i in e, f_j in f : (e_i, f_j) in A

Content

- **Phrase-based Translation Model**
- Distance-based Reordering
- Log-linear Model
- Decoding
- Make Decoding Manageable
- Exercises

Recall: Noisy Channel Model

$$\hat{e} = \underbrace{\operatorname*{argmax}_{e}}_{\text{Decoder}} \underbrace{p(e)}_{\text{Language Model}} \underbrace{p(f|e)}_{\text{Translation Model}}$$

Translation Model

Decoder

Language Model

Phrase-based Model: Decomposition

$$p(f|e) = \prod_{i=1}^{I} \phi(\bar{f}_i|\bar{e}_i)\, d(start_i - end_{i-1} - 1)$$

Phrase-based Model: Decomposition

$$p(f|e) = \prod_{i=1}^{I} \phi(\bar{f}_i|\bar{e}_i)\, d(start_i - end_{i-1} - 1)$$

- f is segmented into *I* phrases

Phrase-based Model: Decomposition

$$p(f|e) = \prod_{i=1}^{I} \phi(\bar{f}_i|\bar{e}_i)\, d(start_i - end_{i-1} - 1)$$

- f is segmented into I phrases
- ϕ is the phrase table translation probability

Phrase-based Model: Decomposition

$$p(f|e) = \prod_{i=1}^{I} \phi(\bar{f}_i|\bar{e}_i)\, d(start_i - end_{i-1} - 1)$$

- f is segmented into I phrases
- ϕ is the phrase table translation probability
- d is the distance-based reordering function.

Phrase-based Model: Decomposition

$$p(f|e) = \prod_{i=1}^{I} \phi(\bar{f}_i|\bar{e}_i)\, d(start_i - end_{i-1} - 1)$$

- f is segmented into I phrases
- ϕ is the phrase table translation probability
- d is the distance-based reordering function.
- $start_i$ is the position of the first word of $\bar{f}_i$

Phrase-based Model: Decomposition

$$p(f|e) = \prod_{i=1}^{I} \phi(\bar{f}_i|\bar{e}_i)\, d(start_i - end_{i-1} - 1)$$

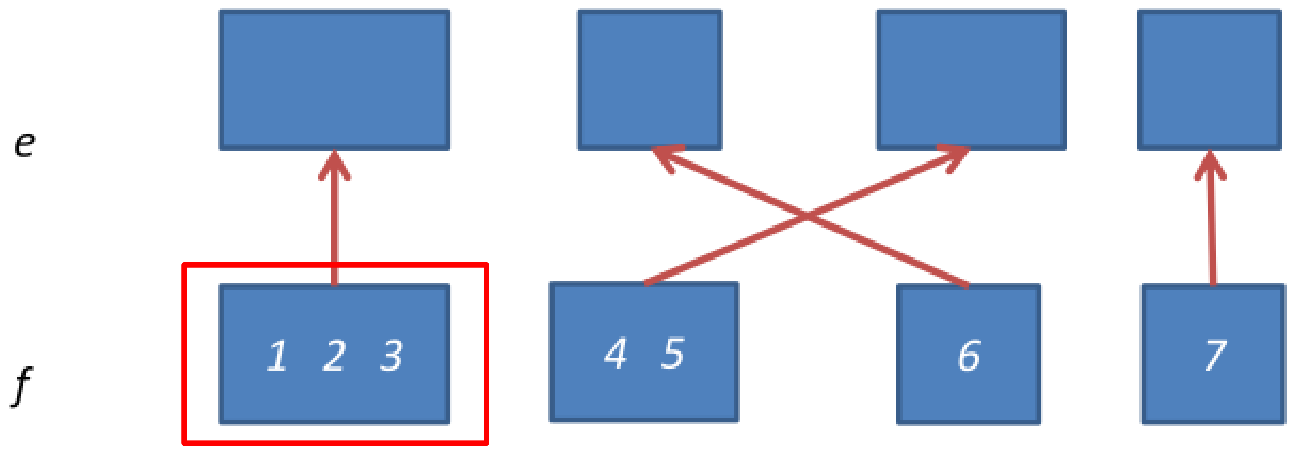
- f is segmented into I phrases
- ϕ is the phrase table translation probability
- d is the distance-based reordering function.
- $start_i$ is the position of the first word of $\bar{f}_i$
- end_{i-1} is the position of the last word in $\bar{f}_{i-1}$

Content

- Phrase-based Translation Model
- **Distance-based Reordering**
- Log-linear Model
- Decoding
- Make Decoding Manageable
- Exercises

Distance based Reordering

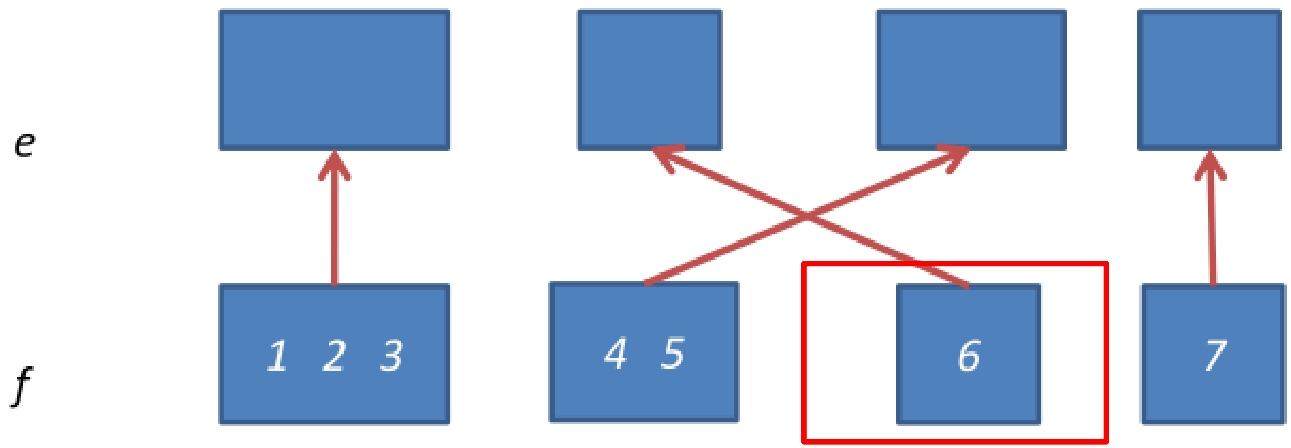
What are the distances associated with each of the English phrases?



i	f_i	$start_i - end_{i-1} - 1$	Distance
1	1-3	?	?
2	6	?	?
3	4-5	?	?
4	7	?	?

Distance based Reordering

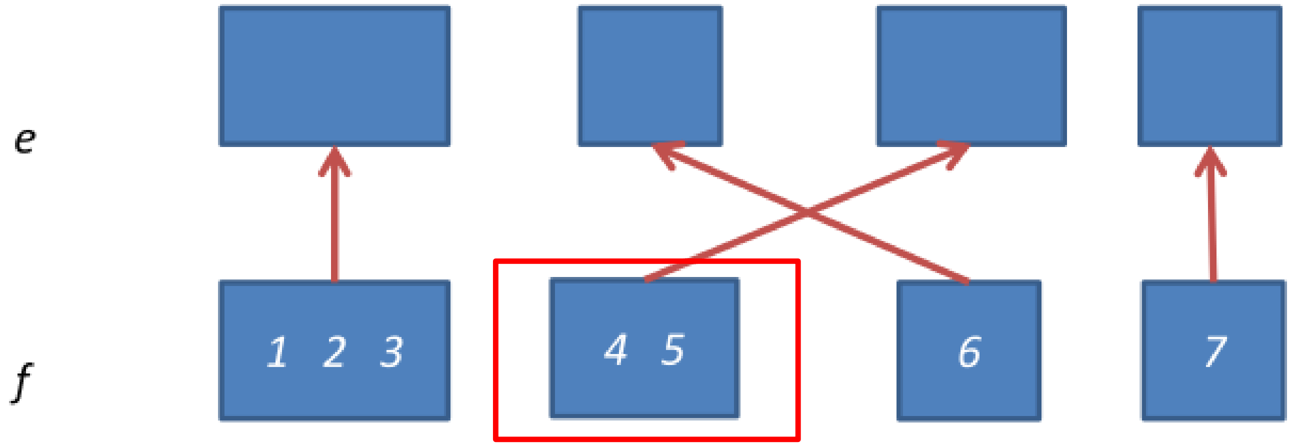
What are the distances associated with each of the English phrases?



i	f_i	$start_i - end_{i-1} - 1$	Distance
1	1-3	1-0-1	0
2	6	?	?
3	4-5	?	?
4	7	?	?

Distance based Reordering

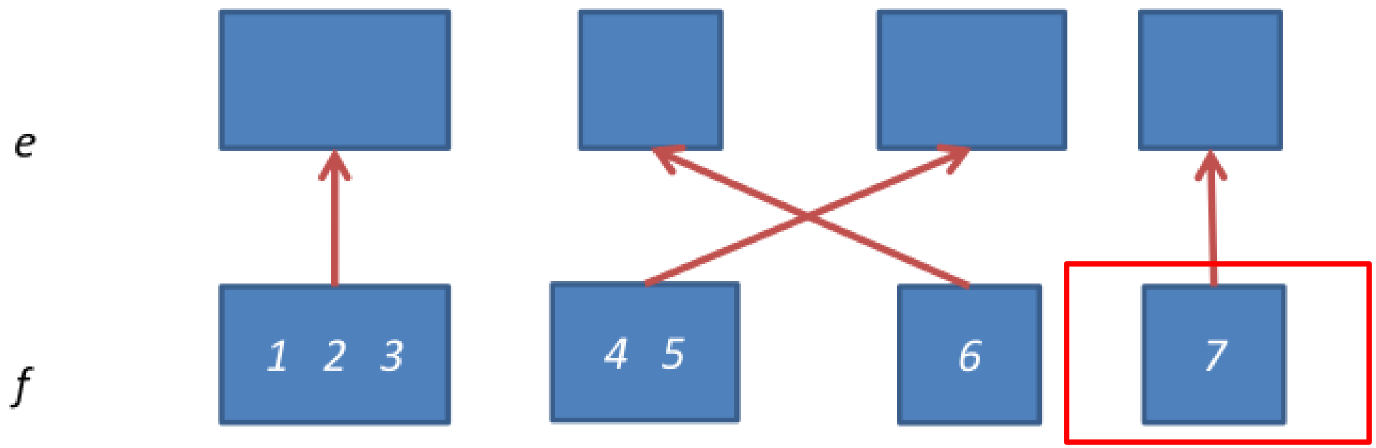
What are the distances associated with each of the English phrases?



i	f_i	$start_i - end_{i-1} - 1$	Distance
1	1-3	1-0-1	0
2	6	6-3-1	2
3	4-5	?	?
4	7	?	?

Distance based Reordering

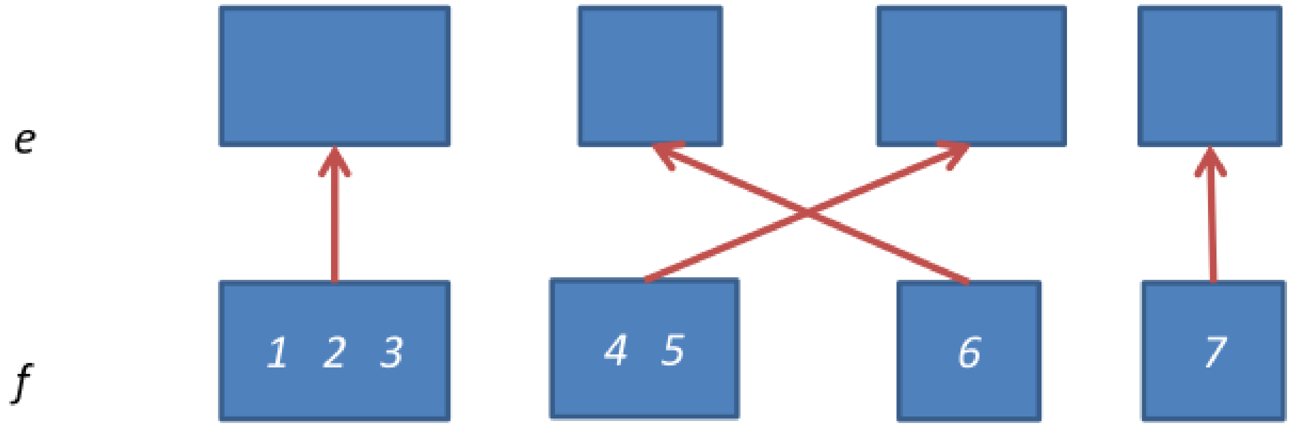
What are the distances associated with each of the English phrases?



i	f_i	$start_i - end_{i-1} - 1$	Distance
1	1-3	1-0-1	0
2	6	6-3-1	2
3	4-5	4-6-1	-3
4	7	?	?

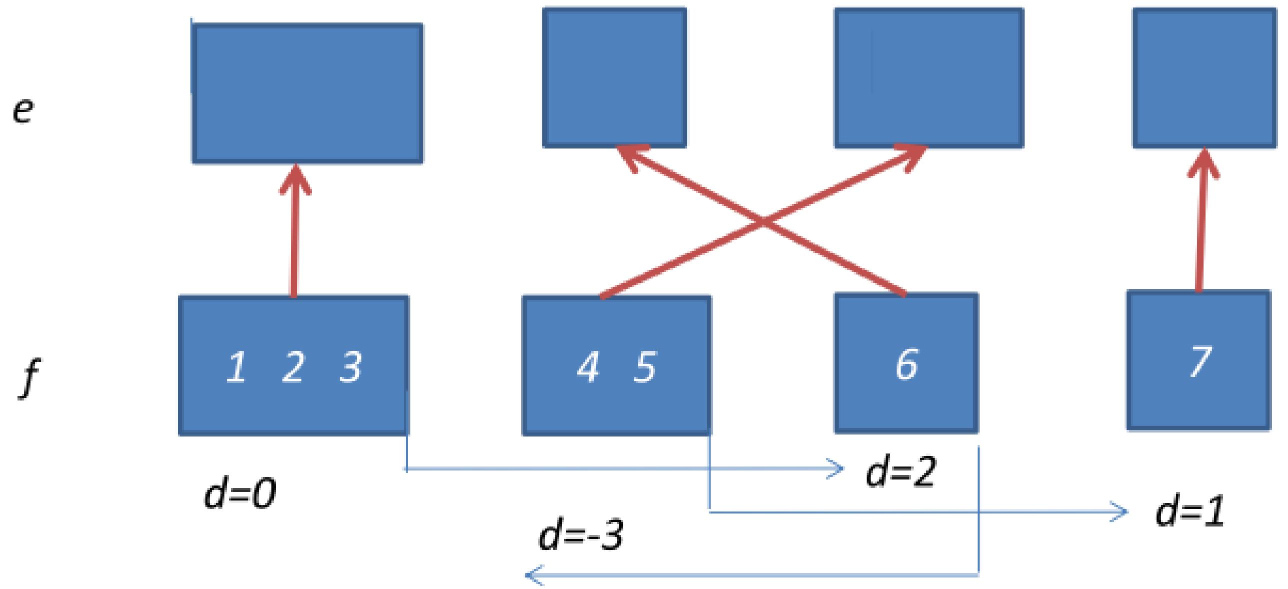
Distance based Reordering

What are the distances associated with each of the English phrases?



i	f_i	$start_i - end_{i-1} - 1$	Distance
1	1-3	1-0-1	0
2	6	6-3-1	2
3	4-5	4-6-1	-3
4	7	7-5-1	1

Distance based Reordering



The distance-based reordering function d is defined so that it penalises reordering over long distances.

Content

- Phrase-based Translation Model
- Distance-based Reordering
- **Log-linear Model**
- Decoding
- Make Decoding Manageable
- Exercises

Phrase-based Model: Extensions

$$p(e|f) = p(e)p(f|e)$$

Rewrite the Phrase-based Model

$$p(e|f) = \prod_{i=1}^{I} \phi(f_i|e_i)\, d(start_i - end_{i-1} - 1) \prod_{i=1}^{|e|} p_{LM}(e_i|e_1 \ldots e_{i-1})$$

3 sub-models: ϕ, d, p_{LM}

Phrase-based Model: Extensions

$$p(e|f) = \prod_{i=1}^{I} \phi(f_i|e_i)\, d(start_i - end_{i-1} - 1) \prod_{i=1}^{|e|} p_{LM}(e_i|e_1 \ldots e_{i-1})$$

We may want to emphasise one sub-model over another.

That is, they have different contribution to the translation.

Phrase-based Model: Extensions

- We can introduce **weights** $\lambda\varphi$, λd, λ_{LM} weighting of components that let us scale the contributions of each of the three components.

- Add weights to yield:

$$\prod_{i=1}^{I} \phi(f_i|e_i)^{\lambda_\phi}\, d(start_i - end_{i-1} - 1)^{\lambda_d} p_{LM}(e_i|e_1 \dots e_{i-1})^{\lambda_{p_{LM}}}$$

Phrase-based Model: Extensions

- Now we have:

$$p(e|f) = \prod_{i=1}^{I} \phi(f_i|e_i)^{\lambda_\phi}\, d(start_i - end_{i-1} - 1)^{\lambda_d} p_{LM}(e_i|e_1 \ldots e_{i-1})^{\lambda_{p_{LM}}}$$

Phrase-based Model: Extensions

- If we take logarithm for both sides:

$$\log p(e|f)$$
$$= \log \prod_{i=1}^{I} \phi(f_i|e_i)^{\lambda_\phi} d(start_i - end_{i-1} - 1)^{\lambda_d} p_{LM}(e_i|e_1 \dots e_{i-1})^{\lambda_{p_{LM}}}$$
$$= \lambda_\phi \log \prod_{i=1}^{I} \phi(f_i|e_i) \quad + \lambda_d \log \prod_{i=1}^{I} d(start_i - end_{i-1} - 1)$$
$$+\lambda_{p_{LM}} \log \prod_{i=1}^{I} p_{LM}(e_i|e_1 \dots e_{i-1})$$
$$= \lambda_\phi \sum_{i=1}^{I} \log \phi(f_i|e_i) + \lambda_d \sum_{i=1}^{I} \log d(start_i - end_{i-1} - 1)$$
$$+\lambda_{p_{LM}} \sum_{i=1}^{I} \log p_{LM}(e_i|e_1 \dots e_{i-1})$$

Phrase-based Model: Extensions

- If we take exponential for both sides:

$$p(e|f) = exp^{\log p(e|f)}$$

$$= \exp(\lambda_\phi \sum_{i=1}^{I} \log \phi(f_i|e_i) + \lambda_d \sum_{i=1}^{I} \log d(start_i - end_{i-1} - 1)$$

$$+\lambda_{p_{LM}} \sum_{i=1}^{I} \log p_{LM}(e_i|e_1 \dots e_{i-1}))$$

$$= \mathbf{\exp}(\sum_{i=1}^{n} \boldsymbol{\lambda_i h_i(e, f)})$$

Log-linear Model

✓ What we end up with is a log-linear model:

$$p(x) = \exp(\sum_{i=1}^{n} \lambda_i h_i(x))$$

where:

- $n = 3$
- $h_1(x) = \log \phi$
- $h_2(x) = \log d$
- $h_3(x) = \log p_{LM}$

Log-linear Model

- What is the advantage of reformulating the translation formula in this way?

Log-linear Model

- What is the advantage of reformulating the translation formula in this way?
- It makes it easier to add in more information sources:

Log-linear Model

- What is the advantage of reformulating the translation formula in this way?
- It makes it easier to add in more information sources:
  - Multiple translation models
  - Multiple language models
  - Linguistic information
  - Lexical probabilities as well as phrase probabilities

Log-linear Model

- Features used in Moses:
  - Bidirectional phrase translation model $\phi(e|f)$, $\phi(f|e)$
  - Bidirectional lexical weighting model $lex(e|f)$, $lex(f|e)$
  - Language model p_{LM}
  - Lexical reordering model $p_o(orientation|f\ , e)$
  - Phrase penalty ρ
  - Word penalty ω

Content

- Phrase-based Translation Model
- Distance-based Reordering
- Log-linear Model
- **Decoding**
- Make Decoding Manageable
- Exercises

Noisy Channel Model Revisited

$$\hat{e} = \underbrace{\operatorname*{argmax}_{e}}\,\underbrace{p(e)}\,\underbrace{p(f|e)}$$

Translation Model

Decoder

Language Model

What is Decoding?

- Process of searching for the best translation among all possible translations:

$$e_{best} = argmax\ _{e} p(e|f)$$

What is Decoding?

$$e_{best} = argmax\ _{e} p(e|f)$$

- Two types of error:
  - the most probable translation is bad - fix the model
  - search does not find the most probably translation - fix the search
- Decoding is evaluated by search error, not quality of translations (although these are often correlated)

Decoding Process

Maria no dio una bofetada a la bruja verde

Build translation left to right:

Select phrase to be translated

Decoding Process

Build translation left to right

- Select phrase to be translated
- Find phrase translation

Decoding Process

Build translation left to right

- Select phrase to be translated
- Find phrase translation
- Add phrase to end of partial translation

Decoding Process

Maria *no* *dio* *una* *bofetada* *a* *la* *bruja* *verde*

Mary

Build translation left to right

- Select phrase to be translated
- Find phrase translation
- Add phrase to end of partial translation
- Mark words as translated

Decoding Process

Maria no dio una bofetada a la bruja verde

Mary did not

One to many translation

Decoding Process

Many to one translation

Decoding Process

Many to one translation

Decoding Process

Reordering

Decoding Process

Maria no dio una bofetada a la bruja verde

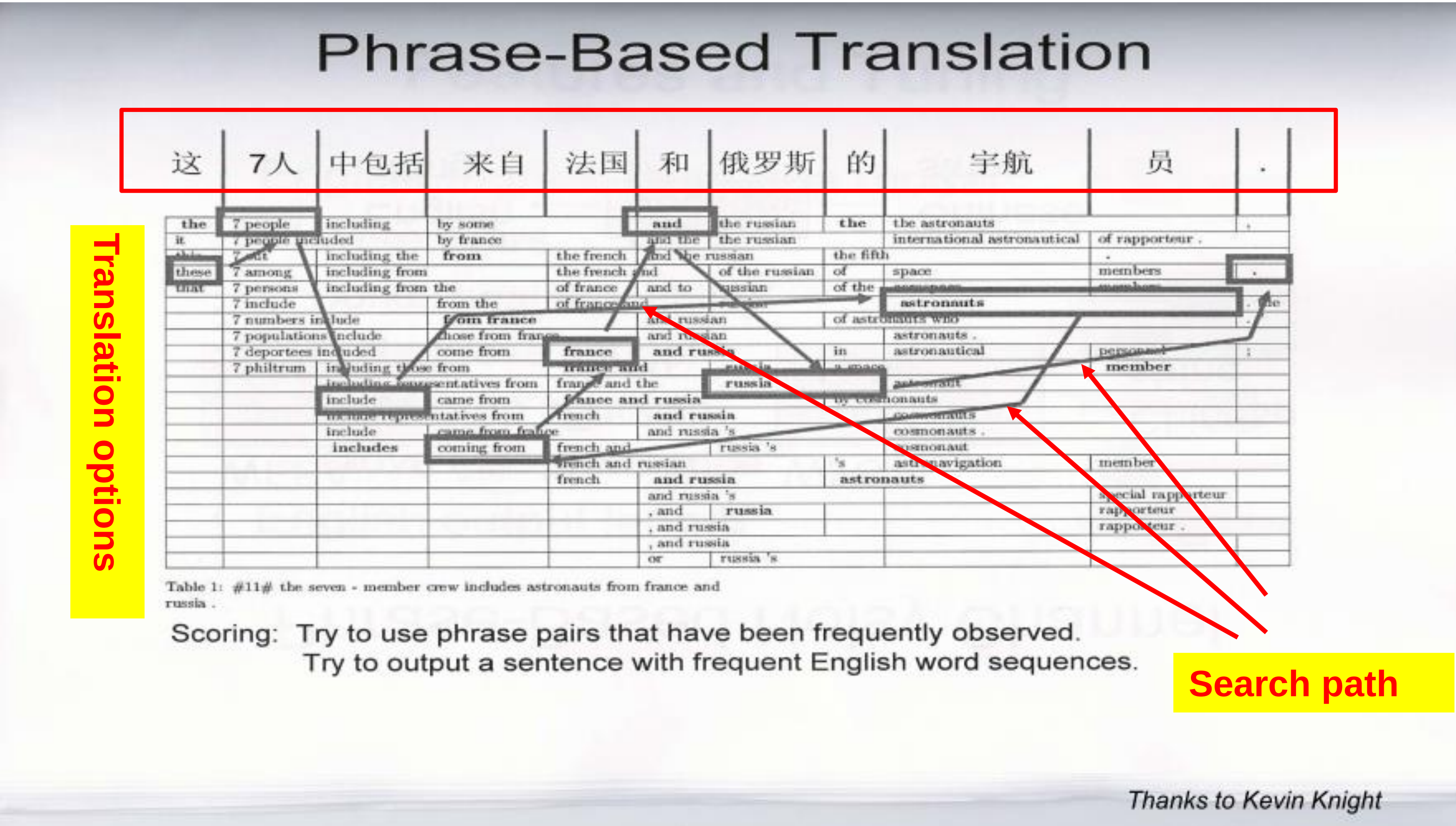
Mary did not slap the green witch

Translation finished!

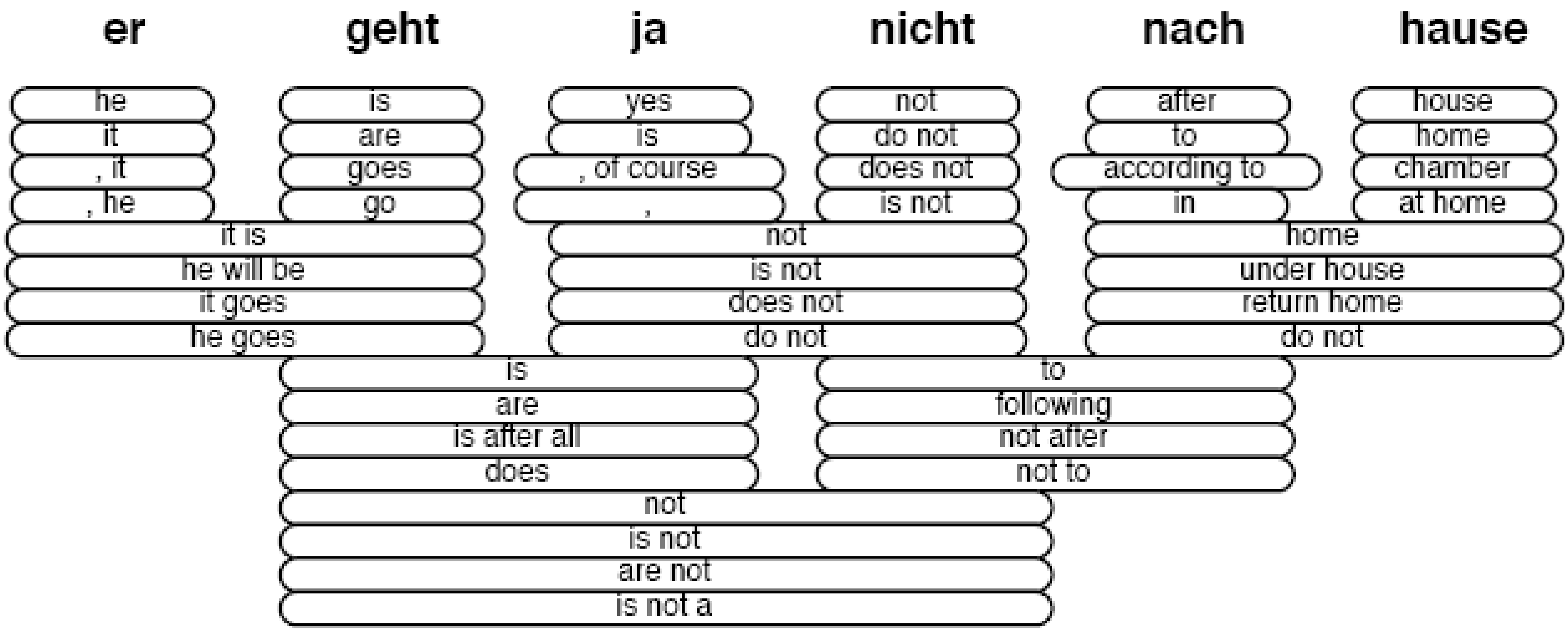
Translation Options

- Many different ways to segment words into phrases
- Many different ways to translate each phrase

Decoding is a Complex Process!

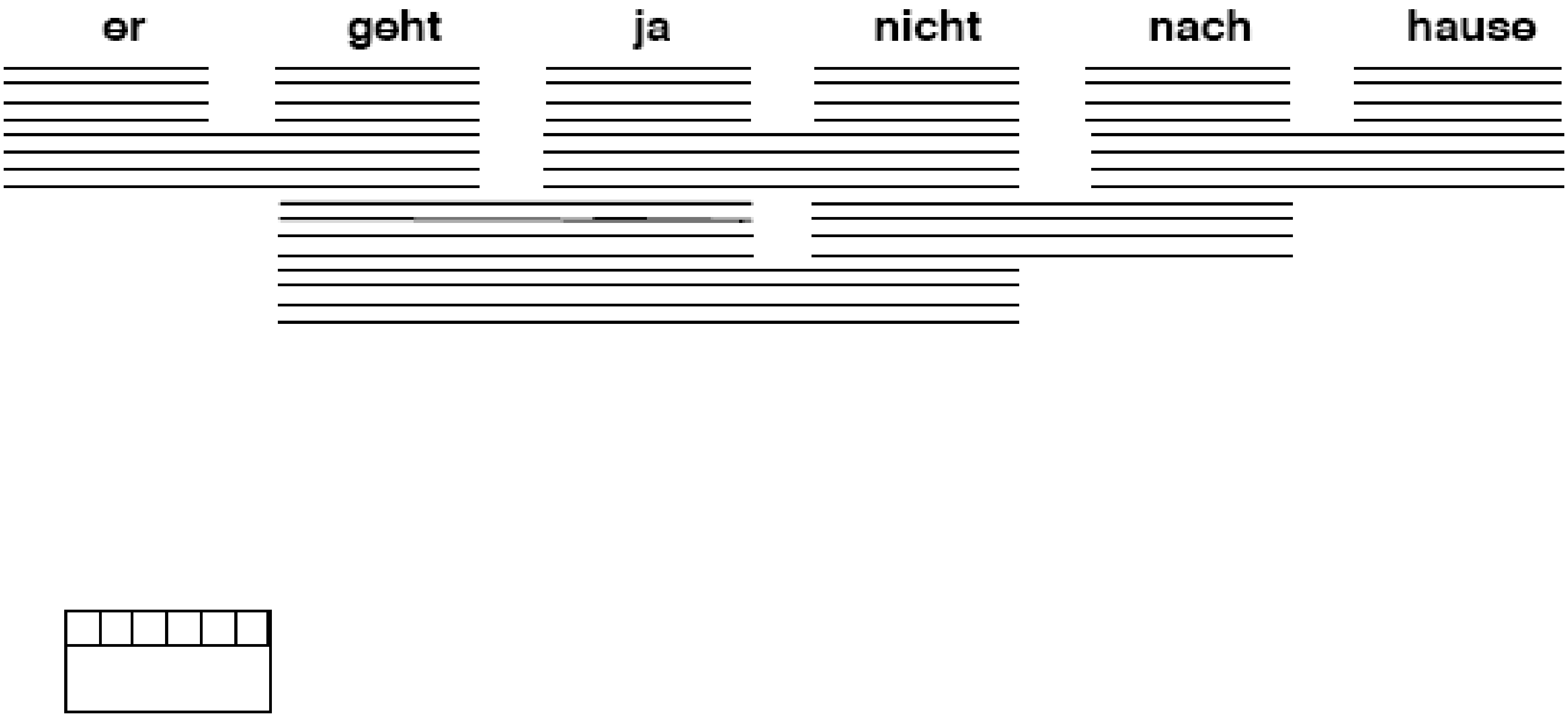


Translation Options



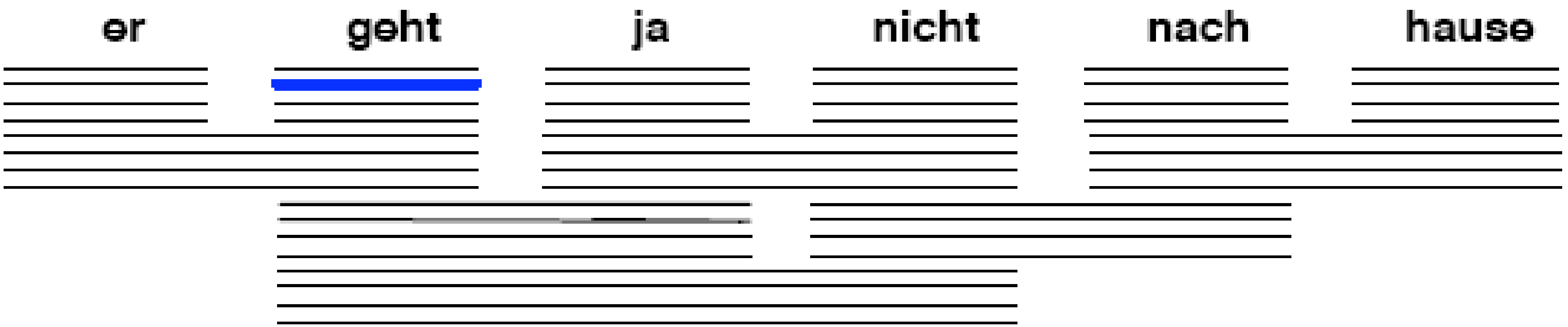
- in Europarl phrase table: 2727 matching phrase pairs for this sentence
- by pruning to the top 20 per phrase, 202 translation options remain

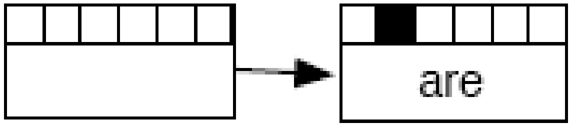
Decoding: Start with Initial Hypothesis



initial hypothesis: no input words covered, no output produced

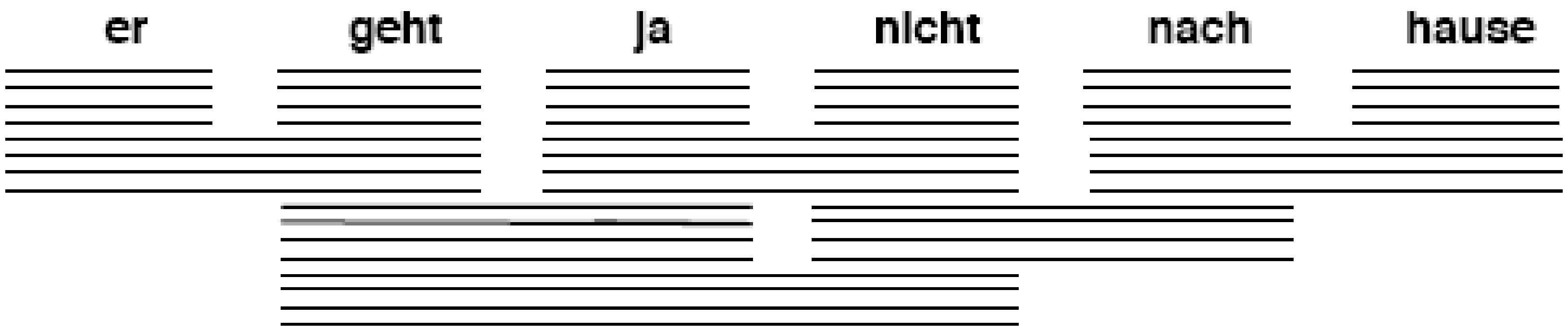
Decoding: Hypothesis Expansion

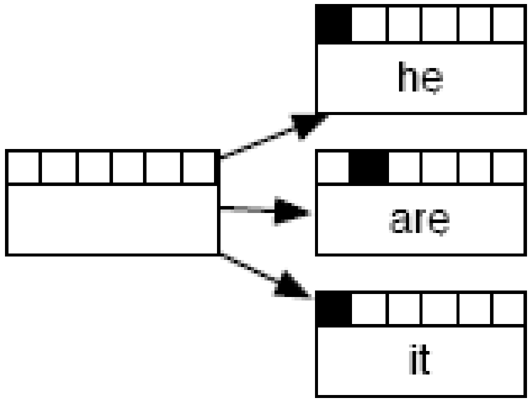




pick any translation option, create new hypothesis

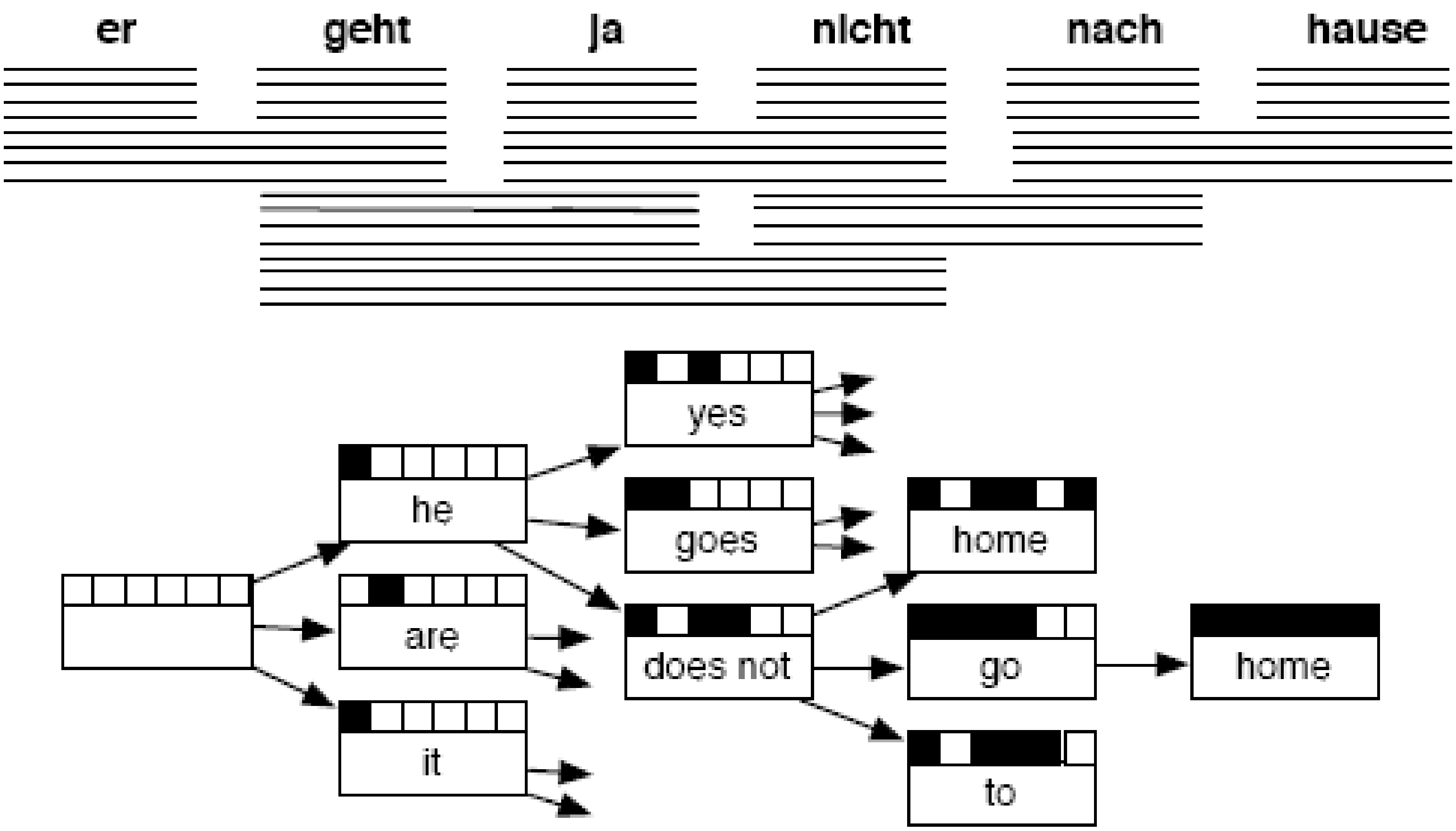
Decoding: Hypothesis Expansion





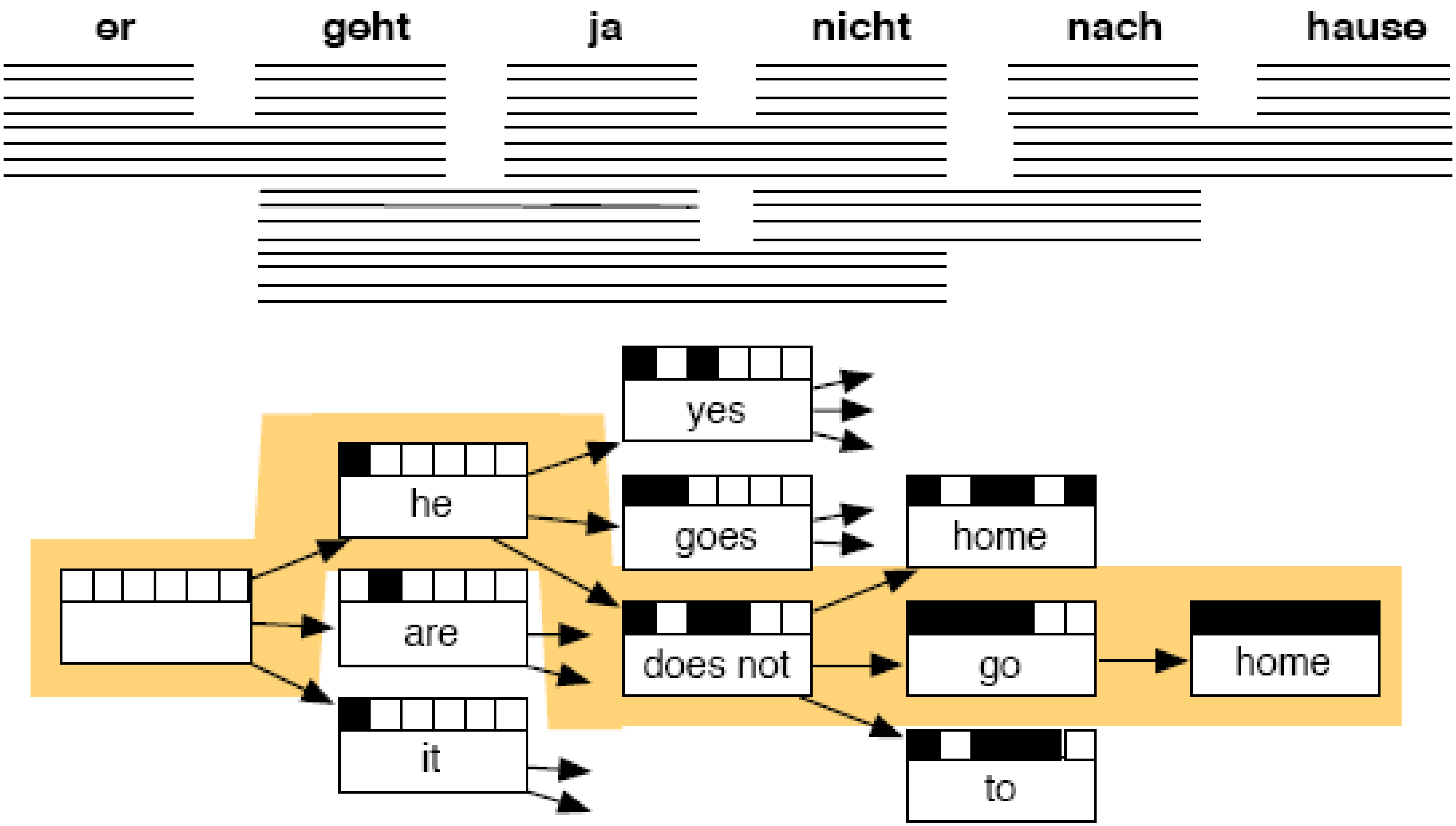
create hypotheses for all other translation options

Decoding: Hypothesis Expansion



also create hypotheses from created partial hypothesis

Decoding: Find Best Path



backtrack from highest scoring complete hypothesis

Content

- Phrase-based Translation Model
- Distance-based Reordering
- Log-linear Model
- Decoding
- **Make Decoding Manageable**
- Exercises

Making decoding manageable

- The decoding problem is NP-complete which means that exhaustively examining all possible translations, scoring them and picking the best is computationally too expensive for an input sentence of even modest length (Koehn, 2010, p.155).

Making decoding manageable

Any idea?

Making decoding manageable

Two strategies:

1. Hypothesis Recombination (risk-free)
2. Pruning the search space (risky)

Hypothesis Recombination

- A translation *hypothesis* is a partial translation.

Hypothesis Recombination

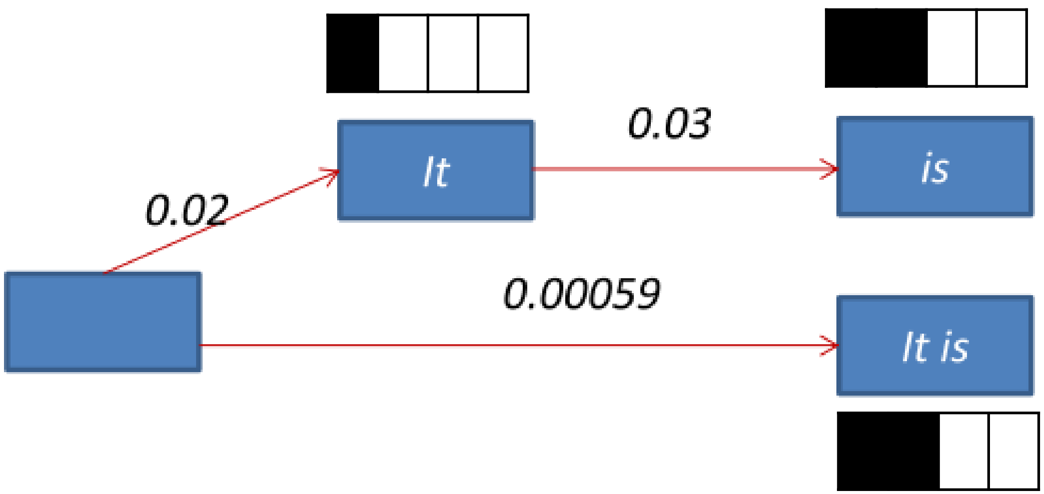
- A translation *hypothesis* is a partial translation.
- We can arrive at the same partial translation in more than one way.

Hypothesis Recombination

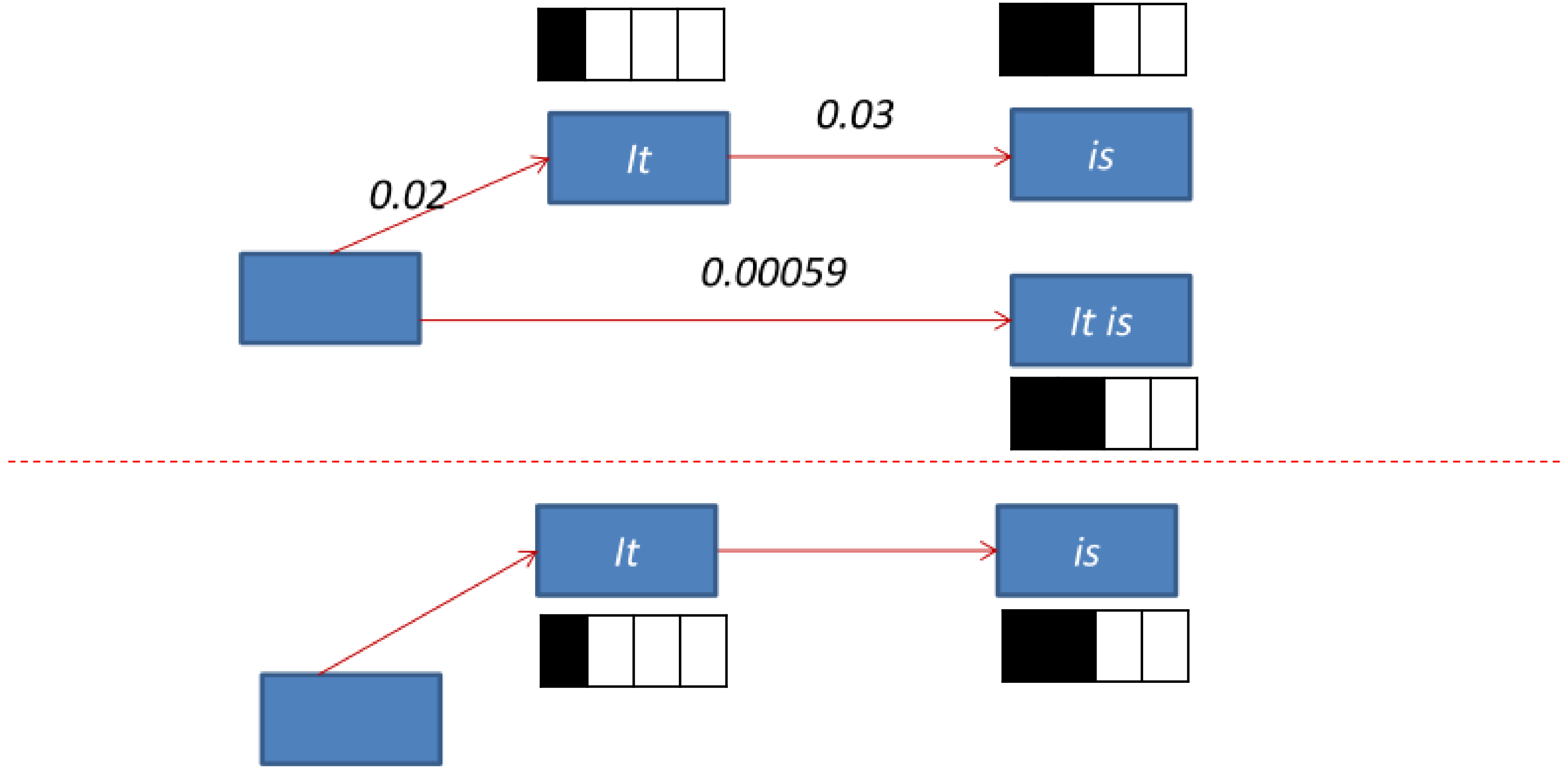
- A translation *hypothesis* is a partial translation.
- We can arrive at the same partial translation in more than one way.
- Hypothesis recombination takes advantage of this by storing only the most likely path associated with a particular hypothesis.

Hypothesis Recombination

- Two hypothesis paths lead to two matching hypotheses
  - ✓ same number of foreign words translated
  - ✓ same English words in the output
  - ✓ different scores



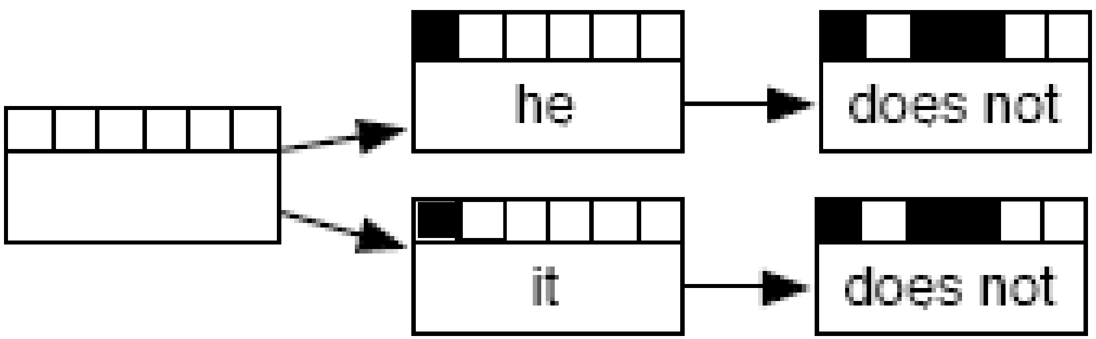
Hypothesis Recombination



Worse hypothesis is dropped

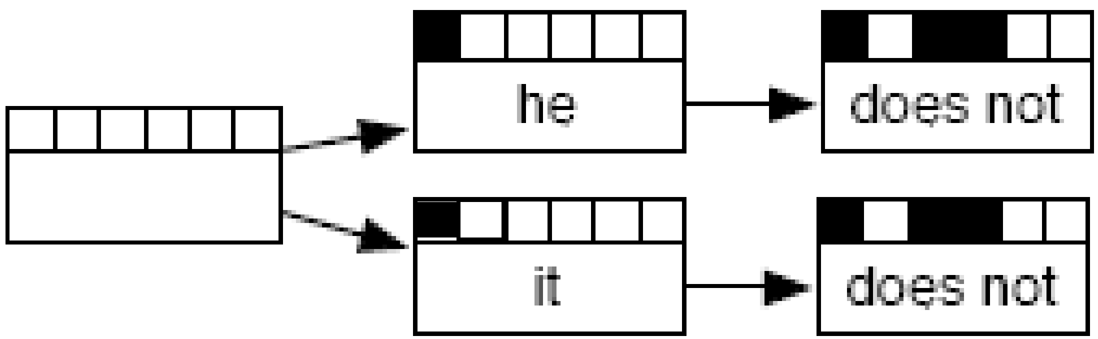
Advanced Hypothesis Recombination

- Two hypothesis paths lead to hypotheses indistinguishable in subsequent search
  - ✓ same number of foreign words translated
  - ✓ same last two English words in output (assuming trigram language model)
  - ✓ same last foreign word translated
  - ✓ different scores

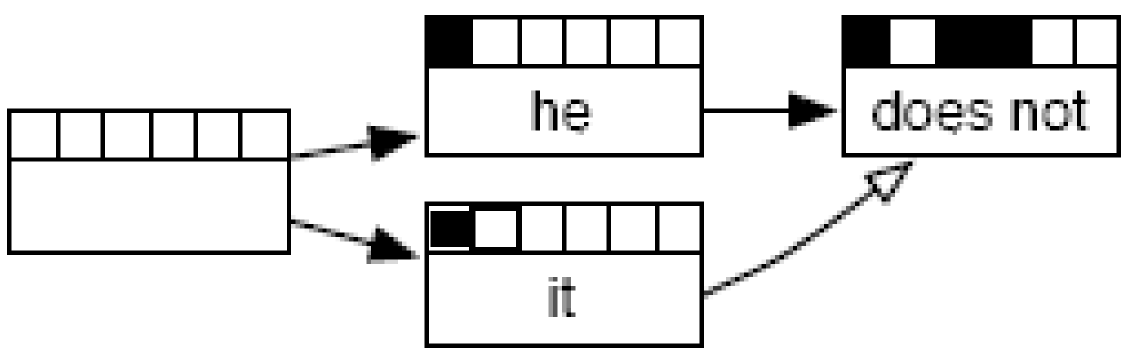


Advanced Hypothesis Recombination

- Two hypothesis paths lead to hypotheses indistinguishable in subsequent search



- Worse hypothesis is dropped



Pruning the search space

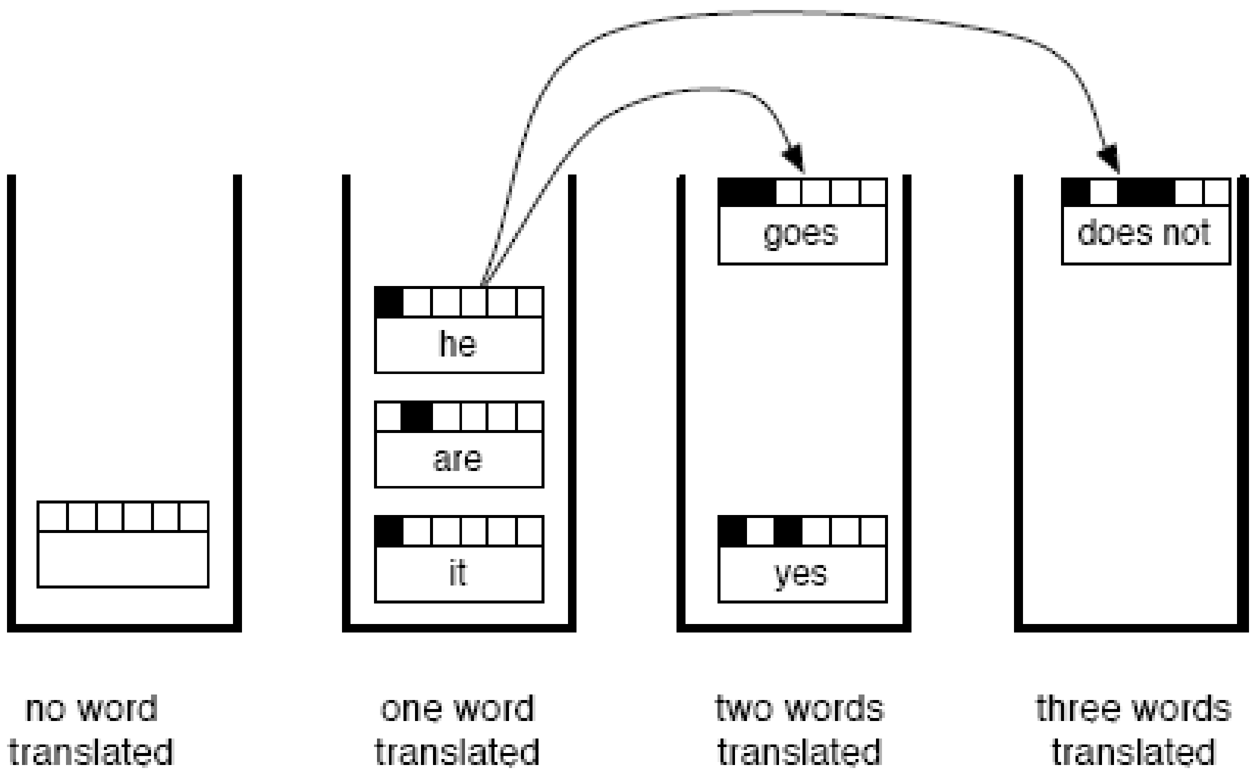
- Recombination reduces search space, but not enough (we still have a NP complete problem on our hands)
- Pruning: remove bad hypotheses early
  - put comparable hypothesis into stacks (hypotheses that have translated same number of input words)
  - limit number of hypotheses in each stack

Pruning the search space

- Pruning is the process of deleting unlikely hypotheses to reduce the search space.
- One way to do this is to store hypotheses in stacks based on the number of words translated.
- Unlikely hypotheses can be pruned from each stack.

Pruning the search space: Stack

- Hypothesis expansion in a stack decoder
  – translation option is applied to hypothesis
  – new hypothesis is dropped into a stack further down



Stack Decoding Algorithm

```
place empty hypothesis into stack 0
for all stacks 0...n − 1 do
    for all hypotheses in stack do
        for all translation options do
            if applicable then
                create new hypothesis
                place in stack
                recombine with existing hypothesis if possible
                prune stack if too big
            end if
        end for
    end for
end for
```

Pruning the search space

Two types of pruning strategies

1. Histogram pruning: keep a maximum of *m* hypotheses in a stack

Pruning the search space

Two types of pruning strategies

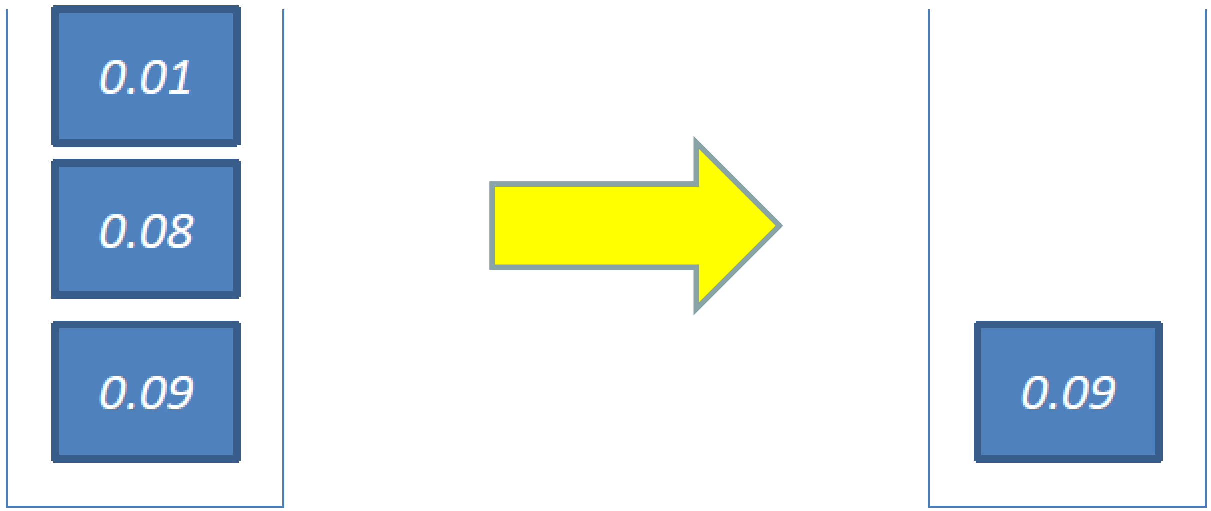
1. Histogram pruning: keep a maximum of m hypotheses in a stack
2. Threshold or beam pruning: keep only those hypotheses that are within a threshold α of the best hypothesis ($\alpha \times$ best_score ($\alpha < 1$)). Any hypothesis that is α times worse than the best hypothesis is pruned.

Histogram Pruning

- Keep a maximum of *m* hypotheses

Histogram Pruning

- How many hypotheses will be pruned if $m = 1$?

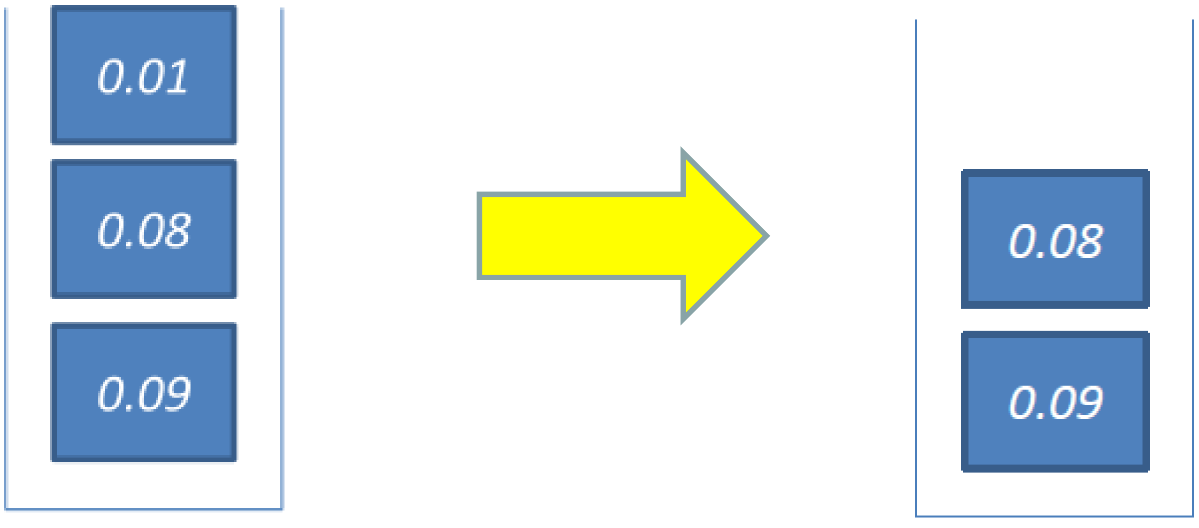


Threshold Pruning

- Keep those hypotheses that only are within a threshold α of the best hypothesis.

Threshold Pruning

- How many hypotheses will be pruned if we prune all hypotheses that are at least *0.5* times worse than the best hypothesis?



Threshold Pruning

- Threshold pruning is more flexible than histogram pruning since it takes into account the difference between the scores of the best and worst hypothesis.

Example

Given a partial phrase table:

ta	*he*	0.4

xihuan	*likes*	0.4
xihuan	*likes to*	0.6

youyong	*swimming*	0.2
youyong	*swim*	0.8

ta xihuan	*he likes*	0.2
ta xihuan	*he likes to*	0.8

xihuan youyong	*likes swimming*	0.3
xihuan youyong	*likes to swim*	0.7

Considering we have the following input sentence:

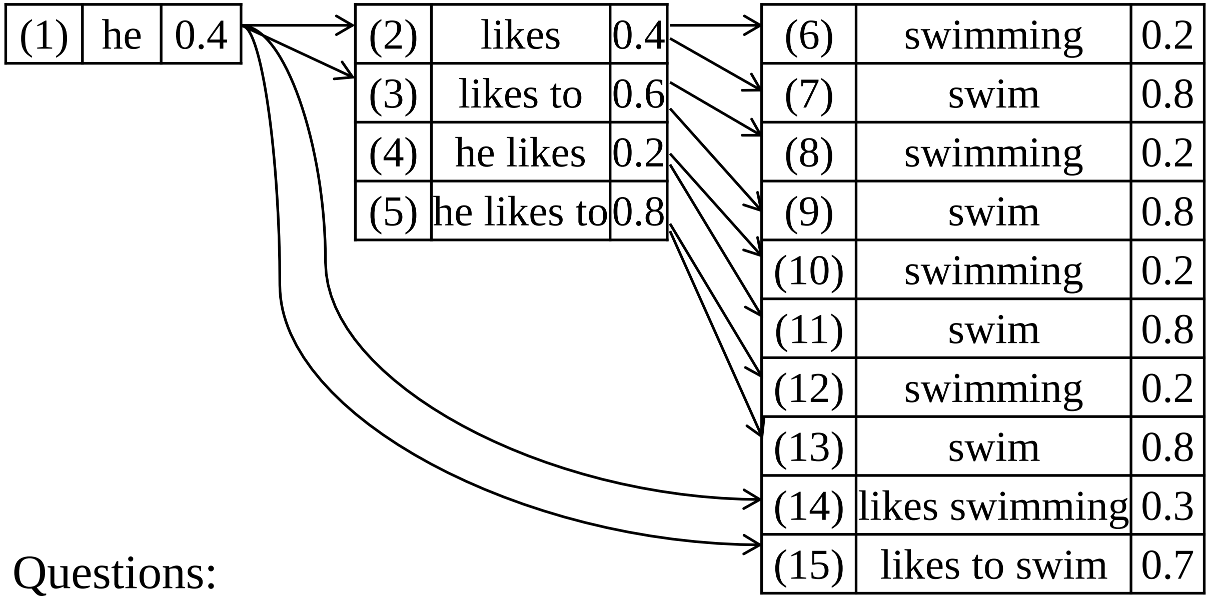
ta xihuan youyong

Assuming that:

- Only monotone word order is permitted;
- Language model is ignored.

Then we have the following searching diagram:

Example



Questions:

1) Calculate all possible hypotheses, and indicate which hypothesis provides the optimal translation for the input sentence.
2) Give all groups of hypotheses which can be recombined and indicate which hypothesis should be selected to represent each group;
3) Assume histogram pruning after recombination, where the maximum number of hypotheses in each stack is 2. Indicate which hypotheses will be pruned;
4) Assuming threshold pruning after recombination, where the threshold is 0.5, please indicate which hypotheses will be pruned.

Solution

Q1:

(1) he, 0.4
(2) he likes, 0.16
(3) he likes to, 0.24
(4) he likes, 0.2
(5) he likes to, 0.8
(6) he likes swimming, 0.032
(7) he likes swim, 0.128
(8) he likes to swimming, 0.048
(9) he likes to swim, 0.192
(10)he likes swimming, 0.04
(11)he likes swim, 0.16
(12)he likes to swimming, 0.16
(13)he likes to swim, 0.64
(14)he likes swimming, 0.12
(15)he likes to swim, 0.28

Solution

Q2:

- { (2), (4) } => (4), he likes, 0.2
- { (3), (5) } => (5), he likes to, 0.8
- { (6), (10), (14) } => (14), he likes swimming, 0.12
- { (7), (11) } => (11), he likes swim, 0.16
- { (8), (12) } => (12), he likes to swimming, 0.16
- { (9), (13), (15) } => (13), he likes to swim, 0.64

Solution

Q2:

- { (2), (4) } => (4), he likes, 0.2
- { (3), (5) } => (5), he likes to, 0.8
- { (6), (10), (14) } => (14), he likes swimming, 0.12
- { (7), (11)) } => (11), he likes swim, 0.16
- { (8), (12) } => (12), he likes to swimming, 0.16
- { (9), (13), (15)} => (13), he likes to swim, 0.64

Q3:

After recombination, we have only 11, 12, 13, 14 in the last stack:

- {(11) and (14)} or {(12) and (14)} will be pruned.

Q4:

After recombination, we have only 11, 12, 13, 14 in the last stack:

Pruning value = 0.64 * 0.5 = 0.32

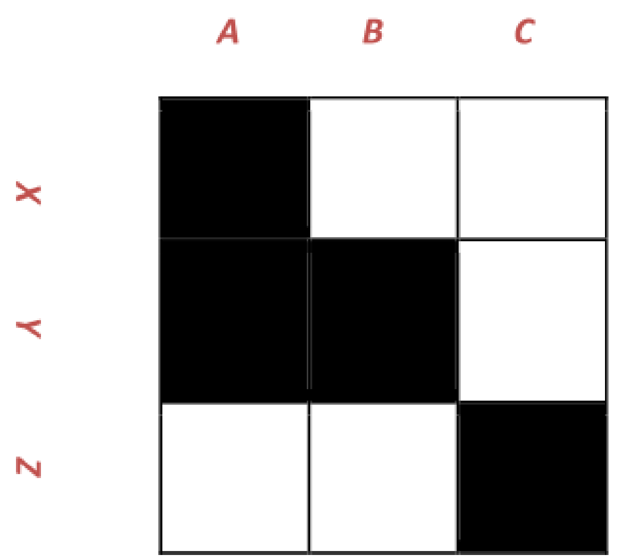
- (12), (14) and (11) will be pruned.

Content

- Phrase-based Translation Model
- Distance-based Reordering
- Log-linear Model
- Decoding
- Make Decoding Manageable
- **Exercises**

Exercise 1

List all phrase pairs that are consistent with the following word alignment:

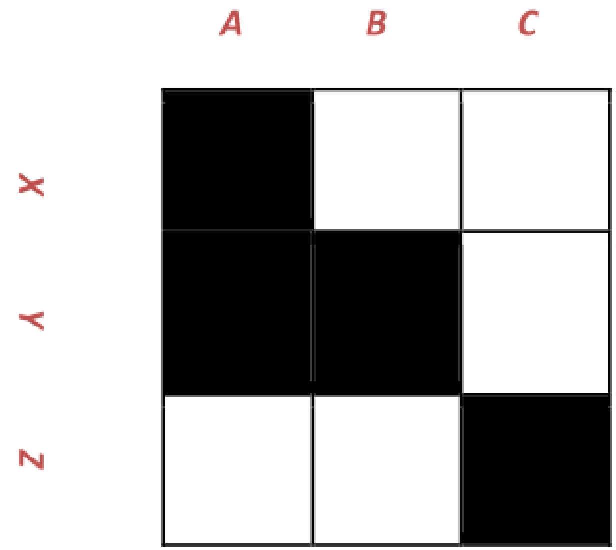


Solution 1

X Y| A B

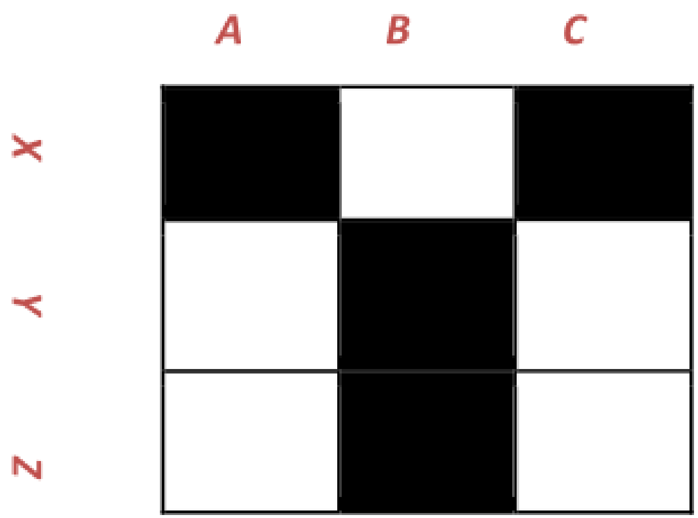
X Y Z | A B C

Z | C



Exercise 2

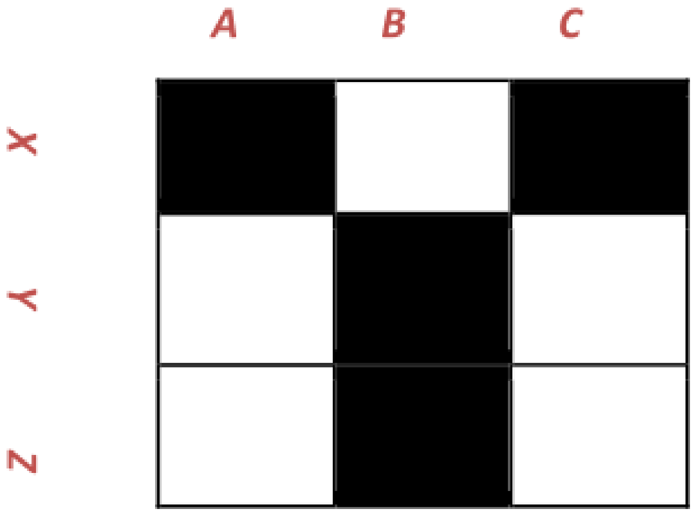
List all phrase pairs that are consistent with the following word alignment:



Solution 2

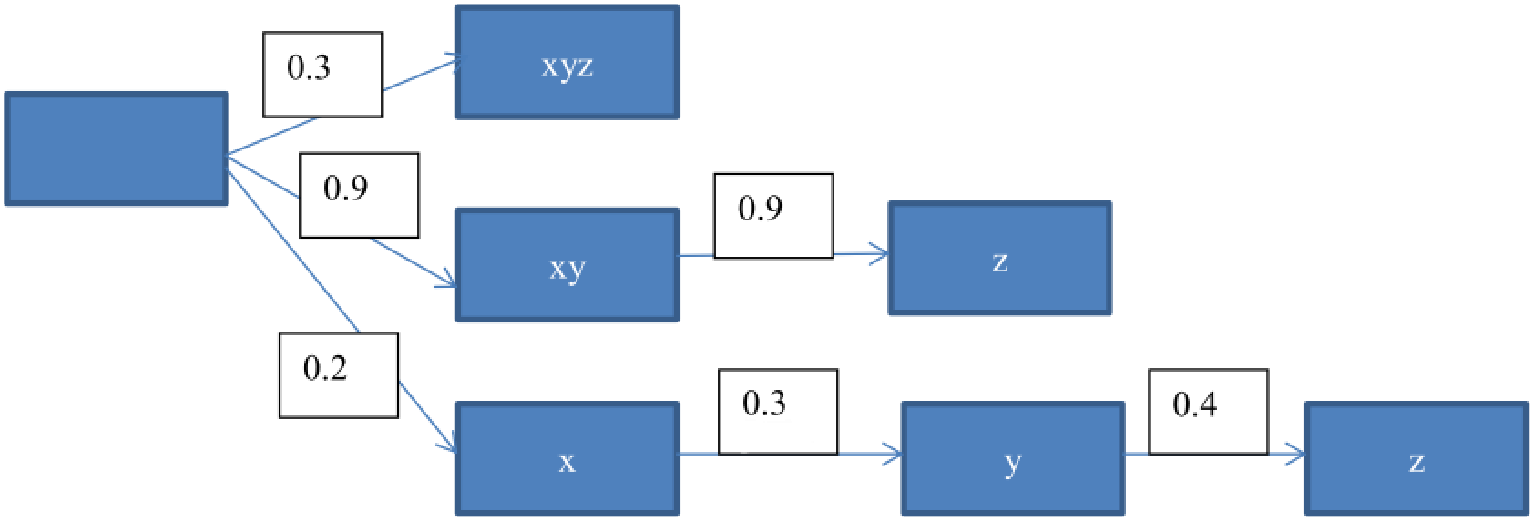
X Y Z | A B C

Y Z | B

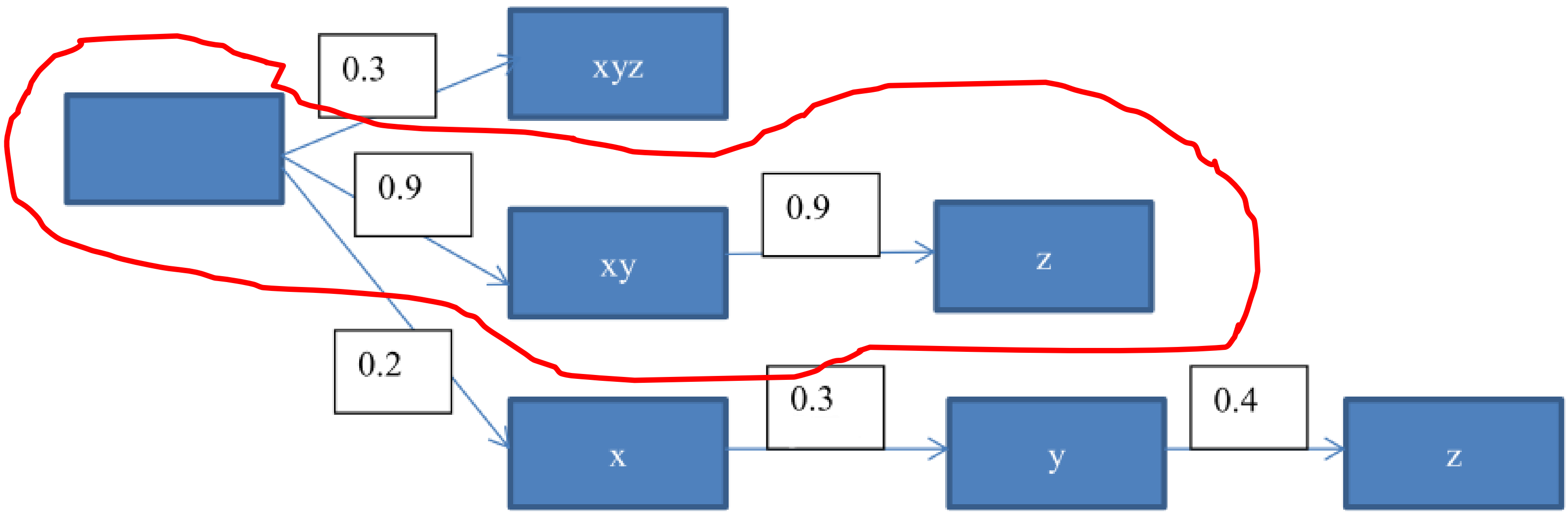


Exercise 3

Which hypothesis will remain after hypothesis recombination?



Solution 3



After recombination:

Discussion