UPPSALA UNIVERSITET

Machine Translation
Tuning and factored translation

Sara Stymne
Uppsala University

Slides mainly from Philipp Koehn and Jörg Tiedemann

UPPSALA
UNIVERSITET

Tuning

Log-linear model

Weights in log-linear models, which is a weighted combination of many components

$$f(s,t) = \sum_i \lambda_i h_i(s,t)$$

$h_i(s,t)$ are feature functions such as

- translation model
- language model
- distortion model

λ_i are weights

- weights are used to tune the importance of each feature function

Feature weights

Contribution of feature h_k determined by weight λ_k

Methods for setting the feature weights:

- manually — try a few, take best
- automatically — tune with an optimization algorithm

How to learn weights

- set aside a development corpus
- set the weights, so that optimal translation performance on this development corpus is achieved
- requires automatic scoring method

Weight optimization

- Setting the feature weights is an optimization problem: $\Lambda_{best} = \text{argmax}_{\Lambda} G(E, T_{\Lambda}(F))$
- Find weight vector $\Lambda_{best} = (\lambda'_1 \cdot\cdot\cdot \lambda'_m)$ that maximizes some gain function G
- The gain function G compares a set of reference sentences E to a set of translated sentences $T_{\Lambda}(F)$
- Which gain function? Our evaluation metric (Bleu)!

UPPSALA UNIVERSITET

Discriminative vs Generative Models

Generative models

- translation process is broken down into steps
- each step is modeled by a probability distribution
- each probability distribution is estimated from the data by maximum likelihood

Discriminative models

- model consists of a number of features
- each feature has a weight, measuring its value for judging a translation as correct
- supervised learning: directly tune model parameters (feature weights)
  towards optimal performance wrt. the evaluation metric on development data

UPPSALA UNIVERSITET

Discriminative training (1)

Employ development corpus

- different from training corpus for phrase extraction
- small (maybe 2000 sentences)
- different from the held-out test set which is used to finally evaluate the translation quality

Translate development corpus using model with current feature weights,
output N -best list of translations (N = 100, 1000, . . .)

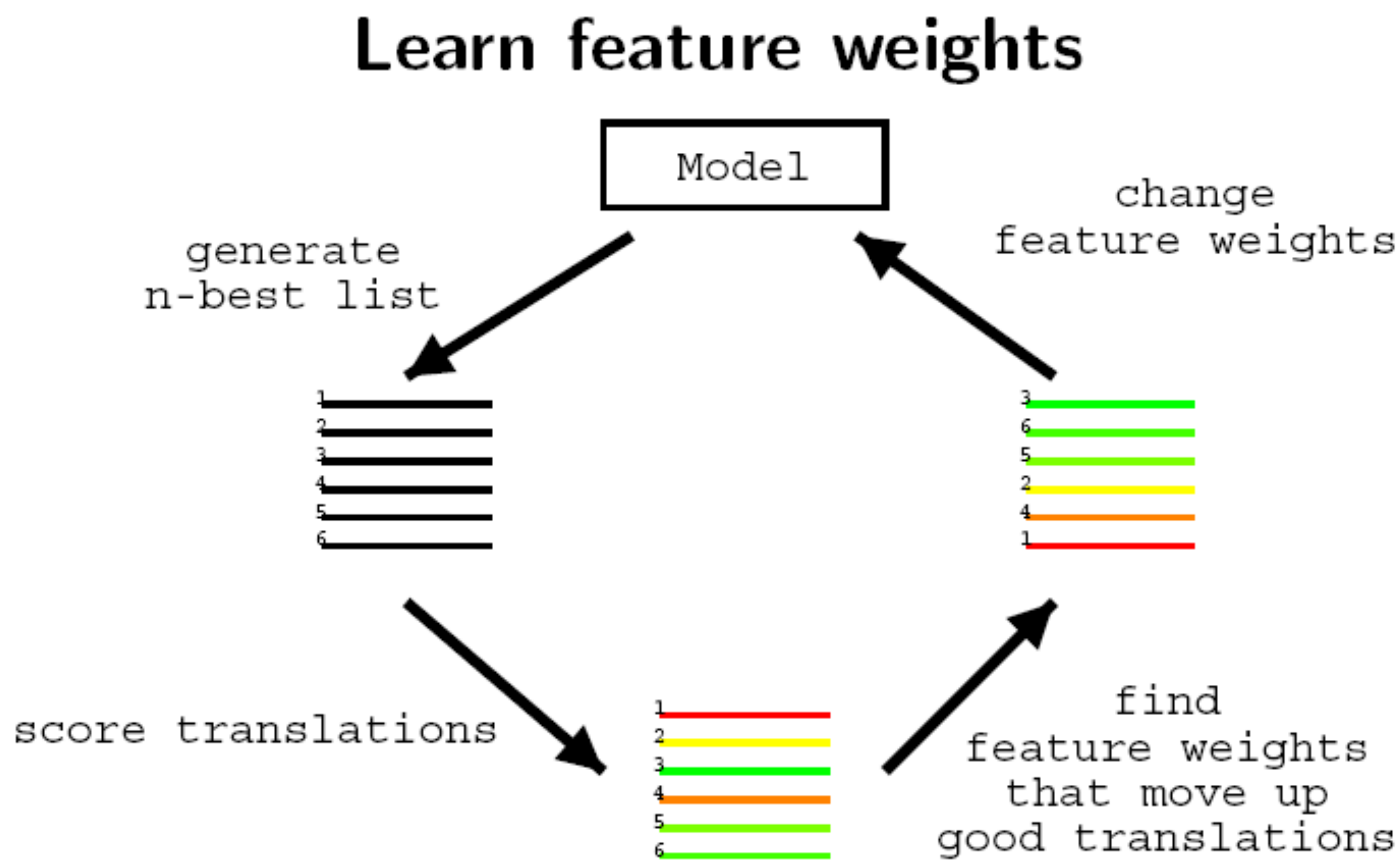
Evaluate translations with the gain function

Adjust feature weights to increase the gain

Iterate translation, evaluation, and adjustment of feature weights for a number of times

UPPSALA UNIVERSITET

Discriminative training (2)



UPPSALA UNIVERSITET

Optimizations on N-best lists (1)

- Task: find weights so that the model ranks best translations first
- Input: er geht ja nicht nach Hause, Ref: he does not go home

Translation	Feature 1	Feature 2	Model score	Gain
he is not go home	-0.5	−3	-0.7	0.3
it is not under house	−2	−2	-0.8	0.2
he does not go home	−4	-1.5	**-1.1**	**1.0**
it is not packing	−3	−3	-1.2	0.0
he is not for home	−5	−6	-2.2	0.2

$\lambda_1 = 0.2$, $\lambda_2 = 0.2$

Try to find values of weights so that the best hypothesis, in bold, is moved up according to model score

UPPSALA UNIVERSITET

Optimizations on N-best lists (2)

- Task: find weights so that the model ranks best translations first
- Input: er geht ja nicht nach Hause, Ref: he does not go home

Translation	Feature 1	Feature 2	Model score	Gain
he is not go home	-0.5	−3	−925	0.3
it is not under house	−2	−2	-0.7	0.2
he does not go home	−4	-1.5	**-0.65**	**1.0**
it is not packing	−3	−3	-1.05	0.0
he is not for home	−5	−6	-2.05	0.2

$\lambda_1 = 0.05$, $\lambda_2 = 0.3$

UPPSALA UNIVERSITET

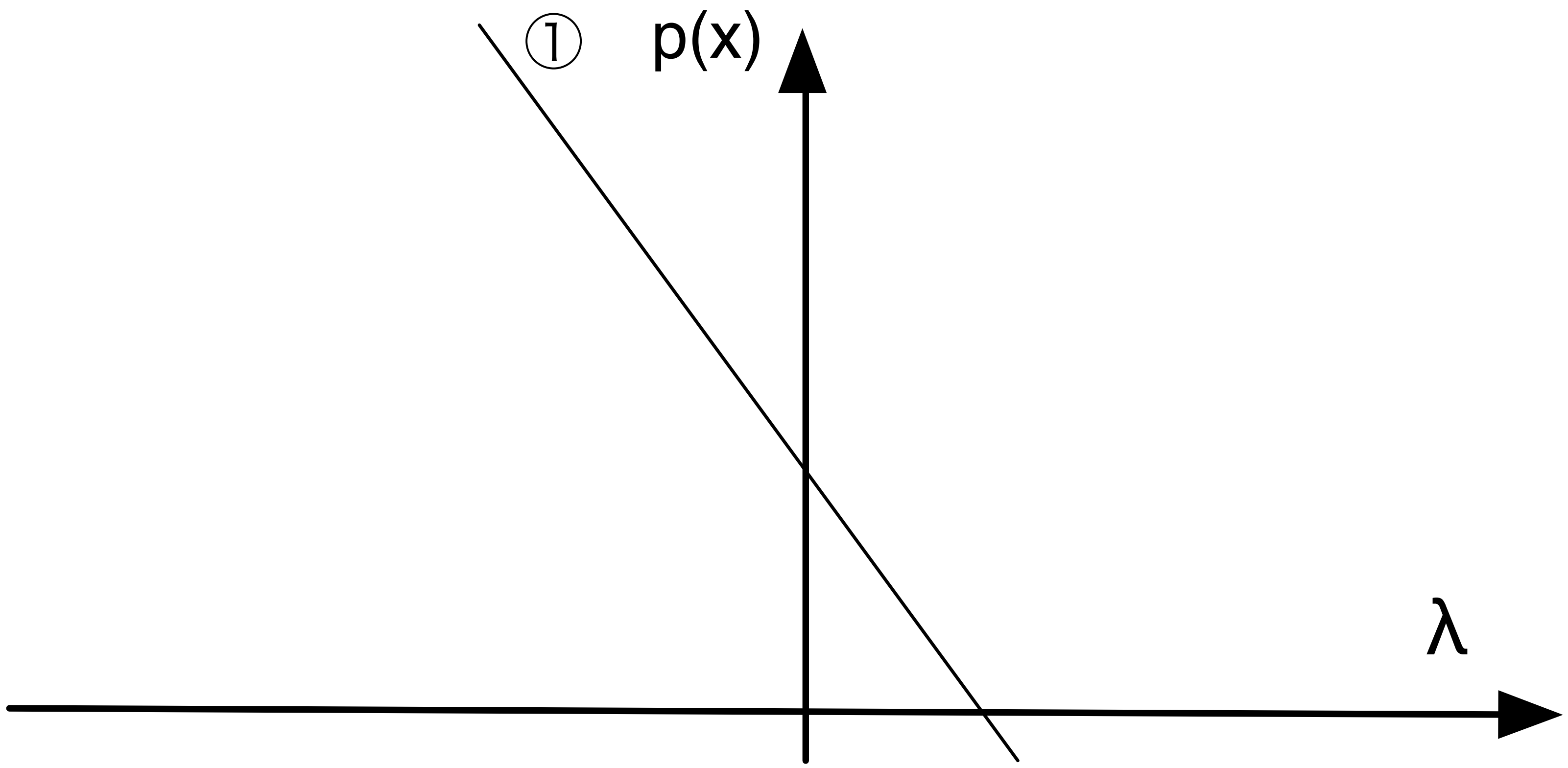
Minimum Error rate training

Line search for best feature weights

```
given: sentences with n-best lists of translations
iterate n times
   randomize starting feature weights
          for each feature
                 find best feature weight
                 update if different from current
return best feature weights found in any iteration
```

UPPSALA UNIVERSITET

One translation for one sentence

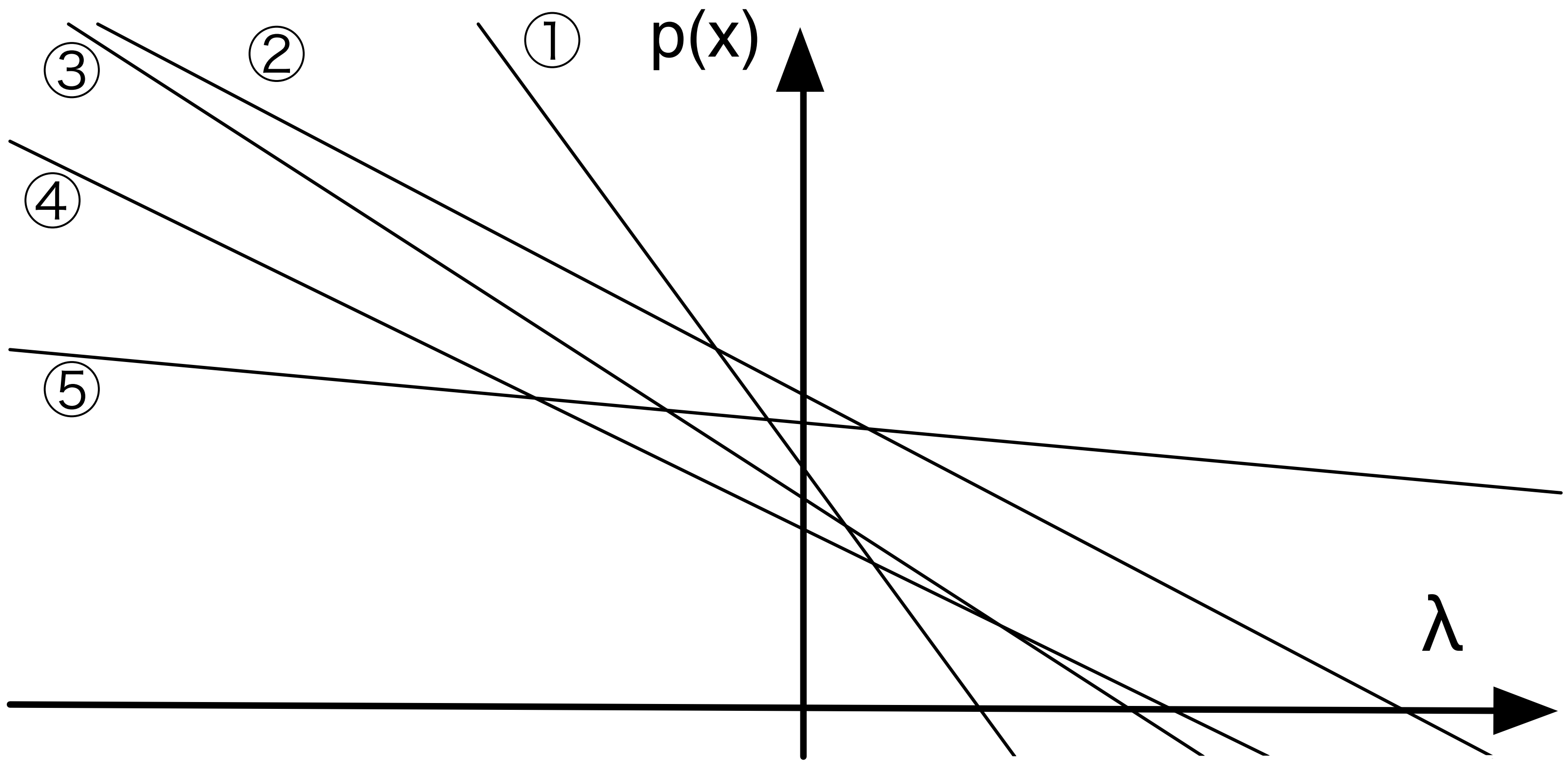


- Probability of one translation $p(\mathbf{e}_i|\mathbf{f})$ is a function of λ

$$p(\mathbf{e}_i|\mathbf{f}) = \lambda a_i + b_i$$

UPPSALA UNIVERSITET

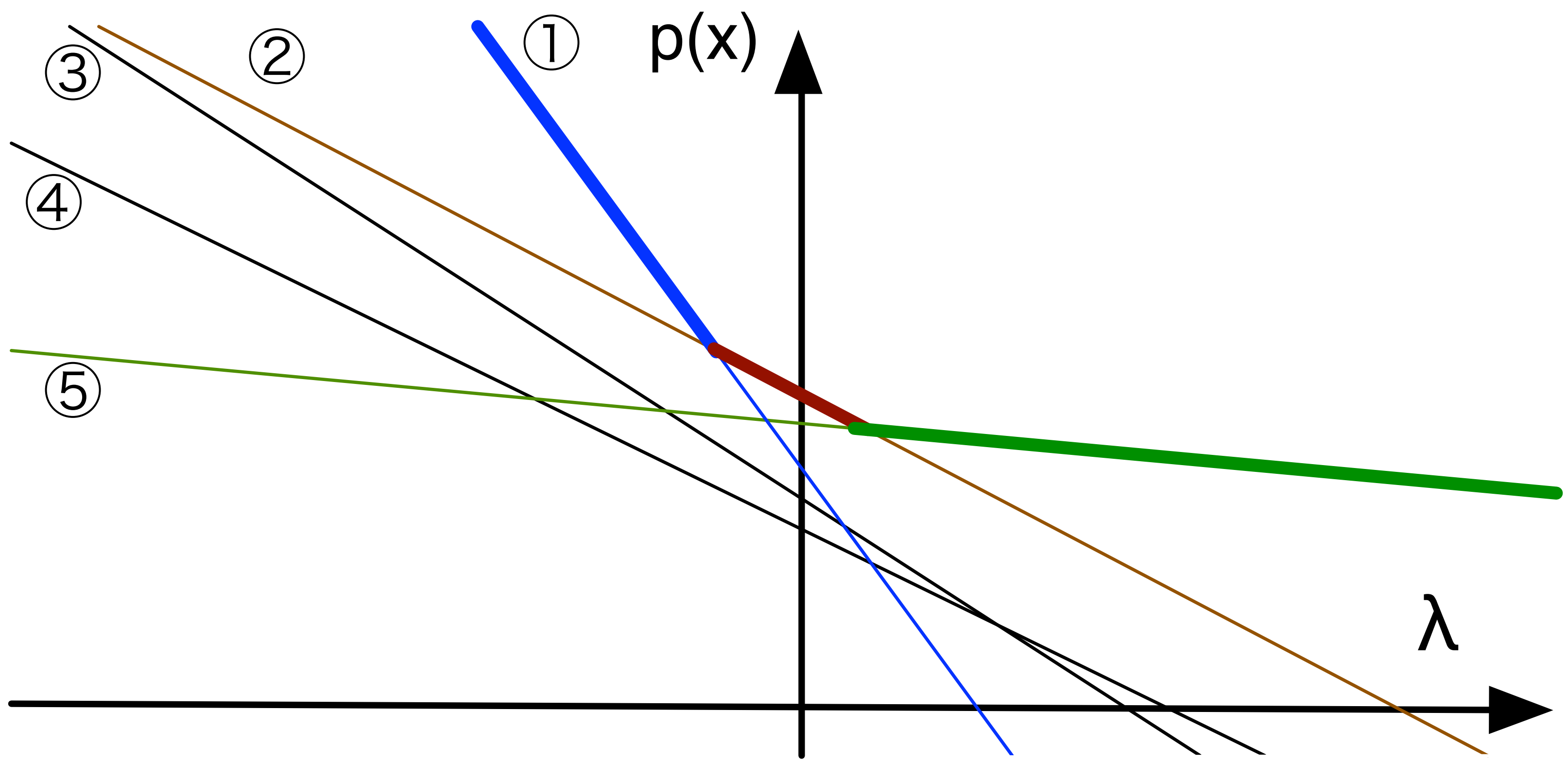
N-best translation for one sentence



- Each translation is a different line

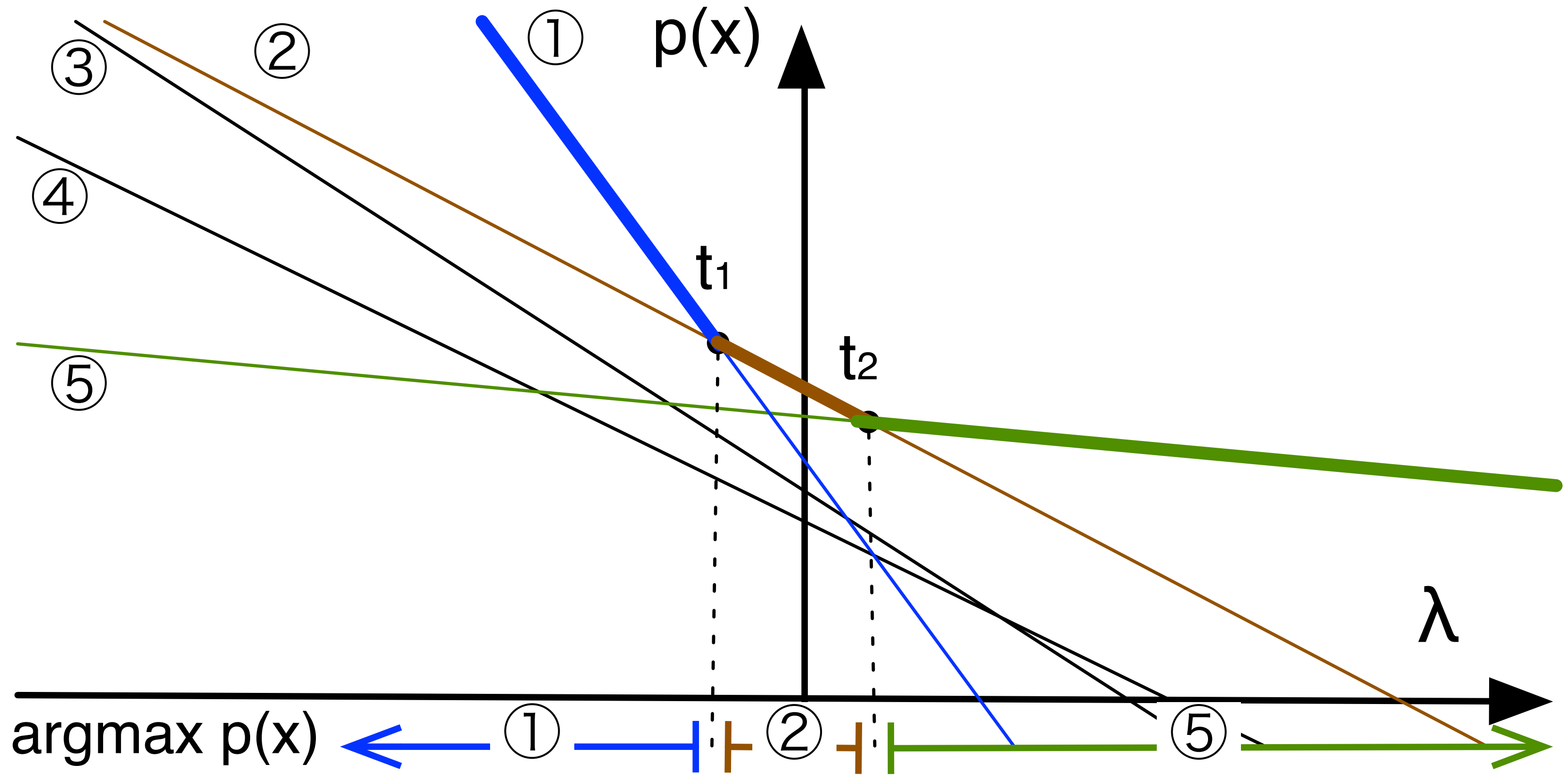
UPPSALA UNIVERSITET

Upper envelope



- Highest probability translation depends on λ

Threshold points



- There are one a few threshold points t_j where the model-best line changes

UPPSALA UNIVERSITET

Finding the optimum value for λ

Real-valued λ can have infinite number of values

But only on threshold points, one of the model-best translation changes

$\Rightarrow$ Algorithm:

- find the threshold points
- for each interval between threshold points
  - find best translations
  - compute error-score
- pick interval with best error-score

UPPSALA UNIVERSITET

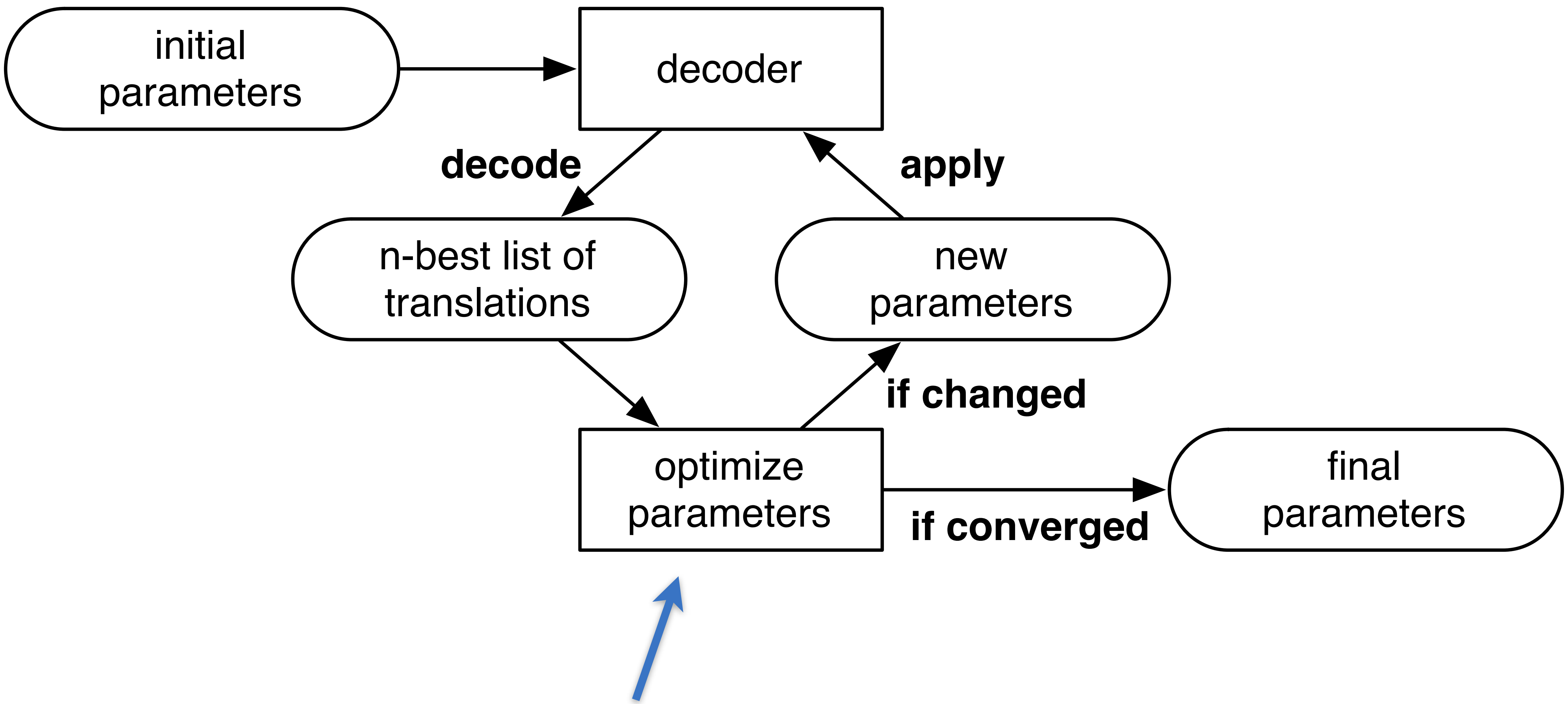
Experimental setup (1)

- Training data for translation model: 10s to 100s of millions of words
- Training data for language model: billions of words
- Parameter tuning
  - set a few weights (say, 10–15)
  - tuning set of 1000s of sentence pairs sufficient
- Finally, test set needed

Experimental setup (2)

- Tuning is non-deterministic and gives different results if you run it several times
- It is good practice to run multiple tuning runs and give the average score
- The method I just outlined is called minimum error rate training (MERT)
  - Works well for a small set of features (20-30)
  - Like the systems we have discussed in the course
  - Default method in Moses
- For larger feature sets we need other methods

Alternative optimization methods



Minimum Error Rate training (MERT)
Pair-wise Ranking Optimisation (PRO)
Margin Infused Relaxed Algorithm (MIRA)

Factored translation

UPPSALA UNIVERSITET

Problems with PBSMT

No use of morphology:

- treat inflectional variants (“look”, “looks”, “looked”) as completely different words!
- in learning translation models: knowing how to translate “look” doesn’t help to translate “looks”

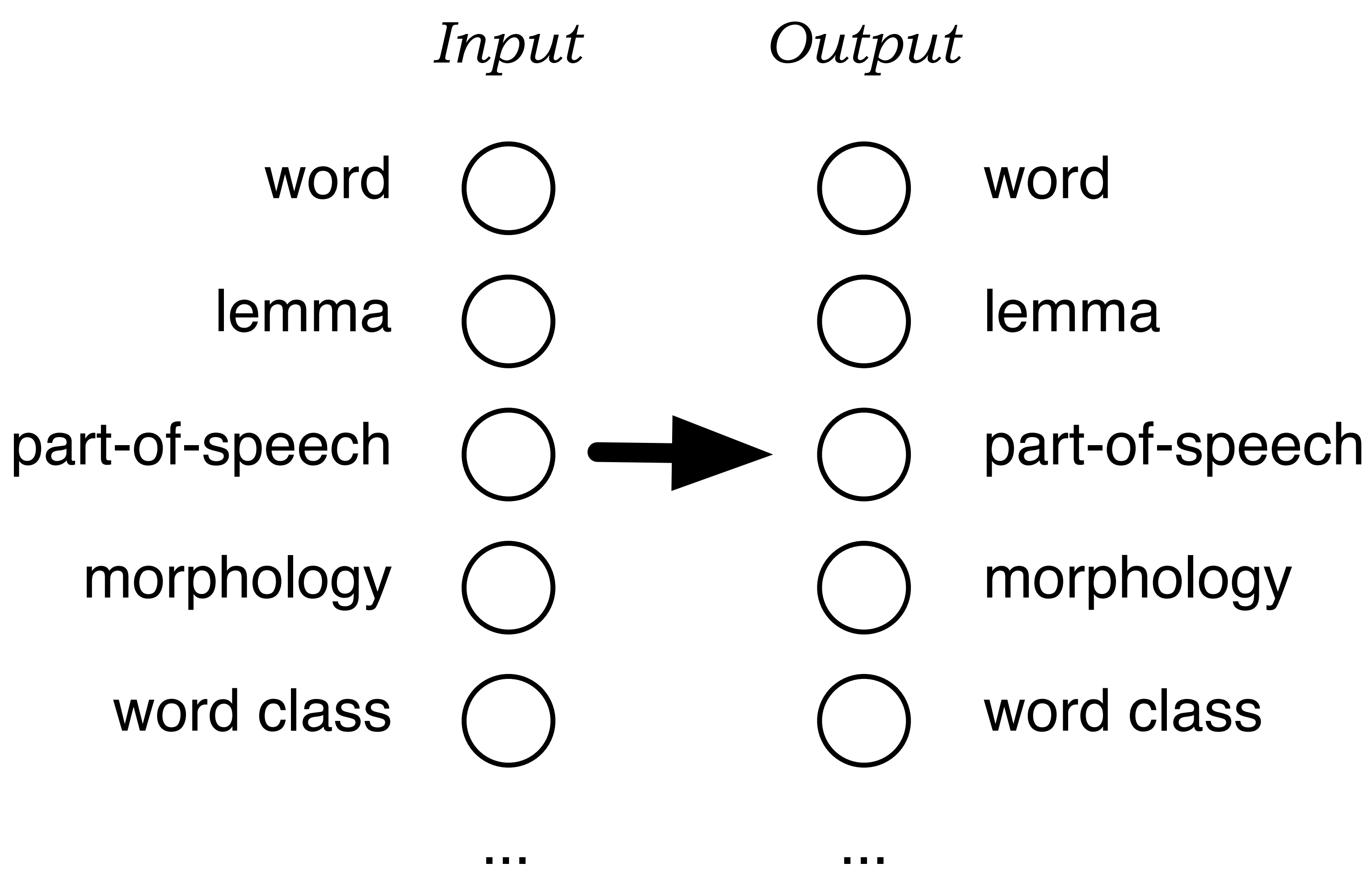
Works fine for English (and reasonable amounts of data)

Problems:

- morphologically rich languages
- sparse data sets
- flexible word order

Factored models (I)

Represent words by factors



UPPSALA UNIVERSITET

Factored models (2)

Morphology

- is productive
- well understood
- generalizable patterns

Factored models

- learn translations of base forms
- learn to map morphology
- learn to generate target surface form

UPPSALA UNIVERSITET

Factored models (3)

Represent words by factors? Why?

- combine scores for translating various factors
- back-off to other factors (lemma)
- use various factors for reordering
- better word alignment (?)

Better generalization

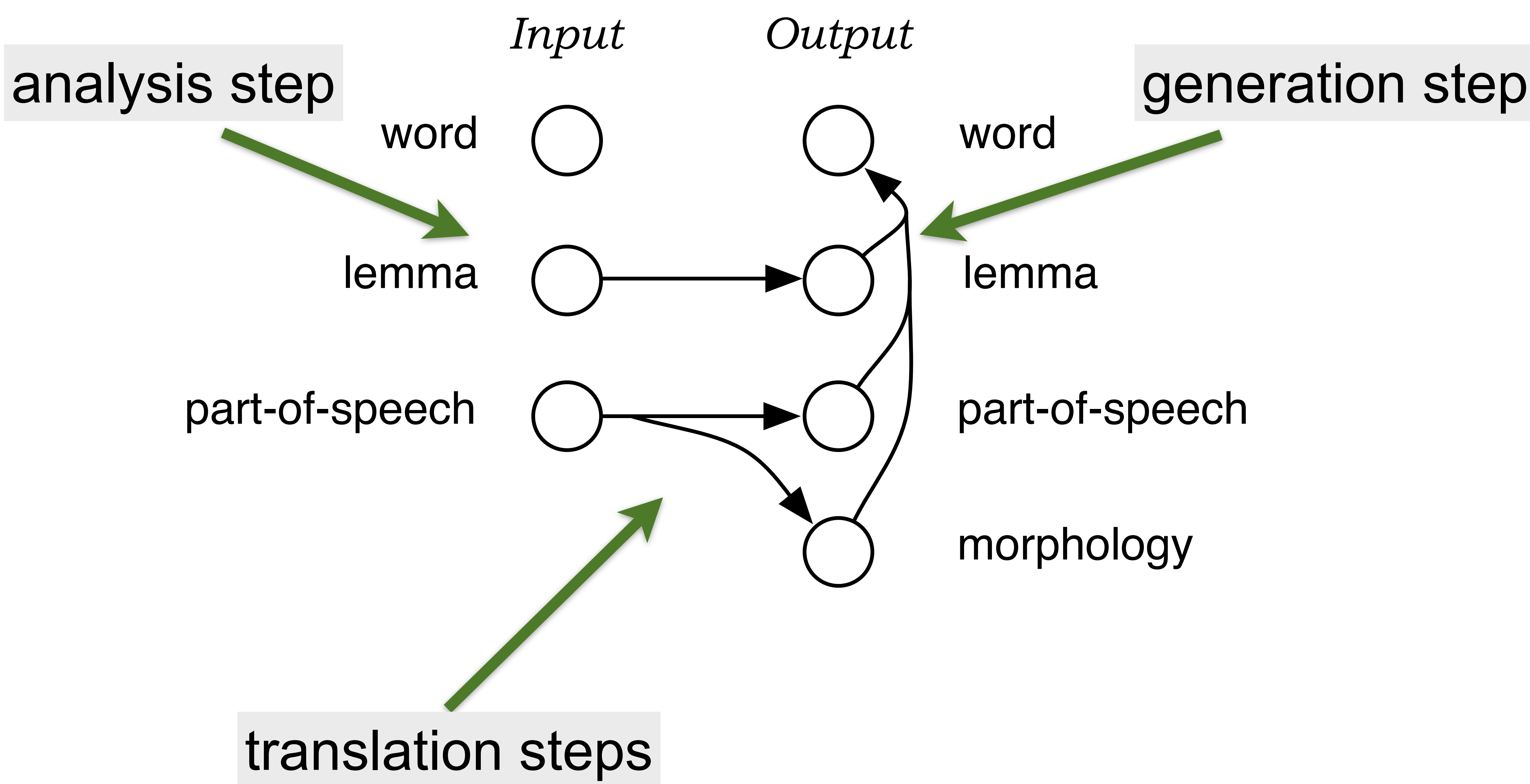
- can translate words that we haven't seen in training
- better statistics for translation options

Richer model (more (linguistic) information)

- PoS, syntactic function, semantic role, ...

UPPSALA UNIVERSITET

Factored model example (I)

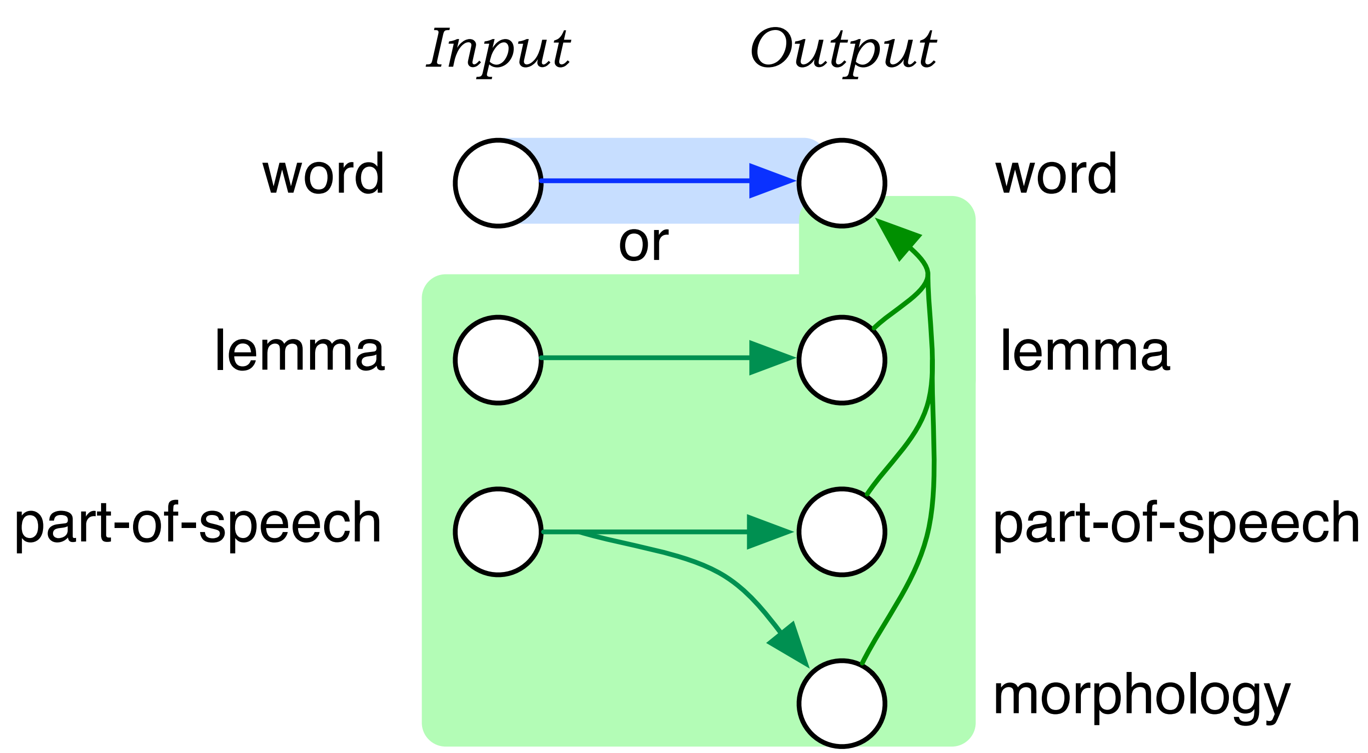


UPPSALA UNIVERSITET

Factored model example (2)

Use benefits of general phrase-based SMT!

- factored models as alternative paths (or backoff)



UPPSALA
UNIVERSITET

Factored model results

Do not always lead to improvement

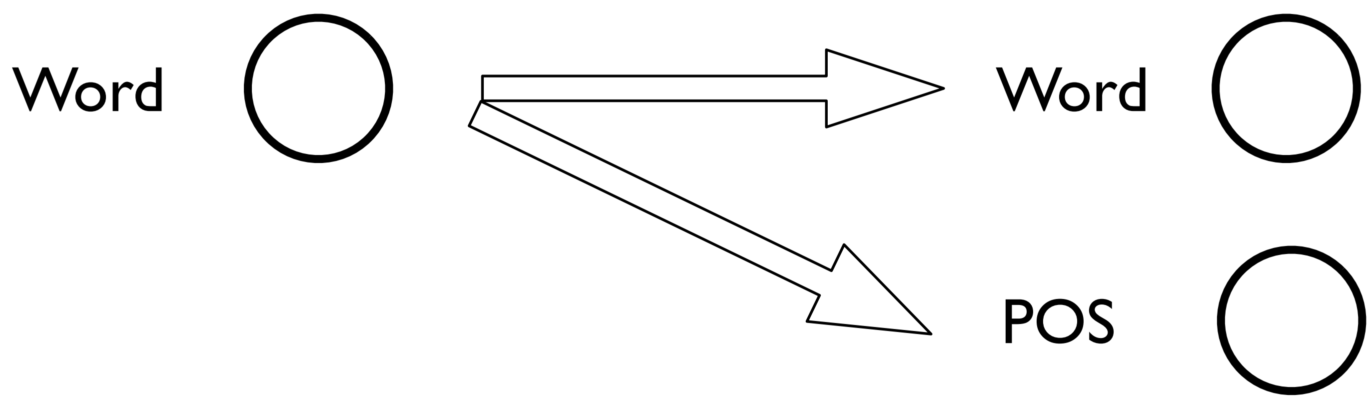
System	In-domain	Out-of-domain
Baseline	**18.19**	**15.01**
With POS LM	19.05	15.03
Morphgen model	14.38	11.65
Both model paths	**19.47**	**15.23**

Complicated models are slow compared to standard PBSMT

UPPSALA UNIVERSITET

Factored model example (3)

Simpler models often more useful!



Useful with POS LM

Simple factored models

Often useful with POS/morphology LMs

- Not much slower than standard models
- Tend to give some improvements to agreement

Improve word order of compounds that have been split

Number of compound modifiers without a head:

- System without POS-model: 136
- System with a POS-model: 6

UPPSALA UNIVERSITET

Factored model in Moses

Full support in Moses:

- *http://www.statmt.org/moses/?n=Moses.FactoredTutorial*

Data Format (example):

```
==> factored-corpus/proj-syndicate.de <==
korruption|korruption|nn|nn.fem.cas.sg floriert|florieren|vvfin|vvfin .|.|per|per

==> factored-corpus/proj-syndicate.en <==
corruption|corruption|nn flourishes|flourish|nns .|.|.
```

- 4 source language factors (word|lemma|pos|morph)
- 3 target language factors (word|lemma|pos)