The operation sequence model: Integrating translation and reordering operations in a single left-to-right model

Nadir Durrani

Helmut Schmid

Alex Fraser

Hinrich Schütze
Center for Information and Language Processing
University of Munich, Germany

Three Statistical Machine Translation (SMT) models

- Phrase-based SMT model
  - Overview, strengths, weaknesses
- N-gram-based SMT model
  - Overview, strengths, weaknesses
- Operation Sequence Model (OSM) SMT model
  - Combines benefits of phrase-based and N-gram-based SMT

Motivation: German-to-English

- Structure of main clauses in German:
  - … V2 MITTELFELD VC
- The “mittelfeld”, what’s between V2 and VC, can be arbitrarily long
- Er V2:hat ein Buch VC:gelesen → He read a book
  - hat … gelesen = read
  - Er hat gestern Nachmittag ein spannendes Buch gelesen
  - Er hat gestern Nachmittag mit seiner kleinen Tochter, die aufmerksam zugehört hat, und seinem Sohn, der lieber am Computer ein Videogame gespielt hätte, ein spannendes Buch gelesen

Motivation: German-to-English

- Er hat ein Buch gelesen → He read a book

- Er hat gestern Nachmittag mit seiner kleinen Tochter, die aufmerksam zugehört hat, und seinem Sohn, der lieber am Computer ein Videogame gespielt haette, ein spannendes Buch gelesen

- We want a model that
  - captures ”hat … gelesen = read”
  - captures the generalization that an arbitrary amount of stuff can occur between V2:hat and VC:gelesen
  - is a simple left-to-right model

Overview

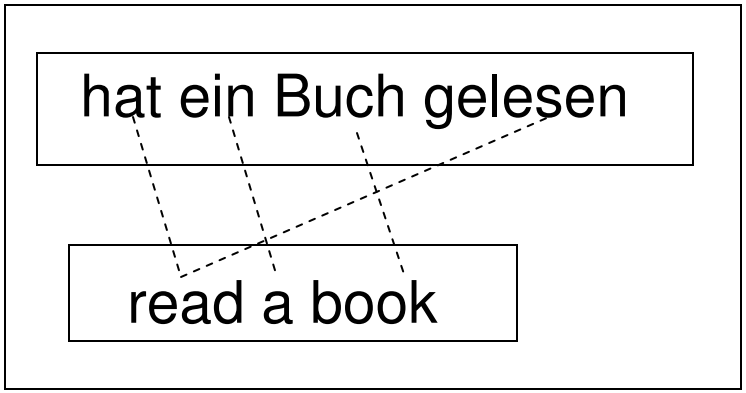
- Operation Sequence Model (OSM), a new SMT model that
  - Captures benefits of existing SMT frameworks
  - Addresses their shortcomings
- Like phrase-based SMT
  - Has the ability to memorize phrase pairs
  - Robust search mechanism
- Like N-gram-based SMT
  - Captures source and target information across phrasal boundaries
  - Does not have spurious phrasal segmentation ambiguity
- Unique property of OSM: Better reordering mechanism
  - Coupling of translation and reordering (like syntax-based SMT)
  - Ability to capture very long distance reordering

Road Map

- Phrase-based SMT (Koehn et. al 2003, Och and Ney 2004)
- N-gram-based SMT (Marino et al. 2006)
- OSM: Operation Sequence Model (Durrani et al. 2011)

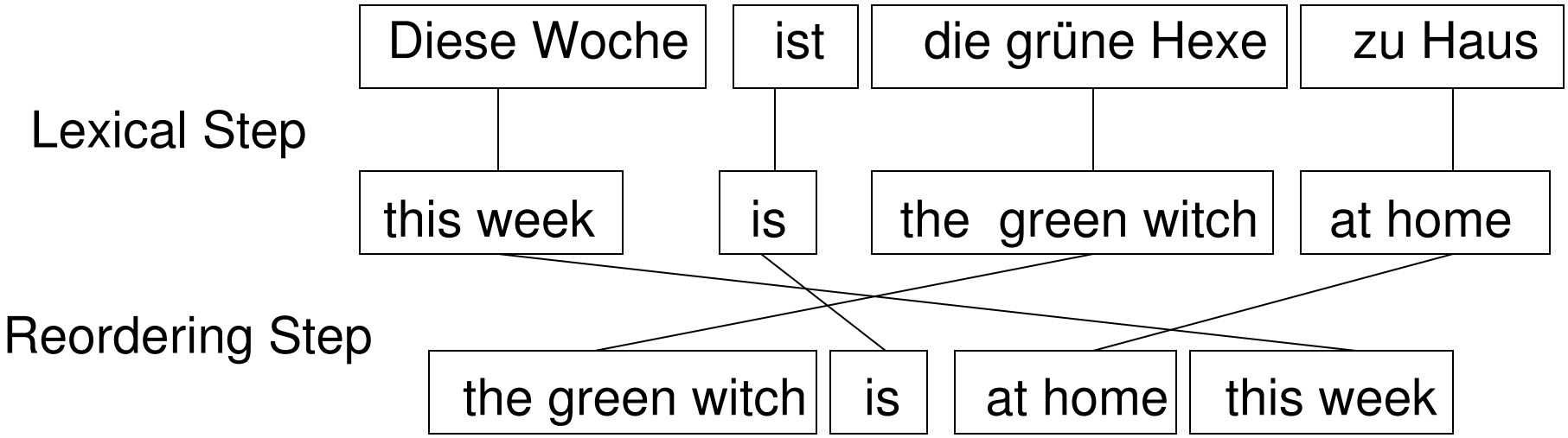
Phrase-based SMT

- Phrase-based SMT is based on phrases
- A phrase is a pair of continuous strings of words (traditionally)
- Many phrases are not phrases in the linguistic sense
- Example of a phrase:



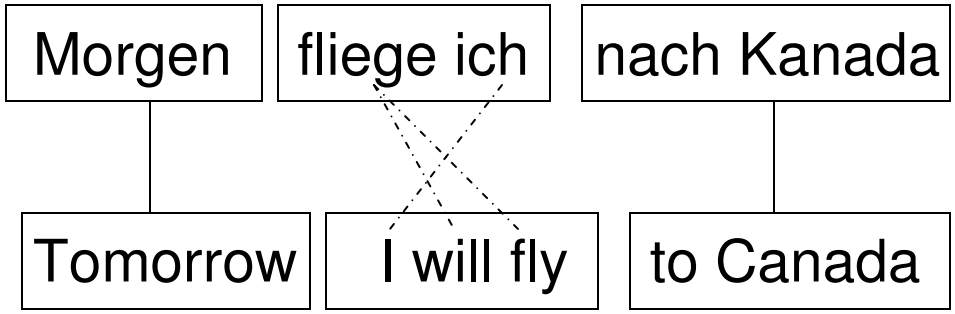
Phrase-based SMT

- State-of-the-art for many language pairs

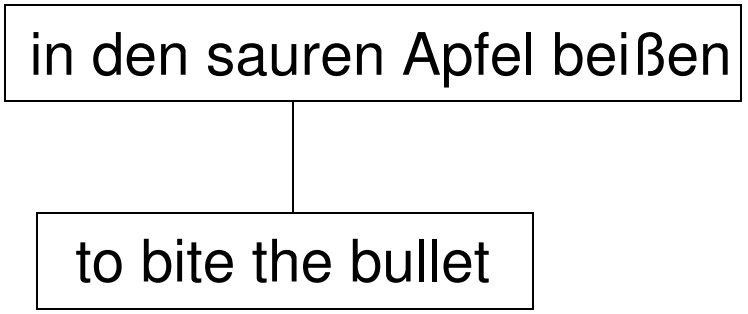


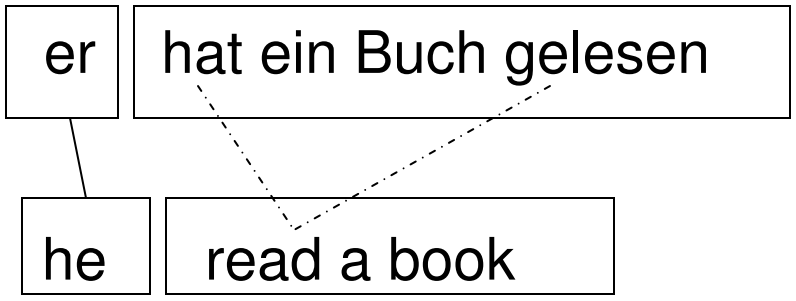
Benefits of phrase-based SMT

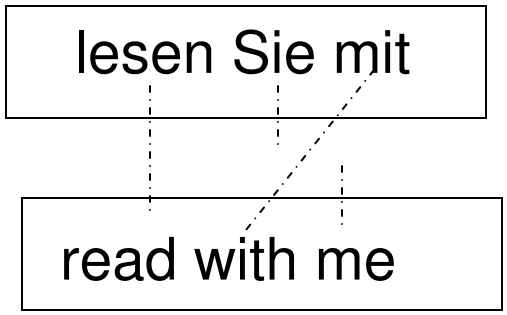
1. Local reordering



2. Idioms







3. Discontinuities in phrases

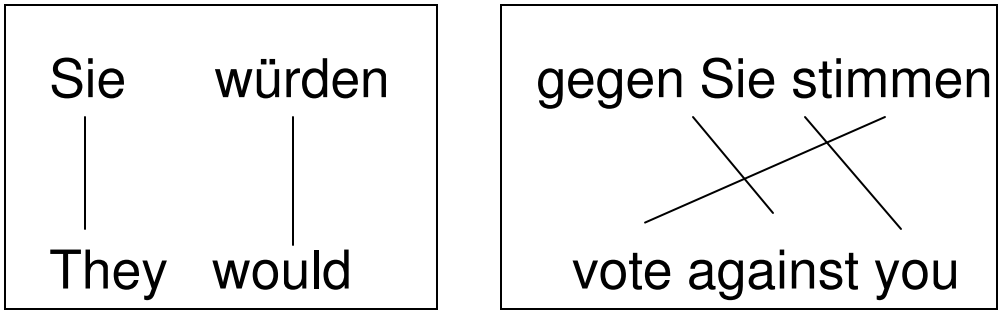
4. Insertions and deletions

Phrase-based SMT: Problems

- Strong phrasal independence assumption
  - Lexical model does not represent dependencies outside phrases
  - Gappy translation units are not allowed outside phrases
  - Deletions and insertions are not handled outside phrases
- Spurious phrasal segmentation in model and search
- Weak reordering model
  - Strongly relies on the language model
  - Hard reordering limit necessary during decoding

Phrase-based model: Problems (1)

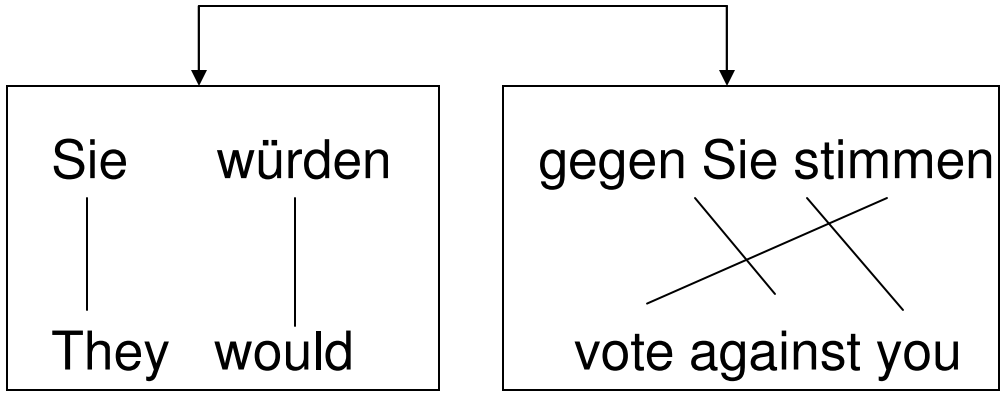
- Strong phrasal independence assumption



p_1 p_2

Phrase-based model: Problems (1)

- Strong phrasal independence assumption



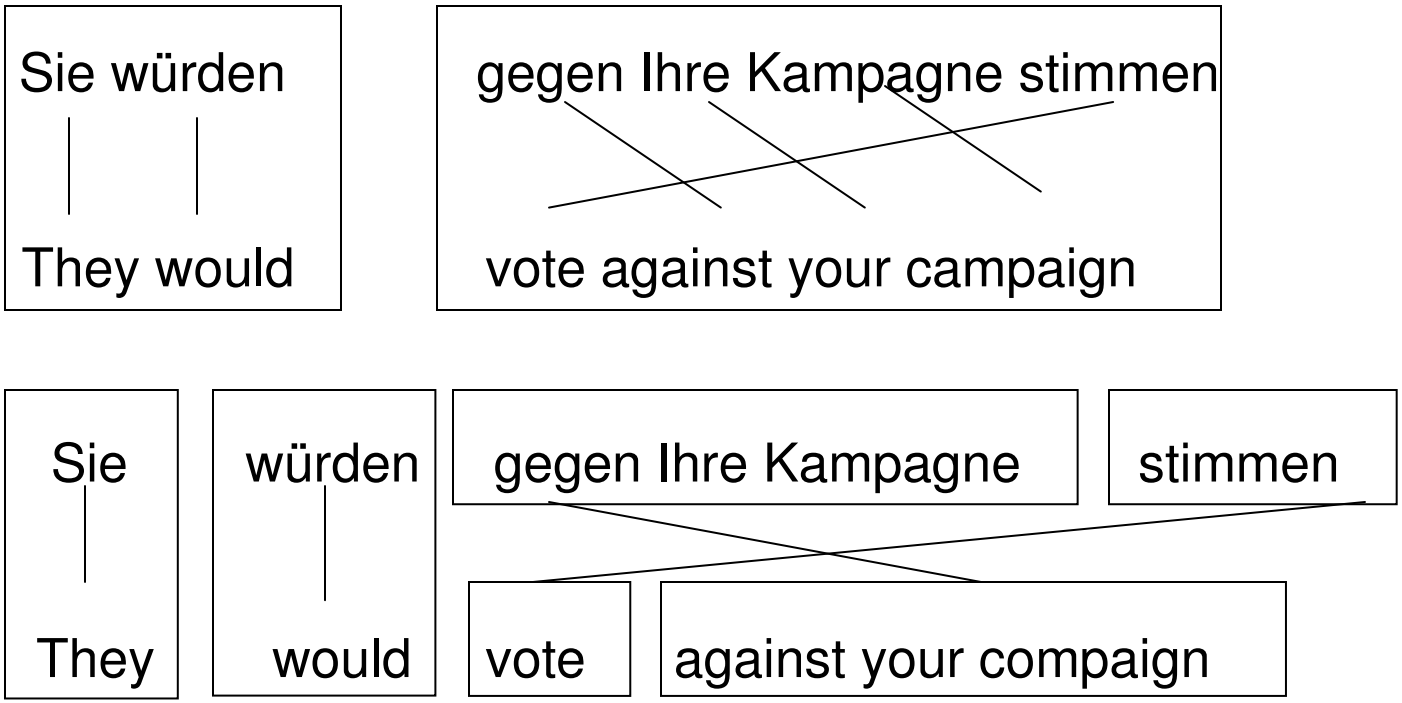
Cannot capture the dependency between parts of verb phrase

würden … stimmen – vote against

(of course, the language model will capture some of these dependencies)

Phrase-based model: Problems (2)

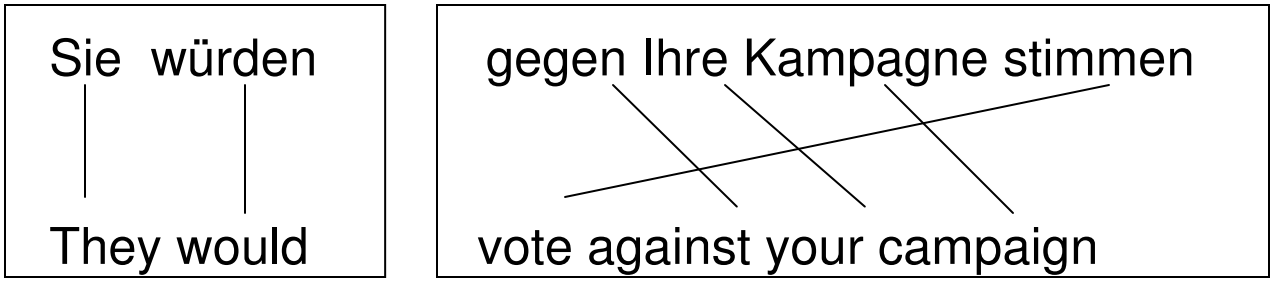
- Spurious phrasal segmentation: Many possible segmentations



Phrase-based model: Problems (3)

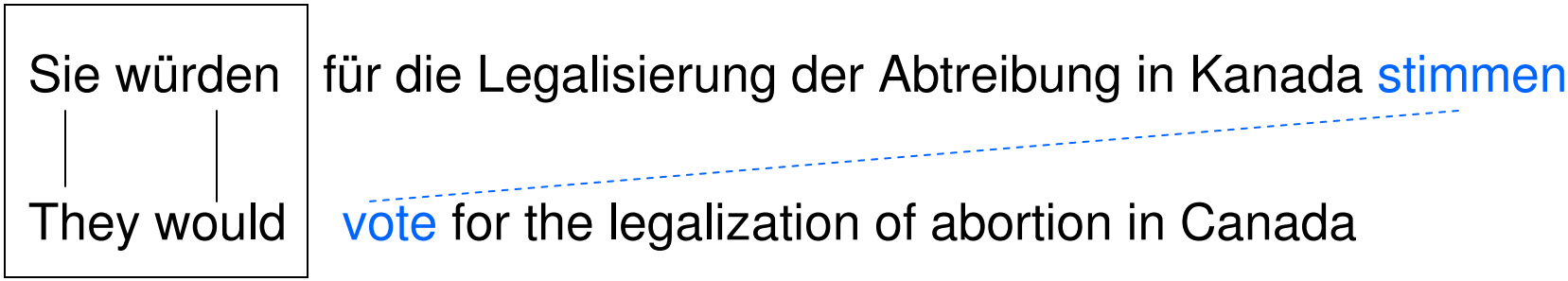
- Long distance reordering is difficult

Training



Phrase-based model: Problems (3)

- Long distance reordering is difficult
  - Have to fall back to smaller phrases at test time due to sparsity
  - Lexical generation model may not have information on how to reorder “stimmen - vote”



(Note: There is work on addressing this, e.g., Galley and Manning, 2008, Green et. al 2010, Bisazza and Federico 2013)

Road Map

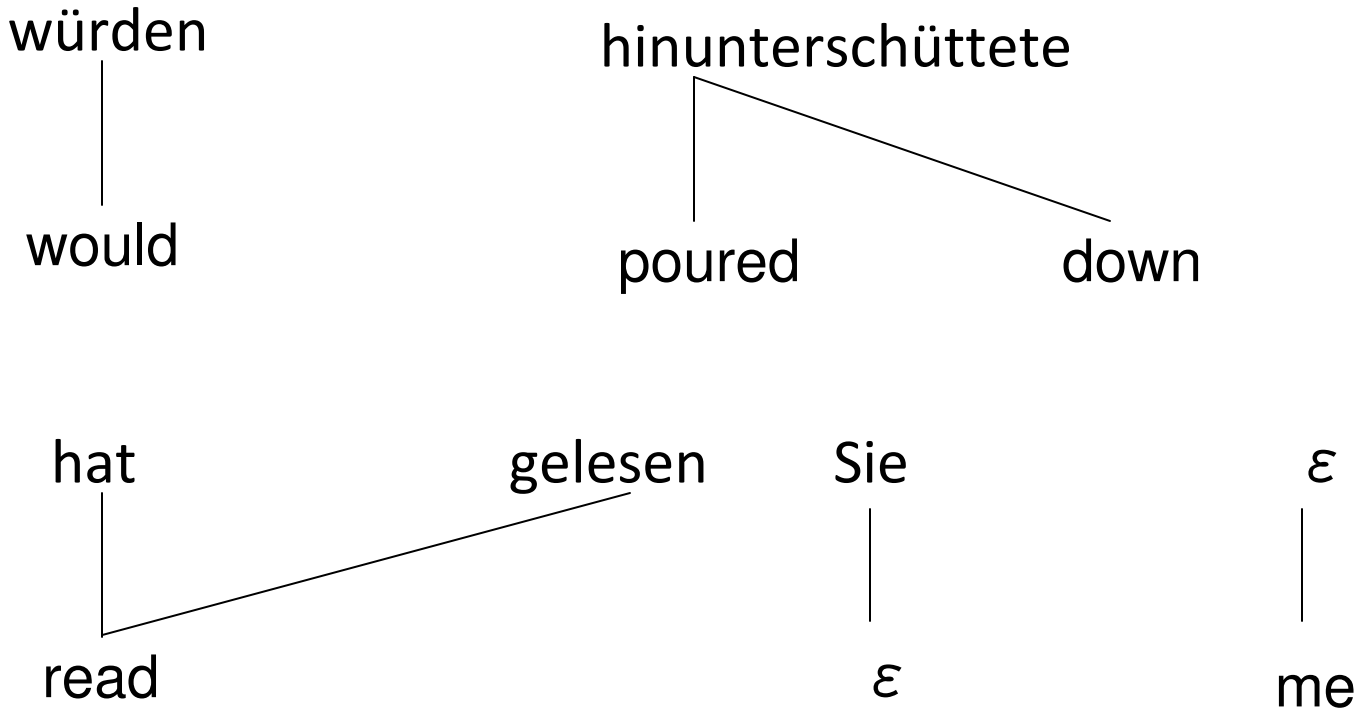
- Phrase-based SMT (Koehn et. al 2003, Och and Ney 2004)
- N-gram-based SMT (Mariño et al. 2006)
- OSM: Operation Sequence Model (Durrani et al. 2011)

N-gram-based SMT

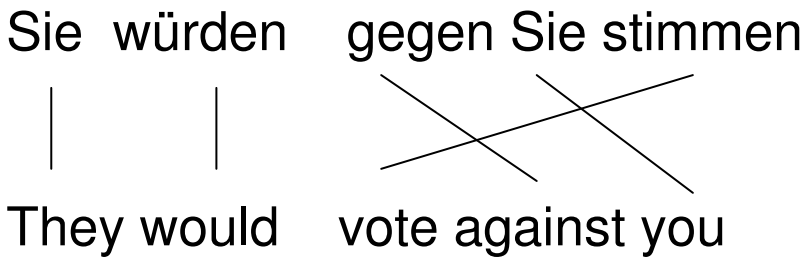
- N-gram-based translation model (Mariño et al. 2006)
  - Translation as sequential generations of MTUs / translation tuples
- Reordering framework (Crego and Mariño 2007)
- Decoders
  - MARIE (Crego 2005, 2007)
  - NCode (Crego et. al. 2011)
- Handling discontinuous units (Crego and Yvon 2009)
- Factored bilingual models (Crego and Yvon 2010)
- CRF-based translation models (Lavergne et. al 2011)
- Continuous space translation models (Le et. al 2012)

MTU = Minimal Translation Unit

- The N-gram-based translation model is based on MTUs
- Definition: MTU = connected component of the alignment bigraph



N-gram-based SMT



Extract Minimal Translation Units (MTUs)

t_1: Sie → They

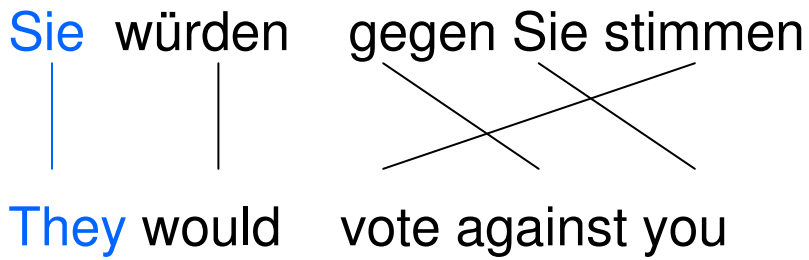
t_2: würden → would

t_3: stimmen → vote

t_4: gegen → against

t_5 : Sie → you

N-gram-based SMT



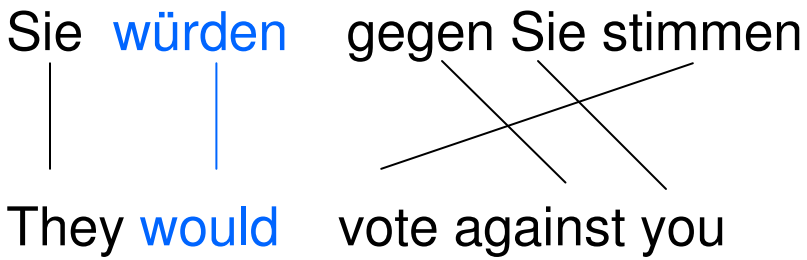
Sie

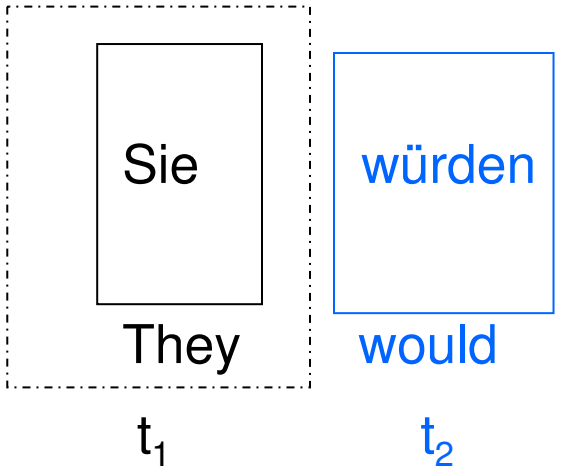
They

t_1

$p(t_1)$

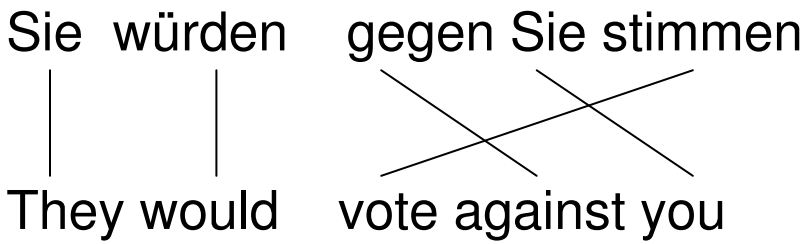
N-gram-based SMT

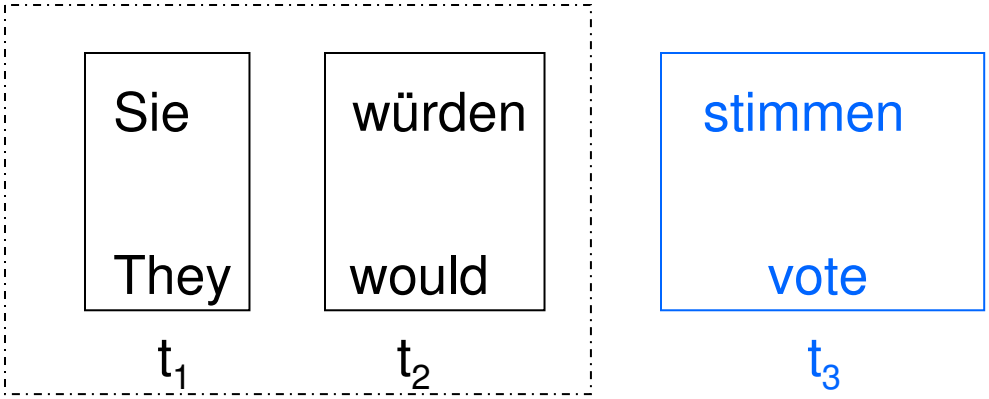




$p(t_1) \times p(t_2|t_1)$

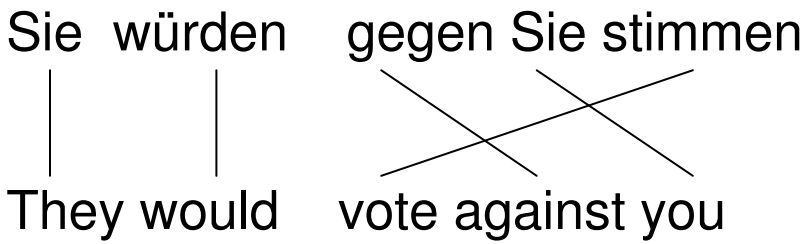
N-gram-based SMT

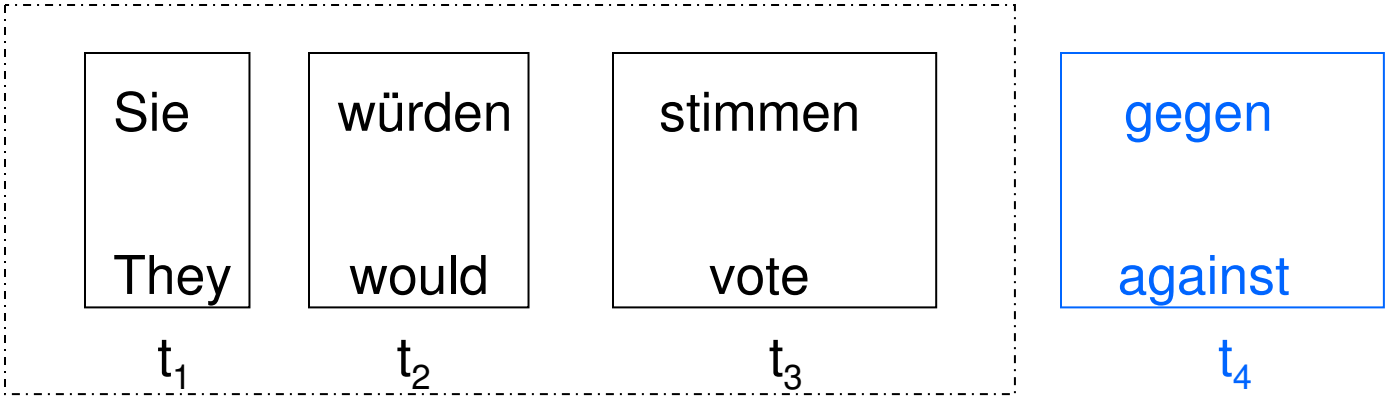




$p(t_1) \times p(t_2|t_1) \times p(t_3| t_1 t_2)$

N-gram-based SMT



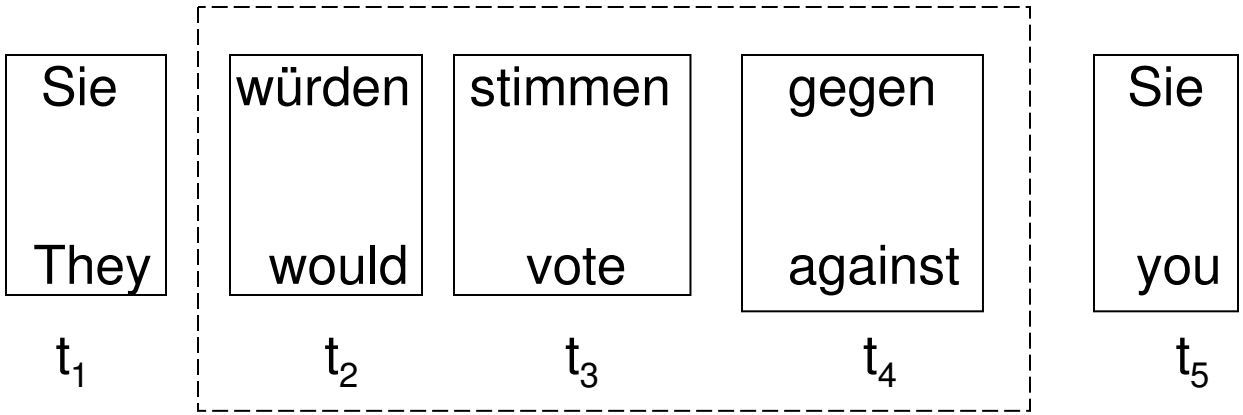


$p(t_1) \times p(t_2 \mid t_1) \times p(t_3 \mid t_1\ t_2) \times p(t_4 \mid t_1\ t_2\ t_3)$

Model

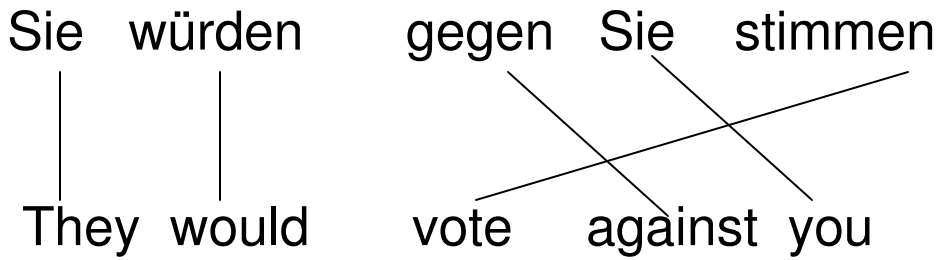
- Joint model over sequences of minimal units

$$p_{tsm}(F, E, A) = p(t_1^J) = \prod_{j=1}^{J} p(t_j | t_{j-n+1}, \ldots, t_{j-1})$$



Context window: 4-gram model

Reordering model and search



- Linearize the source to be in target order
  - gegen Sie stimmen → stimmen gegen Sie
- Learn POS-based rules
  - IN PRP VB → VB IN PRP
- Use POS-based rewrite rules to construct the search graph as preprocessing step

N-gram-based SMT: Summary

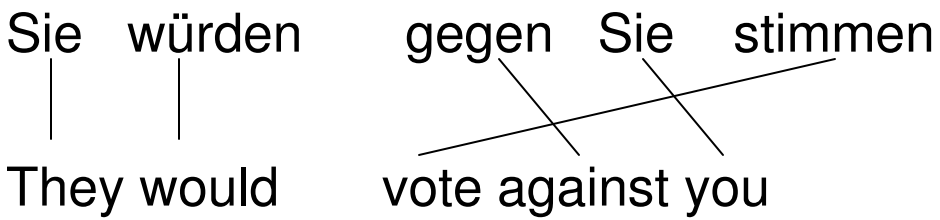
- Translation as sequential generation of a sentence pair
- Markov model
- Each step in the sequence generates one MTU
- POS-based reordering of source

Benefits of N-gram-based SMT

- Avoids spurious phrasal ambiguities
  - Only one way to represent a bilingual sentence pair
- Does not make phrasal independence assumption
  - Capture source and target context across phrasal boundaries
- Estimates better translation model
  - Takes advantage of well-known smoothing methods (Kneser-Ney)

Problems of N-gram-based SMT: Weak reordering model

- Training



POS rule learned: IN PRP VB → VB IN PRP

- Reliance on POS tagger is limitation

- POS rules ignore target side
  - Target language model is only used to score precomputed orderings

Problems of N-gram-based SMT: Weak reordering model

- POS rules are sparse and do not capture long-distance phenomena
  - Er V2:hat gestern Nachmittag mit seiner kleinen Tochter, die aufmerksam zugehört hat, und seinem Sohn, der lieber am Computer ein Videogame gespielt haette, ein spannendes Buch VC:gelesen
  - Generalized rules based on parse trees learned in Crego et. al 2007
    - However still suffer sparsity + parse trees are not always available
- In practice, you have to put a limit on the length of a POS rule

N-gram based SMT: Problems with MTU-based search

- We search only on pre-calculated reorderings of minimal translation units (MTUs)
  - Some orderings that can be justified through language model and lexical generation model are never hypothesized
- Reordering and lexical generation are separated out
  - Cannot take advantage of lexical reordering triggers
- Poor translation coverage
  - Infrequent translations “schoss ein Tor – scored a goal” never hypothesized
  - Many good hypotheses are pruned early

Road Map

- Phrase-based SMT (Koehn et. al 2003, Och and Ney 2004)
- N-gram-based SMT (Marino et al. 2006)
- OSM: Operation Sequence Model (Durrani et al. 2011)

Motivation for OSM

- Like phrase-based SMT, OSM
  - Has the ability to memorize dependencies and lexical triggers
  - Can search for all possible reorderings
  - Has a robust search mechanism
- Like N-gram-based SMT, OSM
  - Is based on minimal translation units (MTUs)
  - Takes source and target context into account
  - Does not have the spurious phrasal segmentation problem
- OSM has a strong reordering mechanism.
  - Strongly couples translation and reordering
  - Handles both short and long distance reorderings
  - No hard reordering limit is required

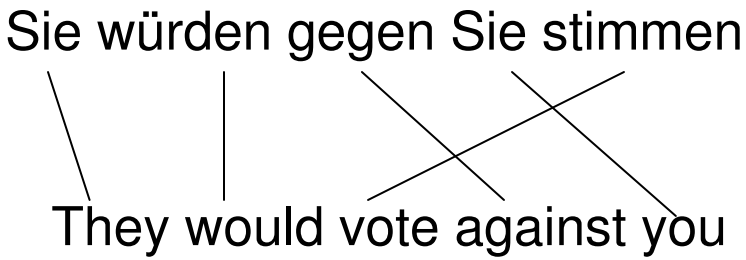
Operation Sequence Model (OSM) in a nutshell

- Aspects that are similar to N-gram-based SMT
  - Translation as sequential generation of a sentence pair
  - Generation is in target order
  - Sequential Markov model
  - MTU-based: Lexical operations generate MTUs
- What's different
  - Sequence of operations (as opposed to sequence of MTUs)
  - Two types of operations
    - Lexical operations
    - Reordering operations

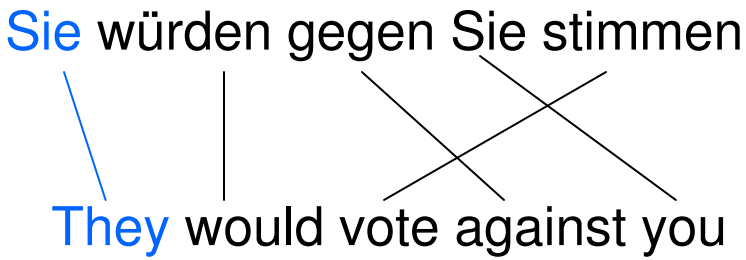
Operation Sequence Model (OSM)

- Introduction of the model: Durrani, Schmid, Fraser, ACL 2011
  - Cept-based decoding
- Phrase-based decoding: Durrani, Fraser, Schmid, NAACL 2013
- Integration into Moses: Durrani, Fraser, Schmid, Hoang, Koehn, ACL 2013

Example

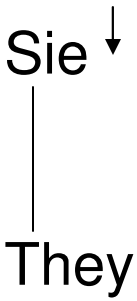


Example

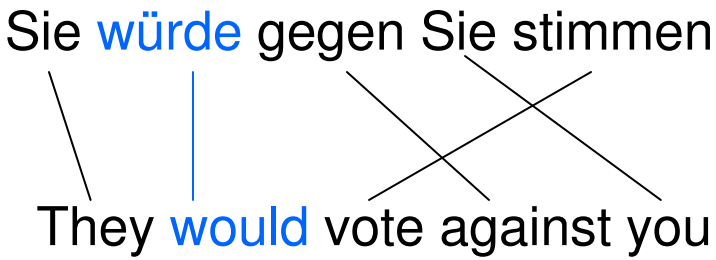


Operations

o_1: Generate (Sie – They)



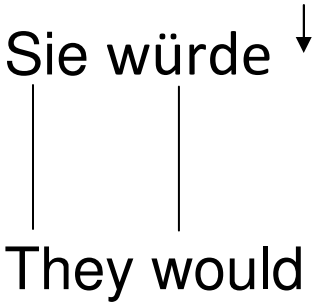
Example



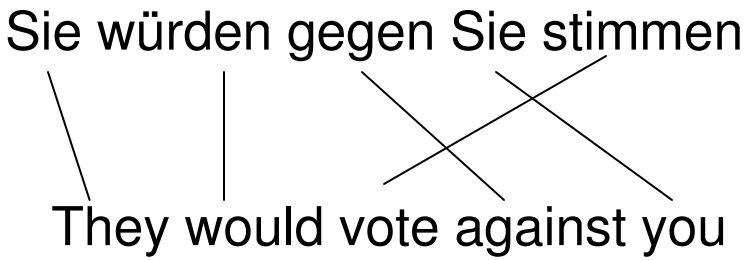
Operations

o_1 Generate (Sie, They)

o_2 Generate (würde, would)

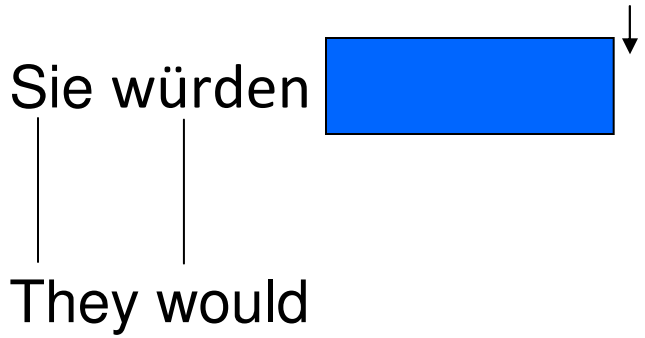


Example

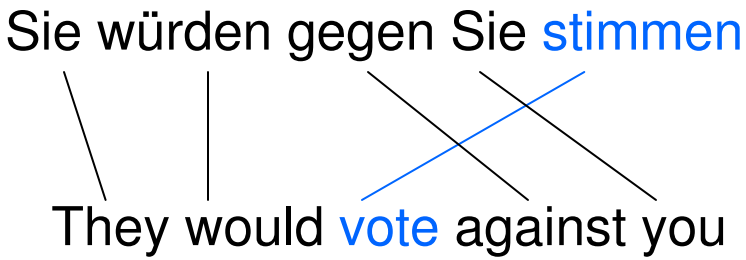


Operations

- o_1 Generate (Sie, They)
- o_2 Generate (würden, would)
- o_3 Insert Gap

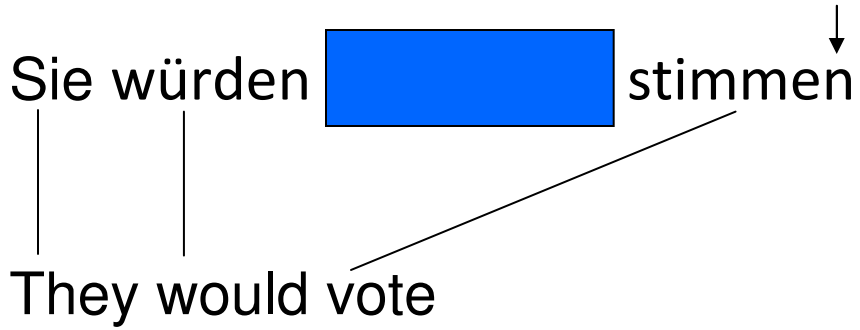


Example

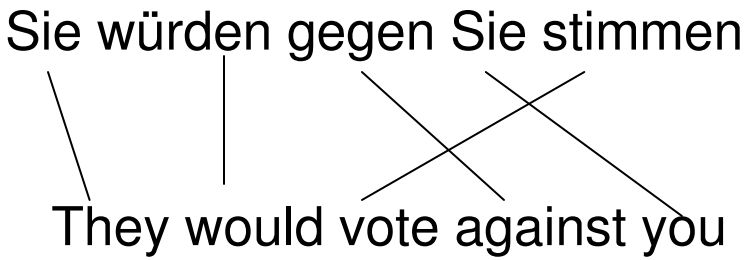


Operations

- o_1 Generate (Sie, They)
- o_2 Generate (würden, would)
- o_3 Insert Gap
- o_4 Generate (stimmen, vote)

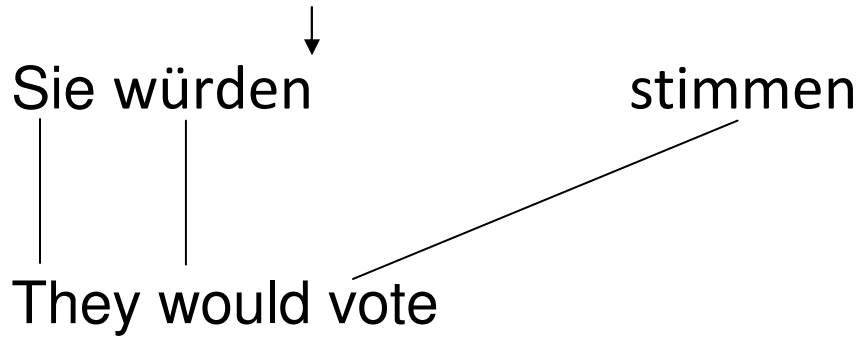


Example

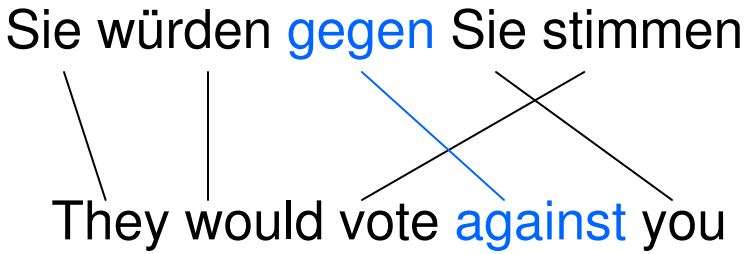


Operations

- o_1 Generate (Sie, They)
- o_2 Generate (würden, would)
- o_3 Insert Gap
- o_4 Generate (stimmen, vote)
- o_5 Jump Back (1)

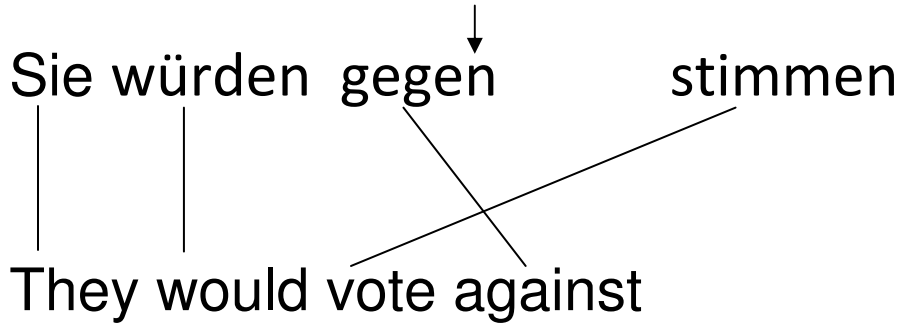


Example

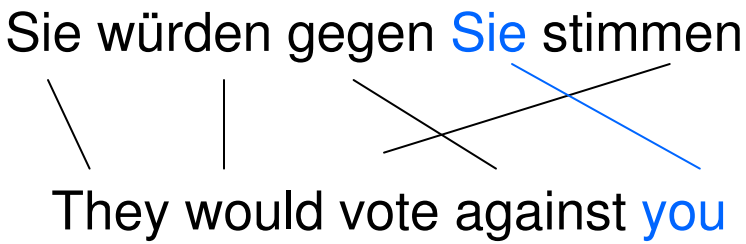


Operations

- o_1 Generate (Sie, They)
- o_2 Generate (würden, would)
- o_3 Insert Gap
- o_4 Generate (stimmen, vote)
- o_5 Jump Back (1)
- o_6 Generate (gegen, against)

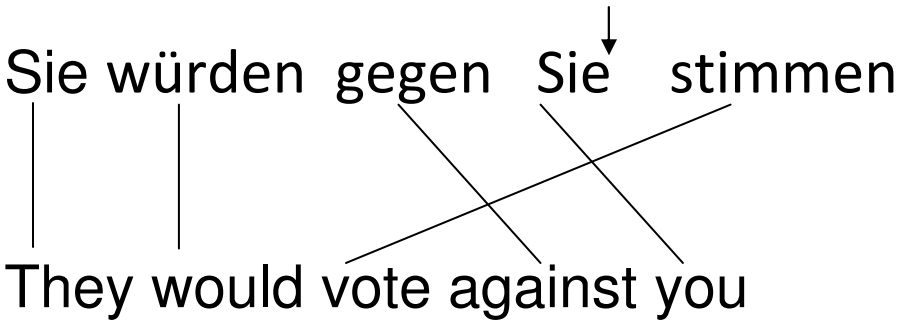


Example



Operations

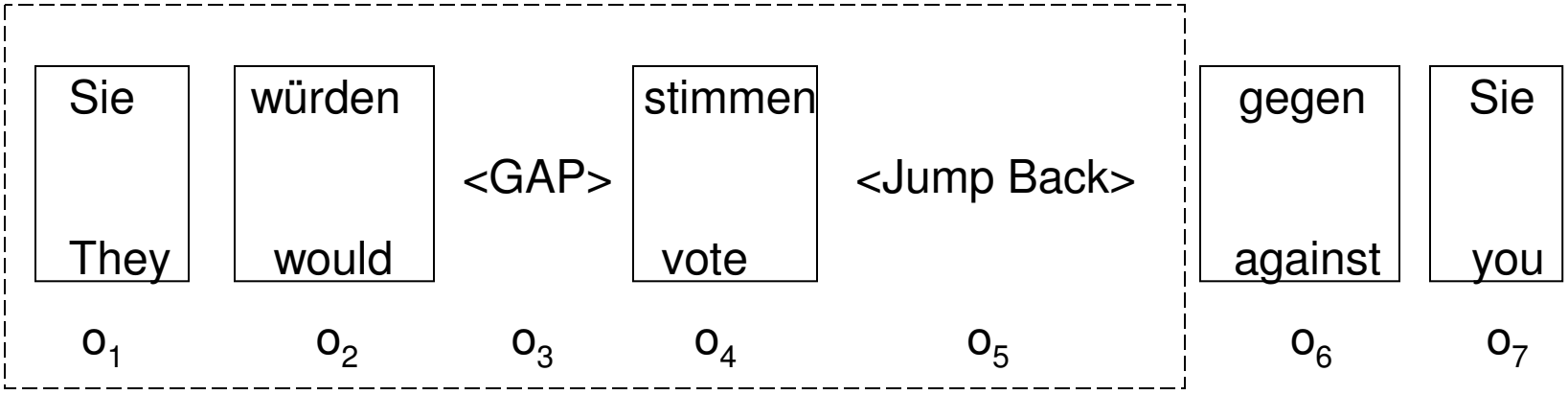
- o_1 Generate (Sie, He)
- o_2 Generate (würde, would)
- o_3 Insert Gap
- o_4 Generate (stimmen, vote)
- o_5 Jump Back (1)
- o_6 Generate (gegen, against)
- o_7 Generate (Sie, you)



Model

- Joint probability model over operation sequences

$$p_{osm}(F, E, A) = p(o_1^J) = \prod_{j=1}^{J} p(o_j | o_{j-n+1}, \ldots, o_{j-1})$$



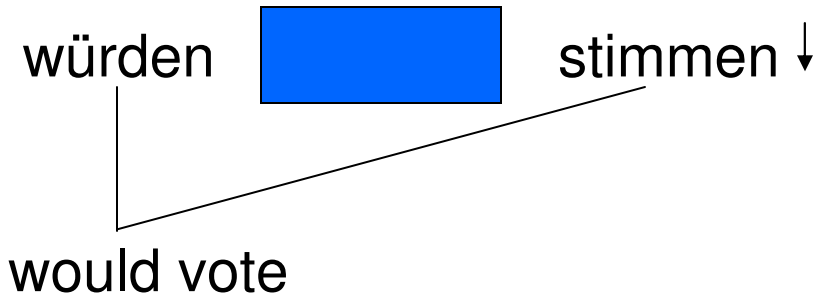
Context window: 9-gram model

Useful properties of OSM

- Capture source and target context across phrasal boundaries
  - Simultaneously generate source and target units
- Model does not have spurious ambiguity
  - Model is based on minimal translation units (MTUs)
- Better reordering mechanism
  - Uniformly handles local and non-local reorderings
  - Strong coupling of lexical generation and reordering

Example of a learned pattern

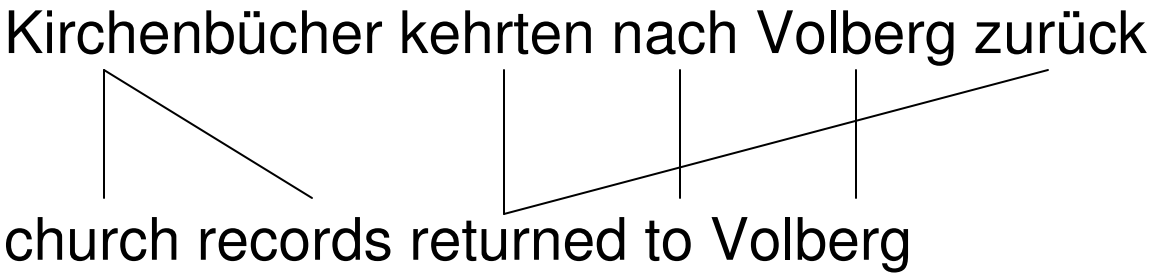
- Operations
  - Generate (würden, would)
  - Insert Gap
  - Generate (stimmen, vote)



- Can generalize to
  - Die Menschen würden dafür stimmen
  - Die Menschen würden gegen meine Außenpolitik stimmen
  - Die Menschen würden für die Legalisierung der Abtreibung in Kanada stimmen

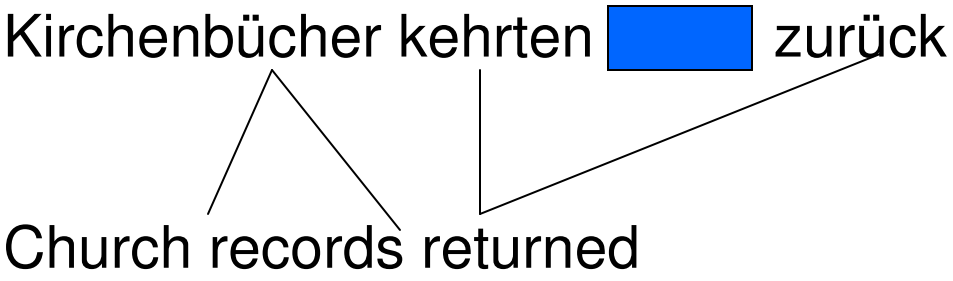
- Equivalent to hierarchical phrase “würden X stimmen – would vote X”

Source side discontinuities



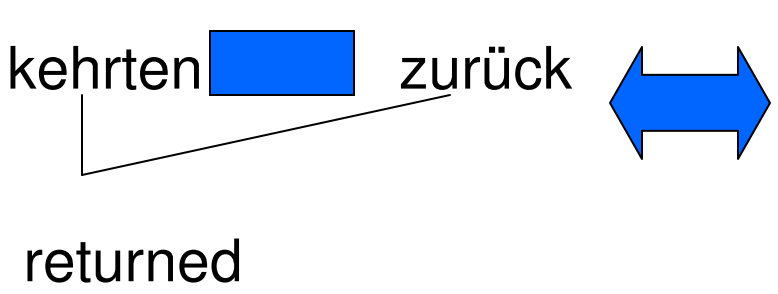
- Operations
  - Generate (Kirchenbücher, church records)
  - Generate (kehrten…[zurück], returned)
  - Insert Gap
  - Continue Source Cept

    …



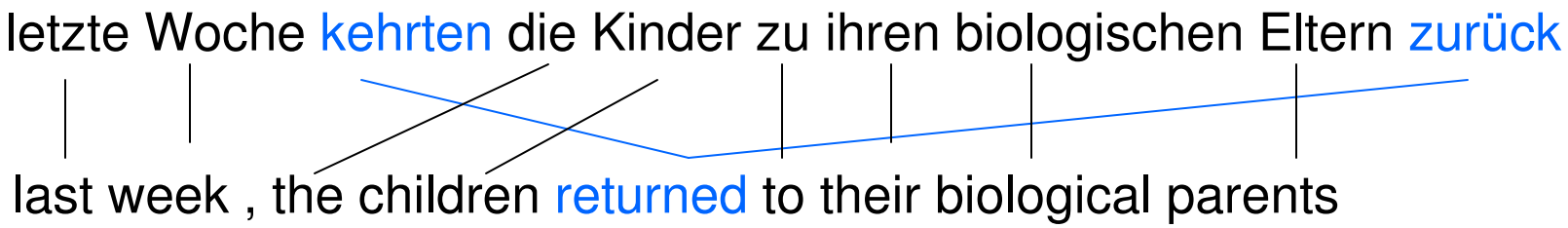
Source side discontinuities: Generalization to unseen sentences

- Operation sequence learned by OSM:



Generate (kehrten…[zurück], returned)
Insert Gap
Continue Source Cept

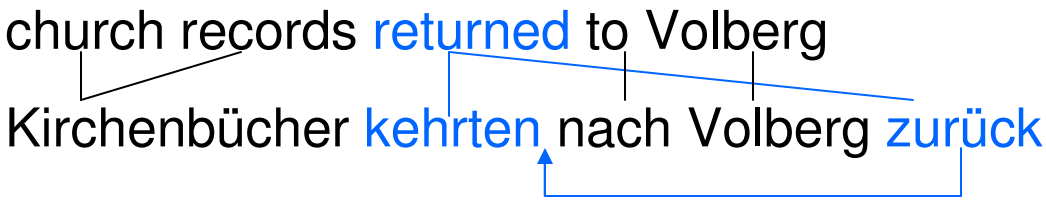
- OSM can generalize to:



Target side discontinuities

- The target side is generated in left-to-right order sequentially.

- Alignments with discontinuous targets are handled by linearization of target.



- Linearize "kehrten…zurück" for the estimation of OSM model

Source deletion and target insertion submodels

- Built-in models for
  - Source word deletion
  - Target word insertion
  - Contextual information: Lexical trigger “Sie” gets deleted in German after verbs

- Example

Source deletion and target insertion submodels

- Built-in models for
  - Source word deletion
  - Target word insertion
  - Contextual information: Lexical trigger “Sie” gets deleted in German after verbs

- Example

  Lesen Sie bitte mit

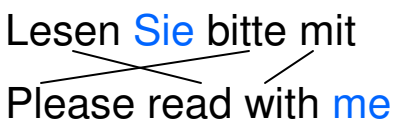
  Please read with me

Operations:

Source deletion and target insertion submodels

- Built-in models for
  - Source word deletion
  - Target word insertion
  - Contextual information: Lexical trigger “Sie” gets deleted in German after verbs

- Example



Operations: Insert Gap

Source deletion and target insertion submodels

- Built-in models for
  - Source word deletion
  - Target word insertion
  - Contextual information: Lexical trigger “Sie” gets deleted in German after verbs

- Example

Operations: Insert Gap → Generate (bitte, Please)

Source deletion and target insertion submodels

- Built-in models for
  - Source word deletion
  - Target word insertion
  - Contextual information: Lexical trigger “Sie” gets deleted in German after verbs

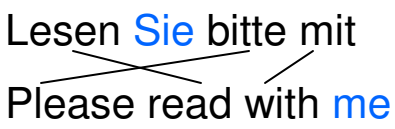
- Example

Operations: Insert Gap → Generate (bitte, Please) → Jump Back(1)

Source deletion and target insertion submodels

- Built-in models for
  - Source word deletion
  - Target word insertion
  - Contextual information: Lexical trigger “Sie” gets deleted in German after verbs

- Example



Operations: Insert Gap → Generate (bitte, Please) → Jump Back(1) →

Generate (Lesen, read)

Source deletion and target insertion submodels

- Built-in models for
  - Source word deletion
  - Target word insertion
  - Contextual information: Lexical trigger “Sie” gets deleted in German after verbs

- Example

  Lesen Sie bitte mit

  Please read with me

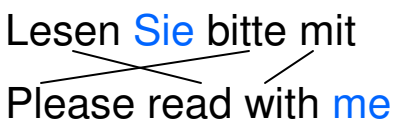
Operations: Insert Gap → Generate (bitte, Please) → Jump Back(1) →

Generate (Lesen, read) → Generate Source Only (Sie)

Source deletion and target insertion submodels

- Built-in models for
  - Source word deletion
  - Target word insertion
  - Contextual information: Lexical trigger “Sie” gets deleted in German after verbs

- Example



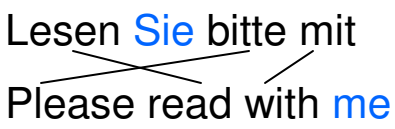
Operations: Insert Gap → Generate (bitte, Please) → Jump Back(1) →

Generate (Lesen, read) → Generate Source Only (Sie) → Jump Forward

Source deletion and target insertion submodels

- Built-in models for
  - Source word deletion
  - Target word insertion
  - Contextual information: Lexical trigger “Sie” gets deleted in German after verbs

- Example

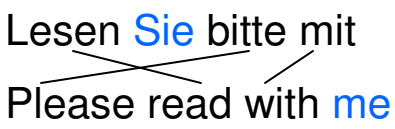


Operations: Insert Gap → Generate (bitte, Please) → Jump Back(1) → Generate (Lesen, read) → Generate Source Only (Sie) → Jump Forward → Generate (mit, with)

Source deletion and target insertion submodels

- Built-in models for
  - Source word deletion
  - Target word insertion
  - Contextual information: Lexical trigger “Sie” gets deleted in German after verbs

- Example



Operations: Insert Gap → Generate (bitte, Please) → Jump Back(1) →

Generate (Lesen, read) → Generate Source Only (Sie) → Jump Forward →

Generate (mit, with) → Generate Target Only (me)

List of Operations

- 5 Translation Operations
  - Generate (X,Y)
  - Continue Source Cept
  - Generate Identical
  - Generate Source Only (X)
  - Generate Target Only (Y)
- 3 Reordering Operations
  - Insert Gap
  - Jump Back (N)
  - Jump Forward

List of Operations

- 5 Translation Operations
  - Generate (X,Y)
  - Continue Source Cept
  - Generate Identical
  - Generate Source Only (X)
  - Generate Target Only (Y)
- 3 Reordering Operations
  - Insert Gap
  - Jump Back (N)
  - Jump Forward

Example

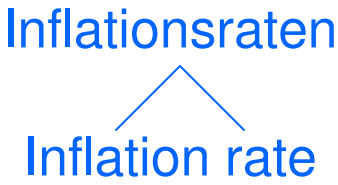
Generate (gegessen , eaten)

List of Operations

- 5 Translation Operations
  - Generate (X,Y)
  - Continue Source Cept
  - Generate Identical
  - Generate Source Only (X)
  - Generate Target Only (Y)
- 3 Reordering Operations
  - Insert Gap
  - Jump Back (N)
  - Jump Forward

Example

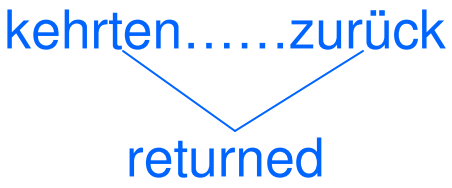
Generate (Inflationsraten , inflation rate)



List of Operations

- 5 Translation Operations
  - Generate (X,Y)
  - Continue Source Cept
  - Generate Identical
  - Generate Source Only (X)
  - Generate Target Only (Y)
- 3 Reordering Operations
  - Insert Gap
  - Jump Back (N)
  - Jump Forward

Example



Generate (kehrten zurück , returned) → Insert Gap → **Continue Source Cept**

List of Operations

- 5 Translation Operations
  - Generate (X,Y)
  - Continue Source Cept
  - Generate Identical
  - Generate Source Only (X)
  - Generate Target Only (Y)
- 3 Reordering Operations
  - Insert Gap
  - Jump Back (N)
  - Jump Forward

Example

Generate Identical

instead of

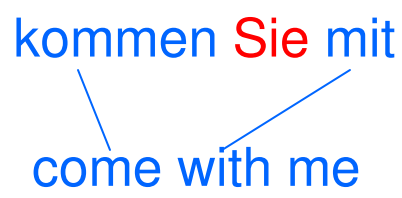
Generate (**Portland** , Portland)

If count (**Portland**) = 1

List of Operations

- 5 Translation Operations
  - Generate (X,Y)
  - Continue Source Cept
  - Generate Identical
  - Generate Source Only (X)
  - Generate Target Only (Y)
- 3 Reordering Operations
  - Insert Gap
  - Jump Back (N)
  - Jump Forward

Example



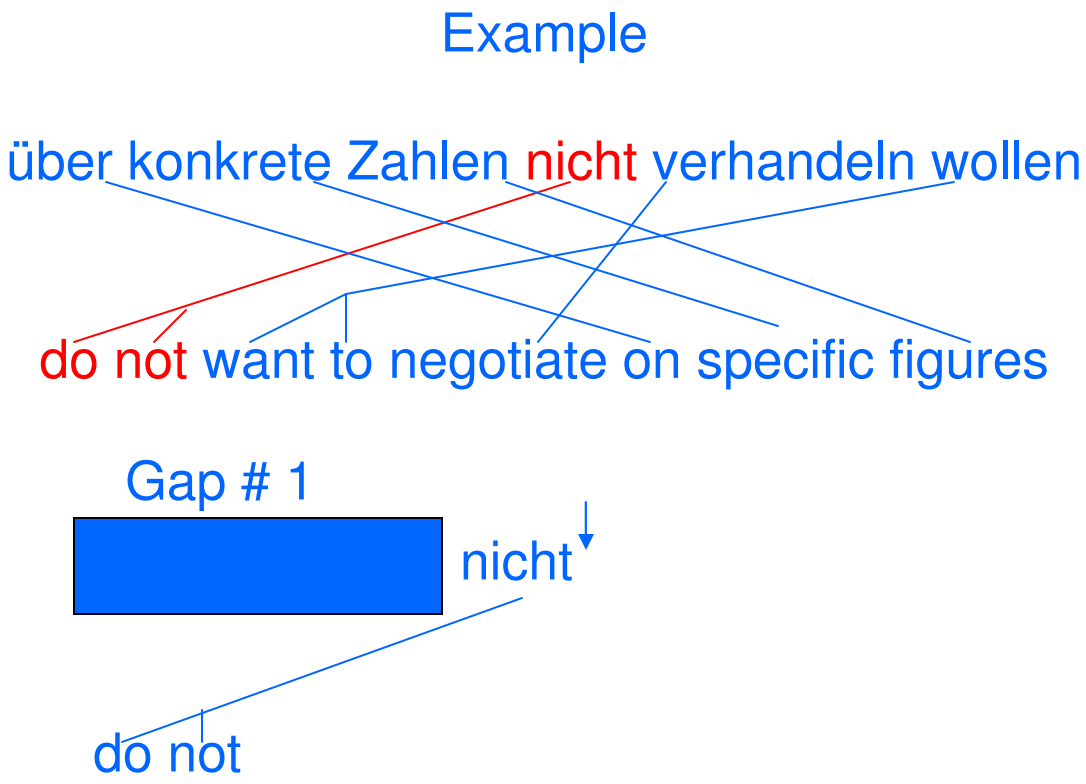
Generate Source Only (Sie)

List of Operations

- 5 Translation Operations
  - Generate (X,Y)
  - Continue Source Cept
  - Generate Identical
  - Generate Source Only (X)
  - Generate Target Only (Y)
- 3 Reordering Operations
  - Insert Gap
  - Jump Back (N)
  - Jump Forward

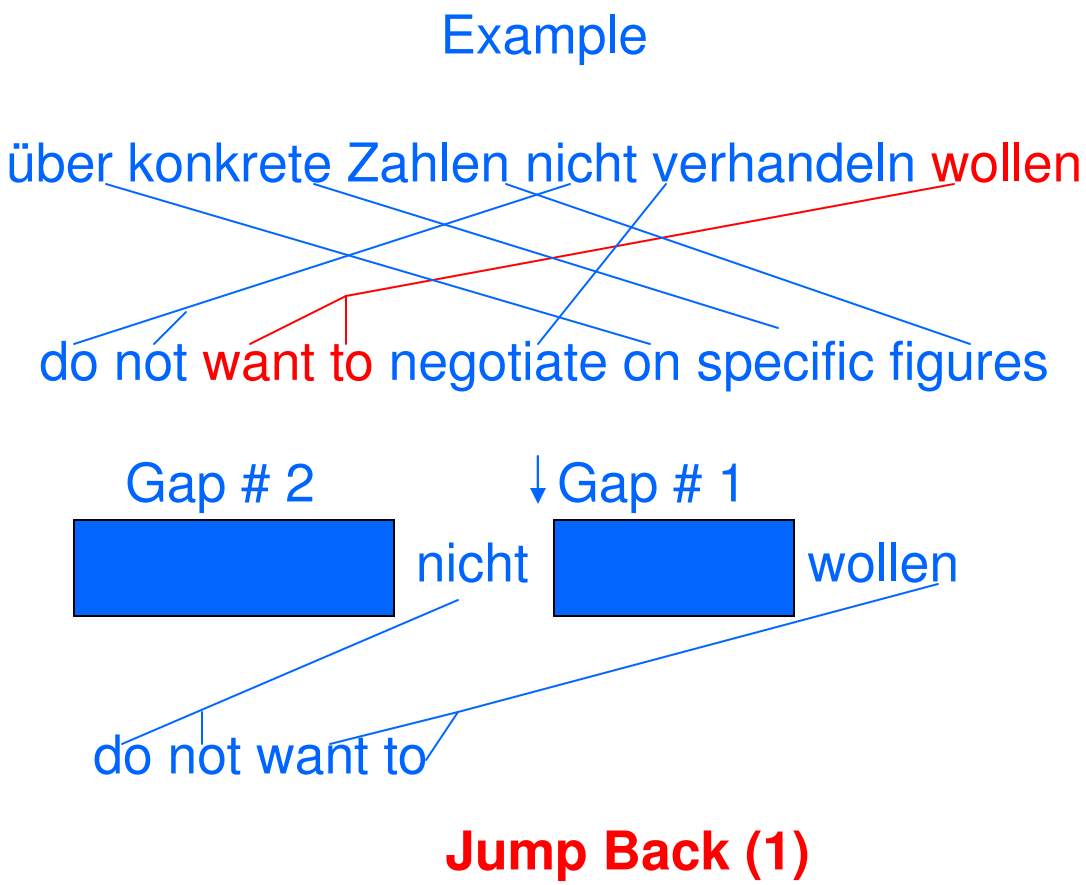
List of Operations

- 5 Translation Operations
  - Generate (X,Y)
  - Continue Source Cept
  - Generate Identical
  - Generate Source Only (X)
  - Generate Target Only (Y)
- 3 Reordering Operations
  - Insert Gap
  - Jump Back (N)
  - Jump Forward



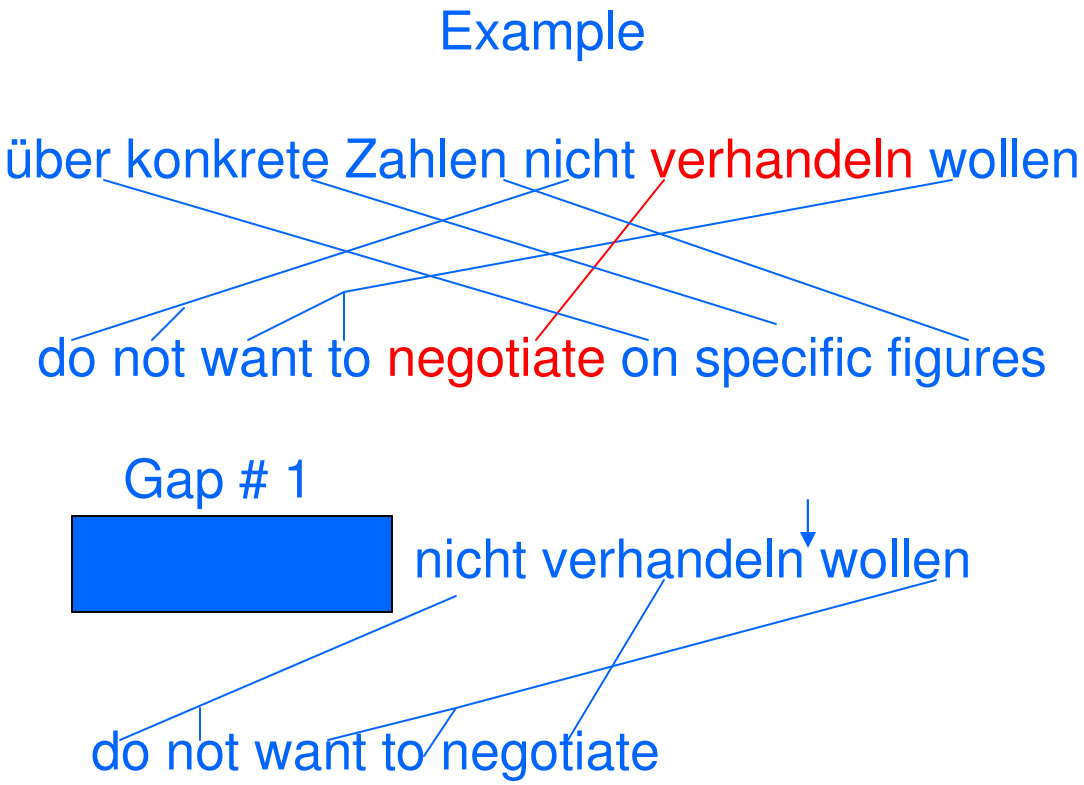
List of Operations

- 5 Translation Operations
  - Generate (X,Y)
  - Continue Source Cept
  - Generate Identical
  - Generate Source Only (X)
  - Generate Target Only (Y)
- 3 Reordering Operations
  - Insert Gap
  - Jump Back (N)
  - Jump Forward



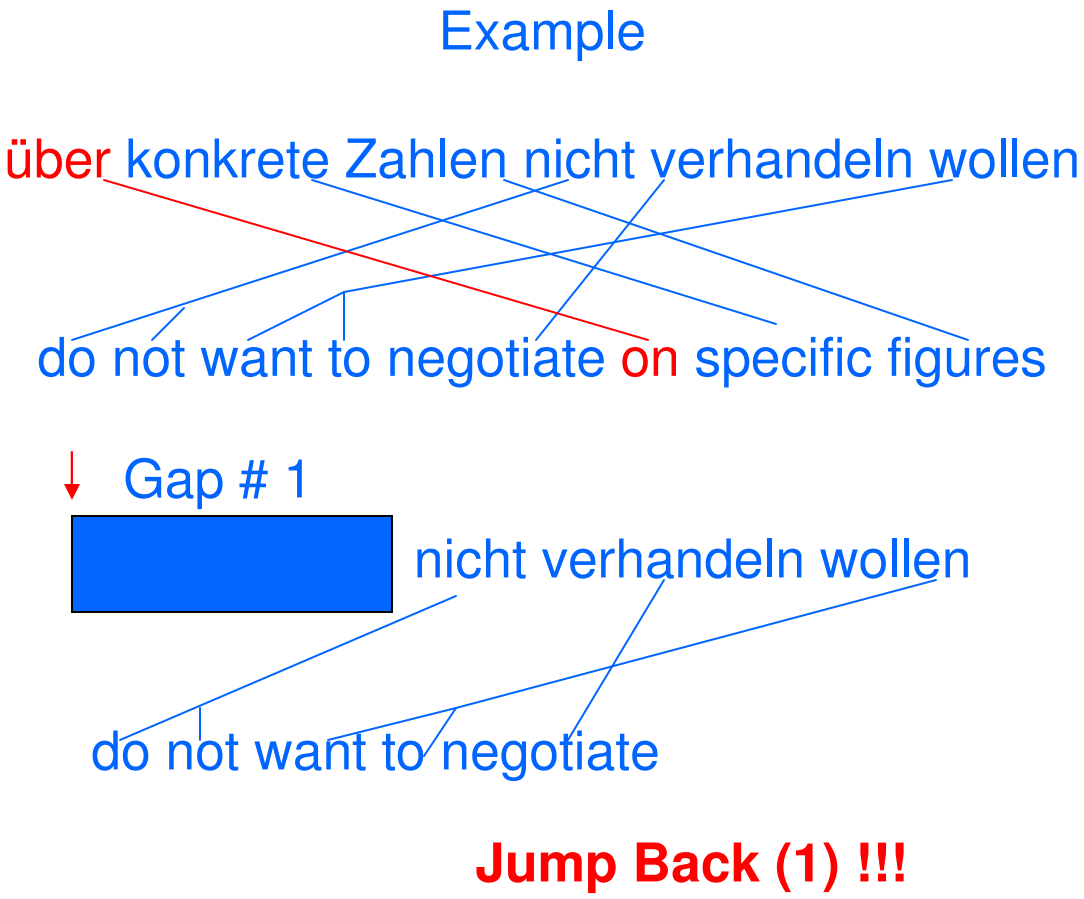
List of Operations

- 5 Translation Operations
  - Generate (X,Y)
  - Continue Source Cept
  - Generate Identical
  - Generate Source Only (X)
  - Generate Target Only (Y)
- 3 Reordering Operations
  - Insert Gap
  - Jump Back (N)
  - Jump Forward



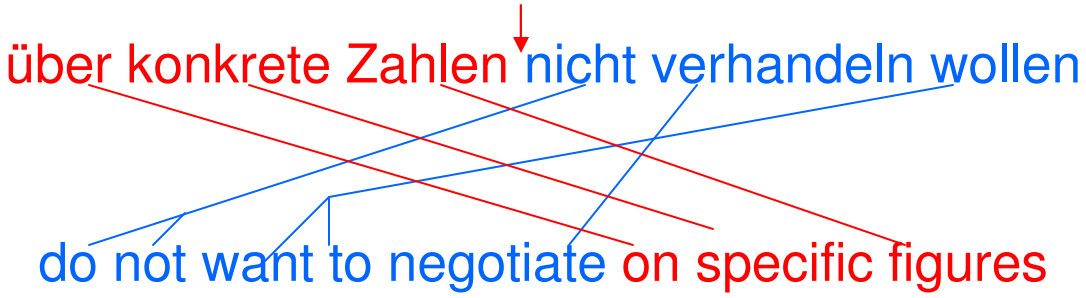
List of Operations

- 5 Translation Operations
  - Generate (X,Y)
  - Continue Source Cept
  - Generate Identical
  - Generate Source Only (X)
  - Generate Target Only (Y)
- 3 Reordering Operations
  - Insert Gap
  - Jump Back (N)
  - Jump Forward



List of Operations

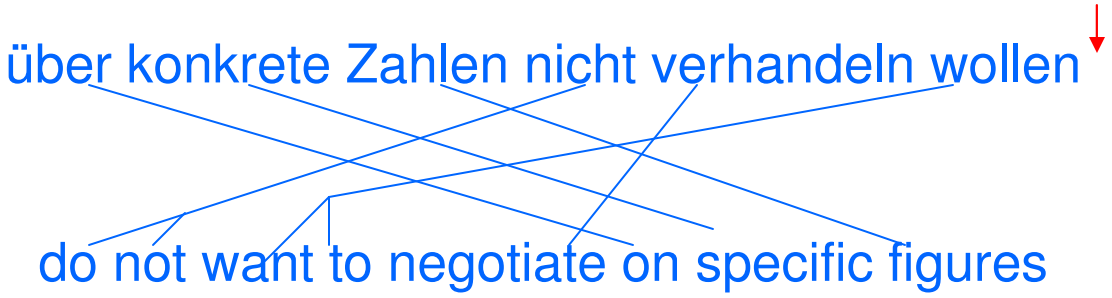
- 5 Translation Operations
  - Generate (X,Y)
  - Continue Source Cept
  - Generate Identical
  - Generate Source Only (X)
  - Generate Target Only (Y)
- 3 Reordering Operations
  - Insert Gap
  - Jump Back (N)
  - Jump Forward



List of Operations

- 5 Translation Operations
  - Generate (X,Y)
  - Continue Source Cept
  - Generate Identical
  - Generate Source Only (X)
  - Generate Target Only (Y)
- 3 Reordering Operations
  - Insert Gap
  - Jump Back (N)
  - Jump Forward

Jump Forward



List of Operations

- 5 Translation Operations
  - Generate (X,Y)
  - Continue Source Cept
  - Generate Identical
  - Generate Source Only (X)
  - Generate Target Only (Y)
- 3 Reordering Operations
  - Insert Gap
  - Jump Back (N)
  - Jump Forward

OSM learns phrases as operation sequences

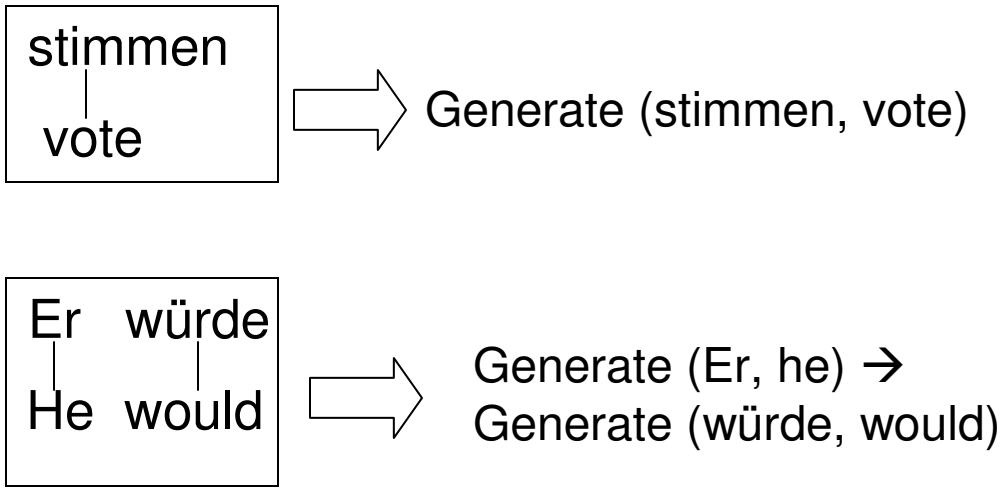
- Although model is based on MTUs, we can memorize phrases.

OSM learns phrases as operation sequences

- Although model is based on MTUs, we can memorize phrases.

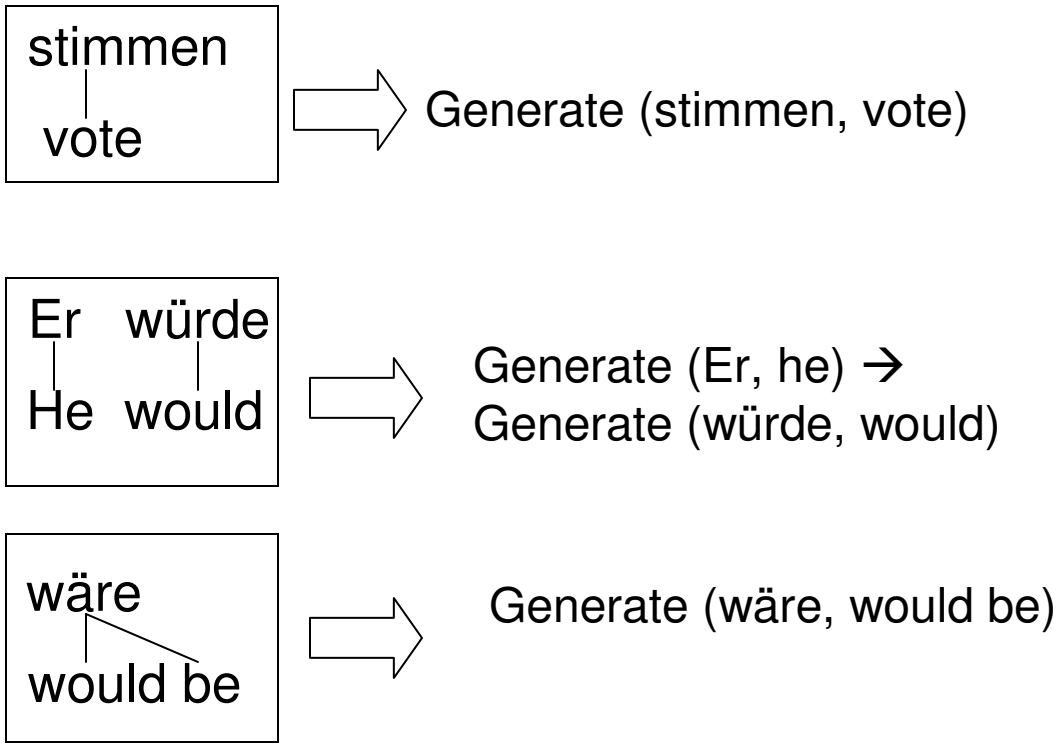
OSM learns phrases as operation sequences

- Although model is based on MTUs, we can memorize phrases.



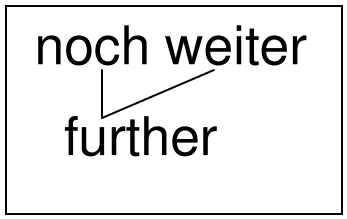
OSM learns phrases as operation sequences

- Although model is based on MTUs, we can memorize phrases.



OSM learns phrases as operation sequences

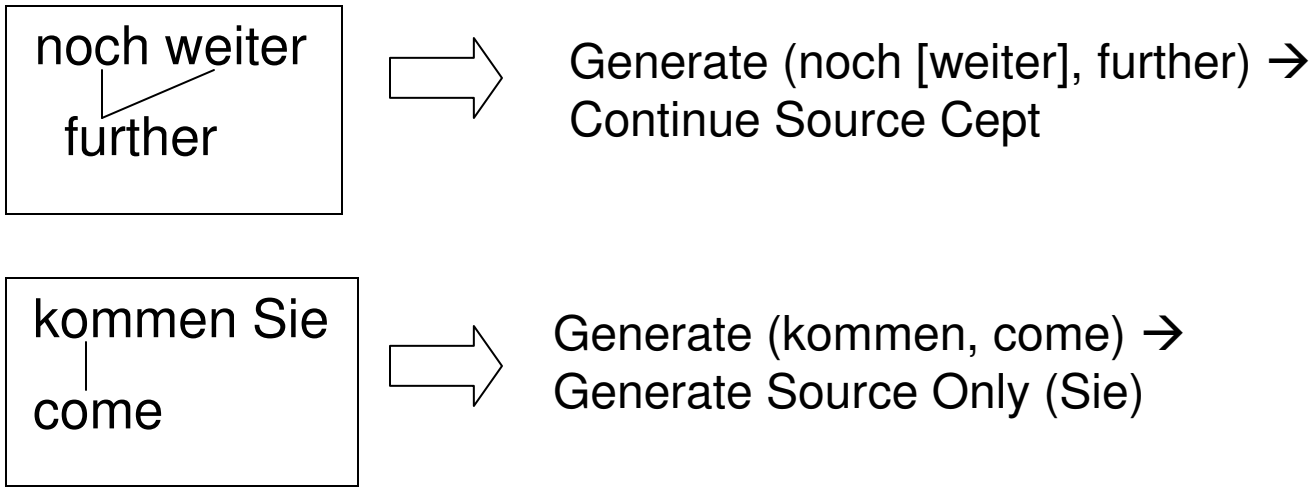
- Although model is based on MTUs, we can memorize phrases



Generate (noch [weiter], further) →
Continue Source Cept

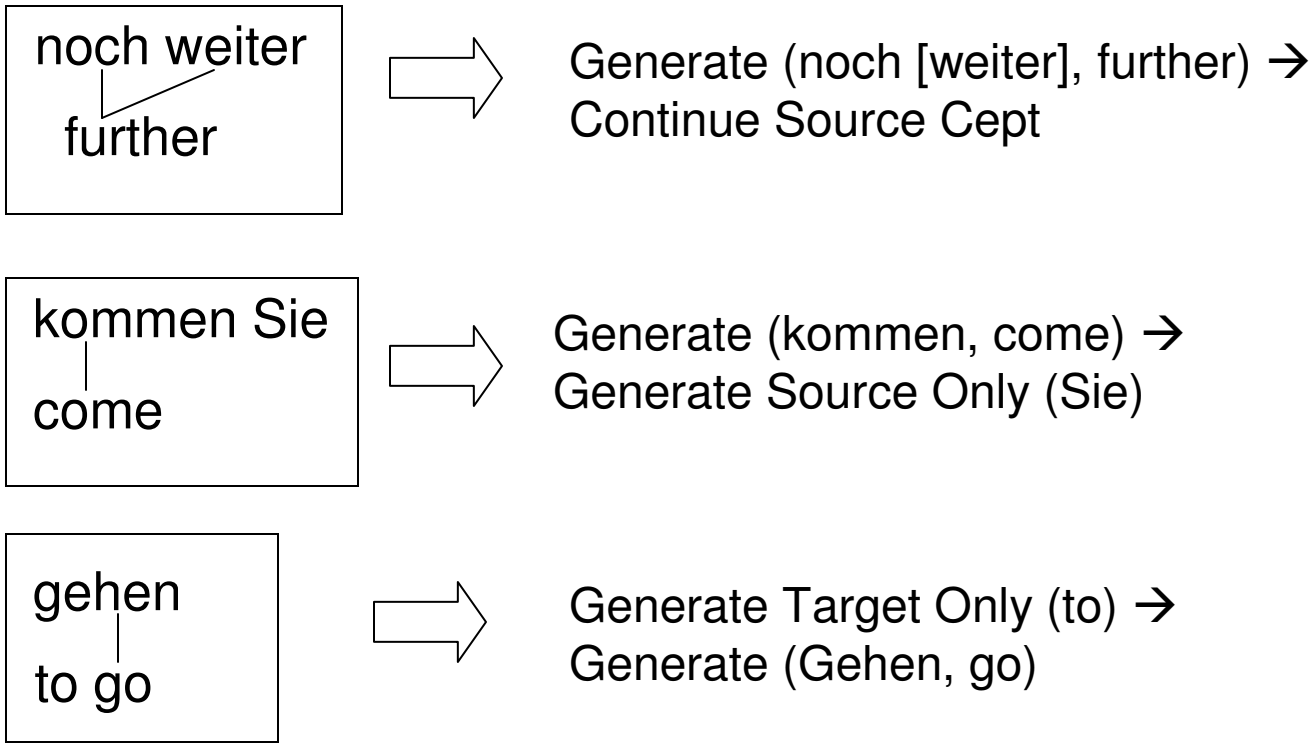
OSM learns phrases as operation sequences

- Although model is based on MTUs, we can memorize phrases



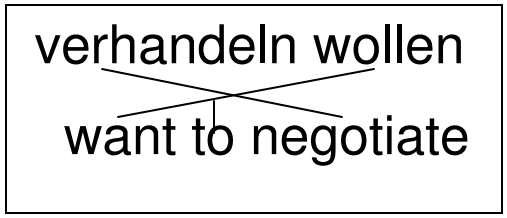
OSM learns phrases as operation sequences

- Although model is based on MTUs, we can memorize phrases



OSM learns phrases as operation sequences

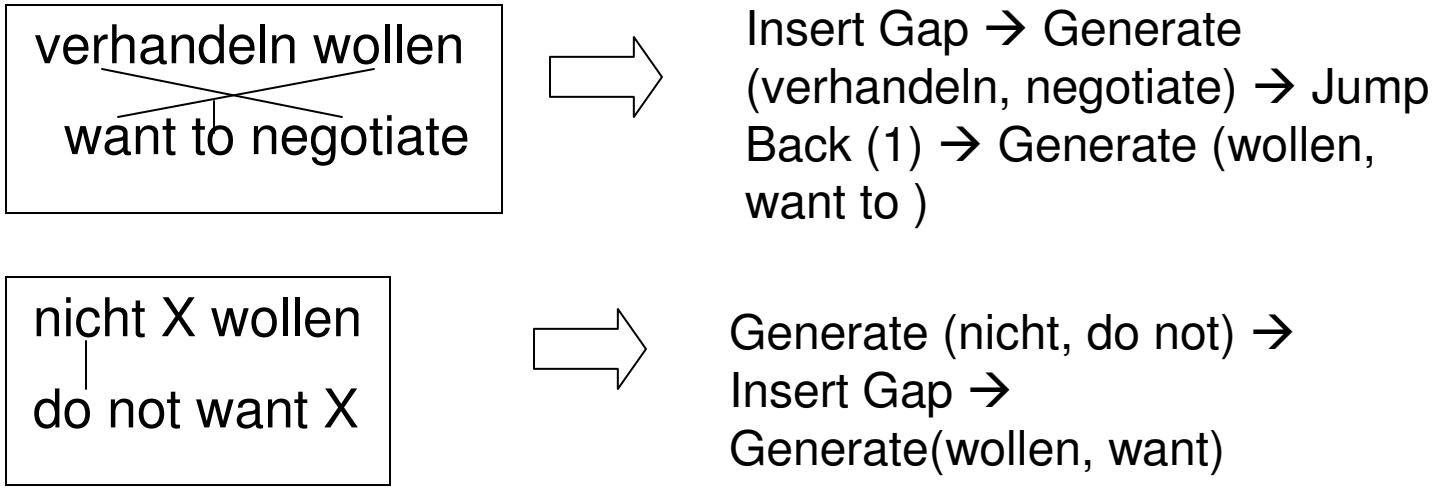
- Although model is based on MTUs, we can memorize phrases



Insert Gap → Generate (verhandeln, negotiate) → Jump Back (1) → Generate (wollen, want to)

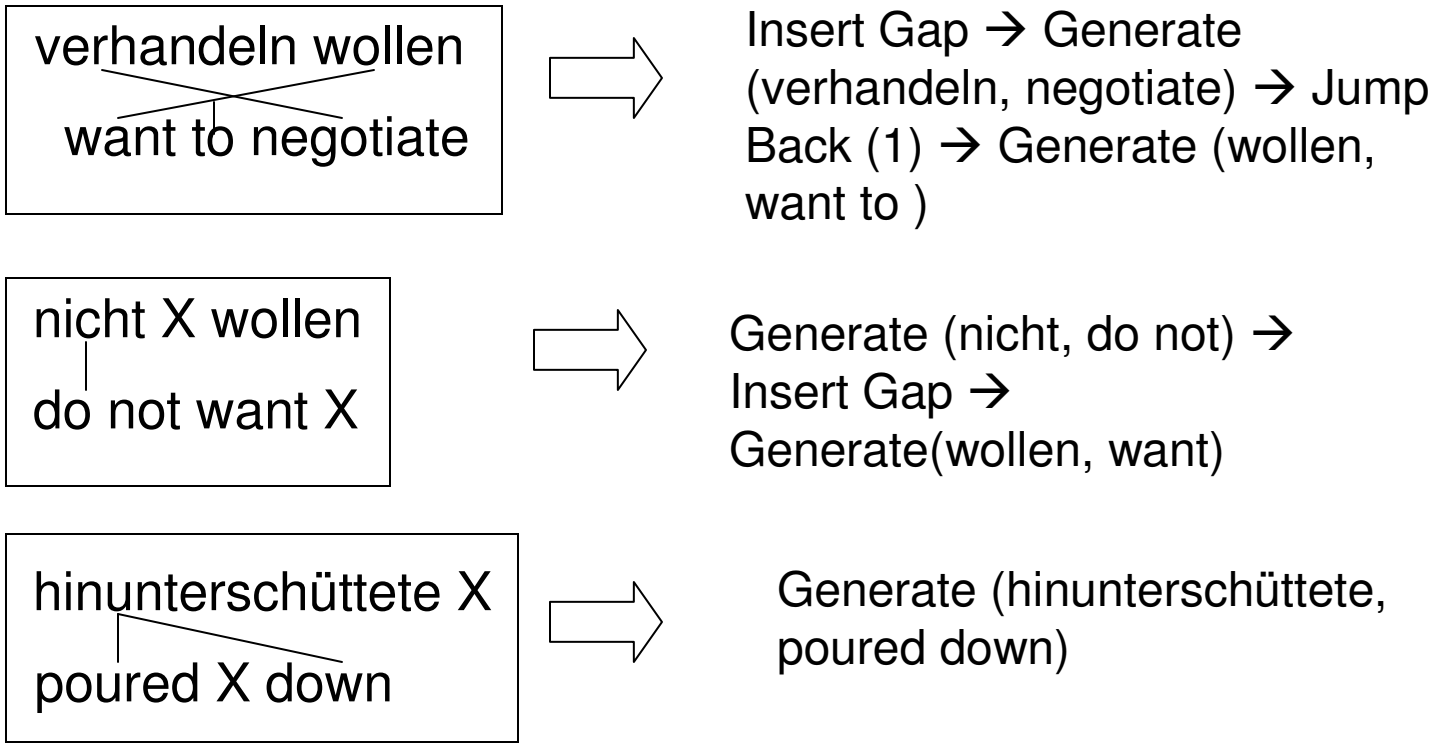
OSM learns phrases as operation sequences

- Although model is based on MTUs, we can memorize phrases



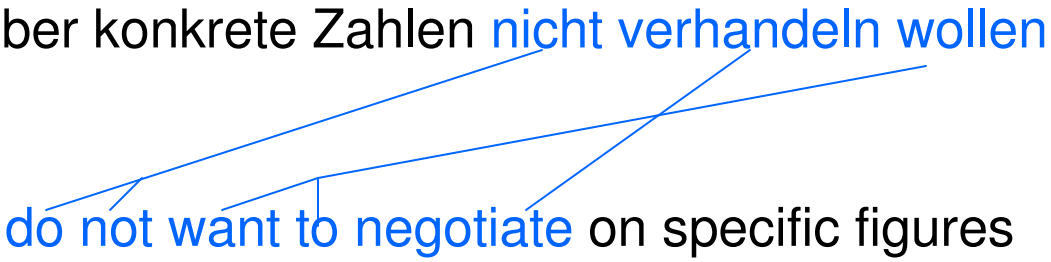
OSM learns phrases as operation sequences

- Although model is based on MTUs, we can memorize phrases



OSM learns phrases as operation sequences

- Although model is based on MTUs, we can memorize phrases



Phrase pair : nicht verhandeln wollen ~ do not want to negotiate
Generate (nicht , do not) → Insert Gap → Generate (wollen , want to) → Jump Back(1)
→ Generate (verhandeln , negotiate)

Phrase pair : nicht X wollen ~ do not want to X
Generate (nicht , do not) → Insert Gap → Generate (wollen , want to)

Phrase pair : verhandeln wollen ~ want to negotiate
Insert Gap → Generate (wollen , want to)
→ Jump Back(1) → Generate (verhandeln , negotiate)

Using OSM in a discriminative model

$$\hat{E} = \arg\max_{E} \left\{ \sum_{j=1}^{J} \lambda_j h_j(F, E) \right\}$$

- Distance-based penalty
- Lexical probability models (Lexical weighting – standard IBM Models)
- Monolingual language model
- Operation sequence model
- Prior probability model
- Gap penalty
- Deletion penalty
- Length-based penalty

Decoders

- Cept-based decoder (or MTU-based decoding)
- Phrase-based decoder
- Moses

Decoding

- MTU-based decoding
  - extends current hypothesis by one MTU
- Phrase-based decoding
  - extends current hypothesis by one phrase
- Phrase-based decoding works better than MTU-based decoding
  - Example: Wie heißen Sie – What is your name
  - MTUs: Wie/What-is, heißen/name, Sie/your
  - “Wie/What-is” is very unlikely, so it will get pruned (or you need a large stack size)
  - The phrase pair: “Wie heißen Sie – What is your name” is likely
    - Makes search easier

Advantages of phrase-based decoding versus MTU-based decoding

- Better future cost estimation
- Better translation coverage
- Lower beam size
- Better handling of unaligned and discontinuous target words

Experimental setup

- Language pairs: German, Spanish and French to English

- Training and test data
  - 8th version of the Europarl corpus
  - Bilingual Data: ~2M parallel sentences (Europarl data + newsdata)
  - Monolingual Data: 22M: news
  - WMT news 2008 for tuning
  - Testing on WMT news 2009-2012

Training & tuning

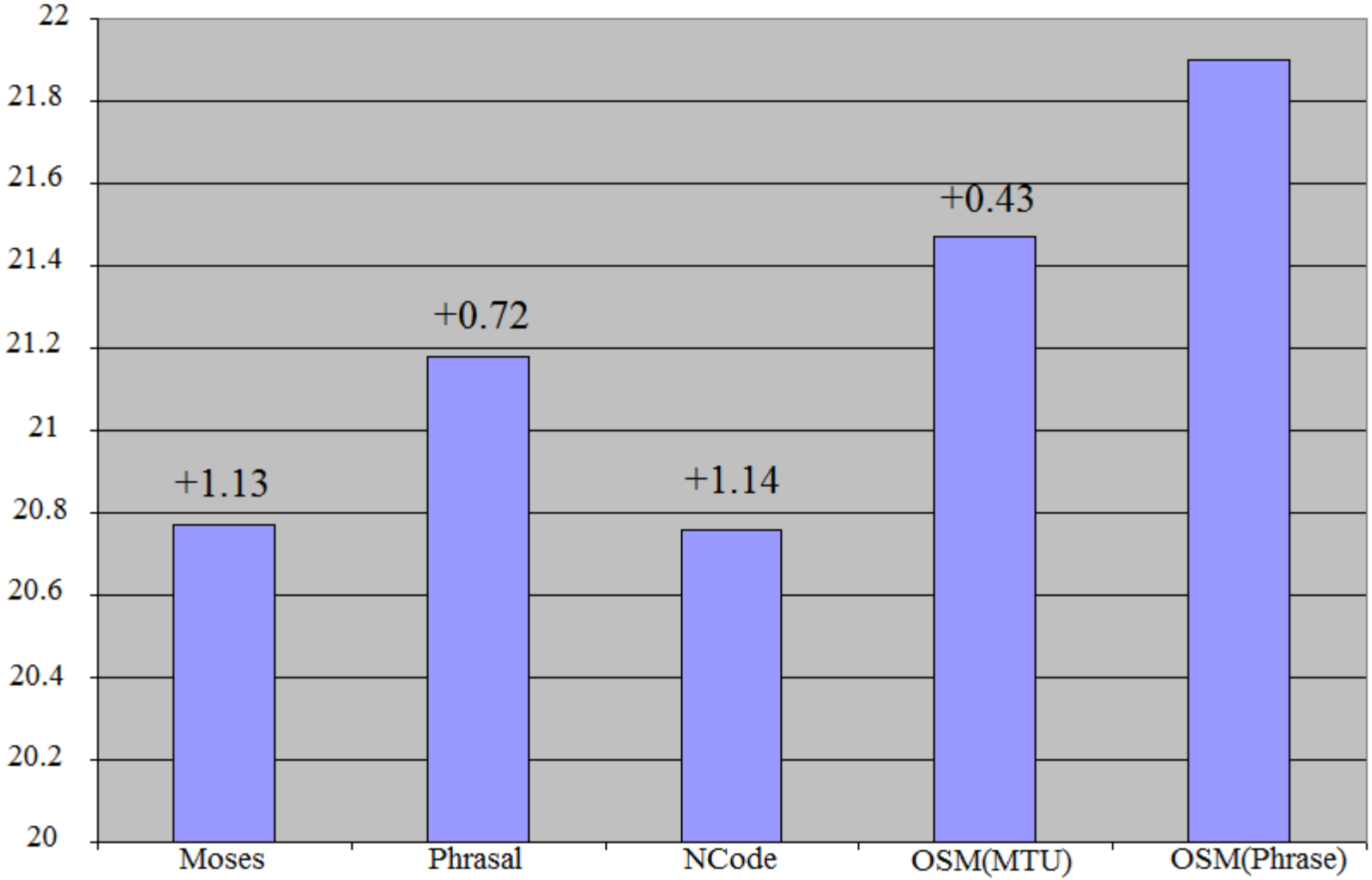
- Giza++ for word alignment symmetrized with GDFA
- Convert word-aligned bilingual corpus into operation corpus
  - Deterministic algorithm
- SRI toolkit to train n-gram language model and OSM model
  - Kneser-Ney smoothing
- Parameter tuning with Z-mert

Baseline systems

- Moses
  - with lexicalized reordering (Koehn et. al 2005)
- Phrasal
  - with hierarchical lexicalized reordering (Galley and Manning 2008)
  - discontinuous phrases (Galley and Manning 2010)
- NCode
  - with lexicalized reordering (Crego and Yvon 2010)

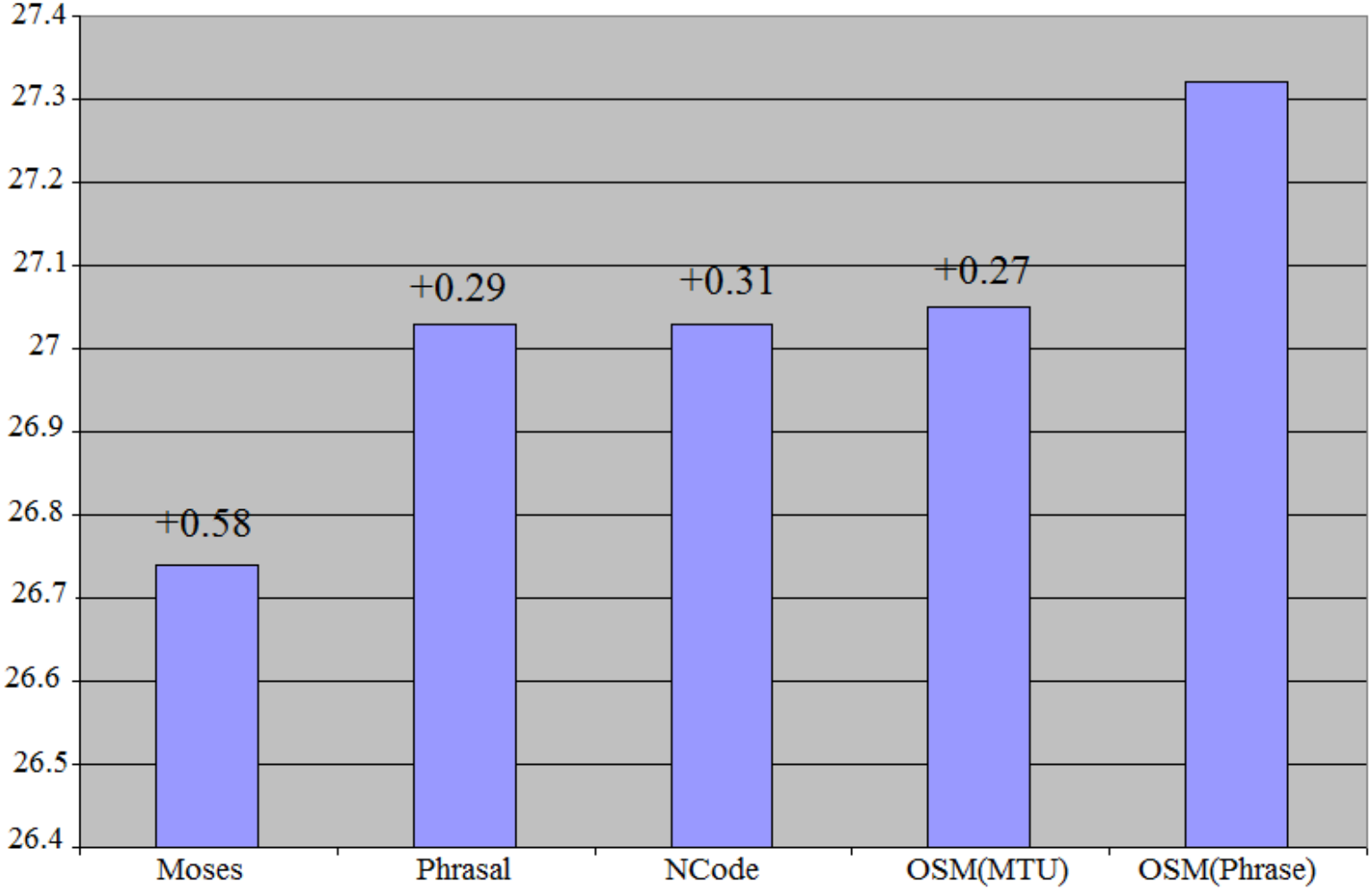
German-to-English

- Significant improvements over all baselines



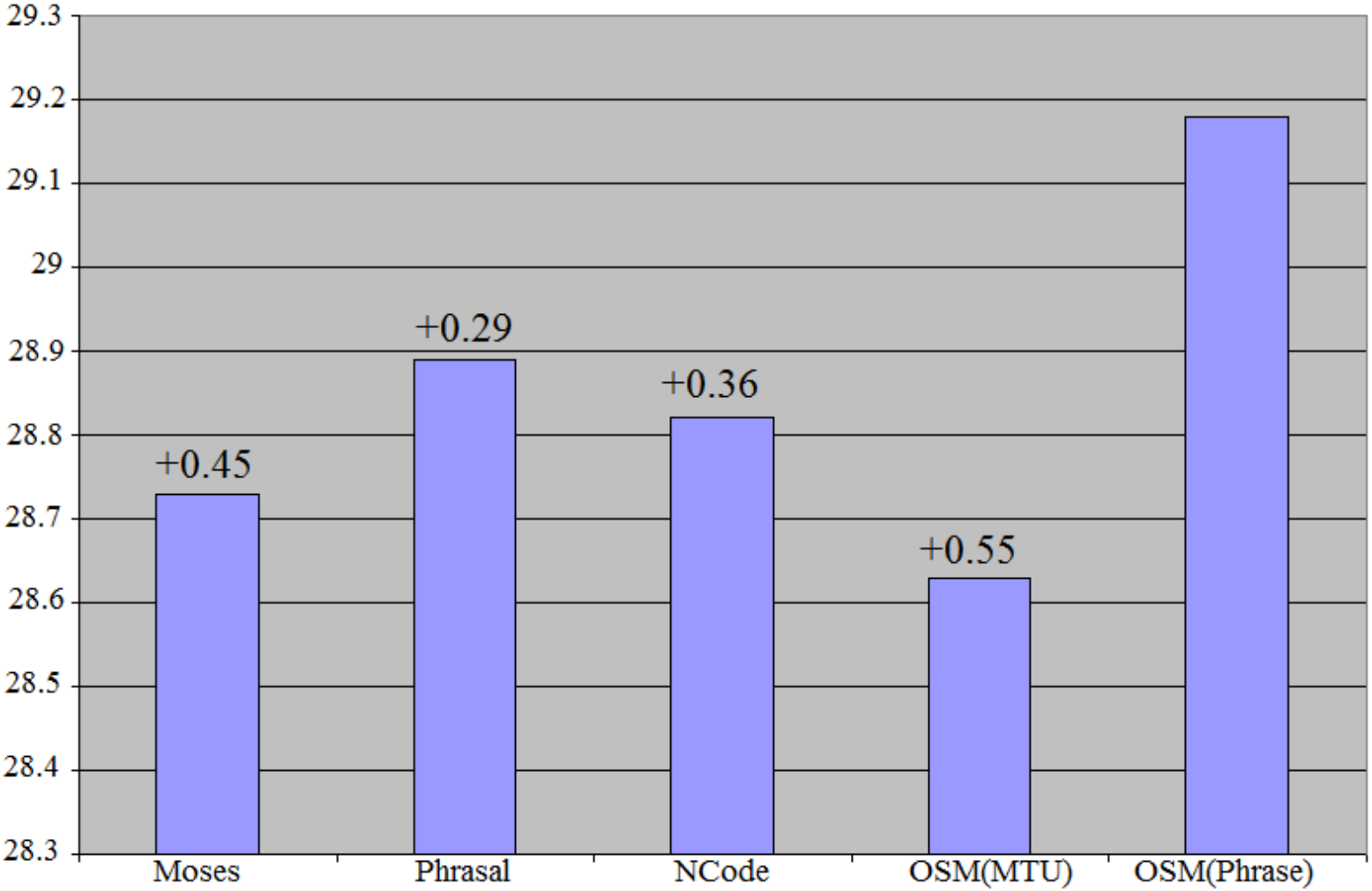
French-to-English

- Significant improvement in 11/16 cases



Spanish-to-English

- Significant improvement in 12/16 cases

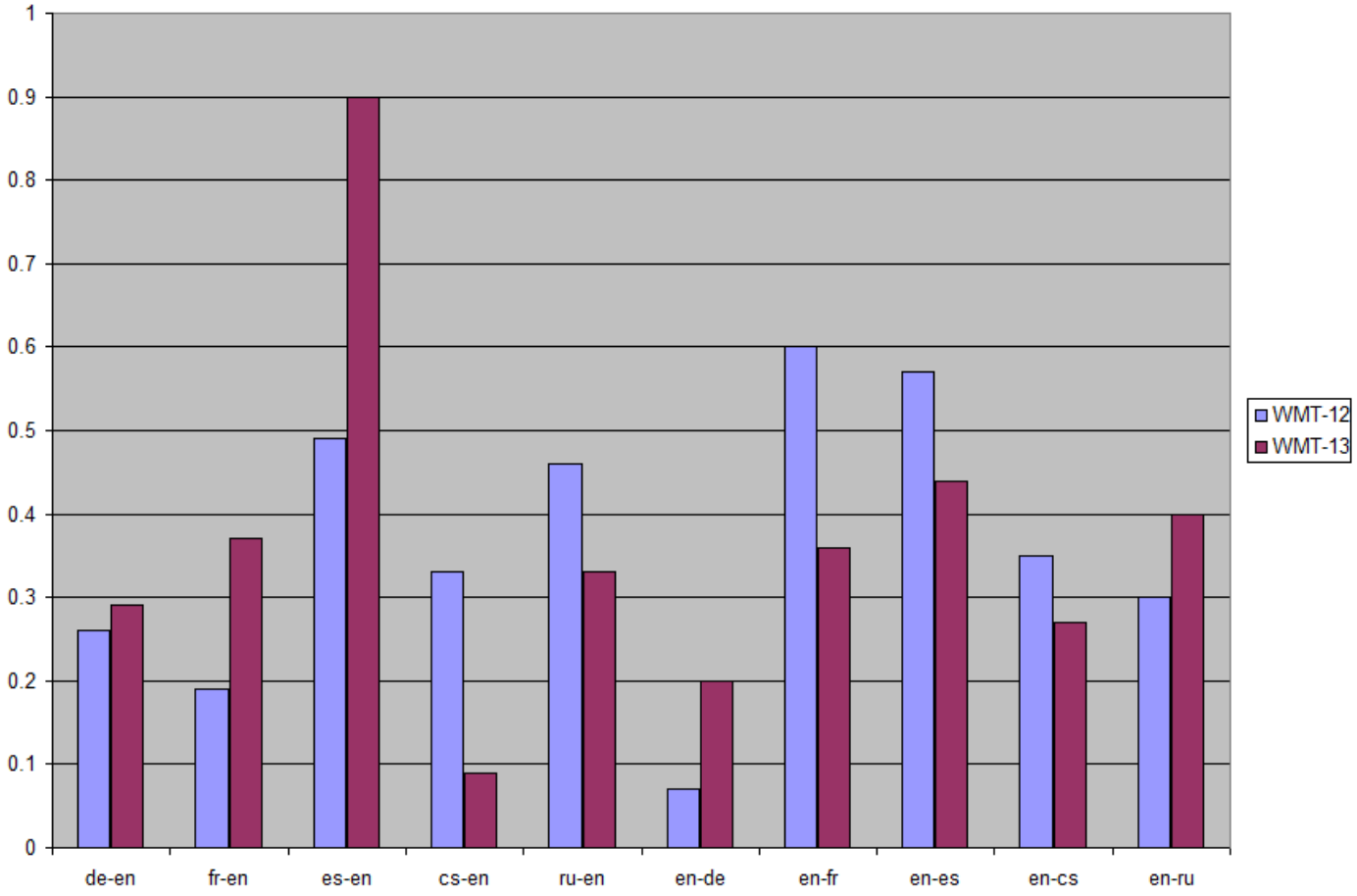


OSM Decoder – Participation in WMT-13

Lang	Evaluation					
	Automatic			Human		
	BLEU	$Rank_u$	$Rank_c$	Win Ratio	$Rank_u$	$Rank_c$
DE-EN	27.6	9/31	8/29	0.562	6-8/17	3-4/14
ES-EN	30.4	6/12	5/12	0.569	3-5/12	1-2/9
CS-EN	26.4	3/11	2/10	0.581	2-3/11	1-2/8
RU-EN	24.5	8/22	8/18	0.534	7-9/19	5-6/12
EN-DE	20.0	6/18	4/14			
EN-ES	29.5	3/13	3/12	0.544	5-6/13	2-3/10
EN-CS	17.6	14/22	2/9	0.517	4-6/12	2-3/5
EN-RU	18.1	6/15	5/13	0.456	9-10/14	
EN-FR	30.0	7/26	6/21	0.541	5-9/16	4-6/8

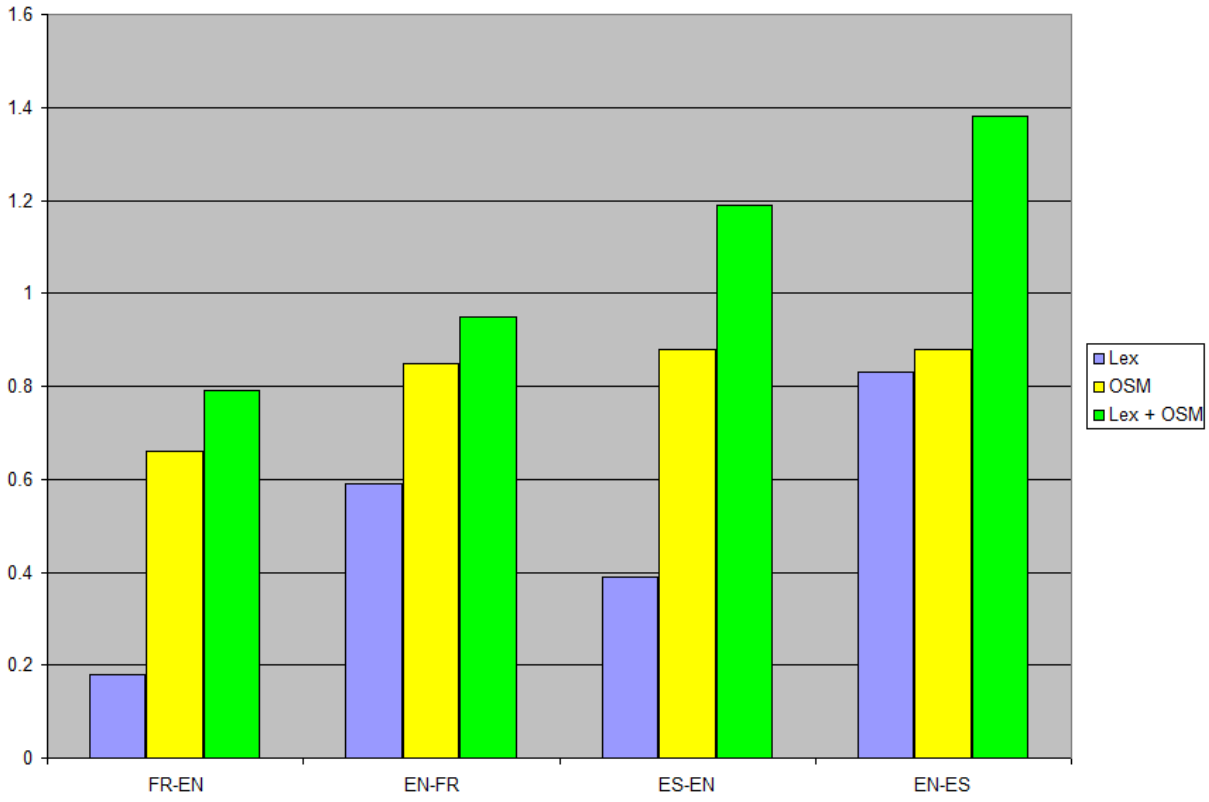
Integration into Moses (Durrani et al., ACL2013)

- Large scale evaluation (WMT-12/13)
- Average gain of +0.40 over submission quality baseline system over 10 language pairs – Significant gains in most cases



Comparison with lexicalized reordering model

- Baseline = Distortion based reordering model
  - OSM outperforms lexical reordering modeling in all four language pairs (yellow vs. purple)
  - Gains are additive (Green)



Summary: The OSM model …

- Integrates translation and reordering in a single generative story
- Uses bilingual context (like N-gram based SMT)
- Has an improved reordering mechanism
  - Models local and non-local dependencies uniformly
  - Takes previous translation decisions into account (like N-gram SMT)
  - Takes previous reordering decisions into account (unlike N-gram SMT)
  - Has ability to memorize lexical triggers (like phrase-based SMT)
  - Considers all possible reorderings during search

Summary: The OSM model …

- Does not have spurious phrasal segmentation (like N-gram SMT)
- Does not need ad-hoc limits during search (unlike N-gram and phrase-based)
- Supports discontinuous translation units
- Handles unaligned translation units through built-in insertion and deletion models

Conclusion

- OSM = union of N-gram-based SMT and phrase-based SMT
- Statistically significant improvements over baseline systems
- Phrase-based decoding+OSM is an effective combination of an MTU-based model and phrase-based-search
- Can be used as a feature in any left-to-right decoding mechanism

- Operation sequence model is available in the latest version of Moses