SYNTAX AND MORPHOLOGY IN MT

David Talbot

Spring 2017

Yandex School of Data Analysis

SYNTAX

- *Syntax* describes how words combine to form sentences
- Syntax determines word-order in most languages

John loves Mary -> loves(John = subject, Mary = object)

Mary loves John -> loves(John = object, Mary = subject)

SYNTAX

- Syntax determines *agreement* in many languages

  John goes home -> agrees_number(John, goes)

  Анна пошла домой -> agrees_gender(Анна, пошла)

- Morphology describes how words change form

  verb conjugations: amo, amas, amat, etc.

  noun declensions: dominus, domine, dominum, etc.

- Syntax determines where agreement occurs
- Morphology is used to mark the agreement

Barack Obama est *né* à Hawaï.

Margaret Thatcher est *née* au Royaume Uni.

- Syntax determines the roles of words in a sentence
- Morphology can mark these syntactic roles explicitly

Barack Obama urodził się *na* Hawa*jach*

Margaret Thatcher urodziła się *w* Wielk*iej* Brytan*ii*

PROBLEMS WITH SYNTAX AND MORPHOLOGY

- Word order differs significantly across languages
- A priori search space is huge (factorial)
- Morphological agreement is difficult to predict
- Both source and target side features needed (more later)

WORD ORDER

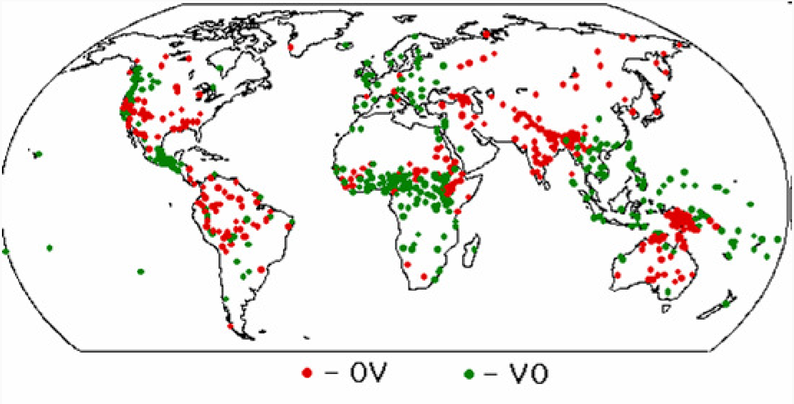
- Word order differs widely across languages

  SOV: John the ball hit (Japanese, Turkish, Uzbek) [45%]

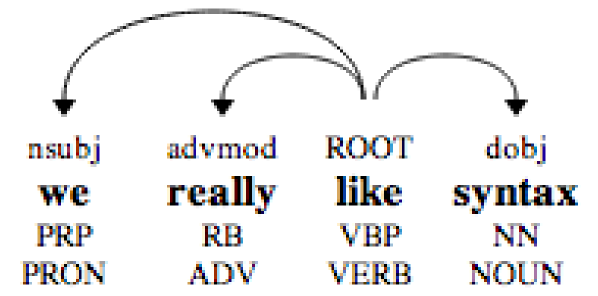
  SVO: John hit the ball (English, Mandarin, Russian) [42%]

  VSO: hit John the ball (Arabic, Tagalog, Welsh) [9%]

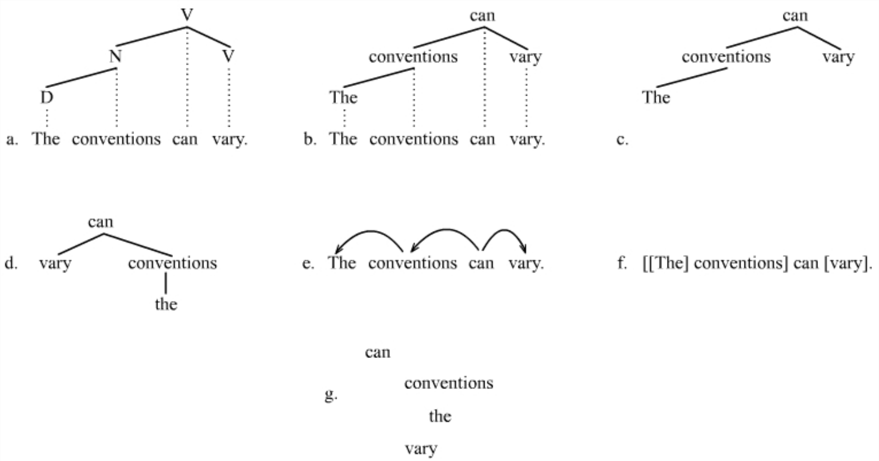
Distribution of OV and VO languages



- Arrows point from head to child
- Labels indicate the 'role' of the child



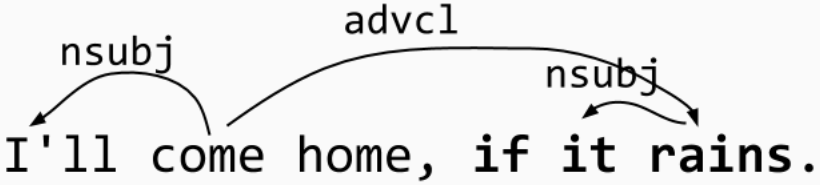
DEPENDENCY TREES



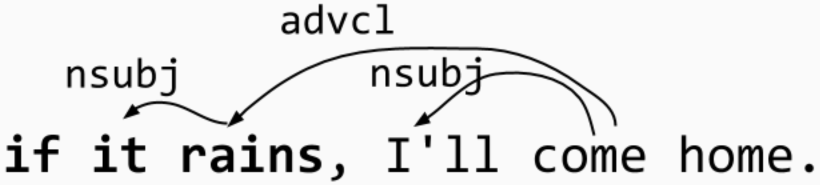
PREDICTING WORD ORDER

- Use source side (English) dependency tree
- Learn / write rules to reorder source words
- Hierarchical (tree) structure of language is more efficient

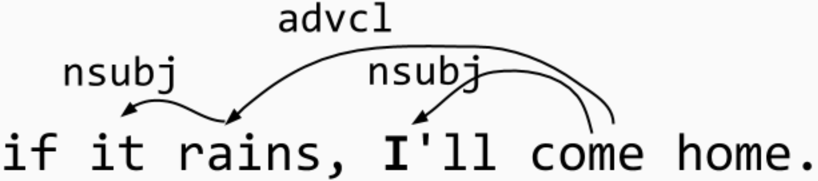
- Re-write rules expressed over roles
  Rule: ’Move advcl before ROOT’



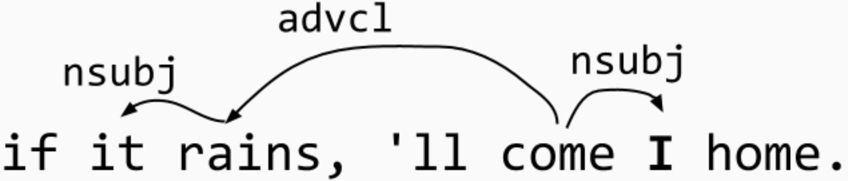
- One operation can move whole clause:
  Rule: ’Move advcl before ROOT’



- Reordering can be applied recursively
  Rule: ’Move nsubj after its head’

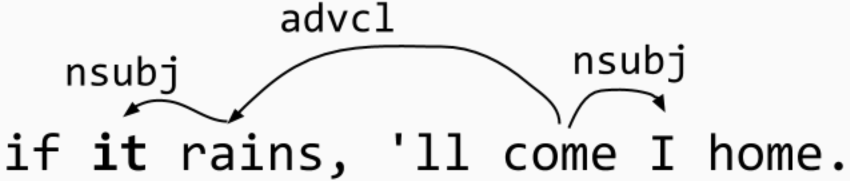


- Reordering can be applied recursively
  Rule: ’Move nsubj after its head’

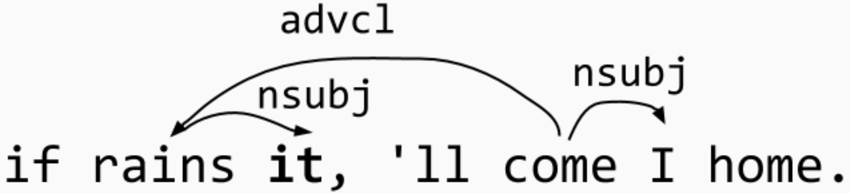


PREDICTING WORD ORDER

- Reordering be done recursively
  Rule: ’Move nsubj after its head’



- Reordering can be applied recursively
  Rule: ’Move nsubj after its head’



- Reorder all source sentences in corpus at the beginning
- Word align the *reordered* corpus
- Extract phrases directly from *reordered* text
- Apply same reordering to any new source sentence to find matching *reordered* phrases for translation

PRE-ORDERING

- Makes word alignment easier (closer to the diagonal)
- Reduces the need for ‘non-contiguous’ phrases
- Amazing results for SVO -> SOV (English -> Japanese)

PRE-ORDERING: JAPANESE EXAMPLE

Source: He went home by car after the class finished.
Reordered: He the class finished after car by home went.

Japanese: Kare wa kurasu ga owatta ato, kuruma de ie ni ikimashita.
Gloss: He [topic] class [subject] finished after car by home [to] went.

- Phrase based system can capture nearby reordering: (by car) -> (car by), (went home) -> (home [to] went)
- Pre-ordering gets long-range reordering
- Pre-ordering increases number of useful phrases in model

PRE-ORDERING

- Where do we get the rules from?
- Can we deal with all reordering on source side?

SOURCE-SIDE ONLY REORDERING?

- Some reordering depends on the choice of translation
- French adjectives mean different things depending on position:
  épisode dernier -> final (last) episode
  dernier épisode -> previous (last) episode
- Keep multiple reorderings of the source
- Translate each reordered version and pick the best one afterwards (using language model)

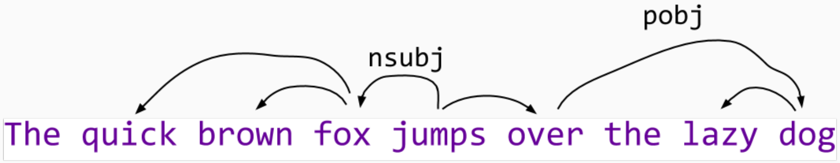
SOURCE-SIDE MORPHOLOGY

- Problem when translating from Russian
- Apply some form of stemming
- Remove those distinctions that don’t affect translation, e.g.
  Cluster ’собака’, ’собаку’, ’собаке’ -> ’собака’
  Cluster ’собак’, ’собаки’, ’собаками’, -> ’собаки’
  But don’t cluster ’собак’, ’собаку’, ’собаками’ together.
- Learn these clusters from data?

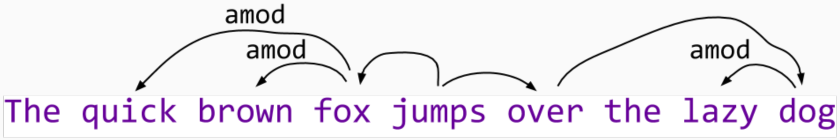
Translating from English to Russian what features can we use to predict the following on the target output?

- The number (singular, plural) of a noun?
- The case (nominative, genitive, etc.) of a noun?
- The gender (masculine, feminine, neuter) of a noun?
- The gender of an adjective?

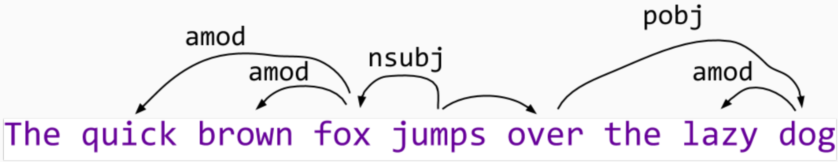
- Predict case from ’nsubj’, ’dobj’, ’pobj’
  fox -> лиса (nominative)
  dog -> собаку (assuming the verb and preposition are ’прыгает через’)



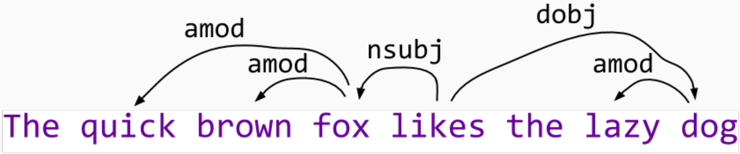
- Predict gender, number, case agreement across ’amod’ arcs (actual values depend on the nouns)
  brown -> feminine, nominative, singular (given лиса)
  lazy -> feminine, accusative, singular (given собаку)



- Changing verb could change case assignment completely
  fox -> лиса (nominative)
  dog -> собаку (assuming the verb and preposition are 'прыгает через')



- Changing verb could change case assignment completely (e.g. inversion)
  fox -> лисе (dative)
  dog -> собака (nominative assuming the verb is ’нравится’)



Building and evaluating a dependency based reorderer

You are provided with:

- Some English sentences with dependency parses and word alignments to Japanese
- Some automatic ’reorderings’ of the English sentences (generated from the word alignments)
- Some scripts and data structures for building a reorderer and evaluating it.

HOMEWORK 2

You are required to do some of the following:

- Implement a recursive algorithm for reordering the nodes of a tree (see reorderer.py)
- Experiment with different implementations of the core reordering function (see reorderer.py)
- Evaluate different reordering algorithms (reverse all nodes, rule-based SOV, machine learned)
- Implement more evaluation metrics (E.g. METEOR)
- Write a report describing what you did and why.

Useful to evaluate reordering separately from translation

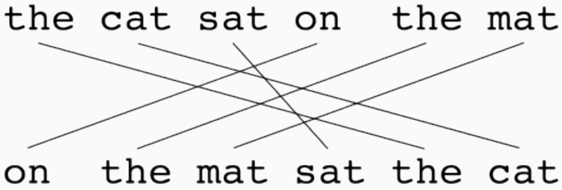
- Generate a 'reference reordering' r for each source sentence by rearranging it according to some word alignments with the target language
- Given a candidate reordering r' compute some distance metric $d(r, r')$
- Correlation metrics such as Kendall's tau can be used e.g.

$$\tau(r, r') = \frac{|\text{ordered pairs} \in r \cap \text{ordered pairs} \in r'|}{|\text{ordered pairs} \in r|}$$

METEOR style score:

- Given a reference r and a candidate r' align r and r'
- Let c be the number of contiguous chunks, i.e. 3 here: (the cat), (sat), (on the mat)
- Let $|r|$ be the number of words in the reference

$$d(r, r') = 1 - \frac{c - 1}{|r| - 1}$$



REFERENCES

- Using a Dependency Parser to Improve SMT for Subject-Object-Verb languages, Xu et al. 2009
- Head Finalization: A Simple Reordering Rule for SOV Languages, Hideki Isozaki et al. 2010
- A Lightweight Evaluation Framework for Machine Translation Reordering Talbot et al. 2011
- Source-Side Classifier Preordering for Machine Translation, Uri Learner and Slav Petrov 2013.