Rule-based Reordering Space in Statistical Machine Translation

Nicolas Pécheux[1,2] *Alexandre Allauzen*[1,2] *François Yvon*[1]
(1) LIMSI-CNRS, (2) Université Paris-Sud

Highlights

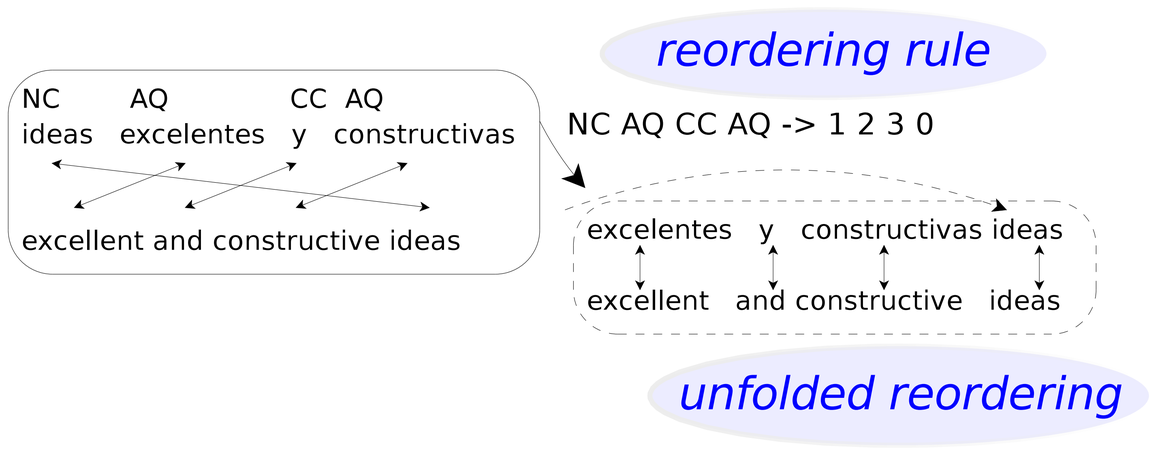
- **How to design the space of reorderings** to be explored in Machine Translation ?
- Computationnal issues, but also a **trade-off between coverage and decoding errors**.
- Empirical and oracle comparison of different reordering spaces.

Data and sytem description

- English-French & English-German
- Ncode, a state-of-the n-gram-based SMT system
- Train: NewsCommentary
  Dev/Test: newstest2009/newstest2010

Reordering lattice generation

- **Reordering in SMT is difficult to study**:
  → preordering + local reordering withing phrases + phrases reordering
- We **split reordering and decoding** in two separate steps.
- Reordering spaces are generated with a (syntactic) rule-based system, compactly encoded in a *lattice*.
- **"Best" reordering** may be derived by unfolding word alignments:
  → (oracle) *unfolded reordering*



From monotone to Oracle Reorderings

Ncode system and *oracle decoding* when:

- no reorderings allowed (monotone (m))
- using our lattice reordering space (l)
- using the unfolded reordering (u)

	dec.	en → fr	en → de
ncode	(m)	19.6± 0.1	12.7± 0.0
	(l)	22.8± 0.1	13.1± 0.1
	(u)	24.8± 0.1	15.8± 0.0
oracle	(m)	70.3	55.7
	(l)	84.3	64.6
	(u)	92.7	81.9

Alternative Tagsets

Reordering rules may use arbitrary categories to increase generalization.

	en → fr	de → en
Standart Part-of-Speech	22.8± 0.1	18.7± 0.1
12 simplified POS	22.7± 0.1	18.7± 0.1
Lexicalized POS for the 50 most frequent words	22.8± 0.1	18.3*± 0.1
50 Brown word classes (classes)	22.8± 0.1	18.2*± 0.2

Reordering Space Trade-off

- Reordering rules may be filtered with a conditional ratio threshold (maxcost), yielding to reordering spaces of increasing sizes.

	maxcost	BLEU ncode	BLEU oracle	#rules	size	coverage (%)
$fr \rightarrow en$	0	20.2*± 0.1	70.5	0	30 / 1	16
	2	21.9*± 0.0	77.2	19k	35 / 11	25
	4	22.5 ± 0.1	84.4	31k	69 / 10^7	40
	8	22.2*± 0.1	89.6	49k	397 / 10^{34}	60

Comparison with MJ-i

- Purely combinatorial MaxJump-i constraints are equivalent to consider all possible reordering rules up to size $i+1$ and yield to larger reordering spaces.

	$en \rightarrow fr$		$de \rightarrow en$	
	BLEU	size	BLEU	size
maxlen=2	22.4 ± 0.1	34 / 10^2	17.6 ± 0.1	34 / 30
MJ-1	22.1*± 0.1	74 / 10^{14}	17.8*± 0.2	19 / 10^{19}
maxlen=3	22.6 ± 0.0	40 / 10^3	18.0 ± 0.1	41 / 10^3
MJ-2	22.3*± 0.1	209 / 10^{23}	18.0 ± 0.1	223 / 10^{31}
maxlen=4	22.9 ± 0.2	43 / 10^4	18.3 ± 0.1	50 / 10^4
MJ-3	22.4*± 0.1	715 / 10^{30}	17.8*± 0.2	768 / 10^{40}

Conclusion

- **Gap to fill by improving the reordering space** at decoding, but maybe not the main possible improvement.
- Competitive results with coarse grained tagsets and automatic word classes in rewriting rules.
- **Larger spaces increase reordering coverage but not overall performance** because of decoding errors.
- The use of **linguistically motivated rules is better than allowing all local permutations**.