MaxForce: Max-Violation Perceptron and Forced Decoding for Scalable MT Training

First Successful Effort of Large-Scale Discriminative Training for MT

Heng Yu — Institute of Computing Tech., CAS
Liang Huang — City University of New York (CUNY)
Haitao Mi — T.J. Watson Research Center, IBM
Kai Zhao — City University of New York (CUNY)

Background

standard perceptron (Liang et al '06)
↓
our violation-fixing perceptron with truly sparse features

training set (>100k sentences)

MERT (2003)
MIRA (2007-12)
PRO (2011)
HOLS (2013)
...

dev set (~1k sents) | test set (~1k sents)

▷ **Why standard perceptron fails**

- b/c it assumes exact search
- but search errors abound in MT
- how to adapt perceptron to MT?

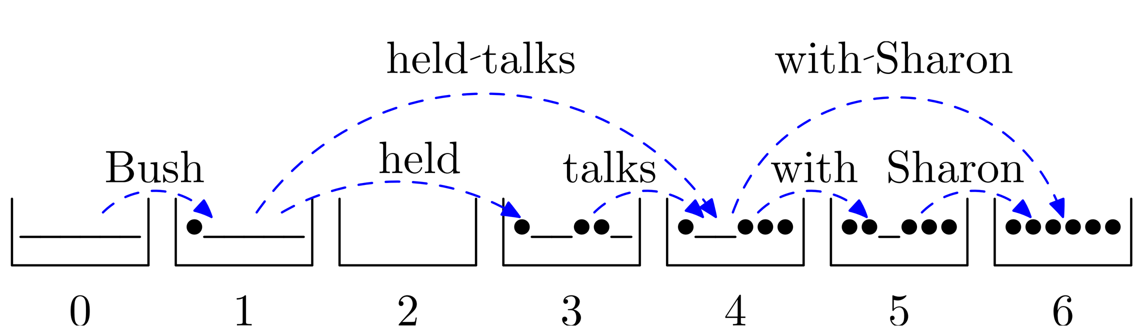
▷ **Small-scale tuning on dev is weak**

- Can't use truly sparse features: only ~10k for MIRA/PRO
- Tends to overfit on dev set

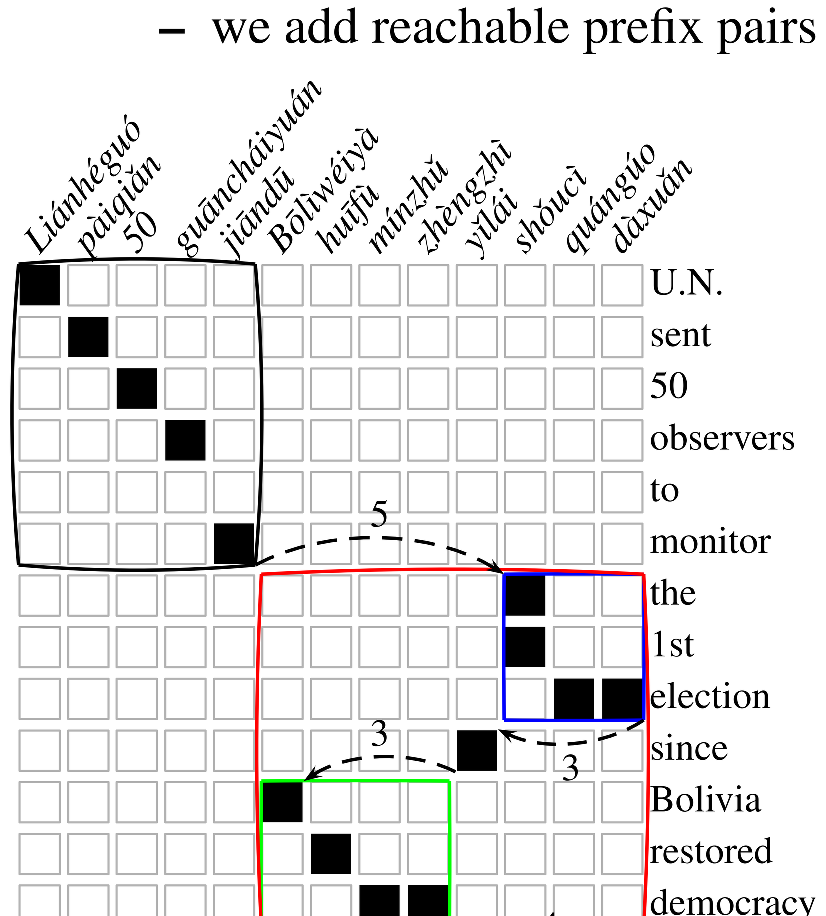
Our Method

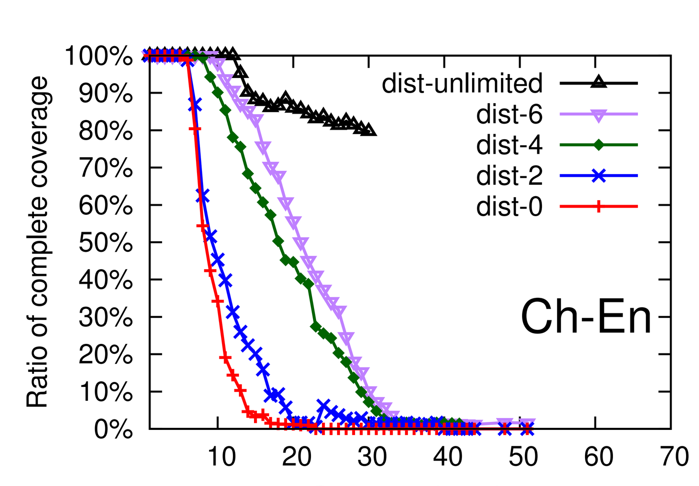
1. use "violation-fixing" perceptron (Huang+ '12) tailored for inexact search
   - fix search errors in the middle of the search
   - "partial updates" instead of "full updates"
2. use forced decoding lattice as the target to update to (latent variables)
3. use parallelized minibatch to speed up learning
4. result: scaled to a large portion of the training data for the first time
   - 20M+ sparse features => **+2.0 BLEU over MERT/PRO**

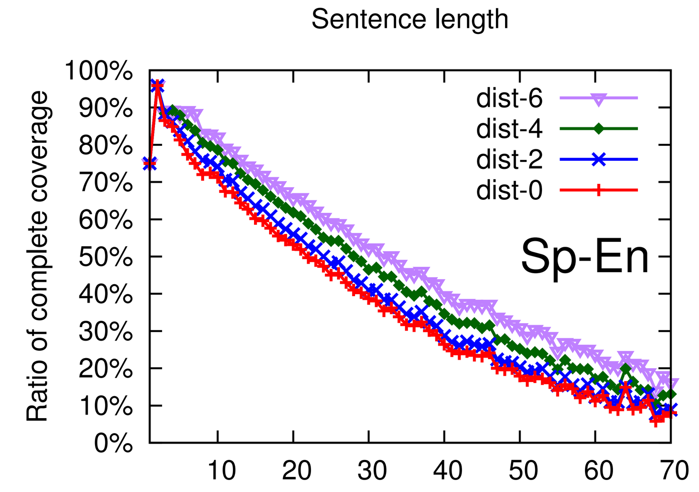
Force Decoding: compute gold-standard (reference-producing) derivations



- many unreachable sentence pairs due to distortion and phrase limits
  – we add reachable prefix pairs



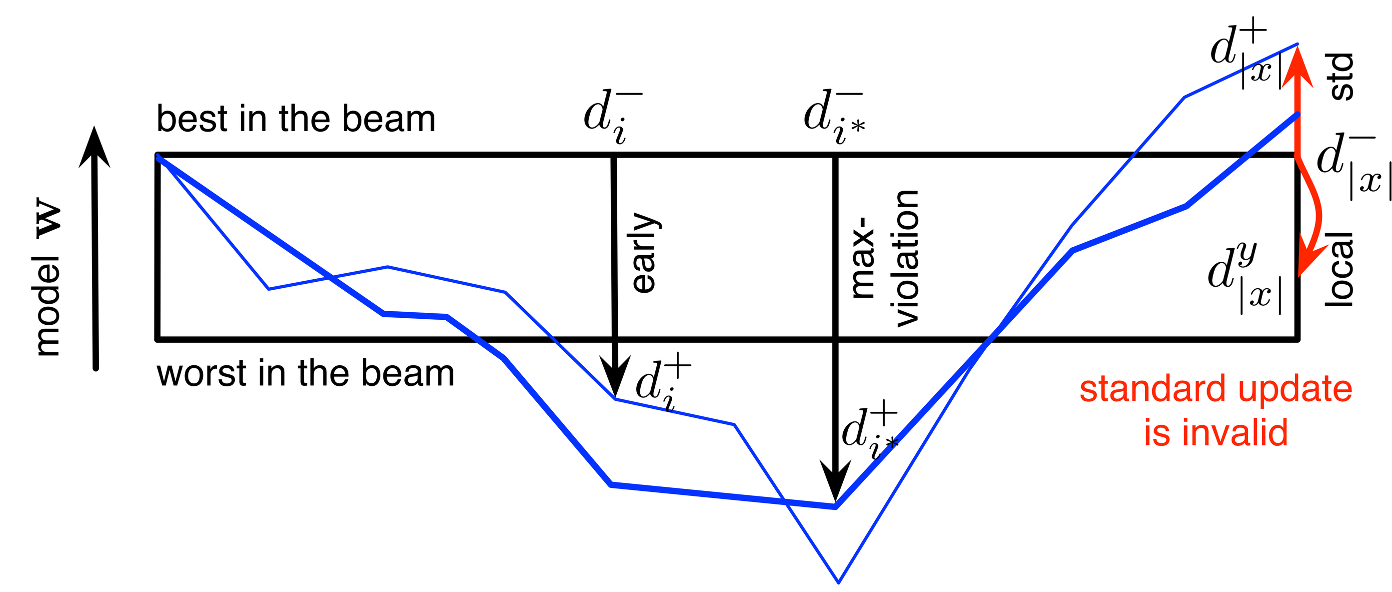




Violation Fixing Perceptron

▷ Violation-Fixing Perceptron (Huang+ 2012) is tailored for inexact search

1. Violation: incorrect prefix scores higher than gold-standard prefix
2. Guaranteed to converge if each update is valid (i.e., on a violation)
3. Examples: early update (Collins+Roark '04) and max-violation (Huang+ '12)
4. Standard update **does not converge** with many invalid updates



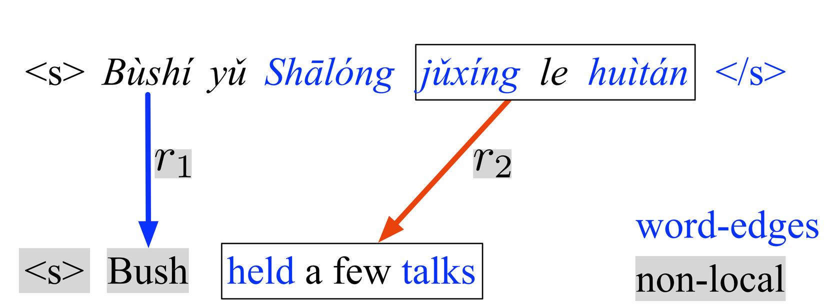
▷ Extend violation-fixing perceptron to handle latent variables (derivations):

1. Early update: update when no correct derivations survive
2. Max-violation: update at the bin where the violation is maximum
3. Standard update ("bold" update in Liang et al '06): **invalid update!**
4. Local update (also from Liang et al): update towards the derivation with highest sentence-level BLEU in the n-best list

Liang et al attribute their failure to gold derivations from "bad" rules. But we give a theoretically sound explanation: search errors cause invalid updates.

Feature Design

1. Dense features: 11 standard phrase-based features from Moses
2. Sparse Features
   - rule-identification features (unique id for each rule)
   - word-edges: lexicalized local translation context within a rule
   - non-local features: dependency between consecutive rules
3. Feature Backoff: Brown clusters, POS tags, Chinese chars/types, etc.



100010 = Shālóng | held
010001 = jǔxíng | talks
rule-bigram = $r_1 \mid r_2$
rule+$w_{-2}w_{-1}$ = r_2 | <s> | Bush

Key References

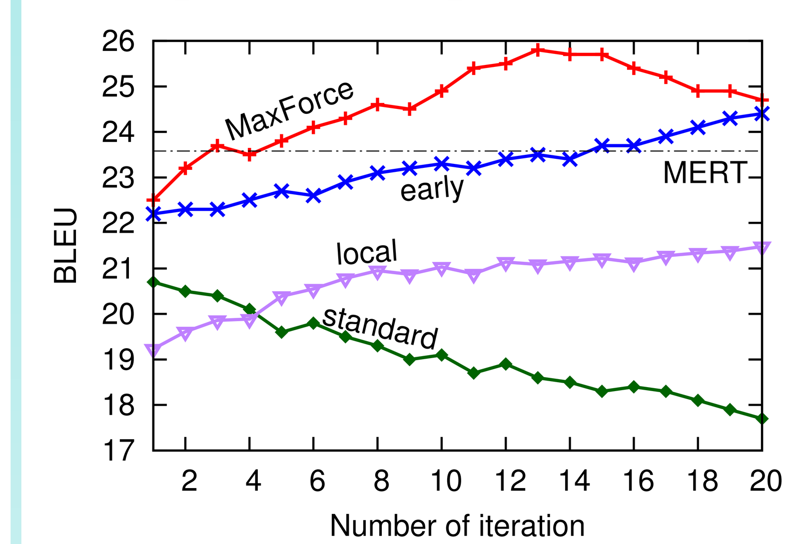
L. Huang, S. Fayong, and Y. Guo. Structured Perceptron with Inexact Search. In *NAACL 2012*.
K. Zhao and L. Huang. Minibatch and Parallelization for Online Large-Margin Structured Learning. In *NAACL 2013*.

Experimental Results

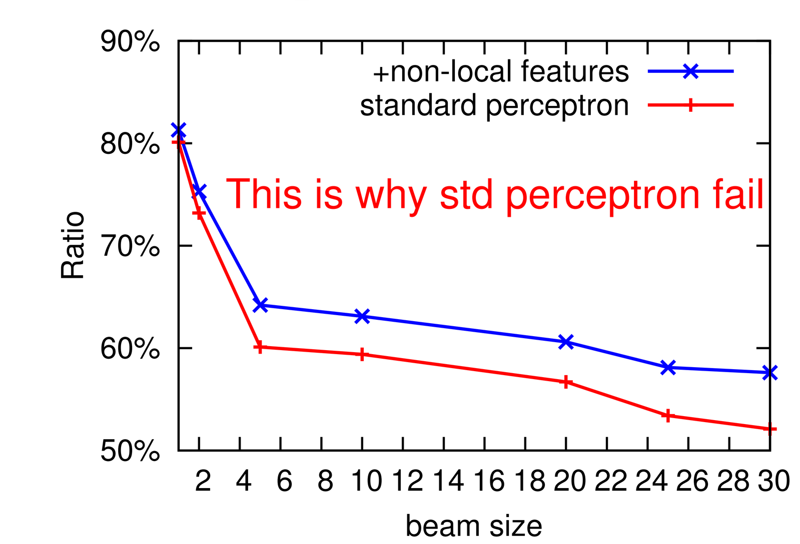
Scale	Lang. Pair	Training Data # sent.	# words	Reachability sent.	+prefix	words	+prefix	feats	ΔBLEU refs	dev/test
small	Ch-En	30K	0.8/1.0M	21.4%	61.3%	8.8%	24.6%	7M	4	+2.2/2.0
large	Ch-En	230K	6.9/8.9M	32.1%	67.3%	12.7%	32.8%	23M	4	+2.3/2.0
large	Sp-En	174K	4.9/4.3M	55.0%		43.9%		21M	1	+1.3/1.1

Overview of all experiments. The ΔBLEU column shows the absolute improvements of our method MaxForce on dev/test sets over MERT. The Chinese datasets also use prefix-pairs in training.

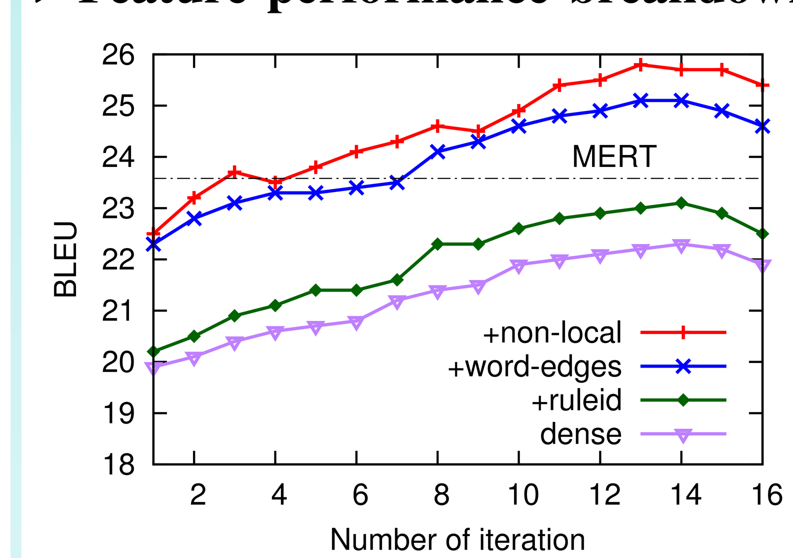
▷ **Comparison of Update Methods**



▷ **Invalid Update % in Standard**



▷ **Feature performance breakdown**

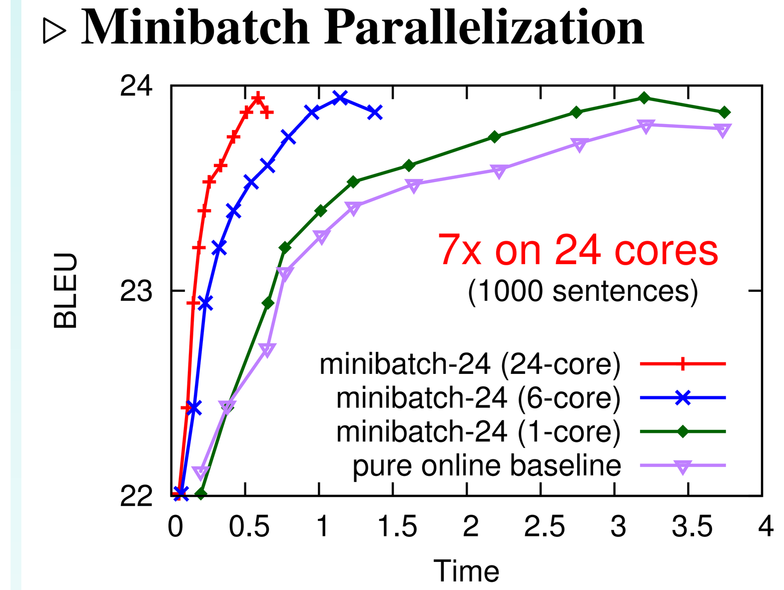


▷ **Feature Counts & Contributions**

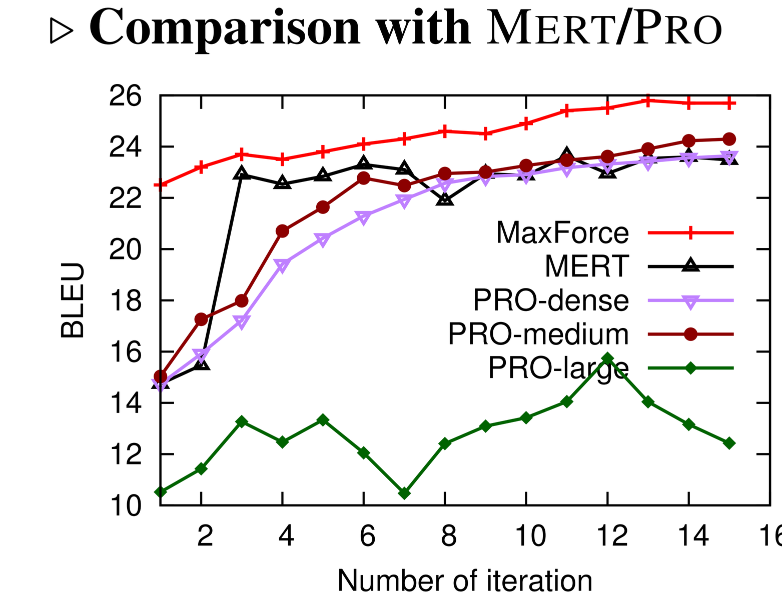
type	count	%	Bleu
dense	11	-	22.3
+ruleid	+9,264	+0.1%	+0.8
+WordEdges	+7,046,238	+99.5%	+2.0
+non-local	+22,536	+0.3%	+0.7
all	7,074,049	100%	25.8

Interestingly, the 0.3% non-local features contribute +0.7 BLEU.

▷ **Minibatch Parallelization**



▷ **Comparison with MERT/PRO**



▷ **Results on Large CH-EN (FBIS)**

system	algorithm	# feat.	dev	test
Moses	MERT	11	25.5	22.5
Cubit	MERT	11	25.4	22.5
Cubit	PRO	11	25.6	22.6
Cubit	PRO	3K	26.3	23.0
Cubit	PRO	36K	17.7	14.3
Cubit	MaxForce	23M	**27.8**	**24.5**

MaxForce is **2.3/2.0** over MERT; 35 hours on 24 cores. MERT: 1 hour.

▷ **Results on SP-EN (with 1-ref)**

system	algorithm	# feat.	dev	test
Moses	MERT	11	27.4	24.4
Cubit	MaxForce	21M	**28.7**	**25.5**

MaxForce is **1.3/1.1** over MERT with 1-ref (δ in 1-ref $\sim 2\delta$ in 4-ref).

Cubit 2.0 will be released at http://acl.cs.qc.edu/.