

SYNTAX FOR STATISTICAL MACHINE TRANSLATION

Franz Josef Och	Dan Gildea	Anoop Sarkar
Kenji Yamada	Sanjeev Khudanpur	Alex Fraser
Shankar Kumar	David Smith	Libin Shen
Viren Jain	Katherine Eng	Zhen Jin

+ Assoc.: Dragomir Radev

Status Report: August 6, 2003

Summary of Workshop Goals

- Improve Chinese-English Statistical Machine Translation
 - Feature functions: simple → complicated (string-based, part of speech based, chunk based, tree based, tree-tree alignment based)
 - Finding feature functions by error analysis and feature hunting based on contrastive errors in the n -best devset
 - Rescoring using Risk Minimization, SVM/perceptron with margin, Min Bayes Risk

Weeks 2 and 3

- Data Preprocessing all over again
- Contrastive error analysis: parse trees of oracle vs. produced (Anoop)
- New oracle based on BLEU rather than WER+PER: BLEU of 45% instead of 38
- Moved to larger dev set (100x100 → 993x1000 → 5765x1000)
- More Features

Experiments: 993 Chinese sents, 1000-best English

Blame	Experiment	Bleu Score		
		Dev (+/-1%)	Test	
Franz	baseline		30.05	
	oracle	45		
Franz	Model1+POS-LM+paren	32.52	32.89	(w/ postprocess)
Anoop	Grand Feature Combination	30.73	32.74	
Kenji	Model1	32.67	32.71	
Franz	missingWords	32.03	32.18	
Viren	8-gram-POS-LM	??	32.10	
Zhen	Parens+Quotes	32.28	32.08	
Zhen	head-word-verb	32.43	31.95	
Katherine	RefWordPenalty	32.29	31.19	
Franz	parsing-prob	31.84	30.98	

Grand Feature Combination

word-level	model1 wordpop-idf MPMQ rev-lm-1 rev-lm-2	IBM model1 prob (sum) prefer words that tend to appear in all 4 references matching parens, matching quotes reverse language model on training reverse language model on training and test
Part of speech	poslm POS-posterior	POS 8gram language model (Eng) prob returned by maxent POS tagger
Tree-based	MConn parserScoreDivLM rightBranch numPPs	balanced co-ordination in Eng tree parser prob / trigram LM prob prefers right branching tree num of prepositional phrases

Ongoing Work

- Lexical dependency features (Drago)
- Tree-to-String alignments (Kenji)
- Tree-to-Tree alignments (Dan)
(oracle expt: prefers oracle 60% of the time)
- Features based on alignment in N-best (Viren, Anoop, etc)
- Perceptron Training (Libin: stay tuned)
- Minimum Bayes Risk (Shankar: stay tuned)

Dependency features

- “small” dependency feature $d(w_1, w_2) = \log \frac{p(w_1 \rightarrow w_2)}{p(w_2 \rightarrow w_1)}$
- Example:
 - $f(\text{”mark”} \rightarrow \text{”achievement”}) = 8$
 - $f(\text{”achievement”} \rightarrow \text{”mark”}) = 4$
 - $d(\text{”mark”}, \text{”achievement”}) = \log(8/4) = \log 2$
- “big” dependency feature $D = f(d_i)$

Example

china 's 14 open border cities achievements remarkable
achievement->china
achievement->14
achievement->open
achievement->border
achievement->city
achievement->remarkable

china 's 14 open border cities building remarkable achievements
city->china
city->14
city->open
city->border
city->build
build->achievement
achievement->remarkable

china 's 14 open border cities , remarkable achievements
,->city
city->china
city->14
city->open
city->border
,->achievement
achievement->remarkable

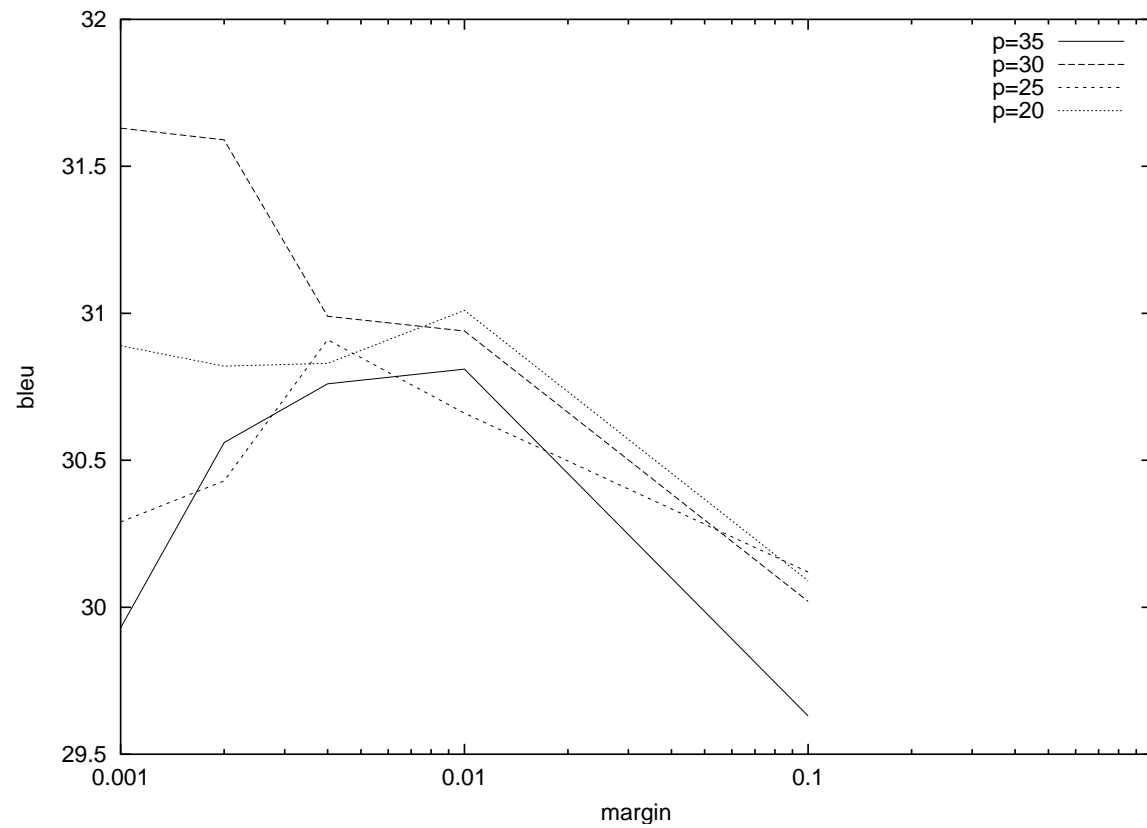
Reranking with Linear Classifiers

- Assumption: The *weight vector* of the separating hyperplane and the *score metrics* are in the same direction.
distance to the hyperplane = quality of a translation
- Large dimensional space. Allowing the use of various features.
- Inseparable in a space with 12 or 20 dimensions
 - Do not use all the translations in the training.
 - *λ -trick*: a new dimension for each sample (Herbrich 02)
 - It still doesn't work. Each sentence requires different *bias*.

Variants of Perceptrons for Reranking

- **Alg 1:** Pairwise translations as samples.
 - $class(T_{ij}, T_{ik}) \in \{-1, +1\}$, for two translations T_{ij} and T_{ik} of a sentence i .
 - T_{ij} and T_{ik} in top and bottom $p\%$.
 - Fast implementation.
- **Alg 2:** Multi-bias perceptron with margin
 - Input: translations $(\mathbf{x}_{ij}, y_{ij})_{i=1..s, j=1..n}$, where \mathbf{x}_{ij} in top or bottom $p\%$.
 - Goal: $\mathbf{w}, b_1, b_2, \dots, b_s$.
 - Update: If $y_{ij}(\mathbf{w}^t \mathbf{x}_{ij}) \leq \tau$, then $\mathbf{w}^{t+1} = \mathbf{w}^t + \eta y_{ij} \mathbf{x}_{ij}$, and $b_i^{t+1} = b_i^t + \eta y_{ij} R^2$.
 - Converge: $t \leq 2(s + 1)((\frac{R}{M})^2) + \frac{\tau}{\eta M^2}$, if the training data is separable with margin M .
 - \mathbf{w} is global for all sentences, providing the direction of the score metric for ranking.

Reranking results on 993 dataset (Alg 2 + λ -trick: 31.66; Baseline: 30.06)



Minimum Bayes-Risk (MBR) Decoders for SMT

- MBR decoders have been shown to optimize performance in speech and language processing tasks using application dependent loss functions
- Goal: Design MBR decoders to optimize translation quality under various loss functions that measure translation performance
 - We assume that we have available
 - * A Baseline Translation/Language Model to give the scores $P((E, A)|F)$
 - * A set \mathcal{E} of most likely translations of F
For each translation E' in this set, we have word-to-word alignments between E' and F and a parse-tree T' .
 - * A parse-tree T_F for the source sentence F
 - * A Loss function $L((E, A, T), (E', A', T'); T_F)$ that measures the quality of E' wrt E using information from word sequences (E, E') , alignments (A, A') and their parse-trees (T, T') and T_F .

– MBR Decoder

$$(\hat{E}, \hat{A}, \hat{T}) = \operatorname{argmin}_{\substack{\{E', A', T'\} \in \mathcal{E} \\ \{E, A\} \in \mathcal{E}}} \sum L((E, A, T), (E', A', T'); T_F) P((E, A)|F)$$

– \mathcal{E} is an N-best List in these experiments

Performance of the MBR Decoder

- Development Set: 993 sentences, 1000-best lists, Baseline from ISI
- Loss functions defined at the sentence level

	Performance Metrics				
Decoder	BLEU	mWER	mPER	Parse-Error	NIST
Baseline	30.73	67.31	43.22	92.8331	9.165
MBR Decoder					
BLEU	31.12	67.47	42.95	-	9.254
WER	30.93	66.68	42.69	-	9.214
PER	30.86	67.01	42.18	-	9.278
Parse-Loss	30.77	67.34	43.22	92.8328	9.176

- Test Set: 878 sentences, 1000-best lists - Best system using Minimum Error Training with all syntactic feature functions

	Performance Metrics		
Decoder	BLEU	mWER	mPER
Baseline	32.9	61.72	38.31
MBR-BLEU	33.2	61.7	38.29