
Statistical MT with Scarce Resources

Sonja Nießen and Hermann Ney

Computational Linguistics, June 2004

presented by Jeremy G. Kahn

SMT with scarce resources

- resources *always* scarce from some POV
- humans and knowledge-based MT use interrelated forms and syntactic transformations to do translation
- SMT doesn't do either, much
- trying here to incorporate some KB expertise to the (now standard) SMT systems

What's new in this paper?

(relative to standard SMT)

- Monolingual word sense disambiguation (WSD) before translation
- Hierarchical lexicon for MT
- harmonization/restructuring questions, verb prefixes
- improve knowledge sense-to-sense translation using other (monolingual) corpora

basically ways of bringing in KB hints to statistical system

Consider the MT pyramid

mapping up to something more like word senses, with little bits up
to syntax

(Monolingual) Word Sense Disambiguation

strategies for word sense disambiguation:

- case (orthographic) information in German
- preferences (indicative over subjunctive, imperative)
- ID related factors within words (in German)
 - base form
 - gender
 - case
 - number

goal: use local context to tease out factors that identify correlations among wordforms.

problem: Introduces sparsity: how to collapse factors to make lexical translation probabilities less sparse

WSD disambiguation issues

Problems in Word sense disambiguation:

- increases vocab size (50% increase in German).
- some lex categories can be collapsed after expansion

blau-adj-masc v. *blau-adj-fem*

([almost] always translates to "blue")

how are these dropped?

“Harmonization” of syntactic structures (s.3)

apparently brute-force

- realignment of questions
- English: removal of supporting ‘do’
- verb prefix reassignment

(details not really described in this paper)

Hierarchical lexicon for MT (s.4)

Instead of just considering the surface words e and f , we'll consider the words *and their readings* (which we will call T).

Given independence assumptions about the alignment,

$$\Pr(f_1^J | a_1^J, e_1^J) = \sum_{T_1^J} \prod_{j=1}^J p(f_j, T_j | e_{a_j}) \quad (1)$$
$$T_j \in \mathcal{T}(f_j)$$

Note most of the time $\mathcal{T}(f_j)$ will be 1.

so: what's $p(f, T|e)$? What does this mean for us?

Generalizing from the simple lexicon

blow out “reading” into a hierarchical lexicon:

$$p(f, T|e) = p_{\Lambda}(f, t_0^n|e) = \frac{\exp(\sum_m \lambda_m h_m(e, f, t_0^n))}{\sum_{\tilde{f}, \tilde{t}_0^n} \exp(\sum_m \lambda_m h_m(e, \tilde{f}, \tilde{t}_0^n))} \quad (2)$$

what is Λ ? weights for lots of different combinations of information over $h_m(e, f, t)$

Dictionary as parameterized space

A word in “traditional” SMT can be considered a value in the vocabulary space vector.

1. only one scalar in this vector can be on
2. each cell corresponds to exactly one surface form and vice versa.

Selecting a likely output word in a probabilistic bilingual lexicon is a function from V_f to a V_e vector.

We’re going to look at generalizing away from these two restrictions, but the V_e dimension will remain the same (for now).

Breaking the assumptions

- We want to allow some bits to represent word *families* (and to allow multiple input f words to set certain bits)
- allow more than one bit to be set

Learned value Λ distinguishes the useful binary features in the input from the non-useful.

What kinds of families are we interested in?

N & N use 3 kinds of parameters tuned by the Λ used here (s.4.2.1) (all are binary $\{0,1\}$ features).

Feature functions in N & N

A generalization of the lexicon: no longer an “impulse” vector!

learning various feature-detector functions h_m over $h_m(e, t_0^n, f)$

Discovering features:

for each e vocabulary item \tilde{e} , and f lemma L , set a bit $_m$ when:

- $m_1 = \{L, \tilde{e}\} : L = t_0$ and $e = \tilde{e}$
- $m_2 = \{T, L, \tilde{e}\} : \text{as above, and } T \text{ is a set of morpho-syntax tags and } T \subseteq t_0^n$
- $m_3 = \{F, T, L, \tilde{e}\} : \text{as above, and } F \text{ is the exact match surface lemma for } f$

Maximize the weights for all these (50K-60K features per class!) across alignments in training.

Bootstrapping from bilingual dictionaries (word pairs)

problem: context missing

- identify which word sense is being translated
- identify groups (“phrase translation”)

algorithm:

- get bilingual corpus.
- do coarse alignment on this corpus
- replace corpus entries with coarse tagging – dropping words!
- generate $p(t_f|t_e)$ probabilities without words

step seems to be characterizing how phrases in one language in general transform to the other (e.g. DNA vs DAN)

Bootstrapping from bilingual dictionaries (2)

- cluster together MWP (syntactic analyzer byproduct) – expand the dictionary pairs.
- examine all possible readings of the dictionary pairs
- use the $p(t_f|t_e)$ rankings to select which meaning you intended
- build new expanded dictionary

Results (1)

On artificially-limited training data:

#train sents		BLEU	m-WER	SSER	ISER
58,000	Baseline	53.7	34.1	30.2	14.1
	Restructuring	56.3	32.5	26.6	12.8
	+ disambig dict				
	+ hierarchical lexicon	57.1	31.8	26.3	11.8
5,000	Baseline	47.4	38.0	37.3	17.4
	Restructuring	52.1	34.7	33.6	15.2
	+ disambig dict				
	+ hierarchical lexicon	52.9	33.9	31.8	13.7

Improved on all measures overall.

Results (3)

#train sents		BLEU	m-WER	SSER	ISER
0	Baseline	23.3	53.6	60.4	29.8
	Restructuring	29.1	50.2	57.8	30.0
	+ disambig dict				
	+ hierarchical lexicon	32.6	48.0	52.8	24.1

Combined improves on all measures overall. Restructuring improves on almost every measure (not ISER 0 ?).

Results (2)

On Nespole! task (very scarce in-domain resources):

	BLEU	m-WER	SSER
Baseline	31.6	50.2	41.1
Restructuring	33.7	45.9	38.1
+ hierarchical lexicon	36.5	44.1	34.3

BLEU, m-WER and SSER all improved.

Discussion

Where could one extend this algorithm?

What other aspects are involved?

Those of us who know FLM: could this be adapted here?