Experiments: Paper Highlights

Departure from the “SMT as a black-box” paradigm:

- Direct SMT tuning for CLIR quality
- New decomposable proxy for retrieval quality to:
  - Explore full decoder search space instead of k-best lists
  - Train faster than k-best reranking frameworks

Structural SVM for SMT

Inject task-specific info via margin-rescaling:

1. Assume unit-decomposable penalty \( \Delta(q, q') \) for producing \( q \) instead of \( q' \):
   - \( \Delta(q, q') = 0 \), if \( q = q' \)
   - Increases as \( q \) gets further away from \( q' \)
2. Closest reachable substitute for reference \( r_f \): \( q_f' = \max_q (-\Delta(q, r_f)) \)
3. Unit-decomposability of \( \Delta \) is necessary for efficient max in the loss:
   \[ \mathcal{L} = \sum_f \max_q(\Delta(q, q_f')) + w \cdot h_q - w \cdot h_q' \]
4. Updates: \( w_{t+1} = w_t - \alpha \cdot \nabla_w \mathcal{L} \)

In CLIR single \( r_f \) does not exist
⇒ a decomposable proxy \( \Delta \) that reflects retrieval-quality is required

Recap: SMT (Baseline)

Translation \( q_f \) of foreign query \( f \):

- Construct \( q_f \) from bilingual translation units (phrases or grammar rules)
- Units carry numerical features \( h_{u,q,f} \)
- Decoding: \( q_f = \arg \max_q w \cdot h_{q,f} \)
- Features must be decomposable over units for efficient \( \arg \max \) \( h_{q,f} = \sum_q h_{u,q,f} \)
- \( w \) is learned to maximize BLEU on human reference translations \( r_f \)

Contribution: Tuning SMT for CLIR

New decomposable penalty \( \Delta \):

- Let \( C^+_f,k \) be docs on \( k \)-th relevance level for query \( f \)
- Relevance score of a translation \( q \) w.r.t. \( C^+_f \):
  \[ S(q, C^+_f) = \sum_{t \in q} \sum_{k \in C^+_f} \omega_k \sum_{d \in \mathbb{BM25}(t,d)} |C_d| \]
  - Relevance level weights \( \omega_k \) are found with grid search
  - BM25 decomposes over terms!
- Novel penalty for Structural SVM
  \[ \Delta(q, C^+_f) = \max_q \left( S_{rel}(q, C^+_f) - S_{rel}(q, C^+_f) \right) \]
  - Define hope, fear & oracle [McAllester and Keshet, 2011]:
    \[ q^{\text{oracle}} = \arg \max_q (\Delta(q, C^+_f)) \]
    \[ q^{\text{hope}} = \arg \max_{q \in E_f} (w \cdot h_q + \Delta(q, C^+_f)) \]
    \[ q^{\text{fear}} = \arg \max_{q \in E_f} (w \cdot h_q - \Delta(q, C^+_f)) \]

Losses to optimize:

- SVM loss:
  \[ \mathcal{L}_{\text{svm}} = \sum_f (w \cdot h_q^{\text{fear}} + \Delta(q^{\text{fear}}, C^+_f)) - w \cdot h_q^{\text{oracle}} \]
- Ramp loss:
  \[ \mathcal{L}_{\text{ramp}} = \sum_f (w \cdot h_q^{\text{fear}} + \Delta(q^{\text{fear}}, C^+_f)) - (w \cdot h_q^{\text{hope}} - \Delta(q^{\text{hope}}, C^+_f)) \]

Experiments: Patent Prior Art Search

Two baseline SMT systems:
- Moses (lattices) & cdec (hypergraphs)
- Train/dev: 1.8M/2k sentences from NTCIR
- Standard dense features and lexical sparse word-to-word mappings
- MIRA weight optimization [Chiang et al., 2008]

CLIR dataset:
- Sampled from the Boost CLIR dataset of JP/EN patents
- Train: 1k queries (5k sentences)
- Dev/test: 400 queries (2k sentences)

Meta-parameters (tuned on dev):
- Rescoring/redecoding: inference with new \( w \) on old/rebuilt MIRA lattices/hypergraphs
- Ramp or SVM losses (\( \mathcal{L}_{\text{ramp}}/\mathcal{L}_{\text{svm}} \))
- Freezing or learning dense features
- \# of iterations
References
