Wild Experimenting in MT

Ondřej Bojar, Aleš Tamchyna and Jan Berka

February 17, 2012

Tamil

Overview:

- A Dravidian language with more than 60 million native speakers.
- Official language in India, Sri Lanka, Singapore.
- Long history and tradition – a classical language.

MT-related properties:

- Uses its own script.
- Written left-to-right.
- Agglutinating language.
- SOV word order.

Components of Phrase-based MT (1/2)

- Word alignment
  - Learned from sentence-aligned parallel data.
  - Example query:
    What is the probability of '*car*' given German '*Auto*'?
  - Implemented in GIZA++.
- Translation model = *phrase table*
  - Trained heuristically based on the word alignment.
  - Example query:
    What is the probability of '*a fast car*' given '*ein schnelles Auto*'?
  - Implementation included in Moses toolkit.
- Language model
  - Trained from target-side monolingual data.
  - How probable are the words '*a fast car*' in an English sentence?
  - Various toolkits exits: SRILM, IRSTLM,...

Components of Phrase-based MT (2/2)

- Feature weights
  - Result of optimization towards a metric of translation quality.
  - Should the decoder trust language model score? How badly should the decoder penalize changes in word order?
  - Optimization algorithms/metrics are an active area of research.
  - Most commonly used is Minimum Error Rate Training (MERT), optimizing for BLEU.
- Decoder
  - Combines all previous steps in a model that generates translations based on input sentences.
  - Searches the hypothesis space for the most adequate translation.
  - Many decoders exist, we will be using Moses.

eman – Experiment MANager

In our SMT playground for `eman`, the components correspond to seeds:

Pipeline Step	**eman Seed**	**Examples of Seed Arguments**
Word alignment	`align`	corpus, source/target language
Translation model	`tm`	`align` step
Language model	`lm`	`srilm` step, target language
Weights tuning	`mert`	`model` step, sentences for tuning
Translation	`translate`	`mert` step, input sentences
Evaluation	`eval(uator)`	`translate` step, MT metric

- User defines steps based on seeds.
- `eman`:
  - Executes the steps.
  - Handles step dependecies, status, cloning,...

Tutorial Outline

- Install `eman`, prepare the environment for experimenting.
- A quick introduction to using `eman`.
- Run a baseline experiment English→Tamil.
- Explore ways to improve the translation quality.

What's wrong with the baseline?

- Tamil is an agglutinating language.
  - One stem/lemma has many forms $\Rightarrow$ *data sparsity*.
  - Word affixes encode a lot of information.
  - This information is mostly represented by syntax in English.
- Tamil has a different word order.
  - English is an SVO language, Tamil is SOV.
  - English *pre*-positions are Tamil *post*-positions.
  - Overall, Tamil constituents tend to be head-final.
- We are using tiny data. Well, that's a technical constaint.

Solutions? (1/4)

We need better word alignment.

- Instead of form→form, let's try stem4→stem4.
- Stemming all words to 4 characters:
  - is crude, linguistically incorrect.
  - almost always improves BLEU score (unless data is really large).

⇒ Reduction of data sparsity outweighs stemming errors.

Solutions? (2/4)

Employ splitting of Tamil affixes.
Align true Tamil stems instead of the crude approximation.

- Will the BLEU score be higher than with stem4$\rightarrow$stem4?

Solutions? (3/4)

Translate into a different language `ta_split` with affixes split from stems.

- Bound morphemes become free.
  ⇒ Word:morpheme ratio more similar to English.
- Data are dramatically less sparse.
- Can we directly compare BLEU with translations into 'normal' Tamil?

Solutions? (4/4)

Treex to the rescue: change source-side word order.

- Less distortion (i.e. need to reorder words when translating).
- Run your Treex reordering scenario on the English data.
- Do a complete training/evaluation pipeline.