Word Confidence Estimation for Machine Translation

Ngoc Quang Luong, Laurent Besacier, Benjamin Lecouteux

Confidence Estimation for Machine Translation

Ngoc-Quang LUONG

Supervisor: Laurent BESACIER, Co-supervisor: Benjamin LECOUTEUX

GETALP Team, LIG, Grenoble, France

INTRODUCTION

Definition:

- Confidence Estimation (CE) is a task of judging automatically each part (e.g. word, segment, or the whole sentence) in the MT hypothesis as correct or incorrect.
- A classifier trained beforehand by a feature set calculates the confidence score for MT hypothesis, then compares it with a threshold. Those with scores exceeding this threshold are categorized in the ***Good*** label set; the rest belongs to the ***Bad*** label set.

Interesting uses of CE:

- Decide whether a given translation is good enough for publishing as is.
- Highlight words that need editing in post-editing tasks.
- Inform readers of portions of the sentence that are not reliable.
- Select the best segments among options from multiple translation systems for MT system combination.

FIRST CONTRIBUTION: WORD CONFIDENCE ESTIMATION (WCE)

1. Feature Extraction

In order to build the binary classifier, we extracted totally 25 features, including four categories:

•System-based (S),

•Lexical (L),

•Syntactic (T)

•And Semantic (M) features

(see Table 1)

ID	Feature name	ID	Feature name
1L	Source POS	14S	Left source context
2S	Source word	15M	Polysemy count
3S	Target Word	16S	Source Language Model
4S	Backoff Behavior	17S	Number of Occurrences
5S	WPP any	18L	Numeric
6L	Target POS	19L	Proper name
7T	Null Link	20S	Left target context
8L	Stop word	21S	Min
9S	Nodes	22S	Target Language Model
10T	Constituent Label	23S	Right source context
11S	Right target context	24T	Distance to Root
12S	Max	25S	WPP $exact$
13L	Punctuation		

Table 1: The various types of features used to train the classifier.

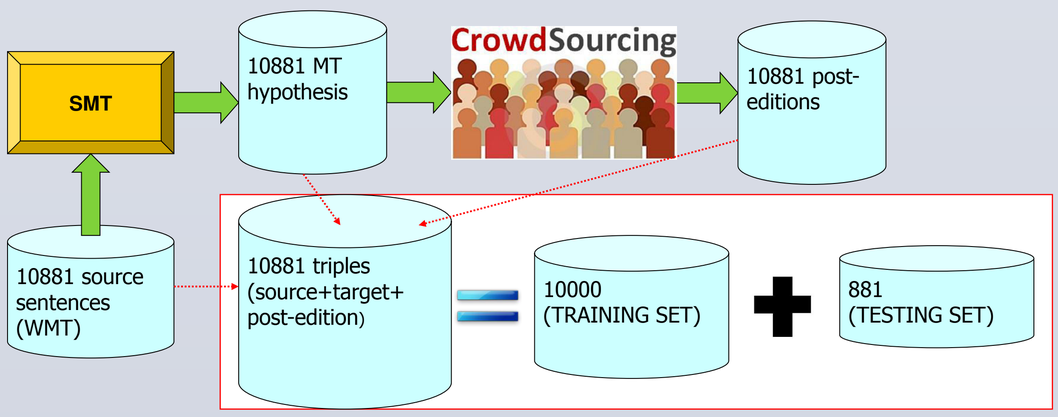
2. Model to train the classifier

- Conditional Random Fields (Lafferty et al., 2001).
- Training algorithm: block- wise coordinate descent (BCD) (Lavergne et al., 2010).

3. French - English SMT System Building

- Decoder: Moses (log-linear model with 14 weighted feature functions.)
- Translation model: Europarl and News parallel corpora (WMT 2010, with 1,638,440 sentences).
- Language model: SRILM, news monolingual corpus (48,653,884 sentences).

4. Corpus Preparation



5. Word Label Setting for Classifier:

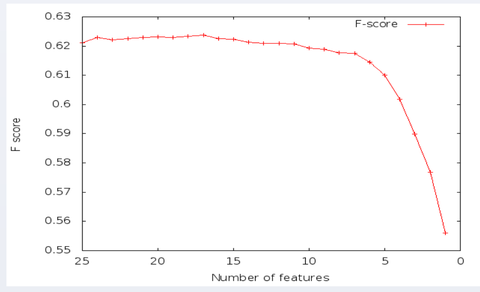
- We tag each word of the sentence in the training set a label by comparing this sentence to its reference. This label is then used to train the classifier.
- Tool used: TERp-A (is a version of TERp)
- Type of edit: I,S,T,Y,P,E (see an example of the setting below).
- Regroup into binary category: E, T and Y => G (85%) AND S, P and I => B (15%)

FIRST EXPERIMENTS AND RESULTS

1. Preliminary experiment with all features

- We track the Precision (Pr), Recall (Rc) and F-score (F) values for *G* and *B* label along threshold variation (from 0.3 to 1.0, step 0.025).
- Compare to 2 baselines: Baseline 1 (all words in each MT hypothesis are classified as good), and Baseline 2 (assigned randomly 85%G + 15%B) (Table 2, left, below).

System	Label	Pr(%)	Rc(%)	F(%)
All features	Good	86.02	88.07	87.04
	Bad	39.11	35.41	37.17
Baseline 1	Good	81.78	100.00	89.98
	Bad	-	0	-
Baseline 2	Good	81.77	85.20	83.45
	Bad	18.14	14.73	16.26



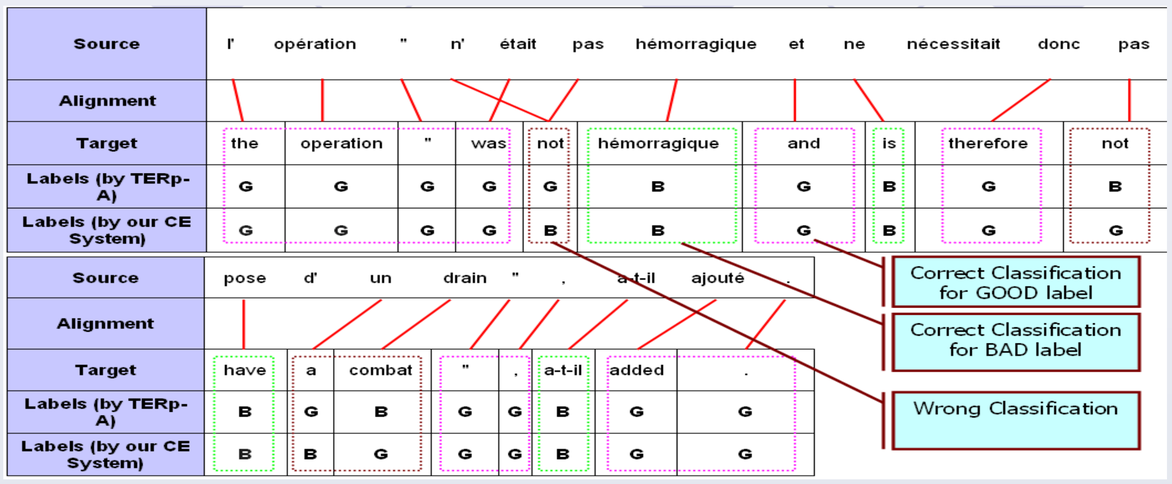


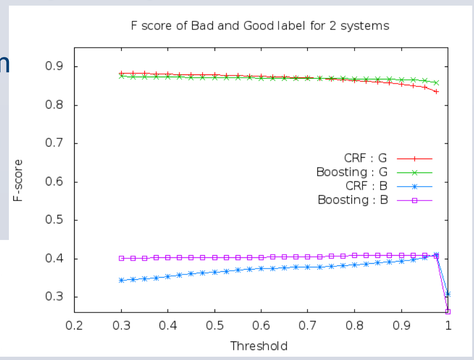
Table 3: Example of all-feature classifier's output

2. Feature Selection

- Objective: to rank our features from most to least important + to find the best performing combination.
- Strategy: "Sequential Backward Selection" algorithm. We start from the full set of N features, and in each step sequentially remove the most useless one.
- Output: The rank of each feature (also its ID in Table 1) + the system's evolution as more and more features are removed (Figure 1, above on the right).

3. Boosting technique to improve the classifier's performance:

- Objective: Take advantage of the sub-models' complementarily when combined.
- How to prepare the training set for Boosting system:
  - Step 1: Starting from 25 features, we build 23 subsets, in which 1 contains all features, 1 contains top 10 in Table 1, and 21 sets of 9 randomly extracted features for each.
  - Step 2: Divide our 10K training set into 10 equal subsets (S1, S2, ..., S10).
  - Step 3: For i:=1 to 10 do
    - Concatenate Sj (j=1..10, j≠i)
    - Train this set by 23 feature above sets (sequentially) => 23 different classifiers.
    - Apply each above classifier to test Si, log the "G" probability (23 in total) to form the training set Di.
  - Step 4: Concatenate Di (i=1..10) to obtain the Boosting training set.
- Testing : Build the test set for Boosting by logging 23 scores (like Step 3) for the usual test set, coming from 23 systems built on the usual training set.
- Comparison of the performance between 2 systems in terms of averaged scores (Table 3, below) or scores along to threshold variation (Figure 2, right).



System	Pr(G)	Rc(G)	F(G)	Pr(B)	Rc(B)	F(B)
Boosting	90.10	84.13	87.01	34.33	49.83	40.03
CRF	86.02	88.07	87.04	39.11	35.41	37.17

CONCLUSION AND ONGOING RESEARCH

- Experimental results show that precision and recall obtained in Good label are very promising, and Bad label reaches acceptable performance.
- A feature selection that we proposed helped to identify the most valuable features, as well as to find out the best performing subset among them.
- The protocol of applying Boosting method exploited effectively the good feature subsets for the system's performance improvement.
- Future work will take a deeper look into the linguistic features of word; experiment the CE at segment level; and find the methodology to conclude the sentence quality relied on the word's and segment's confidence score.