An Efficient Two-Pass Decoder for SMT Using Word Confidence Estimation

Ngoc-Quang Luong **Laurent Besacier** **Benjamin Lecouteux**
LIG, Campus de Grenoble
41, Rue des Mathématiques,
UJF - BP53, F-38041 Grenoble Cedex 9, France
{ngoc-quang.luong,laurent.besacier,benjamin.lecouteux}@imag.fr

Abstract

During decoding, the Statistical Machine Translation (SMT) decoder travels over all complete paths on the Search Graph (SG), seeks those with cheapest costs and backtracks to read off the best translations. Although these winners beat the rest in model scores, there is no certain guarantee that they have the highest quality with respect to the human references. This paper exploits Word Confidence Estimation (WCE) scores in the second pass of decoding to enhance the Machine Translation (MT) quality. By using the confidence score of each word in the *N*-best list to update the cost of SG hypotheses containing it, we hope to "reinforce" or "weaken" them relied on word quality. After the update, new best translations are re-determined using updated costs. In the experiments on our *real WCE scores* and *ideal (oracle) ones*, the latter significantly boosts one-pass decoder by 7.87 BLEU points, meanwhile the former yields an improvement of 1.49 points for the same metric.

1 Introduction

Beside plenty of commendable achievements, the conventional one-pass SMT decoders are still not sufficient yet in yielding human-acceptable translations (Zhang et al., 2006; Venugopal et al., 2007). Therefore, a number of methods to enhance them are proposed, such as: post-editing, re-ranking or re-decoding, etc. Post-editing (Parton et al., 2012) is in fact known to be a human-inspired task where the machine post edits translations in a second automatic pass. In re-ranking (Zhang et al., 2006; Duh and Kirchhoff, 2008; Bach et al., 2011), more features are integrated with the existing multiple model scores for re-selecting the best candidate among *N*-best list. Meanwhile, the re-decoding process intervenes directly into the decoder's search graph (SG), driving it to the optimal path (cheapest hypothesis).
The two-pass decoder has been built by several discrepant ways in the past. Kirchhoff and Yang (2005); Zhang et al. (2006) train additional Language Models (LM) and combine LM scores with existing model scores to re-rank the *N*-best list. Also focusing on the idea of re-ranking, yet Bach et al. (2011); Luong et al. (2014) employ sentence and word confidence scores in the second pass. Meanwhile, Venugopal et al. (2007) do a first pass translation without LM, but use it to score the pruned search hyper-graph in the second pass.
This work concentrates on a second automatic pass where the costs of all hypotheses in the decoder's SG containing words of the *N*-best list will be updated regarding the word quality predicted by Word Confidence Estimation (Ueffing and Ney, 2005) (WCE) system. In single-pass decoding, the decoder searches among complete paths (i.e. those cover all source words) for obtaining the optimal-cost ones. Essentially, the hypothesis cost is a composite score, synthesized from various SMT models (reordering, translation, LMs etc.). Although the *N*-bests beat other SG hypotheses in term of model scores, there is no certain clue that they will be the closest to the human references. As the reference closeness is the users' most pivotal concern on SMT decoder, this work establishes one second pass where model-independent

scores related to word confidence prediction are integrated into the first-pass SG to re-determine the best hypothesis. Inheriting the first pass's N-best list, the second one involves three additional steps:

- Firstly, apply a WCE classifier on the N-best list to assign the quality labels ("Good" or "Bad") along with the confidence probabilities for each word.
- Secondly, for each word in the N-best list, update the cost of all SG's hypotheses containing it by adding the update score (see Section 3.2 for detailed definitions).
- Thirdly, search again on the updated SG for the cheapest-cost hypothesis and trace backward to form the new best translation.

Basically, this initiative originates from an intuition that all parts of hypotheses corresponding to correct (predicted) words should be appreciated while those containing wrong ones must be weakened. The use of novel decoder-independent and objective features like WCE scores is expected to raise up the better candidate, rather than accepting the current sub-optimal one. The new decoder can therefore use both *real* and *oracle* word confidence estimates. In the next section, we introduce the SG's structure. Section 3 depicts our approach about using WCE scores to modify the first-step SG. The experimental settings and results, followed by in-depth analysis and comparison to other approaches are discussed in Section 4 and Section 5. The last section concludes the paper and opens some outlooks.

2 Search Graph Structure

The SMT decoder's Search Graph (SG) can be roughly considered as a "vast warehouse" storing all possible hypotheses generated by the SMT system during decoding for a given source sentence. In this large directed acyclic graph, each hypothesis is represented by a path, carrying all nodes between its begin and end ones, along with the edges connecting adjacent nodes. One hypothesis is called *complete* when all the source words are covered and *incomplete* otherwise. Starting from the empty initial node, the SG is gradually enlarged by expanding hypotheses during decoding. To avoid the explosion of search space, some weak hypotheses can be pruned or recombined. In order to facilitate the access and the cost calculation, each hypothesis **H** is further characterized by the following fields (we can access the value of the field f of hypothesis **H** by using the notion $f(H)$):

- **hyp**: hypothesis ID
- **stack**: the stack (ID) where the hypothesis is placed, also the number of foreign (source) words translated so far.
- **back**: the backpointer pointing to its previous cheapest path.
- **transition** : the cost to expand from the previous hypothesis (denoted by **pre(H)**) to this one.
- **score**: the cost of this hypothesis. Apparently, $score(H) = score(pre(H)) + transition.$
- **out**: the last output (target) phrase. It is worth to accentuate that **out** can contain multiple words.
- **covered**: the source coverage of **out**, represented by the start and the end position of the source words translated into **out**.
- **forward**: the forward pointer pointing to the cheapest outgoing path expanded from this one.
- **f-score**: estimated future cost from this partial hypothesis to the complete one (end of the SG).
- **recombined**: the pointer pointing to its recombined[1] hypothesis.

Figure 1 illustrates a simple SG generated for the source sentence: ***"identifier et mesurer les facteurs de mobilization"***. The attributes **"t"** and **"c"** refer to the transition cost and the source coverage, respectively. Hypothesis **175541** is extended from **57552**, when the three words from 3rd to 5th of the source sentence (*"les facteurs de"*) are translated into *"the factors of"* with the transition cost of -8.5746. Hence, its cost is: $score(175541) = score(57552) + transition(175541) = -16.1014 + (-8.5746) = -24.6760$. Three rightmost hypotheses: **204119**, **204109** and **198721** are complete since they cover all source words. Among them, the cheapest-cost

[1]In the SG, sometimes we recombine hypotheses to reduce the search space in a risk-free way. Two hypotheses can be recombined if they agree in (1) the source word covered so far (2) the last two target words generated, and (3) the end of the last source phrase covered.

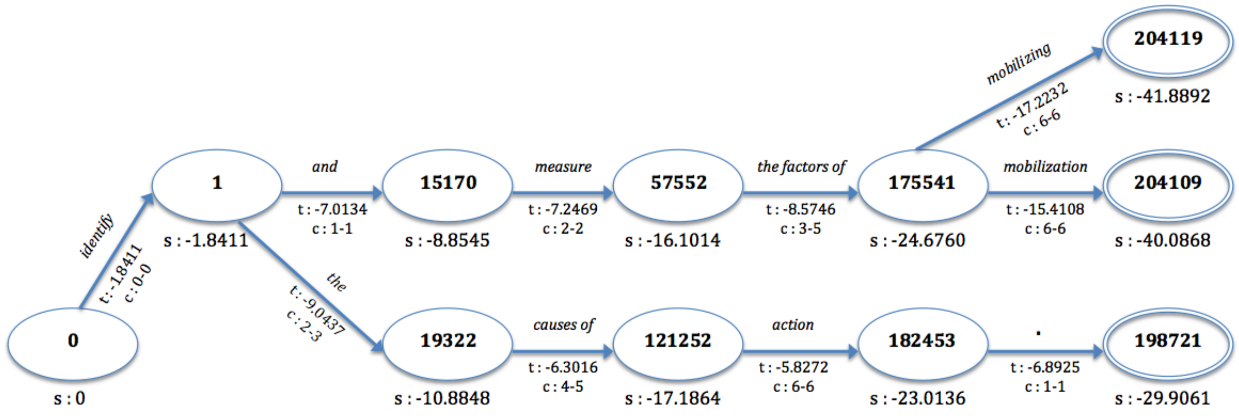

Figure 1: An example of search graph representation

one[2] is **198721**, from which the model-best translation is read off by following the track back to the initial node **0**: ***"identify the causes of action ."***.

3 Our Approach: Integrating WCE Scores into SG

In this section, we present the idea of using additional scores computed from WCE output (labels and confidence probabilities) to update the SG. We also depict the way that update scores are defined. Finally, the detailed algorithm followed by an example illustrates the approach.

3.1 Principal Idea

We assume that the decoder generates N best hypotheses $T = \{T_1, T_2, ..., T_N\}$ at the end of the first pass. Using the WCE system (which can only be applied to sequences of words - and not directly to the search graph - that is why N best hypotheses are used), we are able to assign the *j-th* word in the hypothesis T_i, denoted by t_{ij}, with one appropriate quality label, c_{ij} (e.g. *"G"* (Good: no translation error), *"B"* (Bad: need to be edited)), followed by the confidence probabilities ($P_{ij}(G), P_{ij}(B)$ or $P(G), P(B)$ for short). Then, the second pass is carried out by considering every word t_{ij} and its labels and scores $c_{ij}, P(G), P(B)$. Our principal idea is that, if t_{ij} is a *positive* (good) translation, i.e. c_{ij} = *"G"* or $P(G) \approx 1$, all hypotheses $H_k \in SG$ containing it in the SG should be "rewarded" by reducing their cost. On the contrary, those containing *negative* (bad) translation will be "penalized". Let $reward(t_{ij})$ and $penalty(t_{ij})$ denote the reward or penalty score of t_{ij}. The new **transition** cost of H_k after being updated is formally defined by:

$$transition'(H_k) = transition(H_k) + \begin{cases} reward(t_{ij}) & \text{if } t_{ij} = good\ translation \\ penalty(t_{ij}) & \text{if } otherwise \end{cases} \quad (1)$$

The update finishes when all words in the *N*-best list have been considered. We then re-compute the new score of complete hypotheses by tracing backward via back-pointers and aggregating the **transition cost** of all their edges. Essentially, the re-decoding pass reorders SG hypotheses in term of the more *"G"* words (predicted by WCE system) they contain, the more cost reduction will be made and consequentially, the more opportunity they get to be admitted in the *N*-best list. The re-decoding performance depends largely on the accurateness of confidence scores, or in other words, the WCE quality.

It is vital to note that, during the update process, we might face a phenomena that the word t_{ij} (corresponds to the same source words) occurs in different sentences of the *N*-best list. In this case, for the sake of simplicity, we process it only at its first occurrence (in the highest rank sentence) instead of updating the hypotheses containing it multiple times. In other words, if we meet the exact t_{ij} once again in the next N-best sentence(s), no further score update will be done in the SG.

3.2 Update Score Definitions

Defining the update scores is obviously a nontrivial task as there is no correlation between WCE labels and the SG costs. Furthermore, we have no clue about how proportional the SMT model and

[2]It is important to note that the concept **cheapest cost hypothesis** means that it has the highest model's score value. In other words, the higher the model score value, the "cheaper" the hypothesis is.

WCE (penalty or reward) scores should share in order to ensure that both of them will be appreciated. In this article, we propose several types of update scores, deriving from the global or local cost.

3.2.1 Definition 1: Global Update Score

In this type, an unique score derived from the cost of the current best hypothesis H^* (by the first pass) is used for all updates. We propose to compute this score by two ways: (a) exploiting WCE labels $\{c_{ij}\}$; *or* (b) only WCE confidence probabilities $\{P(G), P(B)\}$ will matter, WCE labels are left aside.

Definition 1a:

$$penalty(t_{ij}) = -reward(t_{ij}) = \alpha * \frac{score(H^*)}{\#words(H^*)} \quad (2)$$

Where $\#words(H^*)$ is the number of target words in H^*, the positive coefficient α accounts for the impact level of this score on the hypothesis's final cost and can be optimized during experiments. Here, $penalty(t_{ij})$ gets negative sign (since $score(H^*) < 0$) and will be added to the transition cost of all hypotheses containing t_{ij} in case where this word is labelled as *"B"*; whereas $reward(t_{ij})$ (same value, opposite sign) is used in the other case.

Definition 1b:

$$\begin{aligned} update(t_{ij}) &= \alpha * P(B) * \frac{score(H^*)}{\#words(H^*)} \\ &\quad - \beta * P(G) * \frac{score(H^*)}{\#words(H^*)} \\ &= (\alpha * P(B) - \beta * P(G)) * \frac{score(H^*)}{\#words(H^*)} \end{aligned} \quad (3)$$

Where $P(G)$, $P(B)$ $(P(G) + P(B) = 1)$ are the probabilities of "Good" and "Bad" class of t_{ij}. The positive coefficient α and β can be tuned in the optimization phase. In this definition, the fact that $update(t_{ij})$ is **a reward** $(reward(t_{ij}))$ or **a penalty** $(penalty(t_{ij}))$ will depend on t_{ij}'s goodness. Indeed, we have: $update(t_{ij}) = reward(t_{ij})$ if $update(t_{ij}) > 0$, which means: $\alpha*[1-P(G)]-\beta*P(G) < 0$ (since $score(H^*) < 0$), therefore $P(G) > \frac{\alpha}{\alpha+\beta}$. On the contrary, if $P(G)$ is under this threshold, $update(t_{ij})$ takes a negative value and therefore becomes a penalty.

3.2.2 Definition 2: Local Update Score

The update score of each (local) hypothesis H_k depends on its current transition cost, even when they cover the same word t_{ij}. Similarly to **Definition 1**, two sub-types are defined as follows:

Definition 2a:

$$penalty(t_{ij}) = -reward(t_{ij}) = \alpha * transition(H_k) \quad (4)$$

Definition 2b:

$$\begin{aligned} update(t_{ij}) &= \alpha * P(B) * transition(H_k) \\ &\quad - \beta * P(G) * transition(H_k) \\ &= (\alpha * P(B) - \beta * P(G)) * transition(H_k) \end{aligned} \quad (5)$$

Where $transition(H_k)$ denotes the current transition cost of hypothesis H_k, and the meanings of coefficient α (**Definition 2a**) or α, β (**Definition 2b**) are analogous to those of **Definition 1a** (**Definition 1b**), respectively.

3.3 Re-decoding Algorithm

The below pseudo-code depicts our re-decoding algorithm using WCE labels (**Definition 1a** and **Definition 2a**).

Algorithm 1 Using WCE labels in **SG** decoding

```
Input: $SG = \{H_k\}, T = \{T_1, T_2, ..., T_N\}, C = \{c_{ij}\}$
Output: $T' = \{T'_1, T'_2, ..., T'_N\}$
1: {Step 1: Update the Search Graph}
2: $Processed \leftarrow \emptyset$
3: for $T_i$ in $T$ do
4:     for $t_{ij}$ in $T_i$ do
5:         $p_{ij} \leftarrow$ position of the source words aligned to $t_{ij}$
6:         if $(t_{ij}, p_{ij}) \in Processed$ then
7:             continue; {ignore if $t_{ij}$ appeared in the previous sentences}
8:         end if
9:         $Hypos \leftarrow \{H_k \in SG | \ out(H_k) \ni t_{ij}\}$
10:        if $(c_{ij} = \text{"}Good\text{"})$ then
11:            for $H_k$ in $Hypos$ do
12:                $transition(H_k) \leftarrow transition(H_k) + reward(t_{ij})$ {reward hypothesis}
13:            end for
14:        else
15:            for $H_k$ in $Hypos$ do
16:                $transition(H_k) \leftarrow transition(H_k) + penalty(t_{ij})$ {penalize hypothesis}
17:            end for
18:        end if
19:        $Processed \leftarrow Processed \cup \{(t_{ij}, p_{ij})\}$
20:    end for
21: end for
22: {Step 2: Trace back to re-compute the score for all complete hypotheses}
23: for $H_k$ in $Final$ (Set of complete hypotheses) do
24:     $score(H_k) \leftarrow 0$
25:     while $H_k \neq$ initial hypothesis do
26:         $score(H_k) \leftarrow score(H_k) + transition(H_k)$
27:         $H_k \leftarrow pre(H_k)$
28:     end while
29: end for
30: {Step 3: Select N cheapest hypotheses and output the new list $T'$}
```

Rank	Cost	Hypotheses + WCE labels						
1	-29.9061	identify	the	cause	of	action	.	
		G	G	G	G	B	B	
2	-40.0868	identify	and	measure	the	factors	of	mobilization
		G	G	G	G	G	G	G

Table 1: The N-best (N=2) list generated by the SG in Figure 1 and WCE labels

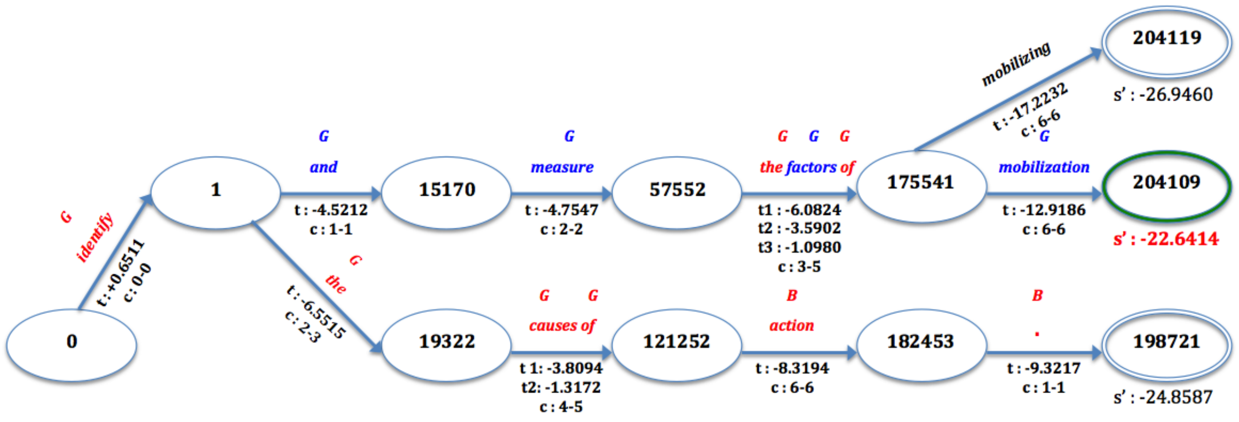

Figure 2: Details of update process for the SG in Figure 1. The first loop (when 1st rank hypothesis is used) is represented in red color, while the second one is in blue. For edges with multiple updates, all transition costs after each update are logged. The winning cost is also emphasized by red color.

The algorithm in case of using WCE confidence probabilities (**Definition 1b** and **Definition 2b**) is essentially similar, except the update step (from line 10 to line 18) is replaced by the following part:

```
for $H_k$ in $Hypos$ do
    $transition(H_k) \leftarrow transition(H_k) + update(t_{ij})$
end for
```

During the update process, the pairs including the visited word t_{ij} and the position of its aligned source words p_{ij} is consequentially admitted to $Processed$, so that all the analogous pairs (t'_{ij}, p'_{ij}) occuring in the latter sentences can be discarded. For each t_{ij}, a list of hypotheses in the SG containing it, called $Hypo$, is formed, and its confidence score c_{ij} (or $P(G)$) determines whether all members H_k in $Hypo$ will be rewarded or penalized. Once having all words in the N-best list visited, we obtain a new SG with updated transition costs for all edges containing them. The last step is to travel over all complete hypotheses (stored in $Final$) to re-compute their scores and then backtrack the cheapest-cost hypothesis to output the new best translation.

These above depictions can be clarified by taking another look at the example in Figure 1: from this SG, the N-best list (for the sake of simplicity, we choose $N = 2$) is generated as the single-pass decoder's result. According to our approach, the second pass starts by tagging all words in the list with their confidence labels, as seen in Table 1. Then, the graph update process is performed for each word in the list, sentence by sentence, which details are tracked in Figure 2. In this example, we apply **Definition 1a** to calculate the reward or penalty score, with $\alpha = \frac{1}{2}$, resulting in: $penalty(t_{ij}) = -reward(t_{ij}) = \frac{1}{2} * \frac{-29.9061}{6} = -2.4922$. Firstly, all hypotheses containing words in the 1st ranked sentence are considered. Since the word *"identify"* is labeled as *"G"*, its corresponding edge (connecting two nodes **0** and **1**) is rewarded and updated with a new cost : $t_{new} = t_{old} + reward = -1.8411 + 2.4922 = +0.6511$. On the contrary, the edge between two nodes **121252** and **182453** is penalized and takes new cost: $t_{new} = t_{old} + penalty = -5.8272 + (-2.4922) = -8.3194$, due to the bad quality of the word *"action"*. Obviously, the edges having multiple considered words (e.g. the one between nodes **19322** and **121252**) will be updated multiple times, and the transition costs after each update can be also observed in Figure 2 (e.g. $t1$, $t2$, etc). Next, when the 2nd-best is taken into consideration, all repeated words (e.g. *"iden-*

tify", *"the"* and *"of"*) are waived since they have been visited in the first loop, whereas the remaining ones are identically processed. The only untouched edge in this SG corresponds to the word *"mobilizing"*, as this word does not belong to the list. Once having the update process finished, the remaining job is to recalculate the final cost for every complete path and returns the new best translation: ***"identify and measure the factors of mobilization"*** (new cost = -22.6414).

4 Experimental Setup

4.1 Datasets and SMT Resources

From a dataset of 10,881 French sentences, we applied a Moses-based SMT system to generate their English hypotheses. Next, human translators were invited to correct MT outputs, giving us the post editions. The set of triples (source, hypothesis, post edition) was then divided into the training set (10000 first triples) and test set (881 remaining ones). The WCE model was trained over all **1-best hypotheses** of the training set. More details on our WCE system can be found in next section.

The N-best list ($N = 1000$) with involved alignment information is also obtained on the test set (1000 * 881 = 881000 sentences) by using Moses (Koehn et al., 2007) options *"-n-best-list"* and *"-print-alignment-info-in-n-best"*. Besides, the SGs are extracted by some parameter settings: *"-output-search-graph"*, *"-search-algorithm 1"* (using cube pruning) and *"-cube-pruning-pop-limit 5000"* (adds 5000 hypotheses to each stack). They are compactly encoded under a plain formatted text file that is convenient to transform into user-defined structures for further processing. We then store the SG for each source sentence in a separated file, and the average size is 43.8 MB.

4.2 WCE scores and Oracle Labels

We employ the Conditional Random Fields (Lafferty et al., 2001) (CRFs) as our machine learning method, with WAPITI toolkit (Lavergne et al., 2010), to train the WCE model. A number of knowledge resources are employed for extracting the system-based, lexical, syntactic and semantic characteristics of word, resulting in the total of 25 major feature types as follows:

- Target Side: target word; bigram (trigram) backward sequences; number of occurrences
- Source Side: source word(s) aligned to the target word
- Alignment Context (Bach et al., 2011): the combinations of the target (source) word and all aligned source (target) words in the window ± 2
- Word posterior probability (Ueffing et al., 2003)
- Pseudo-reference (Google Translate): Does the word appear in the pseudo reference?
- Graph topology (Luong et al., 2013): number of alternative paths in the confusion set, maximum and minimum values of posterior probability distribution
- Language model (LM) based: length of the longest sequence of the current word and its previous ones in the target (resp. source) LM. For example, with the target word w_i: if the sequence $w_{i-2}w_{i-1}w_i$ appears in the target LM but the sequence $w_{i-3}w_{i-2}w_{i-1}w_i$ does not, the n-gram value for w_i will be 3.
- Lexical Features: word's Part-Of-Speech (POS); sequence of POS of all its aligned source words; POS bigram (trigram) backward sequences; punctuation; proper name; numerical
- Syntactic Features: null link (Xiong et al., 2010); constituent label; depth in the constituent tree
- Semantic Features: number of word senses in WordNet.

In the next step, the word's reference labels (or so-called **oracle labels**) are initially set by using TERp-A toolkit (Snover et al., 2008) in one of the following classes: "I' (insertions), "S" (substitutions), "T" (stem matches), "Y" (synonym matches), "P" (phrasal substitutions), "E" (exact matches) and are then regrouped into binary class: *"G"* (good word) or *"B"* (bad word). Once having the prediction model, we apply it on the test set (881 x 1000 best = 881000 sentences) and get needed WCE labels along with confidence probabilities. In term of F-score, our WCE system reaches very promising performance in predicting *"G"* label (**87.65%**), and acceptable for *"B"* label (**42.29%**). Both **WCE** and **oracle** labels will be used in experiments.

4.3 Experimental Decoders

We would like to investigate the WCE's contributions in two scenarios: real WCE and ideal WCE (where all predicted labels are totally identical to the oracle ones). Therefore, we experiment with the seven following decoders:

- **BL**: Baseline (1-pass decoder)
- **BL+WCE(1a, 1b, 2a, 2b)**: four 2-pass decoders, using our estimated WCE labels and confidence probabilities to update the SGs, and the update scores are calculated by **Definition (1a, 1b, 2a, 2b)**.
- **BL+OR(1a, 2a)**: two 2-pass decoders, computing the reward or penalty scores by **Definition (1a, 2a)** on the oracle labels

It is important to note that, when using oracle labels, **Definition 1b** becomes **Definition 1a** and **Definition 2b** becomes **Definition 2a**, since if a word t_{ij} is labelled as "G", then $P(G) = 1$ and $P(B) = 0$, and vice versa. In order to tune the coefficients α and β, we carry out a **2-fold cross validation** on the test set. First, the set is split into two equivalent parts: **S1** and **S2**. Playing the role of a development set, **S1** will train the parameter(s) which then be used to compute the update scores on **S2** re-decoding process, and vice versa. The optimization steps are handled by CONDOR toolkit (Berghen, 2004), in which we vary α and β within the interval $[0.00; 5.00]$ (starting point is 1.00), and the maximum number of iterations is fixed as 50. Test set is further divided to launch experiments in parallel on our cluster using an open-source batch scheduler: OAR (Nicolas and Joseph, 2013). This mitigates the overall processing times on such huge SGs. Finally, the re-decoding results for them are properly merged for evaluation.

5 Results

Table 2 shows the translation performances of all experimental decoders and their percentages of sentences which outperform, remain equivalent or degrade the baseline hypotheses (when match against the references, measured by TER). Results suggest that using **oracle labels** to re-direct the graph searching boosts dramatically the baseline quality. **BL+OR(1a)** augments 7.87 points in BLEU, and diminishes 0.0607 (0.0794) point in TER(TERp-A), compared to **BL**. Meanwhile, with **BL+OR(2a)**, these gains are 7.67, 0.0565 and 0.0514 (in that order). Besides, the contribution of our real WCE system scores seems less prominent, yet positive: the best performing **BL+WCE(1a)** increases 1.49 BLEU points of **BL** (0.0029 and 0.0136 gained for TER and TERp-A). More remarkable, tiny ***p*-values** (in the range $[0.00; 0.02]$, seen on Table 2) estimated between BLEU of each **BL+WCE** system and that of **BL** relying on Approximate Method (Clark et al., 2011) indicate that these performance improvements are significant. Results also reveal that the use of WCE labels are slightly more beneficial than that of confidence probabilities: **BL+WCE(1a)** and **BL+WCE(2a)** outperform **BL+WCE(1b)** and **BL+WCE(2b)**. In both scenarios, we observe that the global update score (**Definition 1**) performs more fruitfully compared to the local one (**Definition 2**).

For more insightful understanding about WCE scores' acuteness, we make a comparison with the best achievable hypotheses in the SG (oracles), based on the "LM Oracle" approximation approach presented in (Sokolov et al., 2012). This method allows to simplify the oracle decoding to the problem of searching for the cheapest path on a SG where all transition costs are replaced by the n-gram LM scores of the corresponding words. The LM is built for each source sentence using uniquely its target post-edition. We update the SG by assigning all edges with the LM back-off score of the word it contains (instead of using the current transition cost). Finally, we combine the oracles of all sentences yielding BLEU oracle of **66.48**.

To better understand the benefit of SG re-decoding, we compare the obtained performances with those from our previous attempt in using WCE for *N*-best list re-ranking (green zone of Table 2). The idea is to build sentence-level features starting from WCE labels, then integrate them with existing SMT model scores to recalculate the objective function value, thus re-order the *N*-best list (Luong et al., 2014). Both approaches are implemented in analogous settings, e.g. identical SMT system, WCE system, and test set. Results suggest that the contribution of WCE in SG re-decoding outperforms that in N-best re-ranking in both "oracle" or real scenarios. **BL+OR(1a)** overpasses its corresponding oracle re-ranker **BL+OR(Nbest_RR)** in 2.08 points of BLEU, diminishes 0.0253 (0.0280) in TER(TERp-A). Meanwhile, **BL+WCE(1a)** wins real WCE re-ranker **BL+WCE(Nbest_RR)** in 1.03 (BLEU), 0.0015 (TER), 0.0103 (TERp-A). These achievements might originate from the following reasons: (1) In re-ranking, WCE scores are integrated at

Systems	Performance			Comparison to BL			p-value
	BLEU ↑	TER ↓	TERp-A ↓	Better (%)	Equivalent (%)	Worse (%)	
BL	52.31	0.2905	0.3058	-	-	-	-
BL+WCE(1a)	**53.80**	**0.2876**	**0.2922**	28.72	57.43	13.85	0.00
BL+WCE(1b)	53.24	0.2896	0.2995	26.45	59.26	14.29	0.00
BL+WCE(2a)	53.32	0.2893	0.3018	23.68	60.11	16.21	0.02
BL+WCE(2b)	53.07	0.2900	0.3006	22.27	55.17	22.56	0.01
BL+OR(1a)	**60.18**	**0.2298**	**0.2264**	62.52	24.36	13.12	-
BL+OR(2a)	59.98	0.2340	0.2355	60.18	28.82	11.00	-
BL+OR(Nbest_RR)	58.10	0.2551	0.2544	58.68	29.63	11.69	-
BL+WCE(Nbest_RR)	52.77	0.2891	0.3025	18.04	68.22	13.74	0.01
Oracle BLEU score	**BLEU = 66.48 (from SG)**						

Table 2: Translation quality of the conventional decoder and the 2-pass ones using scores from real or "oracle" WCE, followed by the percentage of better, equivalent or worse sentences compared to **BL**

sentence level, so word translation errors are not fully penalized; and (2) in re-ranking, best translation selection is limited to *N*-best list, whereas we afford the search over the entire updated SG (on which not only N-best list paths but also those contain at least one word in this list are altered) .

6 Conclusion and perspectives

We have presented a novel re-decoding approach for enhancing the SMT quality. Inherited the result from the first pass (*N*-best list), we predict words' labels and confidence probabilities, then employ them to seek a more valuable (cheaper) path over SGs throughout the re-decoding stage. While "oracle" WCE labels extraordinarily lifts the MT quality up (to reach the oracle score), real WCE achieves also the positive and promising gains. The method sharpens WCE increasing contributions in every aspect of SMT. As future work, we focus on estimating in more detail the word quality using MQM[3] metric as error typology, making WCE labels more impactful. Besides, the update scores used in this article would be further considered towards the consistency with SMT graph scores to obtain a better updated SG.

References

Nguyen Bach, Fei Huang, and Yaser Al-Onaizan. Goodness: A method for measuring machine translation confidence. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 211–219, Portland, Oregon, June 19-24 2011.

Frank Vanden Berghen. *CONDOR: a constrained, non-linear, derivative-free parallel optimizer for continuous, high computing load, noisy objective functions*. PhD thesis, University of Brussels (ULB - Université Libre de Bruxelles), Belgium, 2004.

Jonathan Clark, Chris Dyer, Alon Lavie, and Noah Smith. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *Proceedings of the Association for Computational Lingustics*, 2011.

Kevin Duh and Katrin Kirchhoff. Beyond log-linear models: Boosted minimum error rate training for n-best re-ranking. In *Proc. of ACL, Short Papers*, 2008.

Katrin Kirchhoff and Mei Yang. Improved language modeling for statistical machine translation. In *Proceedings of the ACL Workshop on Building and Using Parallel Texts*, pages 125–128, Ann Arbor, Michigan, June 2005.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 177–180, Prague, Czech Republic, June 2007.

John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting et labeling sequence data. In *Proceedings of ICML-01*, pages 282–289, 2001.

Thomas Lavergne, Olivier Cappé, and François Yvon. Practical very large scale crfs. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 504–513, 2010.

Ngoc Quang Luong, Laurent Besacier, and Benjamin Lecouteux. Word confidence estimation and its integration in sentence quality estimation for machine translation. In *Proceedings of The Fifth International Conference on Knowledge and Systems Engineering (KSE 2013)*, Hanoi, Vietnam, October 17-19 2013.

Ngoc Quang Luong, Laurent Besacier, and Benjamin Lecouteux. Word confidence estimation for smt n-best list re-ranking. In *Proceedings of the Workshop on Humans and Computer-assisted Translation (HaCaT)*, Gothenburg, Sweden, April 2014.

Capit Nicolas and Emeras Joseph. *OAR Documentation - User Guide*. LIG laboratory, Laboratoire d'Informatique de Grenoble Bat. ENSIMAG - antenne de Montbonnot ZIRST 51, avenue Jean Kuntzmann 38330 MONTBONNOT SAINT MARTIN, 2013.

Kristen Parton, Nizar Habash, Kathleen McKeown, Gonzalo Iglesias, and Adrià de Gispert. Can automatic post-editing make mt more meaningful? In *Proceedings of the 16th EAMT*, pages 111–118, Trento, Italy, 28-30 May 2012.

Matthew Snover, Nitin Madnani, Bonnie Dorr, and Richard Schwartz. Terp system description. In *MetricsMATR workshop at AMTA*, 2008.

Artem Sokolov, Guillaume Wisniewski, and Franc ois Yvon. Computing lattice bleu oracle scores for machine translation. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 120–129, Avignon, France, April 2012.

Nicola Ueffing and Hermann Ney. Word-level confidence estimation for machine translation using phrased-based translation models. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 763–770, Vancouver, 2005.

Nicola Ueffing, Klaus Macherey, and Hermann Ney. Confidence measures for statistical machine translation. In *Proceedings of the MT Summit IX*, pages 394–401, New Orleans, LA, September 2003.

Ashish Venugopal, Andreas Zollmann, and Stephan Vogel. An efficient two-pass approach to synchronous-cfg driven statistical mt. In *Proceedings of Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics*, April 2007.

Deyi Xiong, Min Zhang, and Haizhou Li. Error detection for statistical machine translation using linguistic features. In *Proceedings of the 48th Association for Computational Linguistics*, pages 604–611, Uppsala, Sweden, July 2010.

Ying Zhang, Almut Silja Hildebrand, and Stephan Vogel. Distributed language modeling for n-best list re-ranking. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP 2006)*, pages 216–223, Sydney, July 2006.

[3]http://www.qt21.eu/launchpad/content/training