Introduction of a new paraphrase generation tool based on Monte-Carlo sampling

Jonathan Chevelu[1,2] **Thomas Lavergne** **Yves Lepage**[1] **Thierry Moudenc**[2]
(1) GREYC, université de Caen Basse-Normandie
(2) Orange Labs; 2, avenue Pierre Marzin, 22307 Lannion
{jonathan.chevelu,thierry.moudenc}@orange-ftgroup.com,
thomas.lavergne@reveurs.org, yves.lepage@info.unicaen.fr

Abstract

We propose a new specifically designed method for paraphrase generation based on *Monte-Carlo* sampling and show how this algorithm is suitable for its task. Moreover, the basic algorithm presented here leaves a lot of opportunities for future improvement. In particular, our algorithm does not constraint the scoring function in opposite to *Viterbi* based decoders. It is now possible to use some global features in paraphrase scoring functions. This algorithm opens new outlooks for paraphrase generation and other natural language processing applications like statistical machine translation.

1 Introduction

A paraphrase generation system is a program which, given a source sentence, produces a different sentence with almost the same meaning.

Paraphrase generation is useful in applications to choose between different forms to keep the most appropriate one. For instance, automatic summary can be seen as a particular paraphrasing task (Barzilay and Lee, 2003) with the aim of selecting the shortest paraphrase.

Paraphrases can also be used to improve natural language processing (NLP) systems. (Callison-Burch et al., 2006) improved machine translations by augmenting the coverage of patterns that can be translated. Similarly, (Sekine, 2005) improved information retrieval based on pattern recognition by introducing paraphrase generation.

In order to produce paraphrases, a promising approach is to see the paraphrase generation problem as a translation problem, where the target language is the same as the source language (Quirk et al., 2004; Bannard and Callison-Burch, 2005).

A problem that has drawn less attention is the generation step which corresponds to the decoding step in SMT. Most paraphrase generation tools use some standard SMT decoding algorithms (Quirk et al., 2004) or some off-the-shelf decoding tools like MOSES (Koehn et al., 2007). The goal of a decoder is to find the best path in the lattice produced from a paraphrase table. This is basically achieved by using dynamic programming and especially the *Viterbi* algorithm associated with beam searching.

However decoding algorithms were designed for translation, not for paraphrase generation. Although left-to-right decoding is justified for translation, it may not be necessary for paraphrase generation. A paraphrase generation tool usually starts with a sentence which may be very similar to some potential solution. In other words, there is no need to "translate" all of the sentences. Moreover, decoding may not be suitable for non-contiguous transformation rules.

In addition, dynamic programming imposes an incremental scoring function to evaluate the quality of each hypothesis. For instance, it cannot capture some scattered syntactical dependencies. Improving on this major issue is a key point to improve paraphrase generation systems.

This paper first presents an alternative to decoding that is based on transformation rule application in section 2. In section 3 we propose a paraphrase generation method for this paradigm based on an algorithm used in two-player games. Section 4 briefly explain experimental context and its associated protocol for evaluation of the proposed system. We compare the proposed algorithm with a baseline system in section 5. Finally, in section 6, we point to future research tracks to improve paraphrase generation tools.

2 Statistical paraphrase generation using transformation rules

The paraphrase generation problem can be seen as an exploration problem. We seek the best paraphrase according to a scoring function in a space

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 249–252,
Suntec, Singapore, 4 August 2009.

to search by applying successive transformations. This space is composed of states connected by actions. An action is a transformation rule with a place where it applies in the sentence. States are a sentence with a set of possible actions. Applying an action in a given state consists in transforming the sentence of the state and removing all rules that are no more applicable. In our framework, each state, except the root, can be a final state. This is modelised by adding a stop rule as a particular action. We impose the constraint that any transformed part of the source sentence cannot be transformed anymore.

This paradigm is more approriate for paraphrase generation than the standard SMT approach in respect to several points: there is no need for left-to-right decoding because a transformation can be applied anywhere without order; there is no need to transform the whole of a sentence because each state is a final state; there is no need to keep the identity transformation for each phrase in the paraphrase table; the only domain knowledge needed is a generative model and a scoring function for final states; it is possible to mix different generative models because a statistical paraphrase table, an analogical solver and a paraphrase memory for instance; there is no constraint on the scoring function because it only scores final states.

Note that the branching factor with a paraphrase table can be around thousand actions per states which makes the generation problem a difficult computational problem. Hence we need an efficient generation algorithm.

3 Monte-Carlo based Paraphrase Generation

UCT (Kocsis and Szepesvári, 2006) (*Upper Confidence bound applied to Tree*) is a *Monte-Carlo* planning algorithm that have some interesting properties: it grows the search tree non-uniformly and favours the most promising sequences, without pruning branch; it can deal with high branching factor; it is an any-time algorithm and returns best solution found so far when interrupted; it does not require expert domain knowledge to evaluate states. These properties make it ideally suited for games with high branching factor and for which there is no strong evaluation function.

For the same reasons, this algorithm sounds interesting for paraphrase generation. In particular, it does not put constraint on the scoring function. We propose a variation of the UCT algorithm for paraphrase generation named MCPG for *Monte-Carlo based Paraphrase Generation*.

The main part of the algorithm is the sampling step. An episode of this step is a sequence of states and actions, $s_1, a_1, s_2, a_2, \ldots, s_T$, from the root state to a final state. During an episode construction, there are two ways to select the action a_i to perfom from a state s_i.

If the current state was already explored in a previous episode, the action is selected according to a compromise between exploration and exploitation. This compromise is computed using the UCB-Tunned formula (Auer et al., 2001) associated with the RAVE heuristic (Gelly and Silver, 2007). If the current state is explored for the first time, its score is estimated using *Monte-Carlo* sampling. In other word, to complete the episode, the actions $a_i, a_{i+1}, \ldots, a_{T-1}, a_T$ are selected randomly until a stop rule is drawn.

At the end of each episode, a reward is computed for the final state s_T using a scoring function and the value of each (state, action) pair of the episode is updated. Then, the algorithm computes an other episode with the new values.

Periodically, the sampling step is stopped and the best action at the root state is selected. This action is then definitely applied and a sampling is restarted from the new root state. The action sequence is built incrementally and selected after being enough sampled. For our experiments, we have chosen to stop sampling regularly after a fixed amount η of episodes.

Our main adaptation of the original algorithm is in the (state, action) value updating procedure. Since the goal of the algorithm is to maximise a scoring function, we use the maximum reachable score from a state as value instead of the score expectation. This algorithm suits the paradigm proposed for paraphrase generation.

4 Experimental context

This section describes the experimental context and the methodology followed to evaluate our statistical paraphrase generation tool.

4.1 Data

For the experiment reported in section 5, we use one of the largest, multi-lingual, freely available aligned corpus, Europarl (Koehn, 2005). It consists of European parliament debates. We choose

French as the language for paraphrases and English as the pivot language. For this pair of languages, the corpus consists of $1,487,459$ French sentences aligned with $1,461,429$ English sentences. Note that the sentences in this corpus are long, with an average length of 30 words per French sentence and 27.1 for English. We randomly extracted 100 French sentences as a test corpus.

4.2 Language model and paraphrase table

Paraphrase generation tools based on SMT methods need a language model and a paraphrase table. Both are computed on a training corpus.

The language models we use are n-gram language models with back-off. We use SRILM (Stolcke, 2002) with its default parameters for this purpose. The length of the n-grams is five.

To build a paraphrase table, we use the construction method via a pivot language proposed in (Bannard and Callison-Burch, 2005).

Three heuristics are used to prune the paraphrase table. The first heuristic prunes any entry in the paraphrase table composed of tokens with a probability lower than a threshold ϵ. The second, called *pruning pivot heuristic*, consists in deleting all pivot clusters larger than a threshold τ. The last heuristic keeps only the κ most probable paraphrases for each source phrase in the final paraphrase table. For this study, we empirically fix $\epsilon = 10^{-5}$, $\tau = 200$ and $\kappa = 10$.

4.3 Evaluation Protocol

We developed a dedicated website to allow the human judges with some flexibility in workplaces and evaluation periods. We retain the principle of the two-step evaluation, common in the machine translation domain and already used for paraphrase evaluation (Bannard and Callison-Burch, 2005).

The question asked to the human evaluator for the syntactic task is: *Is the following sentence in good French?* The question asked to the human evaluator for the semantic task is: *Do the following two sentences express the same thing?*

In our experiments, each paraphrase was evaluated by two native French evaluators.

5 Comparison with a SMT decoder

In order to validate our algorithm for paraphrase generation, we compare it with an off-the-shelf SMT decoder.

We use the MOSES decoder (Koehn et al., 2007) as a baseline. The MOSES scoring function is set by four weighting factors $\alpha_\Phi, \alpha_{LM}, \alpha_D, \alpha_W$. Conventionally, these four weights are adjusted during a tuning step on a training corpus. The tuning step is inappropriate for paraphrase because there is no such tuning corpus available. We empirically set $\alpha_\Phi = 1$, $\alpha_{LM} = 1$, $\alpha_D = 10$ and $\alpha_W = 0$. Hence, the scoring function (or reward function for MCPG) is equivalent to:

$$R(f'|f, I) = p(f') \times \Phi(f|f', I)$$

where f and f' are the source and target sentences, I a segmentation in phrases of f, $p(f')$ the language model score and $\Phi(f|f', I) = \prod_{i \in I} p(f^i|f'^i)$ the paraphrase table score.

The MCPG algorithm needs two parameters. One is the number of episodes η done before selecting the best action at root state. The other is k, an *equivalence parameter* which balances the exploration/exploitation compromise (Auer et al., 2001). We empirically set $\eta = 1,000,000$ and $k = 1,000$.

For our algorithm, note that identity paraphrase probabilities are biased: for each phrase it is equal to the probability of the most probable paraphrase. Moreover, as the source sentence is the best meaning preserved "paraphrase", a sentence cannot have a better score. Hence, we use a slightly different scoring function:

$$R(f'|f, I) = \min \left(\frac{p(f')}{p(f)} \prod_{\substack{i \in I \\ f^i \neq f'^i}} \frac{p(f^i|f'^i)}{p(f^i|f^i)}, 1 \right)$$

Note that for this model, there is no need to know the identity transformations probability for unchanged part of the sentence.

Results are presented in Table 1. The Kappa statistics associated with the results are 0.84, 0.64 and 0.59 which are usually considered as a "perfect", "substantial" and "moderate" agreement.

Results are close to evaluations from the baseline system. The main differences are from Kappa statistics which are lower for the MOSES system evaluation. Judges changed between the two experiments. We may wonder whether an evaluation with only two judges is reliable. This points to the ambiguity of any paraphrase definition.

System	MOSES	MCPG
Well formed (Kappa)	$64\%(0.57)$	$63\%(0.84)$
Meaning preserved (Kappa)	$58\%(0.48)$	$55\%(0.64)$
Well formed and meaning preserved (Kappa)	$50\%(0.54)$	$49\%(0.59)$

Table 1: Results of paraphrases evaluation for 100 sentences in French using English as the pivot language. Comparison between the baseline system MOSES and our algorithm MCPG.

By doing this experiment, we have shown that our algorithm with a biased paraphrase table is state-of-the-art to generate paraphrases.

6 Conclusions and further research

In this paper, we have proposed a different paradigm and a new algorithm in NLP field adapted for statistical paraphrases generation. This method, based on large graph exploration by *Monte-Carlo* sampling, produces results comparable with state-of-the-art paraphrase generation tools based on SMT decoders.

The algorithm structure is flexible and generic enough to easily work with discontinous patterns. It is also possible to mix various transformation methods to increase paraphrase variability.

The rate of ill-formed paraphrase is high at 37%. The result analysis suggests an involvement of the non-preservation of the original meaning when a paraphrase is evaluated ill-formed. Although the mesure is not statistically significant because the test corpus is too small, the same trend is also observed in other experiments. Improving on the language model issue is a key point to improve paraphrase generation systems. Our algorithm can work with unconstraint scoring functions, in particular, there is no need for the scoring function to be incremental as for *Viterbi* based decoders. We are working to add, in the scoring function, a linguistic knowledge based analyzer to solve this problem.

Because MCPG is based on a different paradigm, its output scores cannot be directly compared to MOSES scores. In order to prove the optimisation qualities of MCPG versus state-of-the-art decoders, we are transforming our paraphrase generation tool into a translation tool.

References

P. Auer, N. Cesa-Bianchi, and C. Gentile. 2001. Adaptive and self-confident on-line learning algorithms. *Machine Learning*.

Colin Bannard and Chris Callison-Burch. 2005. Paraphrasing with bilingual parallel corpora. In *Annual Meeting of ACL*, pages 597–604, Morristown, NJ, USA. Association for Computational Linguistics.

Regina Barzilay and Lillian Lee. 2003. Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. In *HLT-NAACL 2003: Main Proceedings*, pages 16–23.

Chris Callison-Burch, Philipp Koehn, and Miles Osborne. 2006. Improved statistical machine translation using paraphrases. In *HLT-NAACL 2006: Main Proceedings*, pages 17–24, Morristown, NJ, USA. Association for Computational Linguistics.

Sylvain Gelly and David Silver. 2007. Combining online and offline knowledge in UCT. In *24th International Conference on Machine Learning (*ICML*'07)*, pages 273–280, June.

Levente Kocsis and Csaba Szepesvári. 2006. Bandit based monte-carlo planning. In *17th European Conference on Machine Learning, (*ECML*'06)*, pages 282–293, September.

Philipp Koehn, Hieu Hoang, Alexandra Birch Mayne, Christopher Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Annual Meeting of ACL, Demonstration Session*, pages 177–180, June.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of MT Summit*.

Chris Quirk, Chris Brockett, and Bill Dolan. 2004. Monolingual machine translation for paraphrase generation. In Dekang Lin and Dekai Wu, editors, *the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 142–149., Barcelona, Spain, 25-26 July. Association for Computational Linguistics.

Satoshi Sekine. 2005. Automatic paraphrase discovery based on context and keywords between ne pairs. In *Proceedings of International Workshop on Paraphrase (IWP2005)*.

Andreas Stolcke. 2002. Srilm – an extensible language modeling toolkit. In *Proceedings of International Conference on Spoken Language Processing*.