Neural-Driven Search-Based Paraphrase Generation

Betty Fabre
Orange Labs, Lannion
Univ Rennes, IRISA
betty.fabre@orange.com

Jonathan Chevelu
Univ Rennes, IRISA Lannion
jonathan.chevelu@irisa.fr

Tanguy Urvoy
Orange Labs Lannion
tanguy.urvoy@orange.com

Damien Lolive
Univ Rennes, IRISA Lannion
damien.lolive@irisa.fr

Abstract

We study a search-based paraphrase generation scheme where candidate paraphrases are generated by iterated transformations from the original sentence and evaluated in terms of syntax quality, semantic distance, and lexical distance. The semantic distance is derived from BERT, and the lexical quality is based on GPT2 perplexity. To solve this multi-objective search problem, we propose two algorithms: *Monte-Carlo Tree Search For Paraphrase Generation* (MCPG) and *Pareto Tree Search* (PTS). We provide an extensive set of experiments on 5 datasets with a rigorous reproduction and validation for several state-of-the-art paraphrase generation algorithms. These experiments show that, although being non explicitly supervised, our algorithms perform well against these baselines.

1 Introduction

Paraphrase generation, *i.e.* the transformation of a sentence into a well-formed but lexically different one while preserving its original meaning, is a fundamental task of NLP. Its ability to provide diversity and coverage finds applications in several domains like question answering (McKeown, 1979; Harabagiu and Hickl, 2006), machine-translation (Callison-Burch et al., 2006), dialog systems (Yan et al., 2016), privacy (Gröndahl and Asokan, 2019a) or adversarial learning (Iyyer et al., 2018).

The formal definition of paraphrase may vary according to the targeted application and the tolerance we set along several axes, including the *semantic distance* from the source that we want to minimize, the *quality of the syntax*, and the *lexical distance* from the source that we want to maximize to ensure diversity.

The available aligned paraphrase corpora are often biased toward specific problems like *question answering* or *image captioning*. For instance, a transformer or a seq2seq model trained on a question answering corpus will typically turn any input sentence into a question. With the lack of generic aligned datasets, it remains challenging to train generic paraphrase models in a supervised manner.

On the other hand, with the availability of large-scale non-supervised language models like BERT and GPT2, the assessment of a given candidate paraphrase in terms of *semantic distance* from its source and *lexical quality* has become much more tractable.

Leveraging from these metrics, we propose to cast the paraphrase generation task as a multi-criteria search problem. We use PPDB 2.0 (Pavlick et al., 2015), a large-scale database of rewriting rules derived from bilingual corpora, to potentially generate billions of 'naive' candidate paraphrases by edition from the source sentence. To sort the good candidates efficiently from the others, we experiment with two search algorithms. The first one, called *Monte-Carlo Paraphrase Generation* (MCPG), is a variant of the *Monte-Carlo Tree Search* algorithm (MCTS) (Kocsis and Szepesvári, 2006; Gelly and Silver, 2007; Chevelu et al., 2009). The MCTS algorithm is famous for its successes on mastering the – highly combinatorial – game of Go (Gelly and Silver, 2007; Silver et al., 2016).

The second one is a novel search algorithm that we call *Pareto Tree Search* (PTS). In contrast to MCTS which is a single-criterion search algorithm, we designed PTS to retrieve an approximation of the whole Pareto optimal set. This allows for more flexibility on paraphrase generation where the balance between *semantic distance*, *syntax quality*, and *lexical distance* is hard to tune *a priori*. Another difference between MCPG and PTS is that PTS uses a randomized breadth-first exploration policy which proves to be more efficient on this problem.

The main contribution of this article is a study on search-based paraphrase generation through the

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2100–2111
April 19 - 23, 2021.

Source sentence :	he is speaking on june 14 .
PPDB rule	**Edited sentence**
is → is found	he is found speaking on june 14 .
is speaking → 's talking	he 's talking on june 14 .
speaking → speak now	he is speak now on june 14 .
14 → 14th	he is speaking on june 14th .

Table 1: PPDB rules applied to a source sentence

sieve of three criteria: semantic similarity, syntax quality, and lexical distance. We propose and evaluate two search algorithms: MCPG and PTS. We also provide an extensive set of experiments on English datasets with a rigorous reproduction and validation methodology for several state-of-the-art paraphrase generation algorithms.

This article is organized as follows: The candidate paraphrase generation scheme is presented in Section 2. In Section 3, we develop the different criteria we use to qualify correct paraphrases. The two search algorithms (MCPG and PTS) are described in Section 4. Section 5 gives a survey on other state-of-the-art paraphrase generation algorithms. The comparisons with non-supervised and supervised baselines are presented in Section 6 where the methodology is discussed in Section 6.3. We conclude by a data-augmentation experiment in Section 6.6.

2 Paraphrase generation scheme

We model paraphrase generation as a sequence of editions and transformations from a source sentence into its paraphrase. In this work, we only consider local transformations, *i.e.* replacement of certain words or group of words by others that have the same or similar meanings, but the method we propose should work with more sophisticated transformations as well.

The *Paraphrase Database* (PPDB) (Ganitkevitch et al., 2013; Pavlick et al., 2015) is a large collection of scored paraphrase rules that was automatically constructed from various bilingual corpora using a pivot alignment method (Callison-Burch, 2008). The database is divided into increasingly large and decreasingly accurate subsets[1]. We used the XL subset, and we removed the rules labeled as "`Independent`". This left us with a set of 5.5 million rewriting rules. We give some examples of these rules in Table 1.

[1] PPDB is available on `http://paraphrase.org`, and a python interface is available here `https://github.com/erickrf/ppdb`

By iteratively applying the rules from a source sentence like the one in Table 1, we obtain a vast lattice of candidate paraphrases. Some of these candidates like "*he's **talking** on june 14*" are well-formed, but many are syntactically broken, like "*he is **speak now** on june 14*".

The number of rules that apply depends on the source sentence's size and the words it contains. For instance, on the MSRPARAPHRASE dataset (see section 6.2), sentences are quite long and the median number of PPDB-XL rules that apply is around 450. After two rewriting steps, the median number of candidates is around 10^5, and by iterative rewriting, we quickly reach a number of paraphrase candidates that is greater than 10^8.

3 Paraphrase selection criteria

As it depends on the type of text we consider that may be spoken or written, casual or formal; it is not easy to define a *universal semantic distance* or a *universal scale of well formed syntax*. However, recent advances in NLP with neural networks like BERT and GPT2 trained on huge corpora have led to the development of metrics that can act as good proxies for these ideal notions.

For the semantic distance, a quick experiment confirms that the BERT score (Zhang et al., 2019a) performs well on difficult paraphrase identification tasks. The BERT score is an F1-measure over an alignment of the BERT contextual word embeddings of each of the sentences. To assess the sensitivity of this score, we computed the Area Under the ROC curve (AUC) on QQP and PAWS, two difficult paraphrase identification corpora (Kornél Csernai, 2017; Zhang et al., 2019b). On QQP, we obtained 75.2% and on PAWS, we obtained 67.0%. The PAWS corpus being designed to trick paraphrase identification classifiers, 67.0% is a reasonable performance. We hence opted for the BERT score between the source sentence and paraphrase candidate (denoted BERT_S) as our semantic score.

Regarding the syntax quality, the perplexity of GPT2 (Radford et al., 2019) is a good ranking criterion. Although, as illustrated in Table 2, in some cases, a rule-based spell-checker may detect errors that GPT2 would miss (but the reverse is also true). We hence opted for GPT2 as a primary criterion for syntax quality, combined with the LANGUAGETOOL spell-checker (Naber, 2003) that we only used on a second stage for performance reasons.

The lexical distance is important to ensure the

diversity of the produced paraphrases. It is however simple to handle. Some authors use the BLEU surface metric (Miao et al., 2018a), we opted here for the normalized character-level Levenshtein edition distance.

The balance between these criteria is difficult to obtain. Table 2 illustrates their impact on a sentence taken from the train set of MSRPARAPHRASE. The candidate examples in this table underline the tough dilemma between maximizing the semantic similarity (safe and conservative policy) and maximizing the lexical distance (risk-prone). The third and fourth examples underline the utility of the spell-checker: some low-perplexity examples are ill-formed. On the second part of the table, we printed the sentences chosen by our two models: MCPG and PTS.

4 Searching algorithms

Searching for good paraphrases in the large lattice of candidates generated by PPDB is a costly task. We propose two algorithms that share a similar structure: an outer loop explores the lattice at different depths, while an inner loop explores the candidates at each depth. Both algorithms are anytime: they return the best solutions found so far when the time or space budget is depleted.

4.1 MCPG: Monte-Carlo tree search for Paraphrase Generation

Following the idea of Chevelu et al. (2009), we used *Monte-Carlo Tree Search* (MCTS) to explore the PPDB lattice. The three key ingredients of MCTS are: a *bandit* policy at each node of a search-tree to select the most promising paths, randomized rollouts to estimate the quality of these paths, and back-propagation of rewards along the paths to update the bandit. We opted here for a randomized bandit policy called EXP3 (Auer et al., 2002). The MCTS algorithm being not designed for multi-objective problems, we needed to combine semantic similarity $\mathrm{BERT_S}$, syntax correctness GPT2 and surface diversity $\mathrm{Lev_S}$ into a single criterion. We opted for the following polynomial:

$$\alpha \cdot \mathrm{BERT_S} + \beta \cdot \mathrm{Lev_S} \cdot \mathrm{BERT_S} - \gamma \cdot \mathrm{GPT2} \quad (1)$$

where the product $\mathrm{Lev_S} \cdot \mathrm{BERT_S}$ is intended to avoid a trivial maximization of the score by applying a lot of editions to the source sentence. After a few experiments on train sets, we tuned empirically the weights to $\alpha = 3, \beta = 0.5$ and $\gamma = 0.025$ in order to obtain a balance as the one described in Table 2.

4.2 PTS: Pareto Tree Search in the paraphrase lattice

We observed two drawbacks for MCTS.

First, it was designed for combinatorial problems like Go where the evaluation is only possible on the leaves of the search tree. This is not the case for paraphrase generation where the neural models can evaluate any rewriting step and where rewriting from good candidates is more likely to provide good paraphrases than rewriting from bad ones. Secondly, it has been designed for single criterion search which requires fixing the balance between criteria definitively before any paraphrase search begins. This is not very flexible, and it becomes painful when we want to generate sets of candidates.

By plotting the distributions of the scores like on Figure 1, we noticed that most of the candidates were dominated in the Pareto sense: it was possible to eliminate most of the candidates without any hyper-parameter tuning. Hence, we adapted MCPG to explore the paraphrase lattice and recover an approximation of the Pareto front, postponing the balance between criteria as a quick post-optimization stage. This led us to the PTS algorithm described as pseudo-code in Table 3.

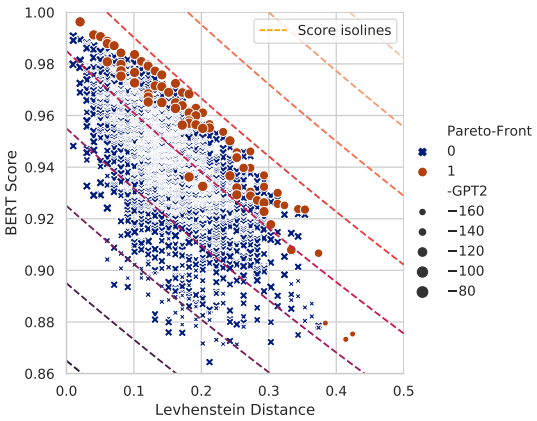

Figure 1: The cloud of candidates generated from the sample of Table 2. The optima of any positive combination of BERT score, normalized Levenshtein distance, and GPT2 perplexity belong to the Pareto front (orange dots). We plotted the projections of MCPG combined score (1) with dashed isolines. The BERT score and Levenshtein distance being clearly anti-correlated, the balance between these two criteria is difficult to tune.

Source	the agency could not say when the tape was made , though the voice says he is speaking on june 14 .				
Policy	Candidate Sentence	BERT_S	GPT2	Lev_S	Errs
high BERT conservative	the agency could not say when the tape was made , though the voice says he is talking on june 14 .	**0.99**	4.23	0.04	0
high Lev risk prone	the organizations and agencies could just 'm saying whenever the tape-based was provided presentation there are , however the express opinions informed that he was found talking pertaining end-june fourteen .	0.60	7.83	**1.0**	0
high GPT2 bad syntax	the agencies was not able say when the tape been given currently undertaking though the voice indicated he talking on june 14 .	0.82	**7.10**	0.55	2
spell & grammar errors	the organisation are incapable of 're saying when the tape is set out , despite the fact that the voice just said he have a conversation on june 14 .	0.73	5.49	0.79	**2**
balanced	the organization could not say when the tape was made , although the voice indicates that he is talking on june 14 .	0.95	4.34	0.28	0
MCPG output	the organization could not say when the tape was made , though the voice indicates that he is talking on june 14 .	0.96	4.36	0.26	0
PTS output	the agency could not say when the tape was made , although the voice says he is speaking on june 14 .	1.00	4.17	0.02	0
Target	the agency could put no exact date on the tape , though the voice says he is speaking on june 14 .	0.87	4.77	0.22	0

Table 2: An example of a source sentence sampled from the MSRPARAPHRASE train set with some representative candidates from its PPDB rewriting graph. For each of the candidates, we computed the BERT score with respect to the source (BERT_S), the normalized GPT2 perplexity (GPT2), the Levenshtein distance from the source sentence (Lev_S), and the number of spell and grammar errors detected (Errs). The PPDB editions are highlighted in green. The detected spell and grammar errors are highlighted in purple. The first candidate maximizes the semantic similarity (BERT score) and is very conservative. The second one maximizes the surface diversity (Lev_S) and takes a lot of risks. The third one shows an ill-formed paraphrase candidate. The fourth candidate emphasizes the utility of the spell-checker. The last candidate, on the fifth row, achieves our equilibrium goal. We show on a second part, the paraphrases generated by our models MCPG an PTS, and on a third part, we give the reference paraphrase from the dataset.

function PTS(input_sentence)
  candidates ← REWRITE(input_sentence)
  depth ← 1
  while time/space < budget **do**
    while time/space < layer-budget **do**
      batch ← SAMPLE(candidates)
      scored ← scored ∪ NN.SCORES(batch)
    end while
    layer_front_set ← PARETO-FRONT(scored)
    candidates ← REWRITE(layer_front_set)
    depth ← depth + 1
  end while
  return PARETO-FRONT(all scored nodes)
end function

Table 3: **Pareto Tree Search** (PTS) algorithm

5 Related work

Rule-based and statistical approaches Following the path of machine translation, the paraphrase generation literature first evolved from laboriously handcrafted linguistic rules (McKeown, 1979; Meteer and Shaked, 1988; Chandrasekar and Srinivas, 1997; Carroll et al., 1999) to more automatic and data-driven rules extraction methods (Callison-Burch et al., 2006; Madnani and Dorr, 2010). Like in machine translation, phrase-level substitution rules can be extracted by sub-sentence alignment algorithms from parallel corpora (Brown et al., 1993). Building such a dedicated corpus being a long and costly task, one usually transforms other corpora through a "pivot representation" (Barzilay and McKeown, 2001; Callison-Burch et al., 2006; Ganitkevitch et al., 2013; Chen et al., 2015). The weakness of these approaches is that phrase-level rewriting rules alone are not able to build coherent sentences. A typical data-driven paraphrase generator used to be a mixture of potentially noisy handcrafted and data-driven rewriting rules coupled with a score that had to be optimized in real-time through dynamic programming. However, dynamic programming methods like *Viterbi* are constrained by the requirement of a score that decomposes into a sum of word-level or phrase-level criteria (Xu et al., 2016). Some attempts were made to relax this constraint with search-based approaches (Chevelu et al., 2009; Daumé et al., 2009), but the global optimized criteria were simplistic, and the obtained solutions were not suitable for practical deployment.

Supervised encoder-decoder approaches Like machine translation, paraphrase generation benefited from deep neural networks and evolved to efficient end-to-end architectures that can both learn to align and translate (Bahdanau et al., 2016; Vaswani et al., 2017). Several papers like (Prakash et al., 2016a; Cao et al., 2017) set the paraphrase generation task as a supervised sequence-to-sequence problem. As confirmed by our experiments in Section 6.4, this approach is efficient for specific types of paraphrases. It is also able to produce relatively long-range transformations, but it requires huge and high-quality sentence-level aligned datasets for training.

The paraphrase generation literature mostly reports results on MSCOCO (Chen et al., 2015) and QQP (Kornél Csernai, 2017) datasets which are built respectively from image captions and semi-duplicate questions. These datasets are very specific: MSCOCO is strongly biased toward image description sentences, and QQP is dedicated to questions. The Transformer model we trained on QQP typically transforms any input into a question. For instance, from "He is speaking on june 14." it gives "Who is speaking on june 14?".

Generative and hybrid approaches Recent approaches rely on conditional generative models like CVAE to allow the generation of paraphrase sets and palliate the lack of supervision data (Gupta et al., 2018; Yang et al., 2019a; Roy and Grangier, 2019; An and Liu, 2019). Others make use of CGANs combined with reinforcement learning (Li et al., 2018; Wang and Lee, 2018). Witteveen and Andrews (2019) propose to simply fine-tune a large non-supervised model like GPT2.

Recent text-generation architectures like the one proposed in (Moryossef et al., 2019; Fu et al., 2019) make a clear separation between a rule-based planning phase and the neural realization. A similar idea was tested by Huang et al. (2019) where PPDB rules are used to control the decoder of a CVAE.

Search-based approaches Search-based methods regained interest in the text generation community for several reasons, including the need for flexibility and the fact that with deep neural-networks, the search evaluation criteria have become more reliable. These methods are often slower than auto-regressive text generation methods, but it is always possible to distillate the models into faster ones like we do in Section 6.6. In (Gröndahl and Asokan, 2019b) they deployed a search-based policy for style imitation, Schwartz and Wolter (2018) and Kumagai et al. (2016) both used MCTS for text generation. Following the same trend, Miao et al. (2018a) proposed to use Metropolis-Hasting sampling (Metropolis et al., 2004) for constrained sentence generation in an algorithm called CGMH.

Starting from the source sentence, the CGMH algorithm samples a sequence of sentences by using local editions: word replacement, deletion, and insertion. For paraphrase generation, CGMH constraints the sentence generation using a matching function that combines a measure of semantic similarity and a measure of English fluency. This model is therefore directly comparable with our MCPG and PTS approaches.

6 Experiments

The evaluation metrics and datasets are described respectively in Section 6.1 and 6.2. We paid attention to set up a rigorous validation protocol for our experiments. The reproducibility and methodology issues that we faced are discussed in Section 6.3. We compare our model with state-of-the-art supervised methods and with CGMH, another search-based algorithm. The technical details on these algorithms are developed in Section 6.4. The results are detailed in Section 6.5.

6.1 Evaluation metrics

We rely on standard machine translation metrics that compare the generated paraphrase to one or several ground-truth references (Olive et al., 2011). We report surface metrics BLEU and TER and semantic metrics METEOR and BERT[2], the average BERT score of the generated sentence with respect to the reference sentences[3].

6.2 Evaluation datasets

Table 4 gives a summary of the datasets we used. On one hand, we have large datasets like MSCOCO or OPUSPARCUS (OPUSPAR.) that are noisy and very specific. On the other hand, we have MSR-PARAPHRASE (MSRPARA.), a high-quality but small human-labeled dataset.

[2]This is the same metric, but this usage of BERT score against **reference** must not be confused with the BERT score against **source** ($BERT_S$) that we use in (1).

[3]We used the scripts from github.com/jhclark/multeval and github.com/Tiiiger/bert_score. For the BERT score we used 'bert-base-uncased_L8_no-idf_version=0.1.2'

Corpus	Size	Len	B>0.75	L<0.25	PPDB@1	PPDB@3
MSCOCO	$2.4 \cdot 10^6$	11	**13.6%**	0.6%	119	$7.4 \cdot 10^4$
MSRPARA.	$7.8 \cdot 10^3$	23	81.8%	18.3%	454	$\mathbf{7.0 \cdot 10^6}$
OPUSPAR.	$4.2 \cdot 10^7$	6	36.0%	0.5%	113	$1.5 \cdot 10^4$
PAWS	$3.6 \cdot 10^5$	20	**100%**	**65.6%**	289	$1.3 \cdot 10^6$
QQP	$1.5 \cdot 10^5$	10	64.1%	21.8%	141	$1.3 \cdot 10^5$

Table 4: **Datasets statistics**. 'Size' is the number of instances. 'Len' is the median number of words per sentence. We also report a rough distribution of the BERT score (B) and the Levenshtein distance (L) computed on the corpora paraphrases pairs. The 'PPDB@X' columns give the median number of candidates that PPDB-XL generates from one sentence respectively at one and three rewriting steps. The size of the PPDB rewriting lattice is correlated to the length of sentences.

Model applied on MSCOCO	*BLEU* ↑
RESIDUAL LSTM (Fu et al., 2020)	23.7
LBOW-TOPK (Fu et al., 2020)	25.3
RESIDUAL LSTM (Prakash et al., 2016a)	37.0
VAE-SVG-EQ (Gupta et al., 2017)	39.6
TRANSFORMER (Egonmwan and Chali, 2019b)	41.8
TRANSSEQ (Egonmwan and Chali, 2019b)	44.5

Model applied on QQP	*BLEU* ↑
RESIDUAL LSTM (Fu et al., 2020)	24.9
CGMH (Miao et al., 2018a) - *weakly-supervised*	18.8
LBOW-TOPK (Fu et al., 2020)	26.2
VAE-SVG-EQ (Gupta et al., 2017)	37.1
TRANSFORMER (Egonmwan and Chali, 2019b)	39.0
TRANSSEQ (Egonmwan and Chali, 2019b)	39.8

Table 5: Inconsistent BLEU scores as reported in several articles on paraphrase generation.

The MSCOCO-2017 set (Chen et al., 2015) contains image captions, assuming that captions associated with the same picture are paraphrases. The strengths of MSCOCO are its size and the fact that each source sentence is associated with four reference paraphrases. However, the sentences are biased towards a descriptive style, and the quality of the paraphrases is often questionable.

The OPUSPARCUS dataset (Creutz, 2018) has been extracted from movies and TV shows subtitles. It contains mostly informal dialogues.

Quora Question Pairs dataset (QQP) (Kornél Csernai, 2017) is a paraphrase identification corpus dedicated to question-answering systems. The MSRPARAPHRASE dataset (Dolan and Brockett, 2005) is mostly build with pieces of news. The sentences of this corpus are quite long. It is a small but high-quality paraphrase identification corpus that was labeled by humans. PAWS$_{wiki}$ (PAWS) (Zhang et al., 2019b; Yang et al., 2019b) is a paraphrase identification corpus that contains several lexically-similar but hard-to-classify pairs like "*Flights to Florida from New York*" and "*Flights from Florida to New York*".

6.3 Methodology and reproducibility issues

In the paraphrase generation literature, most of the papers report results on MSCOCO and QQP corpora. In table 5, we provide the BLEU scores as reported in (Prakash et al., 2016b; Gupta et al., 2017; Fu et al., 2019; Miao et al., 2018b) and (Egonmwan and Chali, 2019a). However, even if the dataset names coincide, and even if each evaluation methodology is correct on its own, the discrepancies between methodologies render these values impossible to compare with each other.

The strange gap between the residual LSTM performance of (Prakash et al., 2016b) and the one reported in (Fu et al., 2019) can be explained by the fact that the first one is using the 2014 version of MSCOCO while the other is using 2017 version. But we also found several other issues: differences in test sets splits, different strategies for sentence length shrinking (or pruning), different vocabulary size, and tokenization strategies.

Regarding the sentence lengths, Prakash et al. (2016b) and Gupta et al. (2017) shrunk all sentences to 15 words. Fu et al. (2019) set the maximum length to 16 while Egonmwan and Chali (2019a) set it to 15 and 10 respectively for the input and target sentences. Knowing that roughly 56% (resp. 5%) of MSCOCO target sentences are strictly longer than 10 (resp. 15) words, these small changes can have a great impact on the results. The vocabulary considered also differs. Fu et al. (2019) used a vocabulary of 8k and 11k tokens from the train sets of QQP and MSCOCO respectively, whereas Egonmwan and Chali (2019a) had a vocabulary of approximately 15k words that was constructed on both the train and test sets.

The scripts used to compute metrics did also differ from one paper to another, and it is known that BLEU scores can vary wildly with different parameterizations (Post, 2018).

We used the code when available and otherwise, we tried as much as possible to reproduce the models of the literature faithfully. This allowed us to have the exact same preprocessing, training and testing pipeline for all our experiments. As a side effect, it gives a grounded benchmark between the methods that we could test.

6.4 Baseline systems implementation

In the next subsections, we present the re-implemented encoder-decoder neural networks architectures and the weakly-supervised paraphrase generator used as baselines in our experiments.

Supervised paraphrase generators As supervised baselines, we trained three neural network architectures that were previously reported to achieve good results on MSCOCO and QQP, in particular, the Seq2Seq architecture, a *Residual* LSTM architecture (Prakash et al., 2016b) and a TRANSFORMER model (Egonmwan and Chali, 2019a). We extended the experiments to the other aligned corpora: MSRPARAPHRASE, OPUSPARCUS and PAWS.

To be more precise, we trained a 4-layers LSTM Seq2Seq with a bidirectional encoder and decoder using attention. This architecture is reported as SEQ2SEQ in the results. We trained a 4-layer Residual LSTM Seq2Seq as introduced by Prakash et al. (2016b) and reproduced by Fu et al. (2019). This architecture is reported as RESIDUAL LSTM in the results. The results we obtained with this model are close to the ones reported by Fu et al. (2019).

Finally, we trained a TRANSFORMER using the *transformer_base* hyper-parameters set from (Vaswani et al., 2017). This architecture is reported as TRANSFORMER BASE in the results.

For all the encoder-decoder experiments, we used the *fairseq* framework (Ott et al., 2019) that implements the SEQ2SEQ and TRANSFORMER architectures. We added our own implementation of the RESIDUAL LSTM architecture.

For preprocessing, we used Moses tokenizer and subword segmentation following Sennrich et al. (2016b) and using the subword-nmt library . The maximum sentence length is set to 1024 tokens which the default setting in *fairseq*. For decoding, we did a beam search with a beam of size 5.

Weakly-supervised paraphrase generator For the weakly-supervised strategy CGMH introduced by Miao et al. (2018b) we used the official code. We managed to reproduce their results on QQP. On our test set we achieve a BLEU score of 22.5 while they reported 18.8. We then extended the experiment to other datasets and metrics.

6.5 Results

Table 6 summarizes the results of the comparison of our models, supervised encoder-decoder neural networks and the weakly-supervised method CGMH. Overall, these results are mixed: it is however important to keep in mind that contrary to the supervised baselines which are retrained for each dataset, the parameters of the CGMH, MCPG and PTS models are left unchanged.

On the MSCOCO and QQP datasets, the supervised baselines achieve clearly better results, but MCPG and PTS achieve better results on OPUSPARCUS and PAWS except with the BERT score for which the TRANSFORMER model achieves similar results. On MSRPARAPHRASE, the encoder-decoder neural networks models perform poorly. This result can be explained by the small number of training examples available on this corpus (See Table 4). On the weakly-supervised side, MCPG and PTS models outperform the CGMH baseline on all corpora except on the MSCOCO dataset where the results are similar.

These results prove that even without a specialized training sets, generic search-based methods are competitive for paraphrase generation. However, it is a fact that encoder-decoder networks have excellent performances for text generation and have the potential to generate more complex paraphrases than those obtained by simple local transformations as in our models.

Training a general – all-purpose – paraphrase generation network would require a huge volume of data. And there is yet much less aligned corpora available for paraphrase than for translation.

6.6 Data-augmentation experiment

In order to get the best of both worlds, one option is to enrich the training set of a TRANSFORMER with the results of a search-based method.

To test this idea, we used our models to augment the MSRPARAPHRASE training set. For that purpose we created new pairs of paraphrases from unused sentences of MSRPARAPHRASE (the pairs labeled as "not paraphrases") using MCPG and PTS. We then trained a new supervised TRANSFORMER models on the augmented training sets.

Having no guarantee that our models generate syntactically perfect sentences, we inverted the pairs, thus taking the generated paraphrases as input and the dataset's sentences as output. This trick, called *back-translation*, forces the target model to generate correct sentences (Sennrich et al., 2016a; Edunov et al., 2018).

We report the results of this experiment in Table 7. The models trained with the augmented

Corpus	Model	*BLEU* ↑	*TER* ↓	*METEOR* ↑	*BERT* ↑
MSCOCO	SEQ2SEQ	**27.5**	**62.3**	**24.3**	0.76
	RESIDUAL LSTM	**26.9**	63.3	**24.2**	0.76
	TRANSFORMER BASE	**26.9**	63.3	**24.2**	0.76
	CGMH	17.3	72.6	21.9	0.7
	MCPG	16.5	73.5	23.2	0.71
	PTS	17.0	69.9	22.8	0.64
QQP	SEQ2SEQ	**29.2**	60.3	30.7	0.8
	RESIDUAL LSTM	**28.4**	**59.1**	30.2	0.8
	TRANSFORMER BASE	**29.1**	**59.5**	30.5	0.8
	CGMH	22.5	65.0	27.0	0.72
	MCPG	24.1	64.5	**31.8**	0.78
	PTS	25.6	**58.7**	31.4	0.78
OPUSPARCUS	SEQ2SEQ	8.4	79.3	13.8	0.69
	RESIDUAL LSTM	8.1	78.6	14.3	0.7
	TRANSFORMER BASE	8.1	84.3	13.9	0.7
	CGMH	7.6	78.9	16.8	0.58
	MCPG	**9.6**	78.6	**23.3**	0.67
	PTS	**9.1**	**70.2**	22.1	0.66
PAWS	SEQ2SEQ	44.0	36.8	39.2	**0.92**
	RESIDUAL LSTM	43.6	37.1	38.9	**0.92**
	TRANSFORMER BASE	42.4	37.6	38.9	**0.92**
	CGMH	15.4	58.1	20.7	0.61
	MCPG	55.5	24.3	**49.2**	**0.93**
	PTS	**57.9**	**21.9**	**48.5**	**0.92**
MSRPARAPHRASE	SEQ2SEQ	11.6	89.5	12.6	0.53
	RESIDUAL LSTM	10.5	93.7	11.2	0.52
	TRANSFORMER BASE	20.7	76.7	21.6	0.65
	CGMH	9.7	72.9	15.4	0.48
	MCPG	**39.3**	52.4	**37.2**	**0.81**
	PTS	**40.3**	**48.4**	36.1	**0.80**

Table 6: **Experiments summary.** Symbol '↑' means that higher value is better. Significantly best values are marked in bold. MCPG and PTS outperform state-of-the-art models on OPUSPAR., PAWS and MSRPARA.. Contrary to the supervised baselines, the parameters of the MCPG and PTS models are left unchanged for each dataset.

Train set	*BLEU*	*TER*	*METEOR*	*BERT*
ORIG.	20.7	76.7	21.6	**0.65**
ORIG. + MCPG	**25.3**	**69.6**	24.5	0.63
ORIG. + PTS	25.3	71.1	**24.6**	0.63

Table 7: **Data-augmentation experiment summary**. We trained a TRANSFORMER base model on three versions of the MSRPARAPHRASE set: the original train set (ORIG) and the original train set extended by paraphrasing other sentences from the same distribution with the MCPG (ORIG + MCPG) and PTS (ORIG + PTS) models.

training sets achieved a significant performance gain on BLEU, TER and METEOR.

7 Conclusion

We experimented with two search-based approaches for paraphrase generation. These approaches are pragmatic and flexible. Being generic, our approaches did not overfit on small datasets. We performed extensive experiments with a rigorous evaluation methodology that we applied both on our algorithms and on the other related methods that we tried to reproduce faithfully. These experiments confirm that our two methods, namely MCPG and PTS, are comparable to supervised state-of-the-art baselines despite being less tightly supervised. When compared with CGMH, another search-based and weakly-supervised method, our algorithms proved to be faster and more efficient.

We plan to refine the scoring with deep reinforcement learning techniques and enrich the edition rules with more sophisticated patterns like phrase permutations. Our search algorithms remain slow for a real-time deployment: the current versions are better suited as an offline model for data augmentation. The experiment of Section 6.6 confirms that this application of search-based methods is a promising research avenue. A planning-then-realization hybridization like the one proposed by Moryossef et al. (2019) and Fu et al. (2019) could also be considered for further works.

Acknowledgment

We want to thank the reviewers for their constructive comments. This work was funded by the ANR Cifre convention N°2018/1559 and Orange-Labs Research.

References

Zhecheng An and Sicong Liu. 2019. Towards Diverse Paraphrase Generation Using Multi-Class Wasserstein GAN. *CoRR*, abs/1909.13827.

Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. 2002. The Nonstochastic Multiarmed Bandit Problem. *SIAM Journal on Computing*, 32(1):48–77.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2016. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv:1409.0473 [cs, stat]*. ArXiv: 1409.0473.

Regina Barzilay and Kathleen R. McKeown. 2001. Extracting paraphrases from a parallel corpus. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics - ACL '01*, pages 50–57, Toulouse, France. Association for Computational Linguistics.

Peter E Brown, Vincent J Della Pietra, Stephen A Della Pietra, and Robert L Mercer. 1993. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19(2):50.

Chris Callison-Burch. 2008. Syntactic Constraints on Paraphrases Extracted from Parallel Corpora. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 196–205, Honolulu, Hawaii. Association for Computational Linguistics.

Chris Callison-Burch, Philipp Koehn, and Miles Osborne. 2006. Improved Statistical Machine Translation Using Paraphrases. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 17–24, New York City, USA. Association for Computational Linguistics.

Ziqiang Cao, Chuwei Luo, Wenjie Li, and Sujian Li. 2017. Joint Copying and Restricted Generation for Paraphrase. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 3152–3158.

John Carroll, Guido Minnen, Darren Pearce, Yvonne Canning, Siobhan Devlin, and John Tait. 1999. Simplifying text for language-impaired readers. In *Ninth Conference of the European Chapter of the Association for Computational Linguistics*, pages 269–270, Bergen, Norway. Association for Computational Linguistics.

R. Chandrasekar and B. Srinivas. 1997. Automatic induction of rules for text simplification. *Knowledge-Based Systems*, 10(3):183–190.

Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollar, and C. Lawrence Zitnick. 2015. Microsoft COCO Captions: Data Collection and Evaluation Server. *arXiv:1504.00325 [cs]*. ArXiv: 1504.00325.

Jonathan Chevelu, Thomas Lavergne, Yves Lepage, and Thierry Moudenc. 2009. Introduction of a new paraphrase generation tool based on Monte-Carlo sampling. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 249–252, Suntec, Singapore. Association for Computational Linguistics.

Mathias Creutz. 2018. Open Subtitles Paraphrase Corpus for Six Languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Hal Daumé, John Langford, and Daniel Marcu. 2009. Search-based structured prediction. *Machine Learning*, 75(3):297–325.

Bill Dolan and Chris Brockett. 2005. Automatically Constructing a Corpus of Sentential Paraphrases. In *Third International Workshop on Paraphrasing (IWP2005)*. Asia Federation of Natural Language Processing.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding Back-Translation at Scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

Elozino Egonmwan and Yllias Chali. 2019a. Transformer and seq2seq model for Paraphrase Generation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 249–255, Hong Kong. Association for Computational Linguistics.

Elozino Egonmwan and Yllias Chali. 2019b. Transformer and seq2seq model for Paraphrase Generation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 249–255, Hong Kong. Association for Computational Linguistics.

Yao Fu, Yansong Feng, and John P Cunningham. 2019. Paraphrase Generation with Latent Bag of Words. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d\textquotesingle Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 13645–13656. Curran Associates, Inc.

Yao Fu, Yansong Feng, and John P. Cunningham. 2020. Paraphrase Generation with Latent Bag of Words. *arXiv:2001.01941 [cs]*. ArXiv: 2001.01941.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The Paraphrase Database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764, Atlanta, Georgia. Association for Computational Linguistics.

Sylvain Gelly and David Silver. 2007. Combining online and offline knowledge in uct. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, page 273–280, New York, NY, USA. Association for Computing Machinery.

Tommi Gröndahl and N. Asokan. 2019a. Effective writing style imitation via combinatorial paraphrasing. *CoRR*, abs/1905.13464.

Tommi Gröndahl and N. Asokan. 2019b. Effective writing style imitation via combinatorial paraphrasing. *CoRR*, abs/1905.13464.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. 2017. A Deep Generative Framework for Paraphrase Generation. *arXiv:1709.05074 [cs]*. ArXiv: 1709.05074.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. 2018. A Deep Generative Framework for Paraphrase Generation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Sanda Harabagiu and Andrew Hickl. 2006. Methods for Using Textual Entailment in Open-Domain Question Answering. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 905–912, Sydney, Australia. Association for Computational Linguistics.

Shaohan Huang, Yu Wu, Furu Wei, and Zhongzhi Luan. 2019. Dictionary-Guided Editing Networks for Paraphrase Generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:6546–6553.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial Example Generation with Syntactically Controlled Paraphrase Networks. *arXiv:1804.06059 [cs]*. ArXiv: 1804.06059.

Levente Kocsis and Csaba Szepesvári. 2006. Bandit Based Monte-carlo Planning. In *Proceedings of the 17th European Conference on Machine Learning*, ECML'06, pages 282–293, Berlin, Heidelberg. Springer-Verlag. Event-place: Berlin, Germany.

Kornél Csernai. 2017. First Quora Dataset Release: Question Pairs - Data @ Quora - Quora.

Kaori Kumagai, Ichiro Kobayashi, Daichi Mochihashi, Hideki Asoh, Tomoaki Nakamura, and Takayuki Nagai. 2016. Human-like Natural Language Generation Using Monte Carlo Tree Search. In *Proceedings of the INLG 2016 Workshop on Computational Creativity in Natural Language Generation*, pages 11–18, Edinburgh, UK. Association for Computational Linguistics.

Zichao Li, Xin Jiang, Lifeng Shang, and Hang Li. 2018. Paraphrase Generation with Deep Reinforcement Learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3865–3878, Brussels, Belgium. Association for Computational Linguistics.

Nitin Madnani and Bonnie J. Dorr. 2010. Generating Phrasal and Sentential Paraphrases: A Survey of Data-Driven Methods. *Computational Linguistics*, 36(3):341–387.

Kathleen R. McKeown. 1979. Paraphrasing Using Given and New Information in a Question-Answer System. In *17th Annual Meeting of the Association for Computational Linguistics*, pages 67–72, La Jolla, California, USA. Association for Computational Linguistics.

Marie Meteer and Varda Shaked. 1988. STRATEGIES FOR EFFECTIVE PARAPHRASING. In *Coling Budapest 1988 Volume 2: International Conference on Computational Linguistics*.

Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. 2004. Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, 21(6):1087. Publisher: American Institute of PhysicsAIP.

Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. 2018a. CGMH: Constrained Sentence Generation by Metropolis-Hastings Sampling. *arXiv:1811.10996 [cs, math, stat]*. ArXiv: 1811.10996.

Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. 2018b. CGMH: Constrained Sentence Generation by Metropolis-Hastings Sampling. *arXiv:1811.10996 [cs, math, stat]*. ArXiv: 1811.10996.

Amit Moryossef, Yoav Goldberg, and Ido Dagan. 2019. Step-by-Step: Separating Planning from Realization in Neural Data-to-Text Generation. *arXiv:1904.03396 [cs]*. ArXiv: 1904.03396.

Daniel Naber. 2003. LanguageTool: A Rule-Based Style and Grammar Checker.

Joseph Olive, Caitlin Christianson, and John McCary, editors. 2011. *Handbook of Natural Language Processing and Machine Translation: DARPA Global Autonomous Language Exploitation*. Springer-Verlag, New York.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2015. PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational*

Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers), pages 425–430, Beijing, China. Association for Computational Linguistics.

Matt Post. 2018. A Call for Clarity in Reporting BLEU Scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Aaditya Prakash, Sadid A. Hasan, Kathy Lee, Vivek Datla, Ashequl Qadir, Joey Liu, and Oladimeji Farri. 2016a. Neural paraphrase generation with stacked residual LSTM networks. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2923–2934, Osaka, Japan. The COLING 2016 Organizing Committee.

Aaditya Prakash, Sadid A Hasan, Kathy Lee, Vivek Datla, Ashequl Qadir, Joey Liu, and Oladimeji Farri. 2016b. Neural Paraphrase Generation with Stacked Residual LSTM Networks. *COLING*, page 12.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

Aurko Roy and David Grangier. 2019. Unsupervised Paraphrasing without Translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6033–6039, Florence, Italy. Association for Computational Linguistics.

Tobias Schwartz and Diedrich Wolter. 2018. A Variant of Monte-Carlo Tree Search for Referring Expression Generation. In *KI 2018: Advances in Artificial Intelligence*, pages 315–326. Springer, Cham.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. 2016. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. *arXiv:1706.03762 [cs]*. ArXiv: 1706.03762.

Yaushian Wang and Hung-Yi Lee. 2018. Learning to Encode Text as Human-Readable Summaries using Generative Adversarial Networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4187–4195, Brussels, Belgium. Association for Computational Linguistics.

Sam Witteveen and Martin Andrews. 2019. Paraphrasing with Large Language Models. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 215–220, Hong Kong. Association for Computational Linguistics.

Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. 2016. Optimizing Statistical Machine Translation for Text Simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415.

Zhao Yan, Nan Duan, Junwei Bao, Peng Chen, Ming Zhou, Zhoujun Li, and Jianshe Zhou. 2016. DocChat: An Information Retrieval Approach for Chatbot Engines Using Unstructured Documents. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 516–525, Berlin, Germany. Association for Computational Linguistics.

Qian Yang, Dinghan Shen, Yong Cheng, Wenlin Wang, Guoyin Wang, Lawrence Carin, et al. 2019a. An end-to-end generative architecture for paraphrase generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3123–3133.

Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. 2019b. PAWS-X: A Cross-lingual Adversarial Dataset for Paraphrase Identification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3678–3683, Hong Kong, China. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2019a. BERTScore: Evaluating Text Generation with BERT. *arXiv:1904.09675 [cs]*. ArXiv: 1904.09675.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019b. PAWS: Paraphrase Adversaries from Word Scrambling. In *Proceedings of the 2019 Conference of*

the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 1298–1308, Minneapolis, Minnesota. Association for Computational Linguistics.