Split and Rephrase

Citation for published version:

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published In:
The 2017 Conference on Empirical Methods on Natural Language Processing (EMNLP)

General rights
Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.
Abstract

We propose a new sentence simplification task (Split-and-Rephrase) where the aim is to split a complex sentence into a meaning preserving sequence of shorter sentences. Like sentence simplification, splitting-and-rephrasing has the potential of benefiting both natural language processing and societal applications. Because shorter sentences are generally better processed by NLP systems, it could be used as a preprocessing step which facilitates and improves the performance of parsers, semantic role labelers and machine translation systems. It should also be of use for people with reading disabilities because it allows the conversion of longer sentences into shorter ones. This paper makes two contributions towards this new task. First, we create and make available a benchmark consisting of 1,066,115 tuples mapping a single complex sentence to a sequence of sentences expressing the same meaning. Second, we propose five models (vanilla sequence-to-sequence to semantically-motivated models) to understand the difficulty of the proposed task.

1 Introduction

Several sentence rewriting operations have been extensively discussed in the literature: sentence compression, multi-sentence fusion, sentence paraphrasing and sentence simplification. Sentence compression rewrites an input sentence into a shorter paraphrase (Knight and Marcu, 2000; Cohn and Lapata, 2008; Filippova and Strube, 2008; Pitler, 2010; Filippova et al., 2015; Toutanova et al., 2016). Sentence fusion consists of combining two or more sentences with overlapping information content, preserving common information and deleting irrelevant details (McKeown et al., 2010; Filippova, 2010; Thadani and McKeown, 2013). Sentence paraphrasing aims to rewrite a sentence while preserving its meaning (Dras, 1999; Barzilay and McKeown, 2001; Bannard and Callison-Burch, 2005; Wubben et al., 2010; Mallinson et al., 2017). Finally, sentence (or text) simplification aims to produce a text that is easier to understand (Siddharthan et al., 2004; Zhu et al., 2010; Woodsend and Lapata, 2011; Wubben et al., 2012; Narayan and Gardent, 2014; Xu et al., 2015; Narayan and Gardent, 2016; Zhang and Lapata, 2017). Because the vocabulary used, the length of the sentences and the syntactic structures occurring in a text are all factors known to affect readability, simplification systems mostly focus on modelling three main text rewriting operations: simplifying paraphrasing, sentence splitting and deletion.

We propose a new sentence simplification task, which we dub Split-and-Rephrase, where the goal is to split a complex input sentence into shorter sentences while preserving meaning. In that task, the emphasis is on sentence splitting and rephrasing. There is no deletion and no lexical or phrasal simplification but the systems must learn to split complex sentences into shorter ones and to make the syntactic transformations required by the split (e.g., turn a relative clause into a main clause). Table 1 summarises the similarities and differences between the five sentence rewriting tasks.

Like sentence simplification, splitting-and-rephrasing could benefit both natural language processing and societal applications. Because shorter sentences are generally better processed by NLP systems, it could be used as a preprocess-
Table 1: Similarities and differences between sentence rewriting tasks with respect to splitting (Split), deletion (Delete), rephrasing (Rephr.) and meaning preserving (MPre.) operations (Y: yes, N: No, ?Y: should do but most existing approaches do not).

<table>
<thead>
<tr>
<th>Task</th>
<th>Split</th>
<th>Delete</th>
<th>Rephr.</th>
<th>MPre.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compression</td>
<td>N</td>
<td>N</td>
<td>?Y</td>
<td>Y</td>
</tr>
<tr>
<td>Fusion</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>?Y</td>
</tr>
<tr>
<td>Paraphrasing</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Simplification</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>Split-and-Rephrase</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
</tbody>
</table>

Table 2 Related Work

We briefly review previous work on sentence splitting and rephrasing.

Sentence Splitting. Of the four sentence rewriting tasks (paraphrasing, fusion, compression and simplification) mentioned above, only sentence simplification involves sentence splitting. Most simplification methods learn a statistical model (Zhu et al., 2010; Coster and Kauchak, 2011; Woodsend and Lapata, 2011; Wubben et al., 2012; Narayan and Gardent, 2014) from the parallel dataset of complex-simplified sentences derived by Zhu et al. (2010) from Simple English Wikipedia\(^2\) and the traditional one\(^3\).

For training Split-and-Rephrase models, this dataset is arguably ill suited as it consists of 108,016 complex and 114,924 simplified sentences thereby yielding an average number of simple sentences per complex sentence of 1.06. Indeed, Narayan and Gardent (2014) report that only 6.1\% of the complex sentences are in fact split in the corresponding simplification. A more detailed evaluation of the dataset by Xu et al. (2015) further shows that (i) for a large number of pairs, the

\(^2\)Simple English Wikipedia (http://simple.wikipedia.org) is a corpus of simple texts targeting “children and adults who are learning English Language” and whose authors are requested to “use easy words and short sentences”.

\(^3\)English Wikipedia (http://en.wikipedia.org).
simplifications are in fact not simpler than the input sentence, (ii) automatic alignments resulted in incorrect complex-simplified pairs and (iii) models trained on this dataset generalised poorly to other text genres. Xu et al. (2015) therefore propose a new dataset, Newsela, which consists of 1,130 news articles each rewritten in four different ways to match 5 different levels of simplicity. By pairing each sentence in that dataset with the corresponding sentences from simpler levels (and ignoring pairs of contiguous levels to avoid sentence pairs that are too similar to each other), it is possible to create a corpus consisting of 96,414 distinct complex and 97,135 simplified sentences. Here again however, the proportion of splits is very low. As we shall see in Section 3.3, the new dataset we propose differs from both the Newsela and the Wikipedia simplification corpus, in that it contains a high number of splits. In average, this new dataset associates 4.99 simple sentences with each complex sentence.

Rephrasing. Sentence compression, sentence fusion, sentence paraphrasing and sentence simplification all involve rephrasing. Paraphrasing approaches include bootstrapping approaches which start from slotted templates (e.g., “X is the author of Y”) and seed (e.g., “X = Jack Kerouac, Y = “On the Road””) to iteratively learn new templates from the seeds and new seeds from the new templates (Ravichandran and Hovy, 2002; Duclaye et al., 2003); systems which extract paraphrase patterns from large monolingual corpora and use them to rewrite an input text (Duboue and Chu-Carroll, 2006; Narayan et al., 2016); statistical machine translation (SMT) based systems which learn paraphrases from monolingual parallel (Barzilay and McKeown, 2001; Zhao et al., 2008), comparable (Quirk et al., 2004) or bilingual parallel (Bannard and Callison-Burch, 2005; Ganitkevitch et al., 2011) corpora; and a recent neural machine translation (NMT) based system which learns paraphrases from bilingual parallel corpora (Mallinson et al., 2017).

In sentence simplification approaches, rephrasing is performed either by a machine translation (Coster and Kauchak, 2011; Wubben et al., 2012; Narayan and Gardent, 2014; Xu et al., 2016; Zhang and Lapata, 2017) or by a probabilistic model (Zhu et al., 2010; Woodsend and Lapata, 2011). Other approaches include symbolic approaches where hand-crafted rules are used e.g., to split coordinated and subordinated sentences into several, simpler clauses (Chandrasekar and Srinivas, 1997; Siddharthan, 2002; Canning, 2002; Siddharthan, 2010, 2011) and lexical rephrasing rules are induced from the Wikipedia simplification corpus (Siddharthan and Mandya, 2014).

Most sentence compression approaches focus on deleting words (the words appearing in the compression are words occurring in the input) and therefore only perform limited paraphrasing. As noted by Pitler (2010) and Toutanova et al. (2016) however, the ability to paraphrase is key for the development of abstractive summarisation systems since summaries written by humans often rephrase the original content using paraphrases or synonyms or alternative syntactic constructions. Recent proposals by Rush et al. (2015) and Bingel and Søgaard (2016) address this issue. Rush et al. (2015) proposed a neural model for abstractive compression and summarisation, and Bingel and Søgaard (2016) proposed a structured approach to text simplification which jointly predicts possible compressions and paraphrases.

None of these approaches requires that the input be split into shorter sentences so that both the corpora used, and the models learned, fail to adequately account for the various types of specific rephrasings occurring when a complex sentence is split into several shorter sentences.

Finally, sentence fusion does induce rephrasing as one sentence is produced out of several. However, research in that field is still hampered by the small size of datasets for the task, and the difficulty of generating one (Daume III and Marcu, 2004). Thus, the dataset of Thadani and McKeown (2013) only consists of 1,858 fusion instances of which 873 have two inputs, 569 have three and 416 have four. This is arguably not enough for learning a general Split-and-Rephrase model.

In sum, while work on sentence rewriting has made some contributions towards learning to split and/or to rephrase, the interaction between these two subtasks have never been extensively studied nor are there any corpora available that would support the development of models that can both split and rephrase. In what follows, we introduce such a benchmark and present some baseline models which provide some interesting insights on how to address the Split-and-Rephrase problem.
3 The WEBSPLIT Benchmark

We derive a Split-and-Rephrase dataset from the WEBNLG corpus presented in Gardent et al. (2017).

3.1 The WEBNLG Dataset

In the WEBNLG dataset, each item consists of a set of RDF triples \( M \) and one or more texts \( T_i \) verbalising those triples.

An RDF (Resource Description Format) triple is a triple of the form subject|property|object where the subject is a URI (Uniform Resource Identifier), the property is a binary relation and the object is either a URI or a literal value such as a string, a date or a number. In what follows, we refer to the sets of triples representing the meaning of each item, and \( \{|T_1\}, \{|T_2\} \) and \( \{|T_3\} \) listing possible verbalisations of these meanings.

The WEBNLG dataset consists of 13,308 MR-Text pairs, 7049 distinct MRs, 1482 RDF entities and 8 DBpedia categories (Airport, Astronaut, Building, Food, Monument, SportsTeam, University, WrittenWork). The number of RDF triples in MRs varies from 1 to 7. The number of distinct RDF tree shapes in MRs is 60.

3.2 Creating the WEBSPLIT Dataset

To construct the Split-and-Rephrase dataset, we make use of the fact that the WEBNLG dataset (i) associates texts with sets of RDF triples and (ii) contains texts of different lengths and complexity corresponding to different subsets of RDF triples. The idea is the following. Given a WEBNLG MR-Text pair of the form \( (M, T) \) where \( T \) is a single complex sentence, we search the WEBNLG dataset for a set \( \{(M_1, T_1), \ldots, (M_n, T_n)\} \) such that \( \{M_1, \ldots, M_n\} \) is a partition of \( M \) and \( \{T_1, \ldots, T_n\} \) forms a text with more than one sentence. To achieve this, we proceed in three main steps as follows.

Sentence segmentation We first preprocess all 13,308 distinct verbalisations contained in the WEBNLG corpus using the Stanford CoreNLP pipeline (Manning et al., 2014) to segment each verbalisation \( T_i \) into sentences.

Sentence segmentation allows us to associate each text \( T \) in the WEBNLG corpus with the number of sentences it contains. This is needed to identify complex sentences with no split (the input to the Split-and-Rephrase task) and to know how many sentences are associated with a given set of RDF triples (e.g., 2 triples may be realised by a single sentence or by two). As the CoreNLP sentence segmentation often fails on complex/rare named entities thereby producing unwarranted splits, we verified the sentence segmentations produced by the CoreNLP sentence segmentation module for each WEBNLG verbalisation and manually corrected the incorrect ones.

Pairing Using the semantic information given by WEBNLG RDF triples and the information about the number of sentences present in a WEBNLG text produced by the sentence segmentation step, we produce all items of the form \( ((M_C, C), \{(M_1, T_1) \ldots (M_n, T_n)\}) \) such that:

- \( C \) is a single sentence with semantics \( M_C \).
- \( T_1 \ldots T_n \) is a sequence of texts that contains at least two sentences.
- The disjoint union of the semantics \( M_1 \ldots M_n \) of the texts \( T_1 \ldots T_n \) is the same as the semantics \( M_C \) of the complex sentence \( C \). That is, \( M_C = M_1 \cup \ldots \cup M_n \).

This pairing is made easy by the semantic information contained in the WEBNLG corpus and includes two subprocesses depending on whether complex and split sentences come from the same WEBNLG entry or not.

Within entries. Given a set of RDF triples \( M_C \), a WEBNLG entry will usually contain several alternative verbalisations for \( M_C \) (e.g., \( T_1 \) and \( T_2 \) in Figure 1 are two possible verbalisations of \( M_1 \)). We first search for entries where one verbalisation \( T_C \) consists of a single sentence and another verbalisation \( T \) contains more than one sentence. For such cases, we create an entry of the form \( ((M_C, T_C), \{(M_C, T)\}) \) such that, \( T_C \) is a single sentence and \( T \) is a text consisting of more than one sentence. The second example item for WEB-SPLIT in Figure 1 presents this case. It uses different verbalisations \( T_1 \) and \( T_2 \) of the same meaning representation \( M_1 \) in WEBNLG to construct
a WEBNLG item associating the complex sentence ($T_1^1$) with a text ($T_1^2$) made of three short sentences.

Across entries. Next we create \((M, C) \in \{(M_1, T_1) \ldots (M_n, T_n)\}\) entries by searching for all WEBNLG texts \(C\) consisting of a single sentence. For each such text, we create all possible partitions of its semantics \(M_C\) and for each partition, we search the WEBNLG corpus for matching entries i.e., for a set \(S\) of \((M_i, T_i)\) pairs such that (i) the disjoint union of the semantics \(M_i\) in \(S\) is equal to \(M_C\) and (ii) the resulting set of texts contains more than one sentence. The first example item for WEBSPLIT in Figure 1 is a case in point. \(C (= T_1^1)\) is the single, complex sentence whose meaning is represented by the triple \(M_1\). \((T_2, T_3)\) is the sequence of shorter texts \(C\) is mapped to. And the semantics \(M_2\) and \(M_3\) of these two texts forms a partition over \(M_1\).

Ordering. For each item \((M_C, C) \in \{(M_1, T_1) \ldots (M_n, T_n)\}\) produced in the preceding step, we determine an order on \(T_1 \ldots T_n\) as follows. We observed that the WEBNLG texts mostly\(^5\) follow the order in which the RDF triples are presented. Since this order corresponds to a left-to-right depth-first traversal of the RDF tree, we use this order to order the sentences in the \(T_i\) texts.

### 3.3 Results

By applying the above procedure to the WEBNLG dataset, we create 1,100,166 pairs of the form \(\{(M_C, T_C), \{(M_1, T_1) \ldots (M_n, T_n)\}\}\) where \(T_C\) is a complex sentence and \(T_1 \ldots T_n\) is a sequence of texts with semantics \(M_1, \ldots, M_n\) expressing the same content \(M_C\) as \(T_C\). 1,945 of these pairs were of type “Within entries” and the rest were of type “Across entries”. In total, there are 1,066,115 distinct \(\{T_C, T_1 \ldots T_n\}\) pairs with 5,546 distinct complex sentences. Complex sentences are associated with 192.23 rephrasings in average (min: 1, max: 76283, median: 16). The number of sentences in the rephrasings varies between 2 and 7 with an average of 4.99. The vocabulary size is 3,311.

\(^5\) As shown by the examples in Figure 1, this is not always the case. We use this constraint as a heuristic to determine an ordering on the set of sentences associated with each input.
4 Problem Formulation

The Split-and-Rephrase task can be defined as follows. Given a complex sentence $C$, the aim is to produce a simplified text $T$ consisting of a sequence of texts $T_1 \ldots T_n$ such that $T$ forms a text of at least two sentences and the meaning of $C$ is preserved in $T$. In this paper, we propose to approach this problem in a supervised setting where we aim to maximise the likelihood of $T$ given $C$ and model parameters $\theta$: $P(T|C; \theta)$. To exploit the different levels of information present in the WebSPLIT benchmark, we break the problem in the following ways:

$$P(T|C; \theta) = \sum_{M_C} P(T|C; M_C; \theta)P(M_C|C; \theta)$$

$$= P(T|C; M_C; \theta), \text{if } M_C \text{ is known.}$$

$$= \sum_{M_{1...n}} P(T|C; M_C; M_{1...n}; \theta) \times P(M_{1...n}|C; M_C; \theta)$$

where, $M_C$ is the meaning representation of $C$ and $M_{1...n}$ is a set $\{M_1, \ldots, M_n\}$ which partitions $M_C$.

5 Split-and-Rephrase Models

In this section, we propose five different models which aim to maximise $P(T|C; \theta)$ by exploiting different levels of information in the WebSPLIT benchmark.

5.1 A Probabilistic, Semantic-Based Approach

Narayan and Gardent (2014) describes a sentence simplification approach which combines a probabilistic model for splitting and deletion with a phrase-based statistical machine translation (SMT) and a language model for rephrasing (re-ordering and substituting words). In particular, the splitting and deletion components exploit the deep meaning representation (a Discourse Representation Structure, DRS) of a complex sentence produced by Boxer (Curran et al., 2007).

Based on this approach, we create a Split-and-Rephrase model (aka HYBRIDSIMPL) by (i) including only the splitting and the SMT models (we do not learn deletion) and (ii) training the model on the WebSPLIT corpus.

5.2 A Basic Sequence-to-Sequence Approach

Sequence-to-sequence models (also referred to as encoder-decoder) have been successfully applied to various sentence rewriting tasks such as machine translation (Sutskever et al., 2011; Bahdanau et al., 2014), abstractive summarisation (Rush et al., 2015) and response generation (Shang et al., 2015). They first use a recurrent neural network (RNN) to convert a source sequence to a dense, fixed-length vector representation (encoder). They then use another recurrent network (decoder) to convert that vector to a target sequence.

We use a three-layered encoder-decoder model with LSTM (Long Short-Term Memory, (Hochreiter and Schmidhuber, 1997)) units for the Split-and-Rephrase task. Our decoder also uses the local-p attention model with feed input as in (Luong et al., 2015). It has been shown that the local attention model works better than the standard global attention model of Bahdanau et al. (2014). We train this model (SEQ2SEQ) to predict, given a complex sentence, the corresponding sequence of shorter sentences.

The SEQ2SEQ model is learned on pairs $(C,T)$ of complex sentences and the corresponding text. It directly optimises $P(T|C; \theta)$ and does not take advantage of the semantic information available in the WebSPLIT benchmark.

5.3 A Multi-Source Sequence-to-Sequence Approach

In this model, we learn a multi-source model which takes into account not only the input complex sentence but also the associated set of RDF triples available in the WebSPLIT dataset. That is, we maximise $P(T|C; M_C; \theta)$ (Eqn. 2) and learn a model to predict, given a complex sentence $C$ and its semantics $M_C$, a rephrasing of $C$.

As noted by Gardent et al. (2017), the shape of the input may impact the syntactic structure of the corresponding text. For instance, an input containing a path $(X|P_1|Y)(Y|P_2|Z)$ equating the object of a property $P_1$ with the subject of a property $P_2$ may favour a verbalisation containing a subject relative (“$x$ V1 y who V2 z”). Taking into account not only the sentence $C$ that needs to be rephrased but also its semantics $M_C$ may therefore help learning.

We model $P(T|C; M_C; \theta)$ using a multi-source sequence-to-sequence neural framework (we refer to this model as MULTISEQ2SEQ). The core idea comes from Zoph and Knight (2016) who show that a multi-source model trained on trilingual translation pairs $((f, g), h)$ outperforms sev-
eral strong single source baselines. We explore a similar “trilingual” setting where $f$ is a complex sentence ($C$), $g$ is the corresponding set of RDF triples ($M_C$) and $h$ is the output rephrasing ($T$).

We encode $C$ and $M_C$ using two separate RNN encoders. To encode $M_C$ using RNN, we first linearise $M_C$ by doing a depth-first left-right RDF tree traversal and then tokenise using the Stanford CoreNLP pipeline (Manning et al., 2014). Like in SEQ2SEQ, we model our decoder with the local-

t-specificity model with feed input as in (Luong et al., 2015), but now it looks at both source en-
coders simultaneously by creating separate context vector for each encoder. For a detailed explanation of multi-source encoder-decoders, we refer the reader to Zoph and Knight (2016).

5.4 Partitioning and Generating

As the name suggests, the Split-and-Rephrase task can be seen as a task which consists of two sub-
tasks: (i) splitting a complex sentence into several shorter sentences and (ii) rephrasing the in-
sentence to fit the new sentence distribution. We consider an approach which explicitly mod-
ifies these two steps (Eqn. 3). A first model $P(M_1, \ldots, M_n|C; M_C; \theta)$ learns to partition a set $M_C$ of RDF triples associated with a complex sentence $C$ into a disjoint set $\{M_1, \ldots, M_n\}$ of sets of RDF triples. Next, we generate a rephrasing of $C$ as follows:

$$P(T|C; M_C; M_1, \ldots, M_n; \theta) = \prod_i P(T_i|C; M_i; \theta) \quad (7)$$

where, the approximation from Eqn. 4 to Eqn. 5 derives from the assumption that the generation of $T$ is independent of $M_C$ given $(C; M_1, \ldots, M_n)$. We propose a pipeline model to learn parameters $\theta$. We first learn to split and then learn to generate from each RDF subset generated by the split.

Learning to split. For the first step, we learn a probabilistic model which given a set of RDF triples $M_C$ predicts a partition $M_1 \ldots M_n$ of this set. For a given $M_C$, it returns the partition $M_1 \ldots M_n$ with the highest probability $P(M_1, \ldots, M_n|M_C)$.

We learn this split module using items $(M_C, C), \{(M_i, T_i) | (M_n, T_n)\}$ in the WEBSPLIT dataset by simply computing the probability $P(M_1, \ldots, M_n|M_C)$. To make our model robust to an unseen $M_C$, we strip off named-entities and properties from each RDF triple and only keep the tree skeleton of $M_C$. There are only 60 distinct RDF tree skeletons, 1,183 possible split patterns and 19.72 split candidates in average for each tree skeleton, in the WEB-SPLIT dataset.

Learning to rephrase. We proposed two ways to estimate $P(T_i|C; M_i; \theta)$: (i) we learn a multi-


t-source encoder-decoder model which generates a text $T_i$ given a complex sentence $C$ and a set of RDF triples $M_i \in M_C$ ; and (ii) we approximate $P(T_i|C; M_i; \theta)$ by $P(T_i|M_i; \theta)$ and learn a simple sequence-to-sequence model which, given $M_i$, generates a text $T_i$. Note that as described earlier, $M_i$’s are linearised and tokenised before we input them to RNN encoders. We refer to the first model by SPLIT-MULTISEQ2SEQ and the second model by SPLIT-SEQ2SEQ.

6 Experimental Setup and Results

This section describes our experimental setup and results. We also describe the implementation de-
tails to facilitate the replication of our results.

6.1 Training, Validation and Test sets

To ensure that complex sentences in validation and test sets are not seen during training, we split the 5,546 distinct complex sentences in the WEB-SPLIT data into three subsets: Training set (4,438,
Table 2 shows, for each of the 5 models, a summary of the task and the size of the training corpus. For the models that directly learn to map a complex sentence into a meaning preserving sequence of at least two sentences (HYBRIDSIMPL, SEQ2SEQ and MULTISEQ2SEQ), the training set consists of 886,857 \( \langle C, T \rangle \) pairs with \( C \) a complex sentence and \( T \), the corresponding text. In contrast, for the pipeline models which first partition the input and then generate from RDF data (SPLIT-MULTISEQ2SEQ and SPLIT-SEQ2SEQ), the training corpus for learning to partition consists of 13,051 \( \langle M_C, \langle M_1, \ldots, M_n \rangle \rangle \) pairs while the training corpus for learning to generate contains 53,470 \( \langle M_i, T_i \rangle \) pairs.

### 6.2 Implementation Details

For all our neural models, we train RNNs with three-layered LSTM units, 500 hidden states and a regularisation dropout with probability 0.8. All LSTM parameters were randomly initialised over a uniform distribution within [-0.05, 0.05]. We trained our models with stochastic gradient descent with an initial learning rate 0.5. Every time perplexity on the held out validation set increased since it was previously checked, then we multiply the current learning rate by 0.5. We performed mini-batch training with a batch size of 64 sentences for SEQ2SEQ and MULTISEQ2SEQ, and 32 for SPLIT-SEQ2SEQ and SPLIT-MULTISEQ2SEQ. As the vocabulary size of the WEBSPLIT data is small, we train both encoder and decoder with full vocabulary. We randomly initialise word embeddings in the beginning and let the model train them during training. We train our models for 20 epochs and keep the best model on the held out set for the testing purposes. We used the system of Zoph and Knight (2016) to train both simple sequence-to-sequence and multi-source sequence-to-sequence models\(^6\), and the system of Narayan and Gardent (2014) to train our HYBRIDSIMPL model.\(^7\)

\(^6\)We used the code available at https://github.com/isi-nlp/Zoph_RNN.

\(^7\)We used the code available at https://github.com/shashiongithub/Sentence-Simplification-ACL14.

### 6.3 Results

We evaluate all models using multi-reference BLEU-4 scores (Papineni et al., 2002) based on all the rephrasings present in the Split-and-Rephrase corpus for each complex input sentence.\(^8\) As BLEU is a metric for n-grams precision estimation, it is not an optimal metric for the Split-and-Rephrase task (sentences even without any split could have a high BLEU score). We therefore also report on the average number of output simple sentences per complex sentence and the average number of output words per output simple sentence. The first one measures the ability of a system to split a complex sentence into multiple simple sentences and the second one measures the ability of producing smaller simple sentences.

Table 3 shows the results. The high BLEU score for complex sentences (SOURCE) from the WEBSPLIT corpus shows that using BLEU is not sufficient to evaluate splitting and rephrasing. Because the short sentences have many n-grams in common with the source, the BLEU score for complex sentences is high but the texts are made of a single sentence and the average sentence length is high. HYBRIDSIMPL performs poorly – we conjecture that this is linked to a decrease in semantic parsing quality (DRSs) resulting from complex named entities not being adequately recognised. The simple sequence-to-sequence model does not perform very well neither does the multi-source model trained on both complex sentences and their semantics. Typically, these two models often produce non-meaning preserving outputs (see example in Table 4) for input of longer length. In contrast, the two partition-and-generate models outperform all other models by a wide mar-
gin. This suggests that the ability to split is key to a good rephrasing: by first splitting the input semantics into smaller chunks, the two partition-and-generate models permit reducing a complex task (generating a sequence of sentences from a single complex sentence) to a series of simpler tasks (generating a short sentence from a semantic input).

Unlike in neural machine translation setting, multi-source models in our setting do not perform very well. SEQ2SEQ and SPLIT-SEQ2SEQ outperform MULTISEQ2SEQ and SPLIT-MULTISEQ2SEQ respectively, despite using less input information than their counterparts. The multi-source models used in machine translation have as a multi-source, two translations of the same content (Zoph and Knight, 2016). In our approach, the multi-source is a complex sentence and a set of RDF triples, e.g., \((C; M_C)\) for MULTISEQ2SEQ and \((C; M_t)\) for SPLIT-MULTISEQ2SEQ. We conjecture that the poor performance of multi-source models in our case is due either to the relatively small size of the training data or to a stronger mismatch between RDF and complex sentence than between two translations.

Table 4 shows an example output for all 5 systems highlighting the main differences. HYBRIDSIMPL’s output mostly uses the input words suggesting that the SMT system doing the rewriting has limited impact. Both the SEQ2SEQ and the MULTISEQ2SEQ models “hallucinate” new information (“served as a test pilot”, “born on Nov 18, 1983”). In contrast, the partition-and-generate models correctly render the meaning of the input sentence (SOURCE), performing interesting rephrasings (“X was born in Y” \(\rightarrow\) “X’s birth place was Y”) and split the input sentence into two.

<table>
<thead>
<tr>
<th>SOURCE</th>
<th>HYBRIDSIMPL</th>
<th>SEQ2SEQ</th>
<th>MULTISEQ2SEQ</th>
<th>SPLIT-MULTISEQ2SEQ</th>
<th>SPLIT-SEQ2SEQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alan Shepard was born in New Hampshire and he served as the Chief of the Astronaut Office.</td>
<td>Alan Shepard was born in New Hampshire and he served as of the the chief astronaut office.</td>
<td>Alan Shepard’s occupation was a test pilot. Alan Shepard was born in New Hampshire.</td>
<td>Alan Shepard served as a test pilot. Alan Shepard’s birth place was New Hampshire.</td>
<td>Alan Shepard served as the Chief of the Astronaut Office. Alan Shepard was born in New Hampshire.</td>
<td></td>
</tr>
</tbody>
</table>

Table 4: Example outputs from different models.

We have constructed a new corpus for this task which is built from readily-available data used for NLG (Natural Language Generation) evaluation. Initial experiments indicate that the ability to split is a key factor in generating fluent and meaningful preserving rephrasings because it permits reducing a complex generation task (generating a text consisting of at least two sentences) to a series of simpler tasks (generating short sentences). In future work, it would be interesting to see whether and if so how, sentence splitting can be learned in the absence of explicit semantic information in the input.

Another direction for future work concerns the exploitation of the extended WebNLG corpus. While the results presented in this paper use a version of the WebNLG corpus consisting of 13,308 MR-Text pairs, 7049 distinct MRs and 8 DBpedia categories, the current WebNLG corpus encompasses 43,056 MR-Text pairs, 16,138 distinct MRs and 15 DBpedia categories. We plan to exploit this extended corpus to make available a correspondingly extended WESPLIT corpus, to learn optimised Split-and-Rephrase models and to explore sentence fusion (converting a sequence of sentences into a single complex sentence).

Acknowledgements

We thank Bonnie Webber and Annie Louis for early discussions on the ideas presented in the paper. We thank Rico Sennrich for directing us to multi-source NMT models. This work greatly benefited from discussions with the members of the Edinburgh NLP group. We also thank the three anonymous reviewers for their comments to improve the paper. The research presented in this paper was partially supported by the H2020 project SUMMA (under grant agreement 688139) and the French National Research Agency within the framework of the WebNLG Project (ANR-14-CE24-0033).
References


Hal Daume III and Daniel Marcu. 2004. Generic sentence fusion is an ill-defined summarization task. Technical report, DTIC.


Tomáš Jelinek. 2014. Improvements to dependency parsing using automatic simplification of data. In Proceedings of LREC.


Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In Proceedings of EMNLP.


