Sentence Simplification with Deep Reinforcement Learning

Xingxing Zhang and **Mirella Lapata**
Institute for Language, Cognition and Computation
School of Informatics, University of Edinburgh
10 Crichton Street, Edinburgh EH8 9AB
x.zhang@ed.ac.uk,mlap@inf.ed.ac.uk

Abstract

Sentence simplification aims to make sentences easier to read and understand. Most recent approaches draw on insights from machine translation to learn simplification rewrites from monolingual corpora of complex and simple sentences. We address the simplification problem with an encoder-decoder model coupled with a deep reinforcement learning framework. Our model, which we call DRESS (as shorthand for **D**eep **RE**inforcement **S**entence **S**implification), explores the space of possible simplifications while learning to optimize a reward function that encourages outputs which are simple, fluent, and preserve the meaning of the input. Experiments on three datasets demonstrate that our model outperforms competitive simplification systems.[1]

1 Introduction

The main goal of *sentence simplification* is to reduce the linguistic complexity of text, while still retaining its original information and meaning. The simplification task has been the subject of several modeling efforts in recent years due to its relevance for NLP applications and individuals alike (Siddharthan, 2014; Shardlow, 2014). For instance, a simplification component could be used as a preprocessing step to improve the performance of parsers (Chandrasekar et al., 1996), summarizers (Beigman Klebanov et al., 2004), and semantic role labelers (Vickrey and Koller, 2008; Woodsend and Lapata, 2014). Automatic simplification would also benefit people with low-literacy skills (Watanabe et al., 2009), such as children and non-native speakers as well as individuals with autism (Evans et al., 2014), aphasia (Carroll et al., 1999), or dyslexia (Rello et al., 2013).

The most prevalent rewrite operations which give rise to simplified text include substituting rare words with more common words or phrases, rendering syntactically complex structures simpler, and deleting elements of the original text (Siddharthan, 2014). Earlier work focused on individual aspects of the simplification problem. For example, several systems performed syntactic simplification only, using rules aimed at sentence splitting (Carroll et al., 1999; Chandrasekar et al., 1996; Vickrey and Koller, 2008; Siddharthan, 2004) while others turned to lexical simplification by substituting difficult words with more common WordNet synonyms or paraphrases (Devlin, 1999; Inui et al., 2003; Kaji et al., 2002).

Recent approaches view the simplification process more holistically as a monolingual text-to-text generation task borrowing ideas from statistical machine translation. Simplification rewrites are learned automatically from examples of complex-simple sentences extracted from online resources such as the ordinary and simple English Wikipedia. For example, Zhu et al. (2010) draw inspiration from syntax-based translation and propose a model similar to Yamada and Knight (2001) which additionally performs simplification-specific rewrite operations (e.g., sentence splitting). Woodsend and Lapata (2011) formulate simplification in the framework of Quasi-synchronous grammar (Smith and Eisner, 2006) and use integer linear programming to score the candidate translations/simplifications. Wubben et al. (2012) propose a two-stage model: initially, a standard phrase-based machine translation (PBMT) model is trained on complex-simple sentence pairs. During inference, the K-best outputs of the PBMT model are reranked according

[1] Our code and data are publicly available at https://github.com/XingxingZhang/dress.

Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 584–594
Copenhagen, Denmark, September 7–11, 2017.

to their dis-similarity to the (complex) input sentence. The hybrid model developed in Narayan and Gardent (2014) also operates in two phases. Initially, a probabilistic model performs sentence splitting and deletion operations over discourse representation structures assigned by Boxer (Curran et al., 2007). The resulting sentences are further simplified by a model similar to Wubben et al. (2012). Xu et al. (2016) train a syntax-based machine translation model on a large scale paraphrase dataset (Ganitkevitch et al., 2013) using simplification-specific objective functions and features to encourage simpler output.

In this paper we propose a simplification model which draws on insights from neural machine translation (Bahdanau et al., 2015; Sutskever et al., 2014). Central to this approach is an encoder-decoder architecture implemented by recurrent neural networks. The encoder reads the source sequence into a list of continuous-space representations from which the decoder generates the target sequence. Although our model uses the encoder-decoder architecture as its backbone, it must also meet constraints imposed by the simplification task itself, i.e., the predicted output must be simpler, preserve the meaning of the input, and grammatical. To incorporate this knowledge, the model is trained in a reinforcement learning framework (Williams, 1992): it explores the space of possible simplifications while learning to maximize an expected reward function that encourages outputs which meet simplification-specific constraints. Reinforcement learning has been previously applied to extractive summarization (Ryang and Abekawa, 2012), information extraction (Narasimhan et al., 2016), dialogue generation (Li et al., 2016), machine translation, and image caption generation (Ranzato et al., 2016).

We evaluate our system on three publicly available datasets collated automatically from Wikipedia (Zhu et al., 2010; Woodsend and Lapata, 2011) and human-authored news articles (Xu et al., 2015b). We experimentally show that the reinforcement learning framework is the key to successful generation of simplified text bringing significant improvements over strong simplification models across datasets.

2 Neural Encoder-Decoder Model

We will first define a basic encoder-decoder model for sentence simplification and then explain how to embed it in a reinforcement learning framework. Given a (complex) *source* sentence $X = (x_1, x_2, \ldots, x_{|X|})$, our model learns to predict its simplified *target* $Y = (y_1, y_2, \ldots, y_{|Y|})$. Inferring the target Y given the source X is a typical sequence to sequence learning problem, which can be modeled with attention-based encoder-decoder models (Bahdanau et al., 2015; Luong et al., 2015). Sentence simplification is slightly different from related sequence transduction tasks (e.g., compression) in that it can involve splitting operations. For example, a long source sentence (*In 1883, Faur married Marie Fremiet, with whom he had two sons.*) can be simplified as two sentences (*In 1883, Faur married Marie Fremiet. They had two sons.*). Nevertheless, we still view the target as a sequence, i.e., two or more sequences concatenated with full stops.

The encoder-decoder model has two parts (see left hand side in Figure 1). The *encoder* transforms the source sentence X into a sequence of hidden states $(\mathbf{h}_1^S, \mathbf{h}_2^S, \ldots, \mathbf{h}_{|X|}^S)$ with a Long Short-Term Memory Network (LSTM; Hochreiter and Schmidhuber 1997), while the *decoder* uses another LSTM to generate one word y_{t+1} at a time in the simplified target Y. Generation is conditioned on all previously generated words $y_{1:t}$ and a dynamically created context vector $\mathbf{c}_t$, which encodes the source sentence:

$$P(Y|X) = \prod_{t=1}^{|Y|} P(y_t|y_{1:t-1}, X) \quad (1)$$

$$P(y_{t+1}|y_{1:t}, X) = \text{softmax}(g(\mathbf{h}_t^T, \mathbf{c}_t)) \quad (2)$$

where $g(\cdot)$ is a one-hidden-layer neural network with the following parametrization:

$$g(\mathbf{h}_t^T, \mathbf{c}_t) = \mathbf{W}_o \tanh(\mathbf{U}_h \mathbf{h}_t^T + \mathbf{W}_h \mathbf{c}_t) \quad (3)$$

where $\mathbf{W}_o \in \mathbb{R}^{|V| \times d}$, $\mathbf{U}_h \in \mathbb{R}^{d \times d}$, and $\mathbf{W}_h \in \mathbb{R}^{d \times d}$; $|V|$ is the output vocabulary size and d the hidden unit size. $\mathbf{h}_t^T$ is the hidden state of the decoder LSTM which summarizes $y_{1:t}$, i.e., what has been generated so far:

$$\mathbf{h}_t^T = \text{LSTM}(y_t, \mathbf{h}_{t-1}^T) \quad (4)$$

The dynamic context vector $\mathbf{c}_t$ is the weighted sum of the hidden states of the source sentence:

$$\mathbf{c}_t = \sum_{i=1}^{|X|} \alpha_{ti} \mathbf{h}_i^S \quad (5)$$

whose weights α_{ti} are determined by an *attention* mechanism:

$$\alpha_{ti} = \frac{\exp(\mathbf{h}_t^T \cdot \mathbf{h}_i^S)}{\sum_i \exp(\mathbf{h}_t^T \cdot \mathbf{h}_i^S)} \quad (6)$$

where $\cdot$ is the dot product between two vectors. We use the dot product here mainly for efficiency reasons; alternative ways to compute attention scores have been proposed in the literature and we refer the interested reader to Luong et al. (2015). The model sketched above is usually trained by minimizing the negative log-likelihood of the training source-target pairs.

3 Reinforcement Learning for Sentence Simplification

In this section we present DRESS, our **D**eep **RE**inforcement **S**entence **S**implification model. Despite successful application in numerous sequence transduction tasks (Jean et al., 2015; Chopra et al., 2016; Xu et al., 2015a), a vanilla encoder-decoder model is not ideal for sentence simplification. Although a number of rewrite operations (e.g., copying, deletion, substitution, word reordering) can be used to simplify text, copying is by far the most common. We empirically found that 73% of the target words are copied from the source in the Newsela dataset. This number further increases to 83% when considering Wikipedia-based datasets (we provide details on these datasets in Section 5). As a result, a generic encoder-decoder model learns to copy all too well at the expense of other rewrite operations, often parroting back the source or making only a few trivial changes.

To encourage a wider variety of rewrite operations while remaining fluent and faithful to the meaning of the source, we employ a reinforcement learning framework (see Figure 1). We view the encoder-decoder model as an agent which first reads the source sentence X; then at each step, it takes an action $\hat{y}_t \in V$ (where V is the output vocabulary) according to a policy $P_{RL}(\hat{y}_t|\hat{y}_{1:t-1}, X)$ (see Equation (2)). The agent continues to take actions until it produces an **E**nd **O**f **S**entence (EOS) token yielding the action sequence $\hat{Y} = (\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_{|\hat{Y}|})$, which is also the simplified output of our model. A reward r is then received and the REINFORCE algorithm (Williams, 1992) is used to update the agent. In the following, we first introduce our reward and then present the details of the REINFORCE algorithm.

3.1 Reward

The reward $r(\hat{Y})$ for system output $\hat{Y}$ is the weighted sum of the three components aimed at capturing key aspects of the target output, namely simplicity, relevance, and fluency:

$$r(\hat{Y}) = \lambda^S r^S + \lambda^R r^R + \lambda^F r^F \quad (7)$$

where $\lambda^S, \lambda^R, \lambda^F \in [0, 1]$; $r(\hat{Y})$ is a shorthand for $r(X, Y, \hat{Y})$ where X is the source, Y the reference (or target), and $\hat{Y}$ the system output. r^S, r^R, and r^F are shorthands for simplicity $r^S(X, Y, \hat{Y})$, relevance $r^R(X, \hat{Y})$, and fluency $r^F(\hat{Y})$. We provide details for each reward summand below.

Simplicity To encourage the model to apply a wide range of simplification operations, we use SARI (Xu et al., 2016), a recently proposed metric which compares **S**ystem output **A**gainst **R**eferences and against the **I**nput sentence. SARI is the arithmetic average of n-gram precision and recall of three rewrite operations: addition, copying, and deletion. It rewards addition operations where system output was not in the input but occurred in the references. Analogously, it rewards words retained/deleted in both the system output and the references. In experimental evaluation Xu et al. (2016) demonstrate that SARI correlates well with human judgments of simplicity, whilst correctly rewarding systems that both make changes and simplify the input.

One caveat with using SARI as a reward is the fact that it relies on the availability of multiple references which are rare for sentence simplification. Xu et al. (2016) provide eight references for 2,350 sentences, but these are primarily for system tuning and evaluation rather than training. The majority of existing simplification datasets (see Section 5 for details) have a single reference for each source sentence. Moreover, they are unavoidably noisy as they are mostly constructed automatically, e.g., by aligning sentences from the ordinary and simple English Wikipedias. When relying solely on a single reference, SARI will try to reward accidental n-grams that should never have occurred in it. To countenance the effect of noise, we apply $\text{SARI}(X, \hat{Y}, Y)$ in the expected direction, with X as the source, $\hat{Y}$ the system output, and Y the reference as well as in the reverse direction with Y as the system output and $\hat{Y}$ as the reference. Assuming our system can produce reasonably good simplifications, by swapping the output

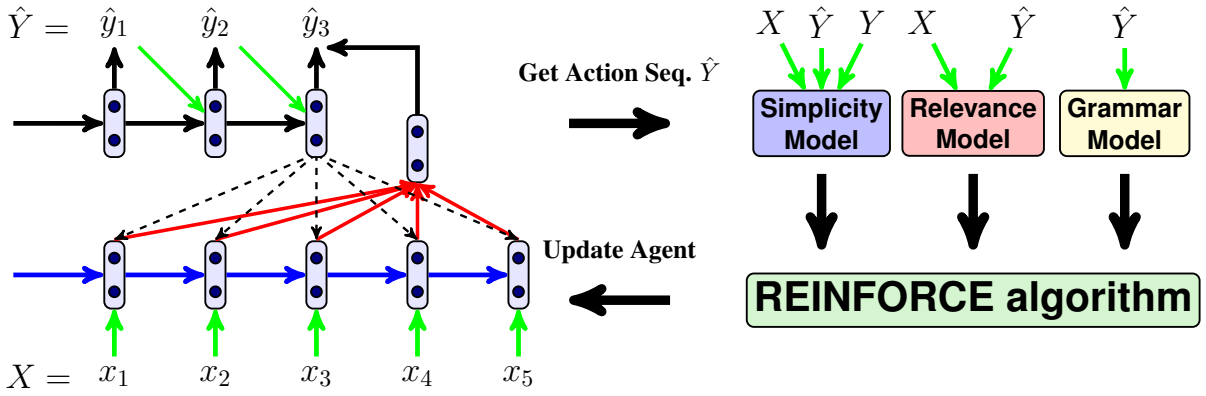

Figure 1: Deep reinforcement learning simplification model. X is the complex sentence, Y the reference (simple) sentence and $\hat{Y}$ the action sequence (simplification) produced by the encoder-decoder model.

and the reference, reverse SARI can be used to estimate how good a reference is with respect to the system output. Our first reward is therefore the weighted sum of SARI and reverse SARI:

$$r^S = \beta \, \text{SARI}(X, \hat{Y}, Y) + (1-\beta) \, \text{SARI}(X, Y, \hat{Y}) \quad (8)$$

Relevance While the simplicity-based reward r^S tries to encourage the model to make changes, the relevance reward r^R ensures that the generated sentences preserve the meaning of the source. We use an LSTM sentence encoder to convert the source X and the predicted target $\hat{Y}$ into two vectors $\mathbf{q}_X$ and $\mathbf{q}_{\hat{Y}}$. The relevance reward r^R is simply the cosine similarity between these two vectors:

$$r^R = \cos(\mathbf{q}_X, \mathbf{q}_{\hat{Y}}) = \frac{\mathbf{q}_X \cdot \mathbf{q}_{\hat{Y}}}{||\mathbf{q}_X|| \, ||\mathbf{q}_{\hat{Y}}||} \quad (9)$$

We use a sequence auto-encoder (SAE; Dai and Le 2015) to train the LSTM sentence encoder on both the complex and simple sentences. Specifically, the SAE uses sentence $X = (x_1, \ldots, x_{|X|})$ to infer itself via an encoder-decoder model (without an attention mechanism). Firstly, an encoder LSTM converts X into a sequence of hidden states $(\mathbf{h}_1, \ldots, \mathbf{h}_{|X|})$. Then, we use $\mathbf{h}_{|X|}$ to initialize the hidden state of the decoder LSTM and recover/generate X one word at a time.

Fluency Xu et al. (2016) observe that SARI correlates less with fluency compared to other metrics such as BLEU (Papineni et al., 2002). The fluency reward r^F models the well-formedness of the generated sentences explicitly. It is the normalized sentence probability assigned by an LSTM language model trained on simple sentences:

$$r^F = \exp\left(\frac{1}{|\hat{Y}|}\sum_{i=1}^{|\hat{Y}|} \log P_{LM}(\hat{y}_i|\hat{y}_{0:i-1})\right) \quad (10)$$

We take the exponential of $\hat{Y}$'s perplexity to ensure that $r^F \in [0, 1]$ as is the case with r^S and r^R.

3.2 The REINFORCE Algorithm

The goal of the REINFORCE algorithm is to find an agent that maximizes the expected reward. The training loss for one sequence is its negative expected reward:

$$\mathcal{L}(\theta) = -\mathbb{E}_{(\hat{y}_1,\ldots,\hat{y}_{|\hat{Y}|})\sim P_{RL}(\cdot|X)}[r(\hat{y}_1, \ldots, \hat{y}_{|\hat{Y}|})]$$

where P_{RL} is our policy, i.e., the distribution produced by the encoder-decoder model (see Equation(2)) and $r(\cdot)$ is the reward function of an action sequence $\hat{Y} = (\hat{y}_1, \ldots, \hat{y}_{|\hat{Y}|})$, i.e., a generated simplification. Unfortunately, computing the expectation term is prohibitive, since there is an infinite number of possible action sequences. In practice, we approximate this expectation with a single sample from the distribution of $P_{LR}(\cdot|X)$. We refer to Williams (1992) for the full derivation of the gradients. The gradient of $\mathcal{L}(\theta)$ is:

$$\nabla\mathcal{L}(\theta) \approx$$
$$\sum_{t=1}^{|\hat{Y}|} \nabla \log P_{RL}(\hat{y}_t|\hat{y}_{1:t-1}, X)[r(\hat{y}_{1:|\hat{Y}|}) - b_t]$$

To reduce the variance of gradients, we also introduce a baseline linear regression model b_t to estimate the expected future reward at time t (Ranzato et al., 2016). b_t takes the concatenation of $\mathbf{h}_t^T$ and $\mathbf{c}_t$ as input and outputs a real value as the expected reward. The parameters of the regressor are

trained by minimizing mean squared error. We do not back-propagate this error to $\mathbf{h}_t^T$ or $\mathbf{c}_t$ during training (Ranzato et al., 2016).

3.3 Learning

Presented in its original form, the REINFORCE algorithm starts learning with a random policy. This assumption can make model training challenging for generation tasks like ours with large vocabularies (i.e., action spaces). We address this issue by pre-training our agent (i.e., the encoder-decoder model) with a negative log-likelihood objective (see Section 2), making sure it can produce reasonable simplifications, thereby starting off with a policy which is better than random. We follow prior work (Ranzato et al., 2016) in adopting a curriculum learning strategy. In the beginning of training, we give little freedom to our agent allowing it to predict the last few words for each target sentence. For every target sequence, we use negative log-likelihood to train the first L (initially, $L = 24$) tokens and apply the reinforcement learning algorithm to the $(L + 1)$th tokens onwards. Every two epochs, we set $L = L - 3$ and the training terminates when L is 0.

4 Lexical Simplification

Lexical substitution, the replacement of complex words with simpler alternatives, is an integral part of sentence simplification (Specia et al., 2012). The model presented so far learns lexical substitution and other rewrite operations *jointly*. In some cases, words are predicted because they seem natural in the their context, but are poor substitutes for the content of the complex sentence. To countenance this, we learn lexical simplifications *explicitly* and integrate them with our reinforcement learning-based model.

We use an *pre-trained* encoder-decoder model (which is trained on a parallel corpus of complex and simple sentences) to obtain probabilistic word alignments, aka attention scores (see α_t in Equation (6)). Let $X = (x_1, x_2, \ldots, x_{|X|})$ denote a source sentence and $Y = (y_1, y_2, \ldots, y_{|Y|})$ a target sentence. We convert X into $|X|$ hidden states $(\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{|X|})$ with an LSTM. Note that $\mathbf{v}_t \in \mathbb{R}^{d\times 1}$ corresponds to the context dependent representation of x_t. Let α_t denote the alignment scores $\alpha_{t1}, \alpha_{t2}, \ldots, \alpha_{t|X|}$. The lexical simplification probability of y_t given the source sentence and the alignment scores is:

$$P_{LS}(y_t|X, \alpha_t) = \text{softmax}(\mathbf{W}_l\, \mathbf{s}_t) \tag{11}$$

where $\mathbf{W}_l \in \mathbb{R}^{|V|\times d}$ and $\mathbf{s}_t$ represents the source:

$$\mathbf{s}_t = \sum_{i=1}^{|X|} \alpha_{ti}\mathbf{v}_i \tag{12}$$

The lexical simplification model on its own encourages lexical substitutions, without taking into account what has been generated so far (i.e., $y_{1:t-1}$) and as a result fluency could be compromised. A straightforward solution is to integrate lexical simplification with our reinforcement learning trained model (Section 3) using linear interpolation, where $\eta \in [0, 1]$:

$$\begin{aligned} P(y_t|y_{1:t-1}, X) = (1 - \eta)\, P_{RL}(y_t|y_{1:t-1}, X) \\ + \eta\, P_{LS}(y_t|X, \alpha_t) \end{aligned} \tag{13}$$

5 Experimental Setup

In this section we present our experimental setup for assessing the performance of the simplification model described above. We give details on our datasets, model training, evaluation protocol, and the systems used for comparison.

Datasets We conducted experiments on three simplification datasets. *WikiSmall* (Zhu et al., 2010) is a parallel corpus which has been extensively used as a benchmark for evaluating text simplification systems (Wubben et al., 2012; Woodsend and Lapata, 2011; Narayan and Gardent, 2014; Zhu et al., 2010). It contains automatically aligned complex and simple sentences from the ordinary and simple English Wikipedias. The test set consists of 100 complex-simple sentence pairs. The training set contains 89,042 sentence pairs (after removing duplicates and test sentences). We randomly sampled 205 pairs for development and used the remaining sentences for training.

We also constructed *WikiLarge*, a larger Wikipedia corpus by combining previously created simplification corpora. Specifically, we aggregated the aligned sentence pairs in Kauchak (2013), the aligned and revision sentence pairs in Woodsend and Lapata (2011), and Zhu's (2010) WikiSmall dataset described above. We used the development and test sets created in Xu et al. (2016). These are complex sentences taken from WikiSmall paired with simplifications provided by Amazon Mechanical Turk workers. The dataset

contains 8 (reference) simplifications for 2,359 sentences partitioned into 2,000 for development and 359 for testing. After removing duplicates and sentences in development and test sets, the resulting training set contains 296,402 sentence pairs.

Our third dataset is *Newsela*, a corpus collated by Xu et al. (2015b) who argue that Wikipedia-based resources are suboptimal due to the automatic sentence alignment which unavoidably introduces errors, and their uniform writing style which leads to systems that generalize poorly. Newsela[2] consists of 1,130 news articles, each rewritten four times by professional editors for children at different grade levels (0 is the most complex level and 4 is simplest). Xu et al. (2015b) provide multiple aligned complex-simple pairs within each article. We removed sentence pairs corresponding to levels 0–1, 1–2, and 2–3, since they were too similar to each other. The first 1,070 documents were used for training (94,208 sentence pairs), the next 30 documents for development (1,129 sentence pairs) and the last 30 documents for testing (1,076 sentence pairs).[3] We are not aware of any published results on this dataset.

Training Details We trained our models on an Nvidia GPU card. We used the same hyperparameters across datasets. We first trained an encoder-decoder model, and then performed reinforcement learning training (Section 3), and trained the lexical simplification model (Section 4). Encoder-decoder parameters were uniformly initialized to $[-0.1, 0.1]$. We used Adam (Kingma and Ba, 2014) to optimize the model with learning rate 0.001; the first momentum coefficient was set to 0.9 and the second momentum coefficient to 0.999. The gradient was rescaled when the norm exceeded 5 (Pascanu et al., 2013). Both encoder and decoder LSTMs have two layers with 256 hidden neurons in each layer. We regularized all LSTMs with a dropout rate of 0.2 (Zaremba et al., 2014). We initialized the encoder and decoder word embedding matrices with 300 dimensional Glove vectors (Pennington et al., 2014).

During reinforcement training, we used plain stochastic gradient descent with a learning rate of 0.01. We set $\beta = 0.1$, $\lambda_S = 1$, $\lambda_R = 0.25$ and $\lambda_F = 0.5$.[4] Training details for the lexical simplification model are identical to the encoder-decoder model except that word embedding matrices were randomly initialized. The weight of the lexical simplification model was set to $\eta = 0.1$.

To reduce vocabulary size, named entities were tagged with the Stanford CoreNLP (Manning et al., 2014) and anonymized with a NE@N token, where NE $\in$ {PER, LOC, ORG, MISC} and N indicates NE@N is the N-th distinct NE typed entity. For example, "John and Bob are ..." becomes "PER@1 and PER@2 are ...". At test time, we de-anonymize NE@N tokens in the output by looking them up in their source sentences. Note that the de-anonymization may fail, but the chance is small (around 2% of the time on the Newsela development set). We replaced words occurring three times or less in the training set with UNK. At test time, when our models predict UNK, we adopt the UNK replacement method proposed in Jean et al. (2015).

Evaluation Following previous work (Woodsend and Lapata, 2011; Xu et al., 2016) we evaluated system output automatically adopting metrics widely used in the simplification literature. Specifically, we used BLEU[5] (Papineni et al., 2002) to assess the degree to which generated simplifications differed from gold standard references and the Flesch-Kincaid Grade Level index (FKGL; Kincaid et al. 1975) to measure the readability of the output (lower FKGL[6] implies simpler output). In addition, we used SARI (Xu et al., 2016), which evaluates the quality of the output by comparing it against the source and reference simplifications.[7] BLEU, FKGL, and SARI are all measured at corpus-level. We also evaluated system output by eliciting human judgments via Amazon's Mechanical Turk. Specifically (self-reported) native English speakers were asked to rate simplifications on three dimensions: *Fluency* (is the output grammatical and well formed?), *Adequacy* (to what extent is the meaning expressed in the original sentence preserved in the output?) and *Simplicity* (is the output simpler than the original sentence?). All ratings were obtained using a five point Likert scale.

Comparison Systems We compared our model against several systems previously proposed in the literature. These include PBMT-R, a mono-

[2] https://newsela.com

[3] If a sentence has multiple references in the development or test set, we use the reference with highest simplicity level.

[4] Weights were tuned on the development set of the Newsela dataset and kept fixed for the other two datasets.

[5] With the default mtevalv13a.pl settings.

[6] FKGL implementation at http://goo.gl/OHP7k3.

[7] We used he implementation of SARI in Xu et al. (2016).

Newsela	BLEU	FKGL	SARI
PBMT-R	18.19	7.59	15.77
Hybrid	14.46	**4.01**	**30.00**
EncDecA	21.70	5.11	24.12
DRESS	23.21	4.13	27.37
DRESS-LS	**24.30**	4.21	26.63

WikiSmall	BLEU	FKGL	SARI
PBMT-R	46.31	11.42	15.97
Hybrid	**53.94**	9.20	**30.46**
EncDecA	47.93	11.35	13.61
DRESS	34.53	**7.48**	27.48
DRESS-LS	36.32	7.55	27.24

WikiLarge	BLEU	FKGL	SARI
PBMT-R	81.11	8.33	38.56
Hybrid	48.97	**4.56**	31.40
SBMT-SARI	73.08	7.29	**39.96**
EncDecA	**88.85**	8.41	35.66
DRESS	77.18	6.58	37.08
DRESS-LS	80.12	6.62	37.27

Table 1: Automatic evaluation on Newsela, WikiSmall, and WikiLarge test sets.

lingual phrase-based machine translation system with a reranking post-processing step[8] (Wubben et al., 2012) and Hybrid, a model which first performs sentence splitting and deletion operations over discourse representation structures and then further simplifies sentences with PBMT-R (Narayan and Gardent, 2014). Hybrid[9] is state of the art on the WikiSmall dataset. Comparisons with SBMT-SARI, a syntax-based translation model trained on PPDB (Ganitkevitch et al., 2013) and tuned with SARI (Xu et al., 2016), are problematic due to the size of PPDB which is considerably larger than any of the datasets used in this work (it contains 106 million sentence pairs with 2 billion words). Nevertheless, we compare[10] against SBMT-SARI, but only models trained on Wikilarge, our largest dataset.

6 Results

Since Newsela contains high quality simplifications created by professional editors, we performed the bulk of our experiments on this dataset. Specifically, we set out to answer two questions: (a) which neural model performs best and (b) how do neural models which are resource lean and do not have access to linguistic annotations fare against more traditional systems. We therefore compared the basic attention-based encoder-decoder model (EncDecA), with the deep reinforcement learning model (DRESS; Section 3), and a linear combination of DRESS and the lexical simplification model (DRESS-LS; Section 4). Neural models were further compared against two strong baselines, PBMT-R and Hybrid. Table 3 shows example output of all models on the Newsela dataset.

[8]We made a good-faith effort to re-implement their system following closely the details in Wubben et al. (2012).

[9]We are grateful to Shashi Narayan for running his system on our three datasets.

[10]The output of SBMT-SARI is publicly available.

Newsela	Fluency	Adequacy	Simplicity	All
PBMT-R	3.56	**3.58****	2.09**	3.08**
Hybrid	2.70**	2.51**	2.99	2.73**
EncDecA	3.63	2.99	2.56**	3.06**
DRESS	3.65	2.94	**3.10**	3.23
DRESS-LS	**3.71**	3.07	3.04	**3.28**
Reference	3.90	2.81**	3.42**	3.38

WikiSmall	Fluency	Adequacy	Simplicity	All
PBMT-R	3.91	**3.74****	2.80**	3.48*
Hybrid	3.26**	3.42	2.82**	3.17**
DRESS-LS	**3.92**	3.36	**3.55**	**3.61**
Reference	3.74*	3.34	3.13**	3.41**

WikiLarge	Fluency	Adequacy	Simplicity	All
PBMT-R	3.68	**3.63***	2.70**	3.34*
Hybrid	2.60**	2.42**	**3.52**	2.85**
SBMT-SARI	3.34**	3.51*	2.77**	3.21**
DRESS-LS	**3.70**	3.28	3.42	**3.46**
Reference	3.79	3.72**	2.86**	3.46

Table 2: Mean ratings elicited by humans on Newsela, WikiSmall, and WkiLarge test sets. Ratings significantly different from DRESS-LS are marked with * ($p < 0.05$) and ** ($p < 0.01$). Significance tests were performed using a student t-test.

The top block in Table 1 summarizes the results of our automatic evaluation. As can be seen, all neural models obtain higher BLEU, lower FKGL and higher SARI compared to PBMT-R. Hybrid has the lowest FKGL and highest SARI. Compared to EncDecA, DRESS scores lower on FKGL and higher on SARI, which indicates that the model has indeed learned to optimize the reward function which includes SARI. Integrating lexical simplification (DRESS-LS) yields better BLEU, but slightly worse FKGL and SARI.

The results of our human evaluation are presented in the top block of Table 2. We elicited judgments for 100 randomly sampled test sentences. Aside from comparing system output (PBMT-R, Hybrid, EncDecA, DRESS, and DRESS-LS), we also elicited ratings for the gold standard Reference as an upper bound. We report results for Fluency, Adequacy, and Simplicity individually and in combination (All is the average rating of the three dimensions). As can be seen, DRESS and DRESS-LS outperform PBMT-R and

Complex	There's just one major hitch: the primary purpose of education is to develop citizens with a wide variety of skills.
Reference	The purpose of education is to develop a wide range of skills.
PBMT-R	It's just one major hitch: the purpose of education is to **make people** with a wide variety of skills.
Hybrid	one hitch the purpose is to develop citizens.
EncDecA	The **key** of education is to develop **people** with a wide variety of skills.
DRESS	There's just one major hitch: the **main goal** of education is to develop **people** with **lots of** skills.
DRESS-LS	There's just one major hitch: the **main goal** of education is to develop citizens with **lots of** skills.
Complex	"They were so burdened by the past they couldn't think about the future," said Barnet, 62, who was president of Columbia Records, the No.1 record label in the United States, before joining Capitol.
Reference	Capitol was stuck in the past. It could not think about the future, Barnett said.
PBMT-R	"They were so **affected** by the past they couldn't think about the future," said Barnett, 62, was president of Columbia Records, before joining Capitol **building**.
Hybrid	'They were so burdened by the past they couldn't think about the future," said Barnett, 62, who was Columbia Records, president of the No.1 record label in the united states, before joining Capitol.
EncDecA	"They were so burdened by the past they couldn't think about the future," said Barnett, who was president of Columbia Records, the No.1 record labels in the United States.
DRESS	"They were so **sicker** by the past they couldn't think about the future," said Barnett, who was president of Columbia Records.
DRESS-LS	"They were so burdened by the past they couldn't think about the future," said Barnett, who was president of Columbia Records.

Table 3: System output for two sentences (Newsela development set). Substitutions are shown in bold.

Hybrid on Fluency, Simplicity, and overall. The fact that neural models (EncDecA, DRESS and DRESS-LS) fare well on Fluency, is perhaps not surprising given the recent success of LSTMs in language modeling and neural machine translation (Zaremba et al., 2014; Jean et al., 2015).

Neural models obtain worse ratings on Adequacy but are closest to the human references on this dimension. DRESS-LS (and DRESS) are significantly better ($p < 0.01$) on Simplicity than EncDecA, PBMT-R, and Hybrid which indicates that our reinforcement learning based model is effective at creating simpler output. Combined ratings (All) for DRESS-LS are significantly different compared to the other models but not to DRESS and the Reference. Nevertheless, integration of the lexical simplification model boosts performance as ratings increase almost across the board (Simplicity is slightly worse). Returning to our original questions, we find that neural models are more fluent than comparison systems, while performing non-trivial rewrite operations (see the SARI scores in Table 1) which yield simpler output (see the Simplicity column in Table 2). Based on our judgment elicitation study, neural models trained with reinforcement learning perform best, with DRESS-LS having a slight advantage.

We further analyzed model performance by computing various statistics on the simplified output. We measured average sentence length and the degree to which DRESS and comparison systems perform rewriting operations. We approximated the latter with Translation Error Rate (TER; Snover et al. 2006), a measure commonly used to automatically evaluate the quality of machine translation output. We used TER to compute the (average) number of edits required to change an original complex sentence to simpler output. We also report the number of edits by type, i.e., the number of insertions, substitutions, deletions, and shifts needed (on average) to convert complex to simple sentences.

As shown in Table 4, Hybrid obtains the highest TER, followed by our models (DRESS and

Models	Len	TER	Ins	Del	Sub	Shft
PBMT-R	23.1	0.13	0.68	0.68	1.50	0.09
Hybrid	12.4	0.90	0.01	10.19	0.12	0.41
EncDecA	17.0	0.36	0.13	5.96	1.69	0.09
DRESS	14.2	0.46	0.07	8.53	1.37	0.11
DRESS-LS	14.4	0.44	0.07	8.38	1.11	0.09
Reference	12.7	0.67	0.40	10.26	3.44	0.73

Table 4: Output length (average number of tokens), TER scores and number of edits by type (Ins*ertions*, Del*etions*, Sub*stitutions*, Sh*ifts*) on the Newsela test set. Higher TER means that more rewriting operations are performed.

DRESS-LS), which indicates that they actively perform rewriting. Perhaps Hybrid is too aggressive when simplifying a sentence, it obtains low Fluency and Adequacy scores in human evaluation (Table 2). There is a strong correlation between sentence length and number of deletion operations (i.e., more deleteions lead to shorter sentences) and PBMT-R performs very few deletions. Overall, reinforcement learning encourages deletion (see DRESS and DRESS-LS), while performing a reasonable amount of additional operations (e.g., substitutions and shifts) compared to EncDecA and PBMT-R.

The middle blocks in Tables 1 and 2 report results on the WikiSmall dataset. FKGL and SARI follow a similar pattern as on Newsela. BLEU scores for PBMT-R, Hybrid, and EncDecA are much higher compared to DRESS and DRESS-LS. Hybrid obtains best BLEU and SARI scores, while DRESS and DRESS-LS do very well on FKGL. In human evaluation, we elicited judgments on the entire WikiSmall test set (100 sentences). We compared DRESS-LS, with PBMT-R, Hybrid, and gold standard Reference simplifications. As human experiments are time consuming and expensive, we did not include other neural models besides DRESS-LS based on our Newsela study which showed that EncDecA is inferior to variants trained with reinforcement learning and that DRESS-LS is the better performing model (however, we do compare *all* models in Table 1). DRESS-LS is significantly better on Simplicity than PBMT-R, Hybrid, and the Reference. It performs on par with PBMT-R on Fluency and worse on Adequacy (but still closer to the human Reference than PBMT-R or Hybrid). When combining all ratings (All in Table 2), DRESS-LS is significantly better than PBMT-R, Hybrid, and the Reference.

The bottom blocks in Tables 1 and 2 report results on Wikilarge. We compared our models with PBMT-R, Hybrid, and SBMT-SARI (Xu et al., 2016). The FKGL follows a similar pattern as in the previous datasets. PBMT-R and our models are best in terms of BLEU while SBMT-SARI outperforms all other systems on SARI.[11] Because there are 8 references for each complex sentence in the test set, BLEU scores are much higher compared to Newsela and WikiSmall. In human evaluation, we again elicited judgments for 100 randomly sampled test sentences. We randomly selected one of the 8 references as the Reference upper bound. On Simplicity, DRESS-LS is significantly better than all comparison systems, except Hybrid. On Adequacy, it is better than Hybrid but significantly worse than other comparison systems. On Fluency, it is on par with PBMT-R[12] but better than Hybrid and SBMT-SARI. On All dimension DRESS-LS significantly outperforms all comparison systems.

7 Conclusions

We developed a reinforcement learning-based text simplification model, which can jointly model simplicity, grammaticality, and semantic fidelity to the input. We also proposed a lexical simplification component that further boosts performance. Overall, we find that reinforcement learning offers a great means to inject prior knowledge to the simplification task achieving good results across three datasets. In the future, we would like to explicitly model sentence splitting and simplify entire documents (rather than individual sentences). Beyond sentence simplification, the reinforcement learning framework presented here is potentially applicable to generation tasks such as sentence compression (Chopra et al., 2016), generation of programming code (Ling et al., 2016), or poems (Zhang and Lapata, 2014).

Acknowledgments We would like to thank Li Dong, Jianpeng Cheng, Shashi Narayan and the EMNLP reviewers for their valuable feedback. We are also grateful to Shashi Narayan for supplying us with the output of his system and Wei Xu for her help with this work. The authors acknowledge the support of the European Research Council (award number 681760).

[11] BLEU and SARI scores reported in Xu et al. (2016) are 72.36 and 37.91, and measured at sentence-level.

[12] We used more data to train PBMT-R and maybe that is why PBMT-R performs better than Xu et al. (2016) reported.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR*, San Diego, CA.

Beata Beigman Klebanov, Kevin Knight, and Daniel Marcu. 2004. Text simplification for information-seeking applications. In *Proceedings of ODBASE*, volume 3290 of *Lecture Notes in Computer Science*, pages 735–747, Agia Napa, Cyprus. Springer.

J. Carroll, G. Minnen, D. Pearce, Y. Canning, S. Devlin, and J Tait. 1999. Simplifying text for language-impaired readers. In *Proceedings of the 9th EACL*, pages 269–270, Bergen, Norway.

R. Chandrasekar, C. Doran, and B. Srinivas. 1996. Motivations and methods for text simplification. In *Proceedings of the 16th COLING*, pages 1041–1044, Copenhagen, Denmark.

Sumit Chopra, Michael Auli, and Alexander M. Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of NAACL: HLT*, pages 93–98, San Diego, CA.

James Curran, Stephen Clark, and Johan Bos. 2007. Linguistically motivated large-scale nlp with c&c and boxer. In *Proceedings of the 45th ACL Companion Volume Proceedings of the Demo and Poster Sessions*, pages 33–36, Prague, Czech Republic.

Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In *Advances in Neural Information Processing Systems*, pages 3079–3087.

Siobhan Devlin. 1999. *Simplifying Natural Language for Aphasic Readers*. Ph.D. thesis, University of Sunderland.

Richard Evans, Constantin Or asan, and Iustin Dornescu. 2014. An evaluation of syntactic simplification rules for people with autism. In *Proceedings of the 3rd Workshop on Predicting and Improving Text Readability for Target Reader Populations (PITR)*, pages 131–140, Gothenburg, Sweden.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The paraphrase database. In *Proceedings of NAACL-HLT*, pages 758–764, Atlanta, GA.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Kentaro Inui, Atsushi Fujita, Tetsuro Takahashi, Ryu Iida, and Tomoya Iwakura. 2003. Text simplification for reading assistance: A project note. In *Proceedings of the 2nd International Workshop on Paraphrasing*, pages 9–16, Sapporo, Japan.

Sébastien Jean, Orhan Firat, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2015. Montreal neural machine translation systems for WMT15. In *Proceedings of the 10th Workshop on Statistical Machine Translation*, pages 134–140, Lisbon, Portugal.

Nobuhiro Kaji, Daisuke Kawahara, Sadao Kurohashi, and Satoshi Sato. 2002. Verb paraphrase based on case frame alignment. In *Proceedings of 40th ACL*, pages 215–222, Philadelphia, PA.

David Kauchak. 2013. Improving text simplification language modeling using unsimplified text data. In *Proceedings of the 51st ACL*, pages 1537–1546, Sofia, Bulgaria.

J Peter Kincaid, Robert P Fishburne Jr, Richard L Rogers, and Brad S Chissom. 1975. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. Technical report, DTIC Document.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Jiwei Li, Will Monroe, Alan Ritter, Dan Jurafsky, Michel Galley, and Jianfeng Gao. 2016. Deep reinforcement learning for dialogue generation. In *Proceedings of the 2016 EMNLP*, pages 1192–1202, Austin, TX.

Wang Ling, Phil Blunsom, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiský, Fumin Wang, and Andrew Senior. 2016. Latent predictor networks for code generation. In *Proceedings of the 54th ACL*, pages 599–609, Berlin, Germany.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 EMNLP*, pages 1412–1421, Lisbon, Portugal.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *ACL System Demonstrations*, pages 55–60.

Karthik Narasimhan, Adam Yala, and Regina Barzilay. 2016. Improving information extraction by acquiring external evidence with reinforcement learning. In *Proceedings of the 2016 EMNLP*, pages 2355–2365, Austin, TX.

Shashi Narayan and Claire Gardent. 2014. Hybrid simplification using deep semantics and machine translation. In *Proceedings of the 52nd ACL*, pages 435–445, Baltimore, MD.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th ACL*, pages 311–318, Philadelphia, PA.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. *ICML (3)*, 28:1310–1318.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *In Proceedings of the EMNLP 2014*, volume 14, pages 1532–43, Doha, Qatar.

MarcAurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. In *Proceedings of ICLR*, San Juan, Puerto Rico.

Luz Rello, Clara Bayarri, Azuki Górriz, Ricardo Baeza-Yates, Saurabh Gupta, Gaurang Kanvinde, Horacio Saggion, Stefan Bott, Roberto Carlini, and Vasile Topac. 2013. Dyswebxia 2.0!: More accessible text for people with dyslexia. In *Proceedings of the 10th International Cross-Disciplinary Conference on Web Accessibility*, pages –, Brazil.

Seonggi Ryang and Takeshi Abekawa. 2012. Framework of automatic text summarization using reinforcement learning. In *Proceedings of the 2012 EMNLP-CoNLL*, pages 256–265, Jeju Island, Korea.

Matthew Shardlow. 2014. A survey of automated text simplification. *International Journal of Advanced Computer Science and Applications*, pages 581–701. Special Issue on Natural Language Processing.

Advaith Siddharthan. 2004. Syntactic simplification and text cohesion. in research on language and computation. *Research on Language and Computation*, 4(1):77–109.

Advaith Siddharthan. 2014. A survey of research on text simplification. *International Journal of Applied Linguistics*, 165(2):259–298.

David A Smith and Jason Eisner. 2006. Quasi-synchronous grammars: Alignment by soft projection of syntactic dependencies. In *Proceedings of the NAACL 206 Workshop on Statistical Machine Translation*, pages 23–30, New York City.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas*, volume 200.

Lucia Specia, Sujay Kumar Jauhar, and Rada Mihalcea. 2012. Semeval-2012 task 1: English lexical simplification. In *In Proceedings of *SEM 2012*, pages 347–355, Montréal, Canada.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112.

D. Vickrey and D. Koller. 2008. Sentence simplification for semantic role labeling. In *Proceedings of ACL-08: HLT*, pages 344–352, Columbus, OH.

Willian Massami Watanabe, Arnaldo Candido Junior, Vinícius Rodriguez de Uzêda, Renata Pontin de Mattos Fortes, Thiago Alexandre Salgueiro Pardo, and Sandra Maria Alu´sio. 2009. Facilita: reading assistance for low-literacy readers. In *Proceedings of the 27th ACM International Conference on Design of Communication*, Bloomington, IN.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.

Kristian Woodsend and Mirella Lapata. 2011. Learning to simplify sentences with quasi-synchronous grammar and integer programming. In *Proceedings of the 2011 EMNLP*, pages 409–420, Edinburgh, Scotland.

Kristian Woodsend and Mirella Lapata. 2014. Text rewriting improves semantic role labeling. *Journal of Artificial Intelligence Research*, 51:133–164.

Sander Wubben, Antal Van Den Bosch, and Emiel Krahmer. 2012. Sentence simplification by monolingual machine translation. In *Proceedings of the 50th ACL*, pages 1015–1024, Jeju Island, Korea.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard S Zemel, and Yoshua Bengio. 2015a. Show, attend and tell: Neural image caption generation with visual attention. *arXiv preprint arXiv:1502.03044*, 2(3):5.

Wei Xu, Chris Callison-Burch, and Courtney Napoles. 2015b. Problems in current text simplification research: New data can help. *Transactions of the Association for Computational Linguistics*, 3:283–297.

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.

Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proceedings of the 39th ACL*, pages 523–530, Toulouse, France.

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*.

Xingxing Zhang and Mirella Lapata. 2014. Chinese poetry generation with recurrent neural networks. In *Proceedings of the 2014 EMNLP*, pages 670–680, Doha, Qatar.

Zhemin Zhu, Delphine Bernhard, and Iryna Gurevych. 2010. A monolingual tree-based translation model for sentence simplification. In *Proceedings of the 23rd COLING*, pages 1353–1361, Beijing, China.