How May I Help You?
Using Neural Text Simplification to Improve Downstream NLP Tasks

Hoang Van, Zheng Tang, and Mihai Surdeanu
Department of Computer Science, University of Arizona
{vnhh, zhengtang, msurdeanu}@email.arizona.edu

Abstract

The general goal of text simplification (TS) is to reduce text complexity for human consumption. In this paper, we investigate another potential use of neural TS: assisting machines performing natural language processing (NLP) tasks. We evaluate the use of neural TS in two ways: simplifying input texts at prediction time and augmenting data to provide machines with additional information during training. We demonstrate that the latter scenario provides positive effects on machine performance on two separate datasets. In particular, the latter use of TS significantly improves the performances of LSTM (1.82–1.98%) and SpanBERT (0.7–1.3%) extractors on TACRED, a complex, large-scale, real-world relation extraction task. Further, the same setting yields significant improvements of up to 0.65% matched and 0.62% mismatched accuracies for a BERT text classifier on MNLI, a practical natural language inference dataset.

1 Introduction

The goal of text simplification (TS) is to reduce text complexity (while preserving meaning) such that the corresponding text becomes more accessible to human readers. Previous works explored how TS can assist children (Kajiwara et al., 2013), non-native speakers (Pellow and Eskenazi, 2014), and people with disabilities (Rello et al., 2013). While this can be achieved in a variety of approaches (Sikka et al., 2020), most TS research has focused on two major approaches: rule-based and neural sequence-to-sequence (seq2seq). Since 2017, there is a significant increase of neural seq2seq TS methods (Zhang and Lapata, 2017; Zhao et al., 2018; Kriz et al., 2019; Maddela et al., 2020; Jiang et al., 2020).

In this paper, we analyze another potential use of the latter TS direction: assisting machines performing natural language processing (NLP) tasks. To this end, we investigate two possible directions: (a) using TS to simplify input texts at prediction time, and (b) using TS to augment training data for the respective NLP tasks. We empirically analyze these two directions using two neural TS systems (Martin et al., 2019; Nisioi et al., 2017), and two NLP tasks: relation extraction using the TACRED dataset (Zhang et al., 2017), and multi-genre natural language inference (MNLI) (Williams et al., 2017). Further, within these two tasks, we explore three methods: two relation extraction approaches, one based on LSTMs (Hochreiter and Schmidhuber, 1997) and another based on transformer networks, SpanBERT (Joshi et al., 2020), and one method for MNLI also based on transformer networks, BERT (Devlin et al., 2018).

Our analysis shows that simplifying texts at prediction times does not improve results, but using TS to augment training data consistently helps in all configurations. In particular, after augmented data is added, all approaches outperform their respective configurations without augmented data on both TACRED (0.7–1.98% in F1) and MNLI (0.50–0.65% in accuracies) tasks. The reproducibility checklist and the software are available at this link: https://github.com/vanh17/TextSiM.

2 Related Work

Recent work have effectively proven the practical application of neural networks and neural deep learning approaches to solving machine learning problems (Ghosh et al., 2021; Blalock et al., 2020; Yin et al., 2017).

With respect to input simplification, several works have utilized TS as a pre-processing step for downstream NLP tasks such as information extraction (Miwa et al., 2010; Schmidek and Barbosa, 2014; Niklaus et al., 2017), parsing (Chandrasekar et al., 1996), semantic role labeling (Vickrey and Koller, 2008), and machine translation (Štajner and Popović, 2016). However, most of them focus on
the use of rule-based TS methods. In contrast, we investigate the potential use of domain-agnostic neural TS systems in simplifying inputs for downstream tasks. We show that, despite the complexity of the tasks investigated and the domain agnosticism of the TS approaches, TS improves both tasks when used for training data augmentation, but not when used to simplify evaluation texts.

On data augmentation for natural language processing downstream tasks, previous work show significant benefits of introducing noisy data on the machine performance (Van et al., 2021; Kobayashi, 2018). Previous efforts used TS approaches, e.g. lexical substitution, to augment training data for downstream tasks such as text classification (Zhang et al., 2015; Wei and Zou, 2019). However, these methods focused on replacing words with thesaurus-based synonyms, and did not emphasize other important lexical and syntactic simplification. Here, we use two out-of-the-box neural TS systems that apply both lexical and syntactic sentence simplification for data augmentation, and show that our data augmentation consistently leads to better performances. Note that we do not use rule-based TS systems because they have been proven to perform worse than their neural counterparts (Zhang and Lapata, 2017; Nisioi et al., 2017). Further, rule-based TS systems are harder to build in a domain-independent way due to the many linguistic/syntactic variations across domains.

3 Approach

We investigate the impact of text simplification on downstream NLP tasks in two ways: (a) simplifying input texts at prediction time, and (b) augmenting training data for the respective NLP tasks. We discuss the settings of these experiments next.

3.1 Input Simplification at Prediction Time

We pose the run-time input simplification problem as a transparent data pre-processing problem. That is, given an input data point, we simplify the text while keeping the native format of the task, and then feed the modified input to the actual NLP task. For example, for the TACRED sentence "the CFO Douglas Flint will become chairman, succeeding Stephen Green is leaving for a government job.", which contains a per:title relation between the two entities Douglas Flint and chairman, our approach will first simplify the text to "the CFO Douglas Flint will become chairman, and Stephen Green is leaving to take a government job.". Then we generate a relation prediction for the simplified text using existing relation extraction classifiers.

3.2 Data Augmentation for Training

Here, we augment training data by simplifying the text of some original training examples, and appending it to the original training dataset. First, we sample which examples should be used for augmentation with probability \( p \). Second, once an example is selected for augmentation, we generate an additional example with the text portion simplified using TS. For example, for the data in section 3.1, we generate an additional training data with the corresponding simplified text. \( p \) is a hyper parameter that we tuned for each task (see next section).

4 Experimental Setup

NLP tasks and methods: We evaluate the impact of TS on two NLP tasks: (a) relation extraction (RE) using the TACRED dataset (Zhang et al., 2017), and (b) natural language inference (NLI) on the MNLI dataset (Williams et al., 2017).

TACRED is a large-scale RE dataset with 106,264 examples built on newswire and web text.
We train three approaches for these two tasks. First, for TACRED, we use a classifier based on LSTM \(^1\) (Hochreiter and Schmidhuber, 1997), and a second based on SpanBERT \(^2\) (Joshi et al., 2020). For MNLI, we trained a BERT-based classifier \(^3\) (Devlin et al., 2018). For reproducibility, we use the default settings and general hyper parameters recommended by the task and creators of the transformer networks (Zhang et al., 2017; Joshi et al., 2020; Devlin et al., 2018). Through this, we aim to separate potential improvements of our approaches from those coming from improved configurations.

**Text simplification methods:** For TS, we use two out-of-the-box neural seq2seq TS approaches: ACCESS (Martin et al., 2019), and NTS (Nisioi et al., 2017). Tables 1 and 2 show the BLEU scores (Papineni et al., 2002) between original and simplified text generated by these two TS systems for the two tasks. The tables highlight that both systems change the input texts, with ACCESS being more aggressive.

**Evaluation measures:** We directly followed the evaluation measures proposed by the original task organizers (Zhang et al., 2017; Williams et al., 2017). Specifically, we used these main metrics: (a) F1 on TACRED relation extraction, and (b)

---

\(^1\)https://github.com/yuhaozhang/tacred-relation
\(^2\)https://huggingface.co/SpanBERT/spanbert-large-cased
\(^3\)https://huggingface.co/bert-base-cased
Train Data Sets

<table>
<thead>
<tr>
<th>ACCESS</th>
<th>Simplified m/mm acc</th>
<th>Original m/mm acc</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Original</td>
<td>71.25/71.43</td>
<td>82.89/83.10</td>
</tr>
<tr>
<td>2 Original Swapped with 10% Simplified</td>
<td>71.76 ± 0.13/</td>
<td>83.00 ± 0.03/</td>
</tr>
<tr>
<td>3 Original Swapped with 20% Simplified</td>
<td>72.12 ± 0.08</td>
<td>83.25 ± 0.05</td>
</tr>
<tr>
<td>4 5% Simplified + Original (AD)</td>
<td>71.30 ± 0.15/</td>
<td>83.47 ± 0.04/</td>
</tr>
<tr>
<td>5 10% Simplified + Original (AD)</td>
<td>71.81 ± 0.07/</td>
<td>82.81 ± 0.05/</td>
</tr>
<tr>
<td>6 15% Simplified + Original (AD)</td>
<td>72.10 ± 0.07</td>
<td>83.13 ± 0.06</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NTS</th>
<th>Simplified m/mm acc</th>
<th>Original m/mm acc</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 Original</td>
<td>33.36/33.53</td>
<td>82.89/83.10</td>
</tr>
<tr>
<td>8 Original Swapped with 10% Simplified</td>
<td>33.39 ± 0.10/</td>
<td>83.28 ± 0.07/</td>
</tr>
<tr>
<td>9 Original Swapped with 20% Simplified</td>
<td>33.46 ± 0.08</td>
<td>83.50 ± 0.11</td>
</tr>
<tr>
<td>10 5% Simplified + Original (AD)</td>
<td>33.71 ± 0.08/</td>
<td>82.60 ± 0.14/</td>
</tr>
<tr>
<td>11 10% Simplified + Original (AD)</td>
<td>33.90 ± 0.11/</td>
<td>82.79 ± 0.09</td>
</tr>
<tr>
<td>12 15% Simplified + Original (AD)</td>
<td>33.50 ± 0.09/</td>
<td>83.92 ± 0.05</td>
</tr>
</tbody>
</table>

Table 5: Matched (m) and mismatched (mm) accuracies on MNLI development using text simplified/augmented by ACCESS (top half) and NTS (bottom half). "Original Swapped with x% Simplified" consists of original data with x% of data points replaced with their simplified form. "x% Simplified + Original" consists of the original data augmented with an additional x% of simplified data. (AD) annotates models using data augmented by neural TS systems during training. Note that our results in the original configuration differ slightly from those in (Devlin et al., 2018). This is likely due to the different hardware and library versions used (Belz et al., 2021).
we could apply a form of quality control, i.e., by accepting only the simplifications that preserve the subject and object of the relation. To illustrate the benefits/dangers of text simplification, we show a few examples where simplification improves/hurts MNLI output in Table 7.

Augmenting training data: As shown in row 3 and 6 in Table 3 and 4, all methods trained on augmented data yield consistent performance improvements, regardless of the RE classifier used (LSTM or SpanBERT) or TS method used (ACCESS or NTS). There are absolute increases of 1.30–1.82% F1 for ACCESS and 0.70–1.98% F1 for NTS on (subtract row 1 from row 3 and row 4 from row 6 for ACCESS and NTS respectively). The best configuration is when the original training data is augmented with all data points that could be simplified while preserving the subject and object of the relation (rows 4 and 8 in the two tables). These results confirm that TS systems can provide additional, useful training information for RE methods.

Similarly, on MNLI, the classifier trained using augmented data outperforms the BERT classifier that is trained only on the original MNLI data. For two TS systems, ACCESS and NTS, we observe performance increases of 0.59–0.65% matched accuracy, and 0.50–0.62% mismatched accuracy (compare rows 1 vs. 2, and row 3 vs. 4 in Table 6). This confirms that TS as data augmentation is also useful for NLI.

All in all, our experiments suggest that our data augmentation approach using TS is fairly general. It does not depend on the actual TS method used, and it improves three different methods from two different NLP tasks. Further, our results indicate that our augmentation approach is more beneficial for tasks with lower resources (e.g., TACRED), but its impact decreases as more training data is available (e.g., MNLI).

6 Conclusion

We investigated the effects of neural TS systems on downstream NLP tasks using two strategies: (a) simplifying input texts at prediction time, and (b) augmenting data to provide machines with additional information during training. Our experiments indicate that the latter strategy consistently helps multiple NLP tasks, regardless of the underlying method used to address the task, or the neural approach used for TS.

Acknowledgements

This work was supported by the Defense Advanced Research Projects Agency (DARPA) under the World Modelers and HABITUS programs. Mihai Surdeanu declares a financial interest in lum.ai. This interest has been properly disclosed to the University of Arizona Institutional Review Committee, and is managed in accordance with its conflict of interest policies.

References


R. Chandrasekar, Christine Doran, and B. Srinivas. 1996. Motivations and methods for text simplifica-


