# Using Machine Translation Evaluation Techniques to Evaluate Text Simplification Systems

**Sandeep Mathias, Pushpak Bhattacharyya**
Department of Computer Science and Engineering
IIT Bombay, India
{sam,pb}@cse.iitb.ac.in

**Abstract**

In this paper, we discuss our approaches to find out ways to evaluate automated text simplification systems, based on the grammaticality and simplicity of their output, as well as the meaning preserved from the input, and the overall quality of simplification of the system. In this paper, we discuss existing techniques currently used in the area of machine translation, as well as a novel technique for text complexity analysis, to assess the quality of the text simplification system.

**Keywords:** METEOR, Language modelling, Lexical complexity

## 1. Introduction

While there has been a lot of research in automated text simplification, there has been relatively little work done in evaluating automated text simplification systems. One of the main bottlenecks in coming up with solutions to this problem lies in what should be measured. Should we provide more emphasis on the grammar of the output, or the retention of the meaning of the output, or the difficulty of the output, or a combination of all three? Our paper aims to describe different techniques to solve these problems. Some of them are already in use in the field of machine translation, while others are relatively new techniques specifically for text simplification.

## 2. Problem Statement

The LREC (Language Resources and Evaluation Conference) 2016 had a workshop called the Quality Assessment for Text Simplification (QATS) workshop that had a shared task which aims to find out ways to evaluate various aspects of assessing the output from text simplification systesms. The main aim of the shared task is to find out ways to evaluate different text simplification systems based on their output. For this task, we consider the following questions:

1. How grammatically correct is the output of the system?
2. How simple is the output of the system?
3. How much of the meaning of the input sentence is preserved in the output of the system?
4. How good is the overall quality of the system?

The training data has 505 sentence pairs that are manually scored for various aspects, such as grammaticality, meaning preservation, simplicity, and overall quality. The sentence pairs (or output sentences in the case of grammaticality and simplicity) are classified as either "good", "ok", or "bad". A separate test set of 126 sentence pairs was also used to test the quality of our approaches.

We view each of these questions as individual classification problems, with each instance being classified as either "good", or "ok", or "bad". The following sections will explain our approaches to solve each of these questions.

## 3. Grammaticality

Grammaticality is a means of finding out how grammatically correct a sentence is. Grammaticality is scored based on the quality of the simplified sentence only. To evaluate grammaticality, we make use of language modelling - one of the tasks in machine translation. In machine translation, the language model is used to check how likely a string in the target language is. Hence, we use language modelling to score the grammaticality of the system.

To build the language model, we make use of the Simple English Wikipedia from the English Wikipedia - Simple English Wikipedia (Kauchak, 2013) corpus. The corpus consists of a sentence aligned and a document aligned parallel corpus between articles in English Wikipedia[1] and Simple English Wikipedia[2]. The sentence aligned corpus has sentences in the Simple English Wikipedia aligned with their corresponding sentence in the English Wikipedia. On the other hand, the document aligned corpus has the article in the Simple English Wikipedia aligned with the corresponding article in the English Wikipedia. Since every article in the Simple English Wikipedia has a corresponding article in the English Wikipedia, using the document-aligned Simple English Wikipedia is equivalent to using the entire Simple English Wikipedia (at least upto the point of the corpus' creation).

For our task, we take all the Simple Engilsh Wikipedia articles from the document aligned corpus to calculate the language modelling score. We make use of the SRILM toolkit used for machine translation to train the language model. Using SRILM, we also find out the out of vocabulary words (OOVs), the perplexity (ppl), and the average perplexity per word (ppl1) in each sentence. The following features are used to help classify the how grammatical the output sentence is:

1. Number of words in the sentence
2. Number of OOVs

[1] http://en.wikipedia.org
[2] http://simple.wikipedia.org

3. Log of the probabililty score (Language model score for the sentence)

4. Perplexity of the sentence

5. Average perplexity per word of the sentence

Details of the experiment and results are covered in Section 6.

## 4. Meaning Preservation

The second aspect that we score is the amount of meaning retained in the output, given a particular input sentence. Consider the following sentence:

> ***Warsaw*** *lies on the* ***Vistula*** *River, about 240 miles southeast of the Baltic coast city of Gdansk.*[3]

An example of a good meaning preservation would be ***Warsaw*** *is on the* ***Vistula*** *River, about 240 miles southeast of Gdansk. Gdansk is a Baltic coast city*. However, a sentence with bad meaning preservation would be something like ***Vistula*** *is on the* ***Warsaw*** *River, about 240 miles southeast of the Baltic coast city of Gdansk*. To accomplish this task, we make use of the METEOR (Denkowski and Lavie, 2014) metric, already being used in machine translation. Unlike BLEU (Papineni et al., 2002) , the advantages of using METEOR are as follows:

1. BLEU matches words only if they are completely matched (i.e. their surface forms are the same). For instance, BLEU would score a match of *live* $\rightarrow$ *lives* as 0. However, since both live and lives have the same stem, METEOR will award some value to the match.

2. BLEU does not match synonyms. A word like *jail* $\rightarrow$ *prison* will be scored as 0. However, in METEOR, it would be awarded a score, since prison is a synonym of jail.

3. BLEU does not score paraphrases. At times, the output sentence may have a paraphrase of the reference phrase - for example, *kicked the bucket* $\rightarrow$ *died*. BLEU will score it as 0, but METEOR will award it a score.

METEOR works by identifying all possible matches between the input and simplified sentences. We consider 4 different types of matches, namely

1. **Exact** If the surface words are the same.

2. **Stem** If the stems of the words unmatched by the previous matcher are the same. This is done using the Snowball Stemmer for English (Porter, 2001).

3. **Synonym** If the stems of the words also remain unmatched, the remaining words are matched if they belong to the same synset in WordNet (Fellbaum, 1998).

4. **Paraphrase** Among the remaining unmatched words, match them if they occur as paraphrases in a paraphrase table (Denkowski and Lavie, 2014).

[3]This was one of the sentences in the test set of the shared task.

For each type of match, a weight is used to calculate the score. We use the weights described in (Denkowski and Lavie, 2014) for exact, stem, synonym, and paraphrase matching. Table 1 gives the weights of different types of matches used. We make use of the METEOR scores for estimating the meaning preservation of the input sentence in the simplified output sentence. Here, for each sentence pair, we calculate the METEOR score for the simplified sentence with respect to the original input sentence. We use the METEOR score as the only feature to measure the meaning of the input sentence preserved in the output sentence.

| Type of match | Weight |
|---|---|
| Exact | 1.00 |
| Stem | 0.60 |
| Synonym | 0.80 |
| Paraphrase | 0.60 |

Table 1: Weights of different matchers in METEOR

## 5. Simplicity

Along with measuring the grammaticality of the simplified output sentence, as well as the meaning of the input sentence preserved in the output sentence, we also look at how simple the output sentence is. This is important in judging the quality of the text simplification.

There are many measures to judge the quality of the simplified output. One of the earliest methods was the Flesch Reading Ease Score (FRES) (Flesch, 1948), proposed by Rudolph Flesch in 1948. This approach took into account mainly the average number of words per sentence, and the average number of syllables per word. It used a simple formula to calculate the reading ease of a piece of text, without the use of any extra data. More recently, the presence of data, in the form of the English Wikipedia - Simple English Wikipedia (Kauchak, 2013) gave rise to data driven approaches to simplify texts.

For calculating the complexity of the texts, we take into account mainly two types of complexity, namely structural complexity and lexical complexity. Structural complexity is a measure of how complex the sentence is, based on its parse tree. For structural compexity, we calculate the number of

1. Main clause sentences

2. Sentences from relative clauses

3. Sentences from appositives

4. Sentences from noun and verb participial phrases

5. Sentences from other subordinate clauses

that we can extract from a single input sentence using Michael Heilman's factual statement extractor[4] (Heilman and Smith, 2010).

[4]The system can be downloaded from www.cs.cmu.edu/~ark/mheilman/qg-2010-workshop

Lexical complexity is the complexity of the text based on its vocabulary. It is based on the complexity of the words and phrases used in the text.
We use a unigram and bigram language model of the English Wikipedia - Simple English Wikipedia (Kauchak, 2013) parallel corpus to calculate the lexical complexity of each n-gram. The complexity of an n-gram is comprised of 2 parts, namely the corpus complexity and the syllable count.

1. **Corpus complexity** For each n-gram ($g$) of the sentence, we calculate its corpus complexity (Biran et al., 2011), $C_c(g)$, defined as the ratio of the log likelihood of $g$ in the English corpus to the log likelihood of $g$ in the Simple English corpus. In other words,

$$C_c(g) = \frac{LL(g|normal)}{LL(g|simple)}$$

   Here, we assume that every n-gram in the Simple English corpus has to occur at least once in the English corpus.

2. **Syllable count** We consider that readers read words one syllable at a time. The syllable count, $s(g)$, of an n-gram ($g$) is defined as the sum of syllables of the words in that n-gram.

With these two ideas, we go ahead and calculate the lexical complexity of an n-gram ($g$) as:

$$Lc(g) = s(g) \times C_c(g)$$

Hence, for a given sentence $S$, and an n-gram size, the lexical complexity is given by

$$Lc(S, n) = \sum_g s(g) \times C_c(g),$$

where $g$ is an n-gram of size $n$.
In addition to this, we also attach a weight $W_n$ to the lexical complexity calculated for a particular n-gram. For a given n-gram size of $n$, the weight is $\frac{1}{n}$. This is because the unigrams in the n-gram are added n-times. For example, if $n$ is 2, and we have an n-gram sequence "a b c d e f g ...", unigrams like b, c, d, e, f, etc. get added twice.
Therefore, we can say that the lexical complexity of a sentence is given by

$$L_c(S) = \sum_n W_n \sum_g s(g) \times C_c(g),$$

The final complexity of the text is calculated as the sum of the lexical and structural complexity. We use this value as the feature in calculating the complexity of the simplified sentence.

# 6. Shared Task Results

For each of the tasks - grammaticality, meaning preservation, simplicity - we treat them as a classification problem and classify the outputs of the simplification systems as either "good", "ok", or "bad". We use Bagging with the REPTree classifier in Weka running 10 iterations to train our model. The training data was a set of 505 sentence pairs, and the test data, an additional 126 sentence pairs. Accuracy (Acc.) is the percentage of sentences correctly classified. To calcuate the mean absolute error (MAE), and the root mean square error (RMSE), 100, 50 and 0 were given to the classes "good", "ok", and "bad" respectively. The baseline we use is the majority class of our training data.
The following are the results of various aspects of our task.

## 6.1. Grammaticality

The following table gives the results for our experiments with the grammaticality of the output sentences.

| Experiment | Acc. (%) | MAE | RMSE |
|---|---|---|---|
| Training set - Baseline | 75.64 | 17.23 | 36.96 |
| Training set | 76.04 | 16.63 | 36.01 |
| Test set - Baseline | 76.19 | 18.25 | 21.63 |
| Test set | 72.22 | 21.43 | 25.78 |

Table 2: Results of grammaticality classification

From the results, we can clearly see that existing language modelling toolkits (like SRILM) can provide reasonably accurate results for grammaticality.

## 6.2. Meaning Preservation

The following table gives the results for our experiments in estimating the meaning preservation of the input in the output sentences.

| Experiment | Acc. (%) | MAE | RMSE |
|---|---|---|---|
| Training set - Baseline | 58.21 | 28.61 | 46.94 |
| Training set | 66.34 | 19.50 | 35.25 |
| Test set - Baseline | 57.94 | 28.97 | 35.30 |
| Test set | 63.49 | 20.63 | 26.75 |

Table 3: Results of Meaning Preservation using METEOR

From the above results, we see that the use of METEOR for classifying the output gives satisfactory results.

## 6.3. Simplicity

The following table gives the results for our experiments in estimating the simplicity of the output of the text simplification system.

| Experiment | Acc. (%) | MAE | RMSE |
|---|---|---|---|
| Training set - Baseline | 52.67 | 32.18 | 49.60 |
| Training set | 48.31 | 32.87 | 48.59 |
| Test set - Baseline | 55.56 | 29.37 | 31.22 |
| Test set | 47.62 | 34.13 | 38.85 |

Table 4: Results of Simplicity of output

From the above results, we realize that calculating the simplicity of the output of the system requires a lot of research to solve.

### 6.4. Overall quality

In addition to calculate the above metrics, we also classify the overall quality of the system. Here, we consider two experiments. The first uses only the classes output that we get from the different aspects of the simplification system, namely the class values of grammaticality, simplicity and meaning preservation as features. The second uses the classes output as well as the other values (like OOVs, lexical complexity, etc.) that we used to classify the individual aspects of simplification as features. The following table gives the results of classifying the overall quality of the text simplification system:

| Experiment | Acc. (%) | MAE | RMSE |
|---|---|---|---|
| Training set - Baseline | 43.76 | 33.17 | 46.51 |
| Training Set - Classes | 45.74 | 31.39 | 44.67 |
| Training Set - Values | 56.23 | 23.56 | 36.70 |
| Test Set - Baseline | 43.65 | 28.17 | 40.52 |
| Test Set - Classes | 33.33 | 42.46 | 47.83 |
| Test Set - Values | 39.68 | 34.92 | 42.97 |

Table 5: Overall Quality Classification Results

Training Set - Classes and Test Set - Classes make use of only the classes output from our tasks, like simplicity, meaning preservation and grammaticality. Training Set - Values and Test Set - Values use all the other values calculated, like METEOR Score, lexical complexity, etc. in addition to the class values to help classify the overall quality of the output. One of the factors affecting the low quality of the overall quality classification is the fact that our simplification results are comparatively low as compared to those of meaning preservation and grammaticality.

## 7. Conclusions

Among the three given tasks, we have seen that, for the evaluation of text simplification systems, metrics such as METEOR and techniques like language modelling can achieve good results as compared to more complex tasks, like evaluating the simplicity of the text, which is also why the accuracy in the overall quality classification of the various text simplification systems is quite low.

## 8. Bibliographical References


Biran, O., Samuel, B., and Elhadad, N. (2011). Putting it simply: a context-aware approach to lexical simplification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers*, volume 2, pages 496–501. Association for Computational Linguistics.

Denkowski, M. and Lavie, A. (2014). Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the EACL 2014 Workshop on Statistical Machine Translation*.

Fellbaum, C. (1998). WordNet: An electronic database.

Flesch, R. (1948). A new readability yardstick. *Journal of applied psychology*, 32(3):221–233.

Heilman, M. and Smith, N. A. (2010). Extracting simplified statements for factual question generation. In *Proceedings of QG2010: The Third Workshop on Ques-tion Generation*, page 11.

Kauchak, D. (2013). Improving text simplification language modelling using unsimplified text data. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 1537–1546. Association for Computational Linguistics.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Porter, M. F. (2001). Snowball: A language for stemming algorithms.