Sentence Simplification as Tree Transduction

Dan Feblowitz
Computer Science Department
Pomona College
Claremont, CA
djf02007@mymail.pomona.edu

David Kauchak
Computer Science Department
Middlebury College
Middlebury, VT
dkauchak@middlebury.edu

Abstract

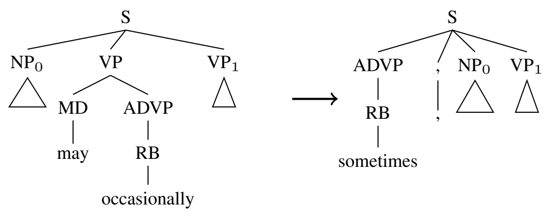
In this paper, we introduce a syntax-based sentence simplifier that models simplification using a probabilistic synchronous tree substitution grammar (STSG). To improve the STSG model specificity we utilize a multi-level backoff model with additional syntactic annotations that allow for better discrimination over previous STSG formulations. We compare our approach to T3 (Cohn and Lapata, 2009), a recent STSG implementation, as well as two state-of-the-art phrase-based sentence simplifiers on a corpus of aligned sentences from English and Simple English Wikipedia. Our new approach performs significantly better than T3, similarly to human simplifications for both simplicity and fluency, and better than the phrase-based simplifiers for most of the evaluation metrics.

1 Introduction

Text simplification is aimed at reducing the reading and grammatical complexity of text while retaining the meaning. Text simplification has applications for children, language learners, people with disabilities (Carroll et al., 1998; Feng, 2008) and in technical domains such as medicine (Elhadad, 2006), and can be beneficial as a preprocessing step for other NLP applications (Vickrey and Koller, 2008; Miwa et al., 2010). In this paper we introduce a new probabilistic model for sentence simplification using synchronous tree substitution grammars (STSG).

Synchronous grammars can be viewed as simultaneously generating a pair of recursively related strings or trees (Chiang, 2006). STSG grammar rules contain pairs of tree fragments called *elementary trees* (Eisner, 2003; Cohn and Lapata, 2009; Yamangil and Shieber, 2010). The leaves of an elementary tree can be either terminal, lexical nodes or aligned *nonterminals* (also referred to as variables or frontier nodes). Because elementary trees may have any number of internal nodes structured in any way STSGs allow for more complicated derivations not expressible with other synchronous grammars.

To simplify an existing tree, an STSG grammar is used as a tree transducer. Figure 1 shows some example simplification STSG rules written in transductive form. As a transducer the grammar rules take an elementary tree and rewrite it as the tree on the right-hand side of the rule. For example, the first rule in Figure 1 would make the transformation



changing “may occasionally” to “sometimes ,” and moving the noun phrase from the beginning of the sentence to after the comma. The indices on the nonterminals indicate alignment and transduction continues recursively on these aligned nonterminals until no nonterminals remain. In the example above, transduction would continue down the tree on the NP and VP subtrees. A probabilistic STSG has a probability associated with each rule.

One of the key challenges in learning an STSG from an aligned corpus is determining the right level of specificity for the rules: too general and they can be applied in inappropriate contexts; too specific, and the rules do not apply in enough contexts. Previous work on STSG learning has regulated the rule specificity based on elementary tree depth (Cohn and Lapata, 2009), however, this approach has not worked well for text simplifica-

S(NP$_0$ VP(MD(may) ADVP(RB(occasionally))) VP$_1$)	$\rightarrow$	S(ADVP(RB(sometimes)) ,(,) NP$_0$ VP$_1$)
NP(NNS$_0$)	$\rightarrow$	NP(NNS$_0$)
NP(JJ$_0$ NNS$_1$)	$\rightarrow$	NP(JJ$_0$ NNS$_1$)
VP(VB$_0$ PP(IN(in) NP$_1$))	$\rightarrow$	VP(VB$_0$ NP$_1$)
VB(assemble),	$\rightarrow$	VB(join)
JJ(small)	$\rightarrow$	JJ(small)
NNS(packs)	$\rightarrow$	NNS(packs)
NNS(jackals)	$\rightarrow$	NNS(jackals)

Figure 1: Example STSG rules representing the maximally general set for the aligned trees in Figure 2. The rules are written in transductive form. Aligned nonterminals are indicated by indices.

tion (Coster and Kauchak, 2011a). In this paper, we take a different approach and augment the grammar with additional information to increase the specificity of the rules (Galley and McKeown, 2007). We combine varying levels of grammar augmentation into a single probabilistic backoff model (Yamangil and Nelken, 2008). This approach creates a model that uses specific rules when the context has been previously seen in the training data and more general rules when the context has not been seen.

2 Related Work

Our formulation is most closely related to the T3 model (Cohn and Lapata, 2009), which is also based on the STSG formalism. T3 was developed for the related problem of text compression, though it supports the full range of transformation operations required for simplification. We use a modified version of their constituent alignment and rule extraction algorithms to extract the basic STSG rules with three key changes. First, T3 modulates the rule specificity based on elementary tree depth, while we use additional grammar annotations combined via a backoff model allowing for a broader range of context discrimination. Second, we learn a probabilistic model while T3 learns the rule scores discriminatively. T3's discriminative training is computationally prohibitive for even modest sized training sets and a probabilistic model can be combined with other probabilities in a meaningful way. Third, our implementation outputs an n-best list which we then rerank based on a trained log-linear model to select the final candidate.

Zhu et al. (2010) suggest a probabilistic, syntax-based approach to text simplification. Unlike the STSG formalism, which handles all of the transformation operations required for sentence simplification in a unified framework, their model uses a combination of hand-crafted components, each designed to handle a different transformation operation. Because of this model rigidity, their system performed poorly on evaluation metrics that take into account the content and relative to other simplification systems (Wubben et al., 2012).

Woodsend and Lapata (2011) introduce a quasi-synchronous grammar formulation and pose the simplification problem as an integer linear program. Their model has similar representational capacity to an STSG, though the learned models tend to be much more constrained, consisting of $<$1000 rules. With this limited rule set, it is impossible to model all of the possible lexical substitutions or to handle simplifications that are strongly context dependent. This quasi-synchronous grammar approach performed better than Zhu et al. (2010) in a recent comparison, but still performed worse than recent phrase-based approaches (Wubben et al., 2012).

A number of other approaches exist that use Simple English Wikipedia to learn a simplification model. Yatskar et al. (2010) and Biran et al. (2011) learn lexical simplifications, but do not tackle the more general simplification problem. Coster and Kauchak (2011a) and Wubben et al. (2012) use a modified phrase-based model based on a machine translation framework. We compare against both of these systems. Qualitatively, we find that phrasal models do not have the representative power of syntax-based approaches and tend to only make small changes when simplifying.

Finally, there are a few early rule-based simplification systems (Chandrasekar and Srinivas, 1997; Carroll et al., 1998) that provide motivation for recent syntactic approaches. Feng (2008) provides a good overview of these.

3 Probabilistic Tree-to-Tree Transduction

We model text simplification as tree-to-tree transduction with a probabilistic STSG acquired from

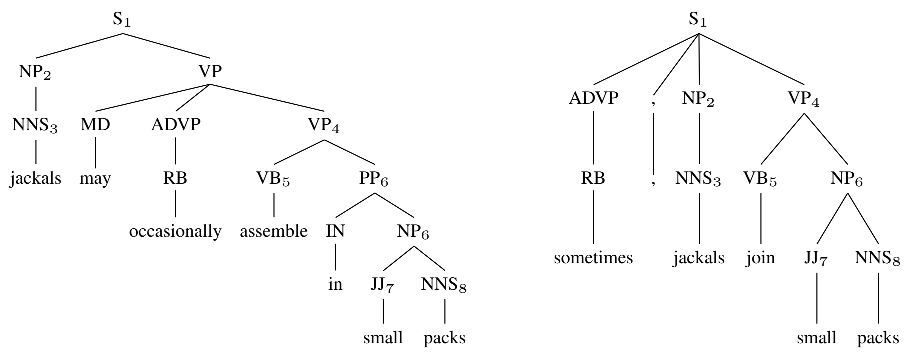


Figure 2: An example pair of constituent aligned trees generated by the constituent alignment algorithm. Aligned constituents are indicated with a shared index number (e.g. NP_2 is aligned to NP_2).

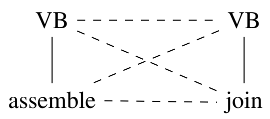
a parsed, sentence-aligned corpus between normal and simplified sentences. To learn the grammar, we first align tree constituents based on an induced word alignment then extract grammar rules that are consistent with the constituent alignment. To improve the specificity of the grammar we augment the original rules with additional lexical and positional information. To simplify a sentence based on the learned grammar, we generate a finite-state transducer (May and Knight, 2006) and use the transducer to generate an n-best list of simplifications. We then rerank the n-best list of simplifications using a trained log-linear model and output the highest scoring simplification. The subsections below look at each of these steps in more detail. Throughout the rest of this paper, we will refer to the unsimplified text/trees as *normal* and the simplified variants as *simple*.

3.1 Rule Extraction

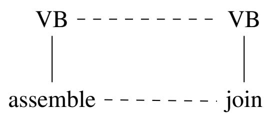
Given a corpus of pairs of trees representing normal and simplified sentences, the first step is to extract a set of basic STSG production rules from each tree pair. We used a modified version of the algorithm presented by Cohn and Lapata (2009). Due to space constraints, we only present here a brief summary of the algorithm along with our modifications to the original algorithm. See Cohn and Lapata (2009) for more details.

Word-level alignments are learned using Giza++ (Och and Ney, 2000) then tree nodes (i.e. constituents) are aligned if: there exists at least one pair of nodes below them that is aligned *and* all nodes below them are either aligned to a node under the other constituent or unaligned. Given the constituent alignment, we then extract the STSG production rules. Because STSG rules can have arbitrary depth, there are often many possible sets of rules that could be extracted from a pair of trees.[1] Following Cohn and Lapata (2009) we extract the *maximally general* rule set from an aligned pair of input trees that is consistent with the alignment: the set of rules capable of synchronously deriving the original aligned tree pair consisting of rules with the smallest depth. Figure 2 shows an example tree pair that has been constituent aligned and Figure 1 shows the extracted STSG rules.

We modify the constituent alignment algorithm from Cohn and Lapata (2009) by adding the requirement that if node b with parent a are both aligned to node z and its parent y, we only align the pairs (a, y) and (b, z), i.e. align the children and align the parents. This eliminates a common occurrence where too many associations are made between a pair of preterminal nodes and their children. For example, for the sentences shown in Figure 2 the word alignment contains "`assemble`" aligned to "`join`". Under the original definition four aligned pairs would be generated:



but only two under our revised definition:



This revised algorithm reduces the size of the alignment, decreasing the number of cases which must be checked during grammar extraction while preserving the intuitive correspondence.

[1]There is always at least one set of rules that can generate a tree pair consisting of the entire trees.

3.2 Grammar Generation

During the production rule extraction process, we select the production rules that are most general. More general rules allow the resulting transducer to handle more potential inputs, but can also result in unwanted transformations. When generating the grammar, this problem can be mitigated by also adding more specific rules.

Previous approaches have modulated rule specificity by incorporating rules of varying depth in addition to the maximally general rule set (Cohn and Lapata, 2009), though this approach can be problematic. Consider the aligned subtrees rooted at nodes (VP_4, VP_4) in Figure 2. An STSG learning algorithm that controls rule specificity based on depth must choose between generating the rule:

$\text{VP}(\text{VB}_0\ \text{PP}(\text{IN}(\text{in})\ \text{NP}_1)) \rightarrow \text{VP}(\text{VB}_0\ \text{NP}_1)$

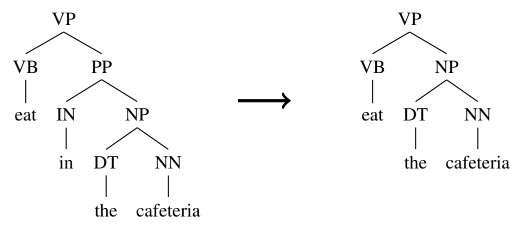
which drops the preposition, or a deeper rule that includes the lexical leaves such as:

$\text{VP}(\text{VB}(\text{assemble})\ \text{PP}(\text{IN}(\text{in})\ \text{NP}_1)) \rightarrow \text{VP}(\text{VB}(\text{join})\ \text{NP}_1)$

or

$\text{VP}(\text{VB}(\text{assemble})\ \text{PP}(\text{IN}(\text{in})\ \text{NP}(\text{JJ}_0\ \text{NNS}_1))) \rightarrow$
$\text{VP}(\text{VB}(\text{join})\ \text{NP}(\text{JJ}_0\ \text{NNS}_1))$

If either of the latter rule forms is chosen, the applicability is strongly restricted because of the specificity and lexical requirement. If the former rule is chosen and we apply this rule we could make the following inappropriate transformation:



simplifying *"eat in the cafeteria"* to *"eat the cafeteria"*.

We adopt a different approach to increase the rule specificity. We augment the production rules and resulting grammar with several parse tree annotations shown previously to improve SCFG-based sentence compression (Galley and McKeown, 2007) as well as parsing (Collins, 1999): parent annotation, head-lexicalization, and annotation with the part of speech of the head word.

Following Yamangil and Nelken (2008), we learn four different models and combine them into a single backoff model. Each model level increases specificity by adding additional rule annotations. Model 1 contains only the original production rules. Model 2 adds parent annotation, Model 3 adds the head child part of speech and Model 4 adds head child lexicalization. The head child was determined using the set of rules from Collins (1999). Figure 3 shows the four different model representations for the VP rule above.

3.3 Probability Estimation

We train each of the four models individually using maximum likelihood estimation over the training corpus, specifically:

$$p(s|n) = \frac{count(s \wedge n)}{count(n)}$$

where s and n are tree fragments with that level's annotation representing the right and left sides of the rule respectively.

During simplification, we start with the most specific rules, i.e. Model 4. If a tree fragment was not observed in the training data at that model level, we repeatedly try a model level simpler until a model is found with the tree fragment (Yamangil and Nelken, 2008). We then use the probability distribution given by that model. A tree fragment only matches at a particular level if all of the annotation attributes match for all constituents. If none of the models contain a given tree fragment we introduce a rule that copies the tree fragment with probability one.

Two types of out-of-vocabulary problems can occur and the strategy of adding copy rules provides robustness against both. In the first, an input contains a tree fragment whose structure has never been seen in training. In this case, copy rules allow the structure to be reproduced, leaving the system to make more informed changes lower down in the tree. In the second, the input contains an unknown word. This only affects transduction at the leaves of the tree since at the lower backoff levels nodes are not annotated with words. Adding copy rules allows the program to retain, replace, or delete unseen words based only on the probabilities of rules higher up for which it does have estimates. In both cases, the added copy rules make sure that any input tree will have an output.

3.4 Decoding and Reranking

Given a parsed sentence to simplify and the probabilistic STSG grammar, the last step is to find the most likely transduction (i.e. simplification) of the input tree based on the grammar. To accomplish this, we convert the STSG grammar into an equivalent finite tree-to-tree transducer: each STSG

Model 1: VP (VB$_0$ PP (IN(in) NP$_1$)) $\rightarrow$ VP (VB$_0$ NP$_1$)
Model 2: VPˆVP (VBˆVP$_0$ PPˆVP (INˆPP (in) NPˆPP$_1$)) $\rightarrow$ VPˆS (VBˆVP$_0$ NPˆVP$_1$)
Model 3: VP[VB]ˆVP (VBˆVP$_0$ PP[NNS]ˆVP (INˆPP (in) NP[NNS]ˆPP$_1$)) $\rightarrow$
VP[VB]ˆS (VBˆVP$_0$ NP[NNS]ˆVP$_1$)
Model 4: VP[VB-assemble]ˆVP (VB[assemble]ˆVP$_0$ PP[NNS-packs]ˆVP (IN[in]ˆPP (in) NP[NNS-packs]ˆPP$_1$)) $\rightarrow$
VP[VB-join]ˆS (VB[join]ˆVP$_0$ NP[NNS-packs]ˆVP$_1$)

Figure 3: The four levels of rule augmentation for an example rule ranging from Model 1 with no additional annotations to Model 4 with all annotations. The head child and head child part of speech are shown in square brackets and the parent constituent is annotated with ˆ.

grammar rule represents a state transition and is weighted with the grammar rule's probability. We then use the Tiburon tree automata package (May and Knight, 2006) to apply the transducer to the parsed sentence. This yields a weighted regular tree grammar that generates every output tree that can result from rewriting the input tree using the transducer. The probability of each output tree in this grammar is equal to the product of the probabilities of all rewrite rules used to produce it.

Using this output regular tree grammar and Tiburon, we generate the 10,000 most probable output trees for the input parsed sentence. We then rerank this candidate list based on a log-linear combination of features:

- The simplification probability based on the STSG backoff model.
- The probability of the output tree's yield, as given by an n-gram language model trained on the simple side of the training corpus using the IRSTLM Toolkit (Federico et al., 2008).
- The probability of the sequence of the part of speech tags in the output tree, as given by an n-gram model trained on the part of speech tags of the simple side of the training corpus.
- A two-sided length penalty decreasing the score of output sentences whose length, normalized by the length of the input, deviates from the training corpus mean, found empirically to be 0.85.

The first feature represents the simplification likelihood based on the STSG grammar described above. The next two features ensure that outputs are well-formed according to the language used in Simple English Wikipedia. Finally, the length penalty is used to prevent both over-deletion and over-insertion of out-of-source phrases. In addition, the length feature mean could be reduced or increased to encourage shorter or longer simplifications if desired.

The weights of the log-linear model are optimized using random-restart hill-climbing search (Russell and Norvig, 2003) to maximize BLEU (Papineni et al., 2002) on a development set.[2]

4 Experiment Setup

To train and evaluate the systems we used the data set from Coster and Kauchak (2011b) consisting of 137K aligned sentence pairs between Simple English Wikipedia and English Wikipedia. The sentences were parsed using the Berkeley Parser (Petrov and Klein, 2007) and the word alignments determined using Giza++ (Och and Ney, 2000). We used 123K sentence pairs for training, 12K for development and 1,358 for testing.

We compared our system (**SimpleTT** – simple tree transducer) to three other simplification approaches:

T3: Another STSG-based approach (Cohn and Lapata, 2009). Our approach shares similar constituent alignment and rule extraction algorithms, but our approach differs in that it is generative instead of discriminative, and T3 increases rule specificity by increasing rule depth, while we employ a backoff model based on grammar augmentation. In addition, we employ n-best reranking based on a log-linear model that incorporates a number of additional features.

The code for T3 was obtained from the authors.[3] Due to performance limitations, T3 was only trained on 30K sentence pairs. T3 was run on the full training data for two weeks, but it never terminated and required over 100GB of memory. The slow algorithmic step is the discriminative training, which cannot be easily parallelized. T3 was tested for increasing amounts of data up to

[2]BLEU was chosen since it has been used successfully in the related field of machine translation, though this approach is agnostic to evaluation measure.

[3]`http://staffwww.dcs.shef.ac.uk/people/T.Cohn/t3/`

30K training pairs and the results on the automatic evaluation measures did not improve.

Moses-Diff: A phrase-based approach based on the Moses machine translation system (Koehn et al., 2007) that selects the simplification from the 10-best output list that is most *different* from the input sentence (Wubben et al., 2012). Moses-Diff has been shown to perform better than a number of recent syntactic systems including Zhu et al. (2010) and Woodsend and Lapata (2011).

Moses-Del: A phrase-based approach also based on Moses which incorporates phrasal deletion (Coster and Kauchak, 2011b). The code was obtained from the authors.

For an additional data point to understand the benefit of the grammar augmentation, we also evaluated a deletion-only system previously used for text compression and a variant of that system that included the grammar augmentation described above. **K&M** is a synchronous context free grammar-based approach (Knight and Marcu, 2002) and **augm-K&M** adds the grammar augmentation along with the four backoff levels.

There are currently no standard evaluation metrics for text simplification. Following previous work (Zhu et al., 2010; Coster and Kauchak, 2011b; Woodsend and Lapata, 2011; Wubben et al., 2012) we evaluated the systems using automatic metrics to analyze different system characteristics and human evaluations to judge the system quality.

Automatic Evaluation

- *BLEU* (Papineni et al., 2002): BLEU measures the similarity between the system output and a human reference and has been used successfully in machine translation. Higher BLEU scores are better, indicating an output that is more similar to the human reference simplification.
- *Oracle BLEU*: For each test sentence we generate the 1000-best output list and greedily select the entry with the highest sentence-level BLEU score. We then calculate the BLEU score over the entire test set for all such greedily selected sentences. The oracle score provides an analysis of the generation capacity of the model and gives an estimate of the upper bound on the BLEU score attainable through reranking.
- *Length ratio*: The ratio of the length of the original, unsimplified sentence and the system simplified sentence.

Human Evaluation

Following previous work (Woodsend and Lapata, 2011; Wubben et al., 2012) we had humans judge the three simplification systems and the human simplifications from Simple English Wikipedia (denoted **SimpleWiki**)[4] based on three metrics: simplicity, fluency and adequacy. Simplicity measures how simple the output is, fluency measures the quality of the language and grammatical correctness of the output, and adequacy measures how well the content is preserved. For the fluency experiments, the human evaluators were just shown the system output. For simplicity and adequacy, in addition to the system output, the original, unsimplified sentence was also shown. All metrics were scored on a 5-point Likert scale with higher indicating better.

We used Amazon's Mechanical Turk (MTurk)[5] to collect the human judgements. MTurk has been used by many NLP researchers, has been shown to provide results similar to other human annotators and allows for a large population of annotators to be utilized (Callison-Burch and Dredze, 2010; Gelas et al., 2011; Zaidan and Callison-Burch, 2011).

We randomly selected 100 sentences from the test set where all three systems made some change to the input sentence. We chose sentences where all three systems made a change to focus on the *quality* of the simplifications made by the systems. For each sentence we collected scores from 10 judges, for each of the systems, for each of the three evaluation metrics (a total of 100*10*3*3 = 9000 annotations). The scores from the 10 judges were averaged to give a single score for each sentence and metric. Judges were required to be within the U.S. and have a prior acceptance rate of 95% or higher.

5 Results

Automatic evaluation

Table 1 shows the results of the automatic evaluation metrics. SimpleTT performs significantly better than T3, the other STSG-based model, and obtains the second highest BLEU score behind only Moses-Del. SimpleTT has the highest oracle BLEU score, indicating that the syntactic model of SimpleTT allows for more diverse simplifications

[4]T3 was not included in the human evaluation due to the very poor quality of the output based on both the automatic measures and based on a manual review of the output.

[5]`https://www.mturk.com/`

System	BLEU	Oracle	Length Ratio
SimpleTT	0.564	**0.663**	0.849
Moses-Diff	0.543	–[*]	0.960
Moses-Del	**0.605**	0.642	0.991
T3	0.244	–[**]	0.581
K&M	0.406	0.602	0.676
augm-K&M	0.498	0.609	0.826
corpus mean	–	–	0.85

Table 1: Automatic evaluation scores for all systems tested and the mean values from the training corpus. [*]Moses-Diff uses the n-best list to choose candidates and therefore is not amenable to oracle scoring. [**]T3 only outputs the single best simplification.

	simplicity	fluency	adequacy
SimpleWiki	3.45	3.93	3.42
SimpleTT	3.55	3.80	3.09
Moses-Diff	3.07	3.64	3.91
Moses-Del	3.19	3.74	3.86

Table 2: Human evaluation scores on a 5-point Likert scale averaged over 100 sentences.

than the phrase-based models and may be more amenable to future reranking techniques. SimpleTT also closely matches the in-corpus mean of the length ratio seen by human simplifications, though this can be partially explained by the length penalty in the log-linear model.

Moses-Del obtains the highest BLEU score, but accomplishes this with only small changes to the input sentence: the length of the simplified sentences are only slightly different from the original (a length ratio of 0.99). Moses-Diff has the lowest BLEU score of the three simplification systems and while it makes larger changes than Moses-Del it still makes much smaller changes than SimpleTT and the human simplifications.

T3 had significant problems with over-deleting content as indicated by the low length ratio which resulted in a very low BLEU score. This issue has been previously noted by others when using T3 for text compression (Nomoto, 2009; Marsi et al., 2010).

The two deletion-only systems performed worse than the three simplification systems. Comparing the two systems shows the benefit of the grammar augmentation: augm-K&M has a significantly higher BLEU score than K&M and also avoided the over-deletion that occurred in the original K&M system. The additional specificity of the rules allowed the model to make better decisions for which content to delete.

Human evaluation

Table 2 shows the human judgement scores for the simplification approaches for the three different metrics averaged over the 100 sentences and Table 3 shows the pairwise statistical significance calculations between each system based on a two-tailed paired t-test. Overall, SimpleTT performed well with simplicity and fluency scores that were comparable to the human simplifications. SimpleTT was too aggressive at removing content, resulting in lower adequacy scores. This phenomena was also seen in the human simplifications and may be able to be corrected in future variations by adjusting the sentence length target.

The human evaluations highlight the trade-off between the simplicity of the output and the amount of content preserved. For simplicity, SimpleTT and the human simplifications performed significantly better than both the phrase-based systems. However, simplicity does come with a cost; both SimpleTT and the human simplifications reduced the length of the sentences by 15% on average. This content reduction resulted in lower adequacy than the phrase-based systems. A similar trade-off has been previously shown for text compression, balancing content versus the amount of compression (Napoles et al., 2011).

For fluency, SimpleTT again scored similarly to the human simplifications. SimpleTT performed significantly better than Moses-Diff and slightly better than Moses-Del, though the difference was not statistically significant.

As an aside, Moses-Del performs slightly better than Moses-Diff overall. They perform similarly on adequacy and Moses-Del performs better on simplicity and Moses-Diff performs worse relative to the other systems on fluency.

Qualitative observations

SimpleTT tended to simplify by deleting prepositional, adjective, and adverbial phrases, and by truncating conjunctive phrases to one of their conjuncts. This often resulted in outputs that were syntactically well-formed with only minor information loss, for example, it converts

> *"The Haiti national football team is the national team of Haiti and is controlled by the Fédération Haïtienne de Football."*

to

Simplicity

	SimpleWiki	Moses-Diff	Moses-Del
SimpleTT		⇐⇐⇐	⇐⇐⇐
SimpleWiki	■	⇐⇐⇐	⇐⇐⇐
Moses-Diff	■	■	⇑

Fluency

	SimpleWiki	Moses-Diff	Moses-Del
SimpleTT		⇐	
SimpleWiki	■	⇐⇐⇐	⇐
Moses-Diff	■	■	

Adequacy

	SimpleWiki	Moses-Diff	Moses-Del
SimpleTT	⇑⇑	⇑⇑⇑	⇑⇑⇑
SimpleWiki	■	⇑⇑⇑	⇑⇑⇑
Moses-Diff	■	■	

Table 3: Pairwise statistical significance test results between systems for the human evaluations based on a paired t-test. The number of arrows denotes significance with one, two and three arrows indicating $p < 0.05$, $p < 0.01$ and $p < 0.001$ respectively. The direction of the arrow points towards the system that performed better.

> *"The Haiti national football team is the national football team of Haiti."*

which only differs from the human reference by one word.

SimpleTT also produces a number of interesting lexical and phrasal substitutions, including:

football striker	→	*football player*
football defender	→	*football player*
in order to	→	*to*
known as	→	*called*
member	→	*part*

T3, on the other hand, tended to over-delete content, for example simplifying:

> *"In earlier times, they frequently lived on the outskirts of communities, generally in squalor."*

to just

> *"A lived"*.

As we saw in the automatic evaluation results, the phrase-based systems tended to make fewer changes to the input and those changes it did make tended to be more minor. Moses-Diff was more aggressive about making changes, though it was more prone to errors since the simplifications chosen were more distant from the input sentence than other options in the n-best list.

6 Conclusions and Future work

In this paper, we have introduced a new probabilistic STSG approach for sentence simplification, SimpleTT. We improve upon previous STSG approaches by: 1) making the model probabilistic instead of discriminative, allowing for an efficient, unified framework that can be easily interpreted and combined with other information sources, 2) increasing the model specificity using four levels of grammar annotations combined into a single model, and 3) incorporating n-best list reranking combining the model score, language model probabilities and additional features to choose the final output. SimpleTT performs significantly better than previous STSG formulations for text simplification. In addition, our approach was rated by human judges similarly to human simplifications in both simplicity and fluency and it scored better than two state-of-the-art phrase-based sentence simplification systems along many automatic and human evaluation metrics.

There are a number of possible directions for extending the capabilities of SimpleTT and related systems. First, while some sentence splitting can occur in SimpleTT due to sentence split and merge examples in the training data, SimpleTT does not explicitly model this. Sentence splitting could be incorporated as another probabilistic component in the model (Zhu et al., 2010). Second, in this work, like many previous researchers, we assume Simple English Wikipedia as our target simplicity level. However, the difficulty of Simple English Wikipedia varies across articles and there are many domains where the desired simplicity varies depending on the target consumer. In the future, we plan to explore how varying algorithm parameters (for example the length target) affects the simplicity level of the output. Third, one of the benefits of SimpleTT and other probabilistic systems is they can generate an n-best list of candidate simplifications. Better reranking of output sentences could close this gap across all these systems, without requiring deep changes to the underlying model.

References

Or Biran, Samuel Brody, and Noemie Elhadad. 2011. Putting it simply: A context-aware approach to lexical simplification. In *Proceedings of ACL*.

Chris Callison-Burch and Mark Dredze. 2010. Creating speech and language data with Amazon's Mechanical Turk. In *Proceedings of NAACL-HLT Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk.*

John Carroll, Gido Minnen, Yvonne Canning, Siobhan Devlin, and John Tait. 1998. Practical simplification of English newspaper text to assist aphasic readers. In *Proceedings of AAAI Workshop on Integrating AI and Assistive Technology*.

Raman Chandrasekar and Bangalore Srinivas. 1997. Automatic induction of rules for text simplification. In *Knowledge Based Systems*.

David Chiang. 2006. An introduction to synchronous grammars. Part of a tutorial given at ACL.

Trevor Cohn and Mirella Lapata. 2009. Sentence compression as tree transduction. *Journal of Artificial Intelligence Review*.

Michael Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.

William Coster and David Kauchak. 2011a. Learning to simplify sentences using Wikipedia. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*.

William Coster and David Kauchak. 2011b. Simple English Wikipedia: A new text simplification task. In *Proceedings of ACL*.

Jason Eisner. 2003. Learning non-isomorphic tree mappings for machine translation. In *Proceedings of ACL*.

Noemie Elhadad. 2006. Comprehending technical texts: predicting and defining unfamiliar terms. In *Proceedings of AMIA*.

Marcello Federico, Nicola Bertoldi, and Mauro Cettolo. 2008. IRSTLM: An open source toolkit for handling large scale language models. In *Proceedings of Interspeech*, Brisbane, Australia.

Lijun Feng. 2008. Text simplification: A survey. CUNY Technical Report.

Michel Galley and Kathleen McKeown. 2007. Lexicalized Markov grammars for sentence compression. In *Proceedings of HLT-NAACL*.

Hadrien Gelas, Solomon Teferra Abate, Laurent Besacier, and Francois Pellegrino. 2011. Evaluation of crowdsourcing transcriptions for African languages. In *Interspeech*.

Kevin Knight and Daniel Marcu. 2002. Summarization beyond sentence extraction: a probabilistic approach to sentence compression. *Artificial Intelligence*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of ACL*.

Erwin Marsi, Emiel Krahmer, Iris Hendrickx, and Walter Daelemans. 2010. On the limits of sentence compression by deletion. In *Empirical Methods in NLG*.

Jonathan May and Kevin Knight. 2006. Tiburon: A weighted tree automata toolkit. In *Proceedings of CIAA*.

Makoto Miwa, Rune Saetre, Yusuke Miyao, and Jun'ichi Tsujii. 2010. Entity-focused sentence simplification for relation extraction. In *Proceedings of COLING*.

Courtney Napoles, Benjamin Van Durme, and Chris Callison-Burch. 2011. Evaluating sentence compression: pitfalls and suggested remedies. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*.

Tadashi Nomoto. 2009. A comparison of model free versus model intensive approaches to sentence compression. In *Proceedings of EMNLP*.

Franz Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of ACL*.

Kishore Papineni, Kishore Papineni, Salim Roukos, Salim Roukos, Todd Ward, Todd Ward, Wei jing Zhu, and Wei jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of ACL*.

Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Proceedings of HTL-NAACL*.

Stuart Russell and Peter Norvig. 2003. Artificial intelligence: A modern approach.

David Vickrey and Daphne Koller. 2008. Sentence simplification for semantic role labeling. In *Proceedings of ACL*.

Kristian Woodsend and Mirella Lapata. 2011. Learning to simplify sentences with quasi-synchronous grammar and integer programming. In *Proceedings of EMNLP*.

Sander Wubben, Antal van den Bosch, and Emiel Krahmer. 2012. Sentence simplification by monolingual machine translation. In *Proceedings of ACL*.

Elif Yamangil and Rani Nelken. 2008. Mining wikipedia revision histories for improving sentence compression. In *Proceedings of HLT-NAACL*.

Elif Yamangil and Stuart Shieber. 2010. Bayesian synchronous tree-substitution grammar induction and its application to sentence compression. In *Proceedings of ACL.*

Mark Yatskar, Bo Pang, Cristian Danescu-Niculescu-Mizil, and Lillian Lee. 2010. For the sake of simplicity: Unsupervised extraction of lexical simplifications from Wikipedia. In *Proceedings of HLT-NAACL.*

Omar F. Zaidan and Chris Callison-Burch. 2011. Crowdsourcing translation: Professional quality from non-professionals. In *Proceedings of ACL.*

Zhemin Zhu, Delphine Bernhard, and Iryna Gurevych. 2010. A monolingual tree-based translation model for sentence simplification. In *Proceedings of ICCL.*