Tree Transduction as a Means for Efficient Sentence Simplification

ALEKSANDRA SIDEROVA, University of Twente, The Netherlands

The use of natural language descriptions as a medium for artists and designers to implement text-based interfaces shows promising results and, as such, this paper examines the possibility of using sentence simplification to accomplish that specific goal. In particular, it has the aim of examining the effectiveness of using tree transduction to remove visually irrelevant tokens from a given sentence. Said goal is accomplished through running trials using three LSTM models - for monolingual machine translation, POS tags and tree transduction. The results show that the tree transduction model has the best performance for the metrics of simplicity, grammatical correctness and preservation of meaning, while the POS tags model performs best in terms of efficiency.

Additional Key Words and Phrases: Sentence simplification, natural language processing, animation, text-based interface, tree transduction, POS tags.

1 INTRODUCTION

Natural languages provide an intuitive, innate means for communicating and conveying concepts, including the ideas of designers and artists. As such, an alternative approach to the traditional use of graphical user interfaces has been proposed, namely the possibility of employing text-based interfaces in design applications as a means of visualizing conceptual ideas through the use of natural language [14].

The benefits of such user interfaces, as well as text in general compared to graphical media, have been investigated by researchers prior to the writing of this paper, with findings showing that textual stimuli can significantly influence originality, compared to the presence of exclusively visual stimuli [12]. Additionally, research has shown that text-based interfaces can improve control when animating movement, can expedite the learning process for inexperienced users and they have the potential to bolster creativity when used appropriately [17].

Considering all of the above, text-based interfaces show promising results for furthering the creative vision and control of artists, while simultaneously providing a means to overcome the challenges faced by those who have little experience interacting with technology [20], especially one that is unfamiliar to them, without allowing low literacy (computer or otherwise) and limitations of the medium to interfere with the creative process.

To stimulate the creation of such interfaces, and in particular for animating natural language text for a system [7], such as the one shown in Figure 5, sentence simplification is a necessary process. This is largely due to the fact that such a system uses an SPO module with the aim of transforming natural language text into OOAL (Object-Oriented Animation Language) to generate visuals. This module breaks down sentences into a subject-predicate-object structure, however, it is limited by its inability to process complex sentences. As such, the use of sentence simplification, in such a way that is both efficient and tailored to the task of transforming natural language descriptions into a visual representation, is essential and it is the topic of this paper.

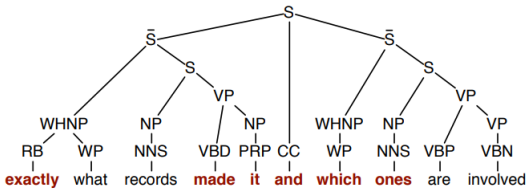

Fig. 1. Part of a sentence, namely "exactly what records made it and which ones are involved", represented as a tree. Source: Cohn and Lapata [2009].

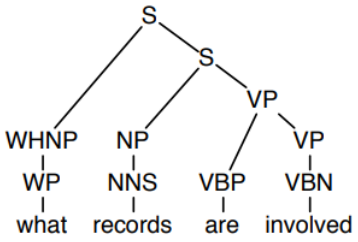

Fig. 2. The sentence shown in Figure 1 after undergoing the process of simplification. Source: Cohn and Lapata [2009].

2 BACKGROUND

This section will describe in more detail terms and tools that will be used to conduct the research and experiments to benefit the understanding of the reader.

2.1 Tree Transduction & Monolingual Machine Translation

Tree transduction is a method that can be used for simplifying sentences, which it accomplishes by representing the components of those sentences as their respective parts of speech (from now on referred to as POS tags), such as noun phrase (NP), verb phrase (VP), etc. [24] and showing their relationships within the sentence by constructing a tree as shown in Figure 1. The figure shows part of a sentence whose components have been tagged. As an example, the phrase "exactly what" is marked as a WHNP (wh-noun phrase), consisting of an RB (adverb) "exactly" and a WP (wh-pronoun) "what" [10].

Using this tree representation, leaves and subtrees can be removed, resulting in a simplified sentence, an example of which can be seen in Figure 2. The inclusion of the dependencies between the different words in the sentence are what makes this method especially promising.

TScIT 37, July 8, 2022, Enschede, The Netherlands
© 2022 University of Twente, Faculty of Electrical Engineering, Mathematics and Computer Science.
Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

The method that tree transduction will be compared to in terms of performance and effectiveness is that of monolingual machine translation. This method works by treating the task of sentence simplification as a monolingual (in the case of this research English-to-English) translation problem from the original complex sentence to its simplified equivalent.

2.2 LSTM

The comparison is done through the widely-used Long Short-Term Memory (LSTM) method, which is capable of solving complex problems, including in the field of natural language processing for machine translation, efficiently and effectively [15], making it a suitable choice for implementing the models.

LSTM is a type of recurrent neural network (RNN) which alleviates the shortcomings of RNNs, such as vanishing gradients, as well as the inability to remember long-term dependencies. It does so by using an efficient gradient-based algorithm, three multiplicative gate units (Forget, Input and Output) and enforcing constant error flow [15]. The structure of an LSTM cell can be seen in Figure 3 [16].

A vanilla RNN and GRU model were also considered for this research, however, ultimately LSTM was decided upon due to the fact that, unlike RNNs, such a structure is capable of remembering long-term dependencies. This is an advantage when it comes to using sequences, which is the case when working with sentences, as each word can depend on the words that come before it. For example, if a sentence starts with the pronoun "We", it follows that any verb in the remainder of the sentence relating to that pronoun must take on its plural form. Additionally, LSTM is preferred to GRU when it comes to obtaining results with a greater accuracy for relatively complex inputs [8].

3 RELATED WORK

Some of the more significant research that is relevant to this paper done on the topics of sentence simplification and tree transduction is presented in this section.

Hassani and Lee [2016] provide a comprehensive survey on systems that make use of natural languages, as well as the problems and requirements present when developing such systems.

Hochreiter and Kepler [1997] introduce in their paper the method of Long Short-Term Memory (LSTM) to overcome the drawbacks of previous recurrent neural networks. Their algorithm is used as a basis for comparing the models in this research paper.

Cohn and Lapata [2009] have formulated an algorithm capable of transforming sentences into trees, which are then used to simplify sentences by methods of not only deletion, but also more complex operations such as substitution and reordering due to the use of synchronous grammar. Furthermore, Feblowitz and Kauchak [2013] have developed a similar model, which performs considerably better than that of Cohn and Lapata.

Zhang and Lapata [2017] present a model for sentence simplification using deep reinforcement learning, which achieves promising results.

Bacciu and Bruno [2018] propose in their paper a deep-learning model which learns a structure-to-substructure transduction model

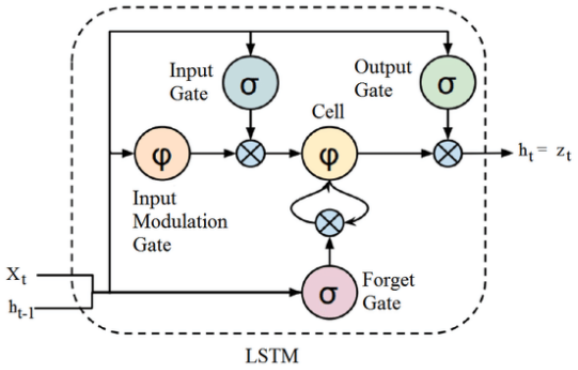

Fig. 3. Diagram showing the structure of an LSTM cell. Source: Kang [2017].

that extends LSTM by taking into consideration the relationships between the nodes in the generated trees. As such, this model is closely related to the work done in this paper.

Alva-Manchego, Scarton and Specia [2020] outline the primary methods for achieving sentence simplification and make a comparison between existing approaches.

4 PROBLEM STATEMENT

Sentence simplification is a topic that has been worked on extensively and has various approaches that have already been designed, tested and shown to work reasonably well [2].

Currently used methods, however, have their limitations. For example, using sequences to represent sentences in natural language can remove a word from its context, which could potentially result in an interpretation that is not entirely relevant to the expected output.

On the other hand, past research has shown that tree transduction can be an effective method for accomplishing the aforementioned task, taking into consideration that sentences represented as tree structures are capable of properly conveying the relationships between the various words in a sentence by using their corresponding parts of speech and the dependencies that exist between them [10].

Considering the promising results this method has shown, this paper has the aim to verify or disprove the effectiveness of tree transduction used in sentence simplification, specifically for the purpose of creating visual representations of text.

This goal differs from general sentence simplification, due to the fact that an overall simplification could include transformations that are not relevant for this specific task, such as replacement of a word with a synonym. Sentence simplification for visuals, on the other hand, does not benefit from such transformations and instead requires visually irrelevant tokens to be removed so the input can be used to generate the image described, rather than incorrectly focusing on the irrelevant aspects of the sentence, such as past tense descriptions that are outdated and, as such, should not be part of the resulting visualization. Hence, this research involves task-oriented sentence simplification with a focus on removing visually irrelevant tokens.

4.1 Research Question

With all of the above taken into consideration, the goal of this research paper is to perform an experiment with the aim of checking whether the inclusion of tree transduction in the implementation of sentence simplification for visual media is an effective improvement upon currently existing algorithms, some of which are elaborated on in Section 3.

This question can be answered by addressing the following two sub-topics, ordered by importance, and their respective sub-questions:

- Correctness – does the use of tree transduction result in a sentence simplified through the removal of visually irrelevant tokens, and if so, to what extent?
- Efficiency – does the use of tree transduction have a significant effect on the performance of sentence simplification algorithms and, if yes, is that effect positive or negative?

Thus, this research aims to explore the effectiveness of tree transduction for removing visually irrelevant tokens in order to generate images.

Additionally, should tree transduction prove to be an improvement over existing methods, such results could lead to the implication that visually descriptive sentences potentially have a distinct structure and relationships between the different parts of speech in said sentences.

5 METHODOLOGY

In the following section, the tools and methods used to answer the research question outlined in Section 4.1 are described in detail.

For the purposes of estimating whether tree transduction shows an improvement over machine translation, three models are implemented and compared - one for monolingual machine translation, one that uses only the POS tags of the words in the sentence, and, finally, one that uses both the POS tags and the dependencies between the words. By making the distinction between using POS tags only and tree transduction, the aim is to investigate whether the inclusion of relationships and the structure of the tree make a difference in the performance of the model.

5.1 Data

The primary dataset for the trials will be based on pairs of sentences, original and simplified, taken from a children's blog, due to the simplicity of the text and its visually descriptive nature. The original source of the dataset is the University of Sheffield.

Only very minor cleaning had to be done to the data by removing sentences that are empty or otherwise consisting only of punctuation marks. This decision was made due to the fact that during the experimental stage, the BLEU metric used for evaluation (described in more detail in Section 5.8) requires all data to be non-empty, while the algorithm itself cleans sentences by removing punctuation, meaning that the sentences such as those described above are transformed into empty strings during the task of processing. Additionally, sentences comprised only of punctuation are not relevant for the task of training a model for sentence simplification of visual descriptions, and therefore there is no reason to keep them when they interfere with the algorithm.

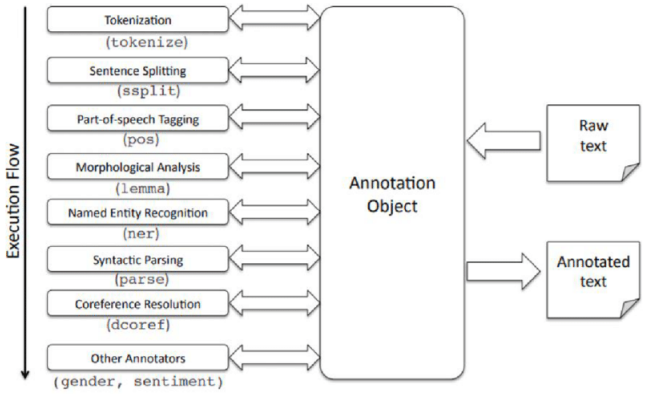

Fig. 4. The pipeline demonstrating the process of Stanford CoreNLP. Source: Manning et al. [2014].

Additionally, sentences presenting speech and dialogue were also removed, due to the fact that their visual representation should remain unchanged, meaning that there is no reason to simplify them. Such sentences were a fairly small proportion of the overall dataset, so their removal did not decrease its size by any significant amount.

The dataset consists of 4185 entries after undergoing the process of cleaning. These pairs of sentences are used to the train three models, one using monolingual machine translation, one using POS tags only and one using tree transduction (i.e. including the dependencies between the words), described in more detail in the following subsections.

The data is split into a training set and a test set, with the test set consisting of 250 entries, while the remaining data is used for training.

In order to mitigate the long waiting time during the tree parsing process, which is the biggest bottleneck time-wise, each sentence in the dataset is parsed once and saved to a binary file using the built-in Pickle library for Python. Whenever the tree structures are needed during the running of the algorithm, the file is read and the data structures are extracted, which is considerably less time-consuming than parsing the tree for each instance of the algorithms. In terms of time measurement, omitting the parsing time makes no difference for visual generation systems, due to the fact that parsing always has to be done for the SPO module, regardless of the method used to simplify the sentences, as seen in Figure 5.

5.2 Stanford CoreNLP

As part of processing the data in order to generate its corresponding trees, this research utilizes the Stanford CoreNLP toolkit, which is widely used for natural language processing with the purpose of annotating text [18]. The process used by the toolkit can be seen in Figure 5.2.

The text is annotated with its corresponding POS tags, which can be divided into clause level, phrase level and word level tags [5], where the word level tags correspond to tags of the leaf level of the tree, while the rest are for the upper levels of the tree, a

Learning rate	SARI score	BLEU score
1e-2	25.179	0.023
1e-3	24.660	0.006
1e-4	24.062	0.000
1e-1	23.874	0.003
1e-6	23.387	0.003
1e-0	23.252	0.001
1e-5	23.072	0.000

Table 1. Results for the learning rate of the monolingual machine translation model.

Learning rate	SARI score	BLEU score
1e-2	28.185	0.428
1e-3	27.464	0.426
1e-4	26.052	0.229
1e-1	23.853	0.001
1e-5	23.470	0.009
1e-6	23.148	0.005
1e-0	23.116	0.007

Table 2. Results for the learning rate of the POS tags model.

Learning rate	SARI score	BLEU score
1e-2	27.870	0.550
1e-3	27.741	0.355
1e-4	25.311	0.096
1e-1	25.073	0.050
1e-6	23.635	0.014
1e-0	23.244	0.000
1e-5	23.072	0.000

Table 3. Results for the learning rate of the tree transduction model.

distinction which proved to be useful for the construction of the tree transduction model.

5.3 Parameters

Hyperparameter tuning is a crucial aspect of the performance of a neural network, including LSTM, so in order to find the most suitable and efficient parameter configuration for the task, the settings recommended by related research were used, in particular for the parameters that are noted to have the largest impact on performance.

According to literature [22], the ideal optimizer for LSTM is either Adam or Nadam, with preference usually given to Nadam due to being the optimizer that converges fastest of the two, at approximately 10 epochs. Additionally, gradient normalization is applied with a threshold $\tau = 1$. The recommended batch size for relatively small datasets, as is the case for this research, is 8.

The choice of activation function was between CRF and softmax, but due to time constraints softmax was chosen despite being the second best option, with more information about the reasoning behind this decision in Section 5.9. The softmax activation function is paired with a sparse categorical cross-entropy function [13].

For the learning rate, there is no agreed upon value that suits any model and, as such, it has to be found through trial and error [21], with the recommended range of values being from 1e-6 to 1 [4]. To find a suitable learning rate, trials were run to compare the results using multiple possible values in the range mentioned above. The results (ordered by highest SARI score) rounded to three decimals for the monolingual machine translation can be found in Table 1. The equivalent scores for the POS tags and tree transduction can be found in Tables 2 and 3, respectively.

The results show that despite the varying methods, all three models have fairly similar rankings. It is worth pointing out that the learning rate was tested using smaller subsets of the dataset, unlike the experiments in Section 6, hence the varying numbers.

For all models, a learning rate of 1e-2 resulted in the best performance with regard to both metrics, and for that reason it was chosen as the parameter for the experiments.

5.4 General Model

In this subsection, elements that are common to all three models are described, while their distinctive aspects are defined in later subsections.

In order to carry out the experiments, an LSTM model is needed. The base model was created [6] and then adjusted accordingly so as to be suited for monolingual machine translation and tree transduction, depending on the purposes of each model. The modifications are described in Section 5.3, Section 5.6 and Section 5.7. The models were implemented in Python using the Keras library, which itself runs on top of Tensorflow and is used for developing deep learning models [9]. The implementation was run through the JupyterLab environment of the University of Twente.

The dataset is modified to fit the tab-delimited text file format used by the algorithm, although no other adjustment of the data is necessary in advance, other than what is described in 5.1. The data is parsed and the sentences are cleaned by transforming them into lowercase and removing punctuation. The resulting sentences are then tokenized, padded and reshaped.

The Input layer is defined using as shape the length of the longest non-simplified sentence. The process continues by creating layers for Embedding, LSTM, RepeatVector, another LSTM layer and finally a TimeDistributed layer.

The layers used are for Input, Embedding, LSTM, RepeatVector, another LSTM layer and finally a TimeDistributed layer. The remaining parameters are described in Section 5.3.

5.5 Monolingual Machine Translation Model

The monolingual machine translation model did not require many modifications. It simply takes the tokenized sentences as input for training and makes its predictions in the form of regular sentences.

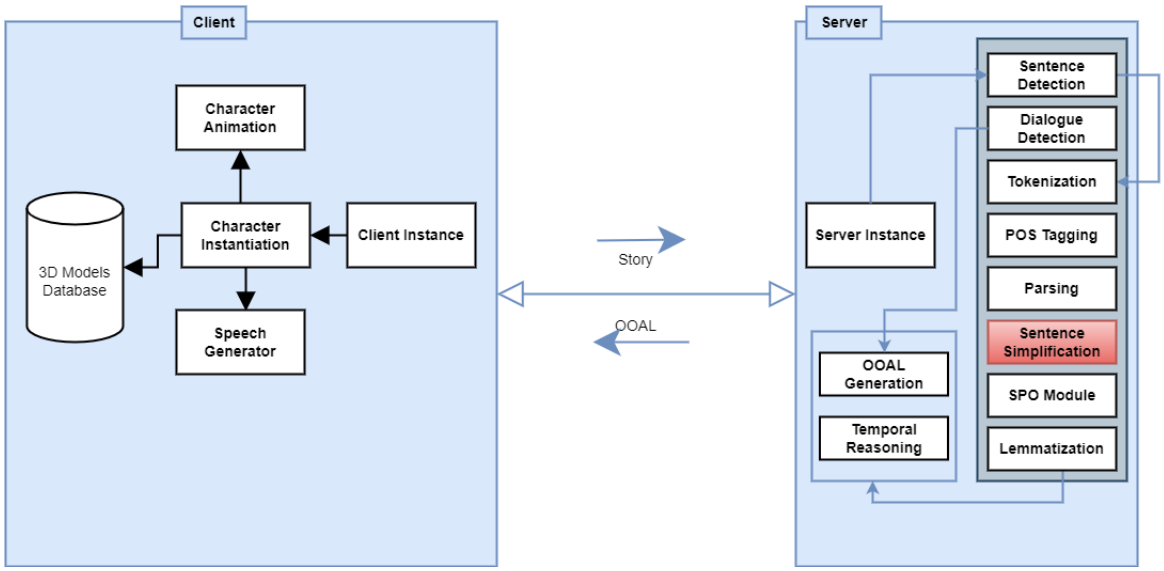

Fig. 5. An overview of a Virtual Reality generation system with the relevant component highlighted in red. Source: Bouali [2022].

The parameters and configurations used are those described in Section 5.3 and Section 5.4.

5.6 POS Tags Model

Following the training of the monolingual translation model, a model using the tags that mark the parts of speech of each word in the sentence is created. However, it does not use the relationships between the different words as input information.

The process for its implementation is considerably more involved, largely due to the fact that the overall process requires performing transformations on the dataset multiple times, a process that is not necessary for training the monolingual model.

In particular, the overall process involves taking each sentence in the dataset, transforming it into a tree, mapping its parts of speech to their equivalent in the original sentence, training the model on the POS tags and finally retrieving the predicted result from the mapping in order for the result to be properly evaluated as a sentence with regard to simplicity, grammatical correctness and preservation of meaning.

The POS tags model uses the Stanford CoreNLP framework [18], described in more detail in Section 5.2 for the purpose of generating trees from the sentences in the dataset and extracting their POS tags. Each tag is then mapped to its respective word in the original sentence, or, in the case of contractions, multiple tags can be mapped to it as needed. As an example, in the first sentence of the dataset, "It was a weekend so she didn't have to go to school.", the contraction "didn't" is treated as two separate words - "did" and "n't", where each has a corresponding POS tag. So, the sentence has 12 words originally, but 13 POS tags in the tree. To facilitate such a mapping, both the POS tags and the original words as leaves are extracted directly from the tree.

An extra step to the mapping process involves resolving the issue of duplicate POS tags. It's possible, even likely, that a sentence can contain multiple of the same parts of speech. Taking as an example the second sentence from the dataset, "Then he swam in the hole and got stuck in the rocks.", the words "swam" and "got" are both verbs in past tense and would therefore be marked as VBD, but when they are retrieved as predictions, a distinction has to be made between them. The solution for this paper is to go over each POS tag in the sentence and, should there be a duplicate, to append a number to it.

The following step is to pass the list of POS tags to the model, which uses the parameters described in Section 5.3 and 5.4. Afterwards the original words are retrieved from the mapping and a sentence is generated based on the parts of speech predicted by the algorithm. In the case that there exists no mapping for a predicted POS tag in the original sentence, in the output it is represented with the token "unknown".

It is worth noting that due to the mapping, the sentence constructed from the prediction can only contain words that were in the original sentence. Unlike sentence simplification for other purposes, here tokens are only removed from the sentence and no other transformations, such as replacement, are performed. Hence, the use of such a mapping creates no issues.

However, the method used to implement this model doesn't make use all the aspects of the tree representation, since it only takes into consideration the POS tags, while in order to truly utilize the tree structure, it is necessary to be able to represent the relationships between the parts of speech of each word.

5.7 Tree Transduction Model

The final model builds upon the one described in Section 5.6 by using the dependencies between the POS tags during its training. Multiple approaches were considered for this, dealing with the encoding of a tree, which is a problem that is still being worked on.

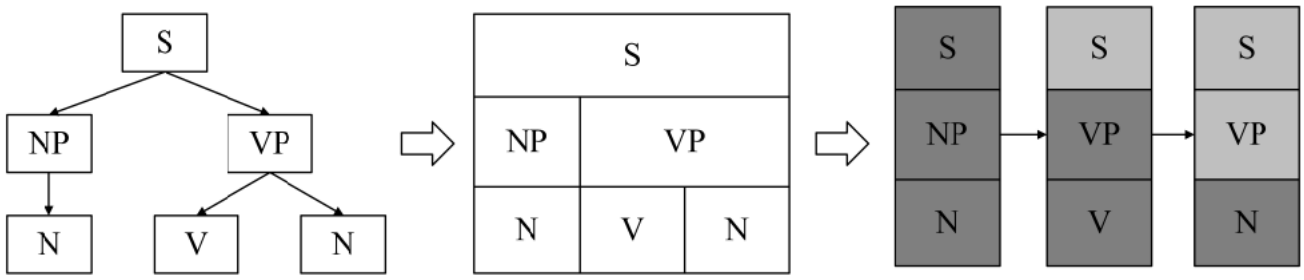

Fig. 6. An example of the correspondence between the parsed tree structure and ON-LSTM cell states. Source: Shen et al. [2018].

```
(ROOT
  (S
    (NP (PRP It))
    (VP
      (VBD was)
      (NP (DT a) (NN weekend))
      (SBAR
        (IN so)
        (S
          (NP (PRP she))
          (VP
            (VBD did)
            (RB n't)
            (VP
              (VB have)
              (S
                (VP
                  (TO to)
                  (VP (VB go) (PP (IN to) (NP (NN school))))))))))
    (. .)))
```

Fig. 7. An example of the string representation of a tree.

One possibility was to use the string representation of a tree, an example of which is depicted in Figure 7, which conveys the different levels of the tree through the use of brackets. Such an implementation was attempted, however, it was ultimately discarded due to the fact that the prediction almost never resulted in a tree structure due to incorrectly placed brackets. For example, the prediction would occasionally end the sequence with an open bracket '(', which is an invalid tree structure. As such, retrieving a sentence from the prediction was not possible in the vast majority of cases.

Additionally, the use of adjacency matrices to represent tree dependencies was briefly considered, however, it was likely a similar problem would occur, in the sense that there is no guarantee that the prediction would be a valid tree structure. For example, it is possible that the resulting adjacency matrix indicates that there is an edge between two sibling nodes, which would not be a tree structure. For that reason, this idea was also discarded without an implementation being attempted.

The solution that was settled on utilizes a method that is used to implement ON-LSTM (ordered neurons LSTM) [23]. The method involves representing the structure of the tree by appending to each leaf-level node the POS tags of its parent and grandparent nodes, up to the root node of the tree. A visual representation of this process can be observed in Figure 6. In order to make the retrieval from the mapping easier, a delimiter '%' is chosen to separate the concatenated POS tags. So, for example, the representation of the tree in Figure 6 would be "N%NP%S V%VP%S N2%VP%S" (N2 is used due to duplicates being handled in the same way as in Section 5.6).

Each sentence is transformed in this way and passed to the model as input. When handling the predictions, which are again in such a form, only the first POS tag of each part of the string (i.e., only the leaves of the tree) is considered. All such tags are taken from the string, their word equivalents are extracted from the mapping and a sentence is constructed.

5.8 Evaluation

The effectiveness of each method will be evaluated through the BLEU (for grammatical correctness and preservation of meaning [25]) and SARI (for evaluating simplicity [25]) metrics, using the EASSE Python package, designed to make evaluation of sentence simplification algorithms standardized and convenient [1].

The BLEU metric outputs results between 0 and 1, with closeness to 1 being an indicator of grammatical correctness and preservation of meaning, though it's unlikely for a simplification to obtain an evaluation of 1 since that would imply that the simplification and the reference are identical [19]. It may also be relevant for this paper to point out that there exists a correlation between the presence of more references per sentence and an increase in the score of the BLEU metric [19], while all sentences in the dataset have only one reference each, which could potentially affect the scoring negatively.

The SARI metric, on the other hand, although not as efficient as BLEU, performs with a considerably higher quality at evaluating simplicity and, similarly to BLEU, a higher score (from 0 to 100) translates to a better end result in terms of simplicity [25].

After all three models are trained, their effectiveness is evaluated by comparing the results of each of them using their BLEU and SARI scores to determine which method is more effective at simplification, grammatical correctness and preservation of meaning.

For efficiency, the three models are compared through measuring the amount of time it takes in seconds for each program to complete its runtime. The time is measured starting from after the import statements up until all metrics are calculated. For the POS tags and tree transduction models, the time it takes to open the Stanford

Model	Average SARI score	Average BLEU score	Average elapsed time (in seconds)
Monolingual Machine Translation	24.979	0.005	156.056
POS tags	33.462	0.033	144.382
Tree transduction	38.712	0.104	156.264

Table 4. Results for the performance of each of the three models.

Model	Lowest SARI score	Lowest BLEU score	Lowest elapsed time (in seconds)
Monolingual Machine Translation	23.981	0.000	147.381
POS tags	33.128	0.000	142.305
Tree transduction	38.316	0.052	147.847

Table 5. Lowest results obtained per metric for each of the three models.

Model	Highest SARI score	Highest BLEU score	Highest elapsed time (in seconds)
Monolingual Machine Translation	26.223	0.007	174.430
POS tags	34.633	0.080	147.832
Tree transduction	40.328	0.220	209.496

Table 6. Highest results obtained per metric for each of the three models.

CoreNLP server is not being considered. Additionally, the time it takes to parse all trees is not taken into account since the parsing of all sentences is only performed once. It is worth pointing out that parsing time will be part of the process in production, however, as mentioned in Section 5.1, parsing is always necessary for such a system, regardless of the method used to simplify the sentences. As such, it can be disregarded when comparing the efficiency of the different approaches.

Additionally, a comparison is made with literature, namely with the work of Alva-Manchego et al. [2019], which presents a benchmark for the evaluation of sentence simplification. The comparison is made with the compiled results for the PWKP dataset instead of the TurkCorpus dataset, since the former uses only one reference per sentence, which is comparable to the structure of the dataset used in this paper, while the latter uses 8 references per sentence [25]. Thus, the results of this paper are more comparable to the PWKP dataset.

5.9 Limitations

The primary limitation of this research is the problem of encoding a tree in such a way that it can feasibly be passed on to the model for training. At the time of the writing of this paper, encoding a tree is a task that is uncommonly performed [3]. As such, it is outside the scope of this research to create such an encoding, which necessitated the use of alternative methods, described in Section 5.7 which might not perform as well.

Another limitation of this research is the relatively small size of the dataset, as it only consists of 4185 entries after cleanup. As a consequence of this, the results obtained through each of the three models are not as accurate as they would be if the dataset was comprised of considerably more entries. As such, the results are not entirely indicative of the capabilities of each algorithm. Regardless, this dataset is relevant to creating visuals and it is big enough to make an informed comparison between the three approaches, so it was deemed suitable for the task.

6 RESULTS

Taking into consideration the proven benefits of tree transduction [10, 11], the progress made in the field of sentence simplification [2, 26], as well as the fact that research combining these subjects with the intent of showing or disproving the efficacy of tree transduction in visual representations is scarce, this paper aims to give a conclusive answer to the research questions described in Section 2 through the results obtained by running experiments with each of the three models described in the above sections.

For each model, 30 experiments were performed and the average, minimum and maximum values were measured in order to make a comparison. The averages can be observed in Table 4, where it can be observed that the tree transduction model scored highest for the BLEU and SARI metrics, but had the worst time performance out of the three models. The POS tags model was the second best for the BLEU and SARI metrics, but was the fastest on average. The monolingual model showed the poorest performance for BLEU and SARI and was second best for elapsed time.

Table 5 shows the lowest scores obtained by each model. Tree transduction again shows the highest results for BLEU and SARI, in the sense that even its lowest scores are still the highest of all three models. However, its fastest time was still the least efficient of the three. The remaining models performed equally poorly for the BLEU metric, while the POS tags model was better than the monolingual one for the other two metrics.

Finally, Table 6 shows the highest scores obtained by each model. Those results further reinforce the notion that the tree transduction model has the best performance for BLEU and SARI and the worst performance time-wise. Notably, the slowest time of the POS tags model was still considerably faster than that of the other models.

Considering the information that can be extracted from the results of the experiments, the tree transduction model showed the best performance for sentence simplification in terms of simplicity, grammatical correctness and preservation of meaning, but it was also the slowest of the three models. The POS tags only model was the fastest of the three, but was not as good at simplifying sentences as the tree transduction model. The monolingual machine translation model performed poorly in terms of simplification and was not as fast as the POS tags model.

Comparing the average results of the tree transduction model to literature [2], it can be observed on Figure 8 that the model's performance can be placed above the UNSUP model, but below the TSM model. However, the size of the dataset, as explained in Section 5.9, should be taken into consideration as an aspect that affects the performance of the tree transduction model.

Model	SARI ↑	BLEU ↑	SAMSA ↑	FKGL ↓
Reference	100.00	100.00	29.91	8.07
Hybrid	**54.67**	53.94	36.04	10.29
Moses	48.99	**55.83**	34.53	11.58
DRESS-LS	40.44	36.32	29.43	8.52
DRESS	40.04	34.53	28.92	8.40
TSM	39.02	37.69	37.39	**6.40**
UNSUP	38.41	38.28	35.81	7.75
PBSMT-R	35.49	46.31	35.63	12.26
QG+ILP	35.24	41.76	**41.71**	7.08
EncDecA	32.26	47.93	35.28	12.12

Fig. 8. The performance of various sentence simplification models on the PWKP dataset. Source: Alva-Manchego et al. [2019].

7 CONCLUSION

In this paper, a comparison was made on the efficiency and efficacy of three methods - monolingual machine translation, POS tags and tree transduction - on sentence simplification for removing visually irrelevant tokens. The results indicate that the tree transduction model is capable of a better performance than the other models in terms of simplification, grammatical correctness and preservation of meaning. However, when a more efficient approach is required, the POS tags model can be more appropriate for the task. The monolingual machine translation model showed poorer results than the other two models and, as such, its usefulness in the field is limited.

These findings can be used as a basis for creating text-based interfaces in an efficient way, as well as for the generation of other visuals such as 3D images, VR, etc. Additionally, the results can be utilized in related research in the field of natural language processing.

8 FUTURE WORK

In this section, a discussion is presented on possible improvements that could be made to the models for future research.

First, it is possible that the tree transduction aspect of the experimental phase could be implemented and carried out in a different, potentially more efficient way. In particular, the possibility of encoding the tree differently and passing it on to the LSTM model exists, however, as mentioned in Section 5.9, to the extent of the author's knowledge, the method of encoding of a tree is an ongoing, still unresolved question, and so implementing it is outside the scope of this research. Should an appropriate method to accomplish tree encoding be developed and demonstrated to work properly, it could serve as basis for further research on the topic.

Another improvement that could be made has to do with the choice of parameters. According to Reimers and Gurevych [2017], the CRF classifier performs better than softmax for tasks that have tag dependencies, which is the case for this task. However, its implementation was judged to be too time-consuming for this research and, as such, could be used as a future improvement.

Another aspect that there was not enough time for is implementing a constraint on the tokens of the predicted output per sentence, in particular for the monolingual machine translation model. The reason being that the other two models effectively place a restriction on which words can be in the final sentence - due to the use of mapping, the words extracted from the POS tags can only be words that were already present in the original sentence. However, such a limitation on the prediction is not used for the monolingual model, as it would require modifying the layers on a lower level with Tensorflow, which is a task outside the time frame of this research.

Additionally, other, albeit smaller, improvements with regard to the efficiency of the POS tags and tree transduction model could be made, in particular during the process of mapping the words of a sentence to its parts of speech and retrieving them afterwards once the stage of prediction has been reached - it may be possible that there are more efficient methods of accomplishing that task.

ACKNOWLEDGMENTS

I would like to thank my supervisor, Nacir Bouali, for giving me the opportunity to work on this topic and for his guidance and assistance during the performance of this research.

REFERENCES

[1] Fernando Alva-Manchego, Louis Martin, Carolina Scarton, and Lucia Specia. 2019. EASSE: Easier Automatic Sentence Simplification Evaluation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*. Association for Computational Linguistics, Hong Kong, China, 49–54. https://doi.org/10.18653/v1/D19-3009

[2] Fernando Alva-Manchego, Carolina Scarton, and Lucia Specia. 2019. Data-Driven Sentence Simplification: Survey and Benchmark. *Computational Linguistics* 46, 1 (2019), 135–187. https://doi.org/10.1162/coli_a_00370

[3] Davide Bacciu and Antonio Bruno. 2018. Text Summarization as Tree Transduction by Top-Down TreeLSTM.

[4] Yoshua Bengio. 2012. Practical recommendations for gradient-based training of deep architectures. *arXiv* (June 2012). https://doi.org/10.48550/arXiv.1206.5533 arXiv:1206.5533

[5] Ann Bies, Mark Ferguson, Karen Katz, and Robert MacIntyre. 2004. Bracketing Guidelines for Treebank II Style Penn Treebank Project. https://www.cis.upenn.edu/~bies/manuals/root.pdf [Online; accessed 24. Jun. 2022].

[6] Nechu BM. 2020. How to build an encoder decoder translation model using LSTM with python and keras. https://towardsdatascience.com/how-to-build-an-encoder-decoder-translation-model-using-lstm-with-python-and-keras-a31e9d864b9b

[7] Nacir Bouali. 2022. Virtual Reality Generation from Natural Language (unpublished). (2022).

[8] Roberto Cahuantzi, Xinye Chen, and Stefan Güttel. 2021. A comparison of LSTM and GRU networks for learning symbolic sequences. *arXiv* (July 2021). https://doi.org/10.48550/arXiv.2107.02248 arXiv:2107.02248

[9] Francois Chollet et al. 2015. *Keras*. https://github.com/fchollet/keras

[10] T. A. Cohn and M. Lapata. 2009. Sentence compression as tree transduction. *Journal of Artificial Intelligence Research* 34 (2009), 637–674. https://doi.org/10.1613/jair.2655

[11] Dan Feblowitz and David Kauchak. 2013. *Sentence Simplification Through Tree Transduction*. Association for Computational Linguisitics, 1–10.

[12] Gabriela Goldschmidt and Anat Litan Sever. 2016. Inspiring design ideas with texts. *Design Studies* 32, 2 (2016), 139–155. https://doi.org/10.1016/j.destud.2010.09.006

[13] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning (Adaptive Computation and Machine Learning series)*. The MIT Press, Cambridge, MA, USA. https://www.amazon.com/Deep-Learning-Adaptive-Computation-Machine/dp/0262035618

[14] Kaveh Hassani and Won-Sook Lee. 2016. Visualizing natural language descriptions. *Comput. Surveys* 49, 1 (2016), 1–34. https://doi.org/10.1145/2932710

[15] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9, 8 (1997), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[16] Eugine Kang. 2017. Long Short-Term Memory (LSTM): Concept. https://medium.com/@kangeugine/long-short-term-memory-lstm-concept-cb3283934359

[17] Sangwon Lee and Jin Yan. 2014. The potential of a text-based interface as a design medium: An experiment in a computer animation environment. *Interacting with Computers* 28, 1 (2014), 85–101. https://doi.org/10.1093/iwc/iwu036

[18] Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CORENLP Natural Language Processing Toolkit. *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations* (2014), 55–60. https://doi.org/10.3115/v1/p14-5010

[19] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu. *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*

(Jul 2002), 311–318. https://doi.org/10.3115/1073083.1073135

[20] Sun Young Park, Brad Myers, and Andrew J. Ko. 2008. Designers' natural descriptions of interactive behaviors. *2008 IEEE Symposium on Visual Languages and Human-Centric Computing* (2008). https://doi.org/10.1109/vlhcc.2008.4639082

[21] Russell Reed and Robert J. Marks I. I. 1999. *Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks (A Bradford Book).* MIT Press, Cambridge, MA, USA. https://www.amazon.com/Neural-Smithing-Supervised-Feedforward-Artificial/dp/0262527014

[22] Nils Reimers and Iryna Gurevych. 2017. Optimal Hyperparameters for Deep LSTM-Networks for Sequence Labeling Tasks. *arXiv* (July 2017). https://doi.org/10.48550/arXiv.1707.06799 arXiv:1707.06799

[23] Yikang Shen, Shawn Tan, Alessandro Sordoni, and Aaron Courville. 2018. Ordered Neurons: Integrating Tree Structures into Recurrent Neural Networks. *arXiv* (Oct. 2018). https://doi.org/10.48550/arXiv.1810.09536 arXiv:1810.09536

[24] Ann Taylor, Mitchell Marcus, and Beatrice Santorini. 2003. The Penn Treebank: An overview. *Treebanks* (Jan 2003), 5–22. https://doi.org/10.1007/978-94-010-0201-1_1

[25] 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 (2016), 401–415. https://doi.org/10.1162/tacl_a_00107

[26] Xingxing Zhang and Mirella Lapata. 2017. *Sentence Simplification with Deep Reinforcement Learning.* Association for Computational Linguistics, 584–594.