Make it Simple: Simplifying English Sentences

Hugo Pitorro Technical University of Munich hugo.pitorro@tum.de
Daniela Amaral Instituto Superior Técnico daniela.amaral@tecnico.ulisboa.pt
Mohammed Saleh Technical University of Munich ge96req@mytum.de

Abstract

Text simplification aims to reduce the complexity of sentences such that they are more readable and understandable. In order to simplify English text, we study different approaches, in particular, we introduce Make it Simple, a model that results from fine-tuning the BART language model and combining it with a controllable mechanism that adjusts sentence-related attributes. Furthermore, we also experimented with direct replacement of complex words to aid the simplification procedure. Finally, our model got close to the current state-of-the-art, with perhaps potential to surpass it, if the future work directions we point out, are to be followed.

1 Introduction

The task of text simplification (TS) is usually posed as: from a source sentence, rewrite it in a way that makes it easier to read, using techniques ranging from lexical simplification to reformulating the grammatical structure of the sentence, paraphrasing wherever need to be. The purpose of this task is to facilitate the reading and understanding of possibly complicated texts for readers who suffer from cognitive disabilities, namely aphasia (Carroll et al., 1998), dyslexia (Rello et al., 2013), autism (Evans et al., 2014), or even second language learners (Paetzold and Specia, 2016).

Furthermore, with governments passing legislation requiring legal and public texts to be available in simplified language (Maaß, 2020), this field is gaining more and more traction as suddenly public entities are forced to translate their entire available information base to a simplified language.

Given that this is an active area of research, our initial focus was on emulating key publications and, possibly, try to build on their work and improve it. During the course of the semester, we have focused mainly on general rewriting of the inputs provided through the fine-tuning of large language models, controlling this rewriting through explicit control tokens and, in a more narrow scope, identifying and replacing possibly lexically complex words in a pre- or post-processing step.

We achieved moderate success in this task as we got close to the state of the art (43.05 SARI (Xu et al., 2016) on the ASSET (Alva-Manchego et al., 2020) dataset). Nonetheless, there is still definitive room for improvement as some topics were left unexplored either due to lack of time, or lack of compute power required to train these large models. Throughout the report, and in a following section on future work, we thoroughly discuss what we consider the bottlenecks in our system to be, and possible ways to tackle them.

2 Related Work

In this section, we describe some of the research conducted. Mainly, in the Sentence Simplification, Controllable Text Generation and Lexical Simplification areas.

2.1 Sentence Simplification

Sentence simplification can be seen as a monolingual machine translation task, where models are trained with aligned pairs of sentences obtained, for example, from Wikipedia articles and their corresponding Simple Wikipedia versions (Zhu et al., 2010; Wubben and Boschand, 2012).

To this purpose, there was some focus on Statistical Machine Translation models (Zhu et al., 2010) but those have since been overcome by large neural models. Among several others: Nisioi et al. (2017) used a vanilla recurrent neural network for text simplification; Zhang and Lapata (2017) combined RNNs and reinforcement learning; Zhao et al. (2018) introduced the transformer architecture for TS and integrated it with a paraphrase dataset (Pavlick and Callison-Burch, 2016); Dong et al. (2019) presented EditNTS, a reinforcement learning model that learns ADD, DELETE and KEEP

operations to simplify text.

2.2 Controllable Text Generation

In the last few years, controllable text generation with conditional training of Seq2Seq models has been applied to different NLP tasks such as summarization (Fan et al., 2018), politeness in machine translation (Sennrich et al., 2016), sentence compression (Fevry and Phang, 2018; Mallinson et al., 2018), along with others.

In the text simplification task, Scarton and Specia (2018) pioneered this idea of controllable tokens to generate simplified sentences by embedding a grade level token into a Seq2Seq model. ACCESS (Martin et al., 2020a) leveraged a similar approach by conditioning the sentences on number of characters, character-level Levenshtein similarity, word frequency and syntactic complexity.

Furthermore, building on ACCESS, Martin et al. presented MUSS (Martin et al., 2020b) an unsupervised multilingual model that fine-tunes a BART language model instead of the original Transformer architecture. Recently, Sheang and Saggion (2021) reiterated this idea, but with the newer T5 (Raffel et al., 2019) model and an additional token encoding the number of words ratio between sentences, becoming the current state of the art.

2.3 Lexical Simplification

Lexical simplification (LS) is the task of identifying complex words and finding the best candidates to replace them. Early studies on Complex Word Identification (CWI) usually identified complex words based on some word frequency threshold (Biran et al., 2011), word length (Bautista et al., 2009), or even attempting to simplify all words (Thomas and Anderson, 2012). However, Horn et al. (2014) showed that this last approach may ignore a large portion of complex words due to its inability to find simpler alternatives.

Moreover, as Shardlow (2013) states that the simplify-all approach might result in distorted meanings and the more resource-intensive threshold-based approach does not necessarily perform better, novel approaches have been presented (Gooding and Kochmar, 2019) to perform CWI conditioned on the context words lie in, in a sequence modelling fashion.

Detailing complex word replacement, previously rule-based systems usually replace an identified word by it most frequent synonym in WordNet (Thomas and Anderson, 2012). However, more recent approaches are using contextualized word vectors (Qiang et al., 2020), leveraging the power of the powerful neural models currently available.

3 Datasets

Fortunately, when considering text simplification for English sentences, there exists a large amount of data publicly available, mainly due to the existence of a Simple English version of Wikipedia[1] and educational article sources such as Newsela[2].

Besides, there have been numerous efforts aligning these articles, providing the research community with high quality simplification datasets for the English language. Among them, we have WikiLarge (Zhang and Lapata, 2017) and WikiAuto (Jiang et al., 2020) which differ in the way they were constructed and overall size.

Additionally, in order to augment the training dataset, we made use of a paraphrase dataset entitled OpusParcus (Creutz, 2018): it uses differently authored subtitles for movies and TV shows, resulting in aligned paraphrases for scenes with the same meaning.

Moreover, there are two test datasets that are commonly discussed in the literature: TurkCorpus (Xu et al., 2016) and the ASSET (Alva-Manchego et al., 2020) dataset. These last two source the same original sentences, but resort to different simplification techniques in the crowdsourced references they provide. While TurkCorpus only allows for rewriting the original sentence, ASSET is less restrictive and makes it possible to delete expressions or split the ideas over several sentences.

On the other hand, for the lexical simplification approach, we utilized the English dataset from the Complex Word Identification 2018 Shared Task (Yimam et al., 2018), called CWIG3G2 (Yimam et al., 2017).

Concluding, in table 1, we can take note of each dataset's size.

3.1 Newsela

Featuring the Newsela dataset in our project would have been an excellent addition, since it consists of educational articles with professional simplifications. However, Newsela is a corporate entity and their data is not publicly available. We tried to get access to it but, unfortunately, we didn't receive a reply.

[1] https://simple.wikipedia.org/
[2] https://newsela.com/

Name	#Train	#Val	#Test
Wiki_Large	296.402	-	-
Wiki_Auto	604.000	-	-
TurkCorpus	-	2.000	359
ASSET	-	2.000	359
OpusParcus	~500.000	-	-
CWIG3G2	27.299	3328	4252

Table 1: **Datasets.** Train and test sets and the respective number of samples.

4 Evaluation of our system

In order to evaluate our models, we rely on three evaluation metrics: SARI, FKGL and BLEU. We compute them using EASSE (Alva-Manchego et al., 2019), a simplification evaluation library in Python.

4.1 SARI

SARI (Xu et al., 2016) compares system output against both references and the input sentences. It measures the goodness of words that are added, deleted and kept by the systems comparing the output of the simplification model to multiple references and the original sentence, using both n-gram precision and recall.

So far, SARI is the most commonly adopted metric for text simplification in English, and we use it as a reference score of the overall performance.

4.2 FKGL

In order to measure the readability of our systems, we use FKGL (Kincaid et al., 1975). It is computed as a linear combination of sentence length and the number of syllables per word.

Although FKGL does not take into account grammaticality and meaning preservation (Wubben et al., 2012), it is one of the mostly used evaluation metrics for text simplification in English (only). However, due to this limitation it should not be used alone as an evaluation metric.

4.3 BLEU

Usually utilized as a metric to evaluate machine translation systems, BLEU (Papineni et al., 2002), is an N-gram based metric that is supposed to correlate with meaning preservation

Although it has been reported that BLEU doesn't necessarily correlate with the quality of simplifications (Scialom et al., 2021b; Sulem et al., 2018), we found that we should still report it in order to compare our model with other existing work.

5 Fine-Tuning Language Models for Simplification

Similar to a baseline step, we decided to train a basic Seq2Seq model with attention (Bahdanau et al., 2014). However, as it did not amount to good or even promising results, possibly due to implementation related problems, work in this direction was halted. Still, we report this model in our results under the name *Baseline*. Furthermore, for our initial experiments we settled on a subset of the larger WikiAuto dataset as we found it to be enough (circa 50000 sentences) for a proof-of-concept demonstration.

As a next step, and with a large amount of inspiration from the MUSS paper (Martin et al., 2020b), we set out to fine-tune a BART (Lewis et al., 2019) model on the subset of WikiAuto we have mentioned above. We resorted to the HuggingFace[3] `transformers` package, which has all the necessary tools to build and fine-tune our models. Specifically, we have made use of the `facebook/bart-base` model which is adequate to our computing power: the training portion was conducted through an instance of Google Colab Pro[4] as neither of us have a machine capable of training at this model/dataset scale.

From there, we had built a model with a somewhat reasonable behavior, achieving a 38.59 SARI score on the ASSET dataset (more details and comparisons on our all of our model results will be discussed in a future section).

In addition to BART, we also experimented with the T5 (Raffel et al., 2019) model, which should have displayed relatively better results. However, we encountered some overfitting issues in the fine-tuning procedure and tried some known fixes (freezing layers, dropout, etc...) to no avail.

5.1 Ranking candidate sentences

Finally, one other aspect to consider optimizing when it comes to these large language models generation ability, is that they are able to output several candidate sentences. With the use of beam-search to search over the most likely tokens, the model is able to keep track of what sentences observe the highest likelihood and drop them dynamically if a new, more probable, candidate arises. But then, how should we rank these candidates and output a singular, most simple sentence?

[3] https://huggingface.co/

[4] https://colab.research.google.com/

One simple heuristic is to compute the FKGL metric over the candidates and return the lowest score, which should indicate the simpler sentence overall. However, this completely disregards the semantic meaning in each sentence. In Martin et al. (2019), machine translation metrics such as METEOR (Banerjee and Lavie, 2005) and smoothed BLEU (Lin and Och, 2004) showed the largest correlation with meaning preservation. Furthermore, we would want to encourage grammatical structure simplification, which can be measured through SAMSA (Sulem et al., 2018). Given all of these metrics and aspects to optimize, we believe that a harmonic compromise between them could be a plausible solution to the ranking problem.

On the other hand, we also experimented with another idea: drawing inspiration from Scialom et al. (2021b), we tried using a question generation and answering framework on the source and output sentence in order to evaluate their semantic meaning. For simplification, this requires an adaptation of QuestEval (Scialom et al., 2021a) that utilizes BERTScore (Zhang et al., 2019) instead of just plain F1 score of the provided answers. Since it's encouraged for the simplification model to replace words with synonyms, we expect their respective BERT (Devlin et al., 2018) embeddings to be similar.

What we found however is that we introduced a significant overhead in our model's inference time and didn't gain much in terms of results. Using a threshold approach on the candidate sentence's BERTScore, we only managed to achieve the same SARI result we previously had once the threshold had gotten low enough that the output was identical to the one provided by FKGL ranking. Therefore, given the worst efficiency and poor results, we disregarded this question generation/answering idea from our final system.

5.2 Paraphrase dataset augmentation

Closing the fine-tuning section, we should discuss another experiment we have conducted: MUSS (Martin et al., 2020b), one of the most recent successful papers in the TS field, augments the WikiLarge dataset with paraphrase data, easing the rewriting of phrases and expressions. For their work, an unsupervised method was devised for aligning sliding windows of text scraped from the web, computing a similarity score between them for alignment. In the end, they constructed an unsupervised paraphrase dataset of circa 1 million samples, something unfeasible for us to reproduce.

However, instead of gathering the dataset ourselves, we tried to arrange a paraphrase dataset and amplify WikiLarge with further sentence rewriting examples. To this effect, we chose a subset of OpusParcus (Creutz, 2018), resulting in an augmented training dataset of ~500000 aligned sentences. Following a large amount of training time, this idea (which still sounds promising) failed to produce an exciting SARI score, which prompted us to not investigate further due to time constraints.

6 Encoding Simplification With Explicit Tokens

In addition to general fine-tuning, we wanted to control sentence simplification attributes using explicit control tokens. First, we compute them for every sentence pair in the training set and inject the ratio between source and target in the source sentence, such that the model should learn to condition the output on the ratios. As a measure of simplicity, we compute the following tokens:

#Chars `<c_xx>`: the number of characters ratio between source and target sentences. This control token provides information about sentence compression.

LevSim `<lev_xx>`: normalized character-level Levenshtein similarity (Levenshtein, 1966) between the source and target. Quantifies how different the target is to the source sentence.

WordRank `<rank_xx>`: inverse frequency order of all words in the target divided by that of the source. Word frequency are indicators of word complexity.

DepTree `<dep_xx>`: maximum depth of the dependency tree of the target divided by that of the source. This token should provide information about syntactic complexity.

#Words `<rat_xx>`: number of words ratio between source sentence and target sentence. The number of words in the target divided by that of the source.

At inference time, we condition the generation by choosing the values most suitable to the degree of simplification required.

6.1 Hyperparameter search

At inference time, we have to choose fixed ratios that maximize the simplicity for the audience (measured by the SARI score). To this effect, we hand-

crafted different possible values for each token and tried out combinations between them. It is easy to see how this leads to an exponential runtime complexity, which is something we couldn't afford to run several times. Concluding, after trying out several combinations on the ASSET dataset, we settled on the values shown in table 2.

#Chars	LevSim	WordRank	DepTree	#Words	#Syl
c_0.8	lev_0.5	rank_0.8	dep_0.9	rat_0.9	n_syl_1.9

Table 2: Best token combination for the ASSET test set, training on WikiLarge

6.2 Finding new tokens

In addition to the tokens described in Sheang and Saggion (2021), we explored new token ideas that could possibly increase our overall SARI score (42.94 originally). The following options were tested:

#Splits: the number of sentences ratio between source and target. This should provide information about the model's ability to split a long sentence and generating short sentences. Result: $\mathbf{42.43}$.

Average word length: the average word length ratio between the words in source and test sentences. Provides information about word compression. Result: $\mathbf{42.475}$.

#Syl: average number of syllables per word ratio. Should help readability. Result: $\mathbf{43.05}$.

More tokens could be tested, such as the number of polysyllables or the proportion of edited, deleted and added words, for example. However, these tokens would control the sentence compression and readability, which are already controlled to some degree.

Finally, since only one of these tokens produced better results, we only introduce the novel `n_syl` token and add it to our system.

6.3 Ablation study

In this section, we study the impact that each token has over the generated outputs. In figure 1 we present how varying one feature in isolation impacts the others. The displayed results stem from computing these metrics over the ASSET test set outputs and averaging them over all the sentences.

It is observed that some tokens are strongly correlated and that some don't appear to have much influence on the majority of the outputs. For example, if we consider character compression and word ratio, Levenshtein similarity and dependency tree depth, we can clearly see how they are correlated: shorter sentences should observe more rewriting and fewer words overall, while also making an impact on the dependency tree depth. On the other hand, tokens like word complexity and the number of syllables don't seem to be having much effect, failing to vary the output metric by much.

Nonetheless, overall, the tokens seem to be mostly behaving as expected, showing more or less a linear increase if we look at each feature's data separately.

7 Lexical Simplification

Once the simplification model performed up to a decent standard, it was time to experiment with explicit lexical simplification, setting up a pipeline that tried to simplify any complex words. In turn, this implied having two separate models: one for the identification of what words to replace and one for the actual replacement.

7.1 Complex Word Identification

Initially, as a baseline in identifying if a word is complex or not, we tried out a simple heuristic: with access to the Zipf (Zipf, 1949) word frequencies in a large corpus of text, we could go over each word in the provided input sentence and mark it as complex if its frequency was below a certain threshold. Zipf's law is a power law model of how words are distributed in a specific set of texts.

This approach works reasonably well, but it still has its shortcomings. For example, it fails to consider a word's context, which was utilized by Gooding and Kochmar (2019) in order to improve the same CWI task. Thus, following the paper's footsteps, we decided to model this problem as a sequence classification task: we prefix a sentence (similar to the token idea we have described in previous sections) with the word to be labeled and let our model predict if it's complex or not.

Moreover, we should discuss what it entails for a word to be complex. Most of the research available poses the CWI problem as a binary classification task, but complexity is more fine-grained than a binary label. A word isn't equally complex in the eyes of both a native and non-native speaker. In this direction, in recent years there have been some efforts in building regression models that are able to capture these different views. This last task was the objective of the 2021 SemEval Lexical Complexity Prediction Shared Task (Shardlow et al., 2021).

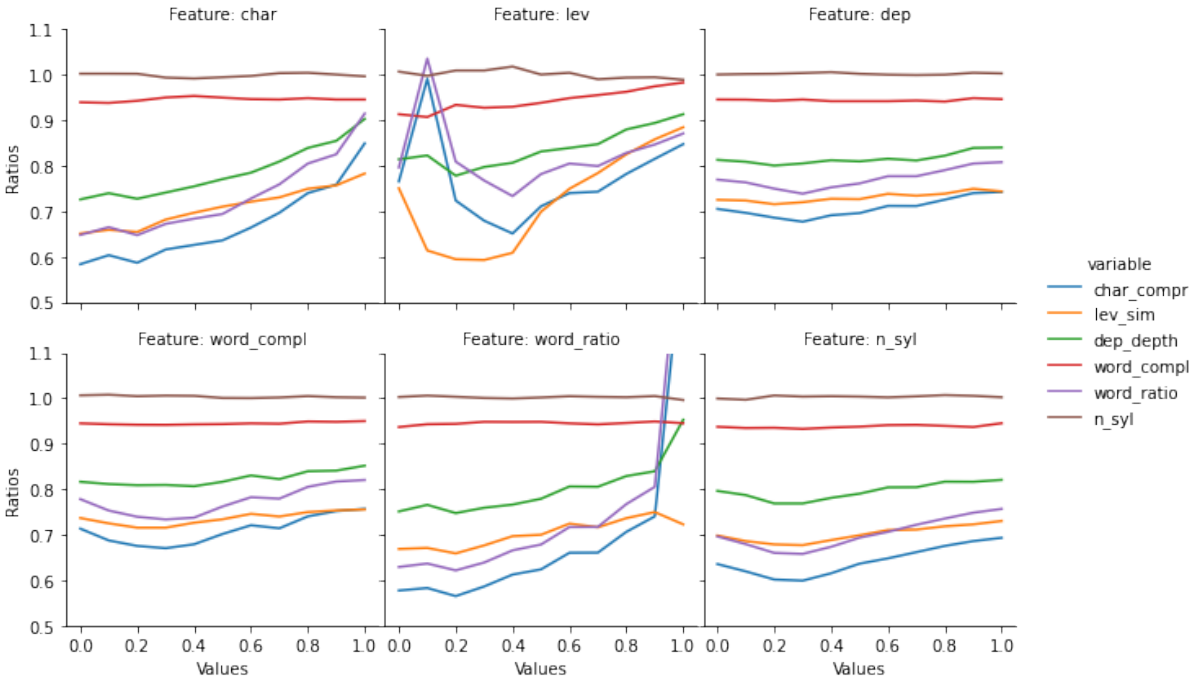

Figure 1: **Ablation study.** Influence that each feature has on the generated outputs and on other features.

Similarly, we adapted our code to perform regression on the CWI dataset from Yimam et al. (2017) that used binary complex annotation for our regression task: however, the dataset has the number of annotators who found the word complex and the number of annotators overall, making it possible for us to model the probability of a word being complex as the regression target.

For the regressor, we resourced the DistilBert (Sanh et al., 2019) model available in the `transformers` package and utilized it in a sequence classification fashion but with a single class, effectively performing regression. After fine-tuning on the binary CWI dataset, we had constructed a model that performed quite well, achieving a Mean Average Error of 0.052 on the test portion of the dataset, even visually evaluating it (see table 3), we can see that it behaves as expected. The benefit of regression is that it allows us to set a threshold that better models our data and, after some tweaking, we ended up settling for a value of 0.2.

7.2 Complex Word Replacement

First, we think that this portion of our system is its biggest bottleneck. Unsurprisingly so, since our time-constrained solution is very simple: after identifying which words to replace, we generate n copies of the source sentence, where n is equal to the amount of complex words; from there we mask each complex word in a corresponding sentence, leveraging DistilRoBERTa (Sanh et al., 2019) to predict the most likely token given the context.

Now, this solution has some merit as the most likely word should be one that occurs very frequently in the associated context and is, therefore, simple. On the other hand, the reasoning behind replacing each word individually is related to damage control, since there are no guarantees that the substitute word is even remotely similar to the original. Particularly, when replacing nouns, the most likely scenario is for the model to find another, more common noun (e.g., neurosurgeon $\rightarrow$ businessman).

To summarize, this is not the optimal setting for this task, but it was the one possible to implement with limited time.

8 Results

Finally, in table 4 we present the results of all developed models, tested with the ASSET and TurkCorpus datasets and trained with WikiLarge or a subset of WikiAuto. For each model, we report the SARI, BLEU and FKGL metrics. Here, our best results were produced by BART with all the tokens available in the literature, plus the novel `n_syl` token.

We also observe disappointing results for T5-based models, especially since it is the main ingre-

Use HTML and CSS markup sparingly and only with good reason.
Stallone also had an cameo appearance in the 2003 French film Taxi 3 as a passenger.
A fee is the price one pays as renumeration for services, especially the honorarium paid to a doctor, lawyer, consultant, or other member of a learned profession.

Table 3: **Identification of complex words**

dient for the current state-of-the-art system (Sheang and Saggion, 2021). Another interesting finding is that CWI decreases the SARI score, while displaying the best FKGL score, emphasizing the increased readability in detriment of meaning preservation.

Furthermore, we only performed the hyperparameter search over the ASSET dataset, which should imply that the token results for the TurkCorpus dataset should be taken with a grain of salt and could substantially improve. Plus, we disregarded testing all the features over the WikiAuto trained model, since it's expected to perform worse than its WikiLarge counterpart.

All in all, we got close to the state-of-the-art with our best model (BART+Tokens+n_syl) achieving a SARI of 43.05 which should only improve with the suggestions from the Future Work Section.

9 Future work

Finally, after having identified some problems, bottlenecks or potential areas of improvement impacting our system's performance, we will proceed to briefly discuss them, in order to facilitate the continuation of our work.

9.1 BRIO-like training procedure

To begin, we have come across the BRIO (Liu et al., 2022) model, that is currently the state-of-the-art in abstractive summarization. Due to the similarity between these two tasks, we think that replicating BRIO's properties will substantially increase our model's performance.

The authors have introduced a novel training procedure that combines contrastive loss and the regular MLE-based cross entropy loss we already utilize. Particularly, the model utilizes the contrastive loss with the assumption that text generation is not a "one correct answer" type of task and thus, assigns probability mass to the supposedly suboptimal candidate outputs.

As previously explained, BART-like models generate their candidate sentences in a token-level autoregressive fashion, using beam-search to limit the output space and, consequently, being able to generate several candidates. Given the MLE assumption, the first output is the most likely, since it produces a sentence similar to the provided reference. On the other hand, this line of thought is flawed since other candidates can be simple paraphrases, our desired outcome.

To combat this, the contrastive loss is based on an arbitrary metric suitable to the task at hand (in our case, SARI). This novel training procedure introduces the possibility of reordering the candidate sentences according to their simplicity, directly coordinating with the token-level generation task to improve the system's performance.

Furthermore, since we already investigated this ranking-type of approach after the fine-tuning procedure, it is reasonable that incorporation during training will improve our system.

9.2 From Single Complex Words to Multi-Word Expressions

In addition to the CWI and replacement, a possible improvement would be to also replace Multi-word Expressions (MWE) as they represent word sets that should be treated as single lexical units.

Since MWEs are most of the time inherently complex, replacing them with a single word synonym, if possible, could facilitate the TS task and mark an improvement in our overall system as shown in both Kochmar et al. (2020) and Gooding et al. (2020).

9.3 Replacement of Complex Words

As mentioned above, replacing complex words is our model's biggest bottleneck, as it just accounts for the context and not for the word itself. In natu-

	Models	ASSET			TurkCorpus		
		SARI ↑	BLEU ↑	FKGL ↓	SARI ↑	BLEU ↑	FKGL ↓
WikiLarge	Baseline	23.221	0.008	5.825	17.830	0.004	35.298
	BART	38.59	84.03	7.294	38.77	73.38	7.08
	T5	36.15	**89.98**	8.37	38.14	80.51	7.52
	T5+Tokens	36.89	84.05	7.92	36.33	72.72	6.71
	BART+Tokens	42.94	67.31	5.1	37.85	65.31	8.09
	BART+Tokens+n_syl	**43.05**	74.64	5.72	38.60	61.15	5.72
	BART+Tokens+n_syl+CWI	42.996	63.65	**5.095**	38.20	53.20	5.34
WikiAuto	Baseline	25.637	0.019	7.439	15.950	0.015	15.732
	BART	39.21	85.88	6.57	36.30	79.44	7.61
	T5	37.654	85.532	6.994	37.339	80.660	7.497

Table 4: **Sentence Simplification in English.** We display the SARI, BLEU and FKGL scores on the ASSET and TurkCorpus test sets. Models were trained both on WikiLarge and a subset of the WikiAuto dataset.

rally occurring sentences, it is completely ordinary for antonyms or words meaning a different thing to be surrounded by the same context (e.g. agreement and disagreement in legal texts), implying that our approach is inherently flawed.

However, previous research we encountered (Biran et al., 2011; Qiang et al., 2020) uses either a rule-based approach, or a similar masking approach, implying that this is a research field left to further explore. After some consideration, we thought that in a future work setting, it might be worth it to explore modelling this problem as an expression paraphrasing problem. Of course, this leads us to the problem of having no suitable datasets.

Nonetheless, it might be possible to adapt current paraphrase datasets by aligning each sentence pair at a token level, hopefully constructing a dataset of expressions that have a similar meaning, while preserving their context which may still be useful.

9.4 Modelling inter-sentence dependencies

Finally, the vast majority of TS related research models the problem in a sentence to sentence scenario. However, in real word applications it is very useful to provide a document level simplification rather than on a sentence level.

Consider, for example, legal text that should still be understood by non-legal experts but is usually too complex and long: if we were to simplify it at the sentence level, we would never achieve optimal simplifications that stem from cutting largely irrelevant portions of text. Thus, confirming that it might be useful to pursue the TS problem from a different prism.

To this effect, Sun et al. (2021) introduced a new dataset D-Wikipedia that provides aligned articles in detriment of aligned sentences, while also introducing a corresponding D-SARI metric, formalizing this TS paradigm. Furthermore, by introducing several baseline models, the authors proved there is some merit to this approach.

Concluding, we should also relate this version of the problem with the abstractive summarization one, which is so very similar. The only additional constraint is that in document-level simplification, we require the resulting sentences to be simple, possibly resulting in larger outputs in order to express ideas which could be condensed otherwise. Nevertheless, emulating successful summarization papers could serve some purpose to this new paradigm.

Acknowledgements

We would like to thank professor Miriam Anschütz for her guidance and feedback during the semester, and also for the opportunity to work on the Text

Simplification problem.

References

Fernando Alva-Manchego, Louis Martin, Antoine Bordes, Carolina Scarton, Benoît Sagot, and Lucia Specia. 2020. ASSET: A dataset for tuning and evaluation of sentence simplification models with multiple rewriting transformations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4668–4679, Online. Association for Computational Linguistics.

Fernando Alva-Manchego, Louis Martin, Carolina Scarton, and Lucia Specia. 2019. Easse: Easier automatic sentence simplification evaluation.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Susana Bautista, Pablo Gervás, and R.Ignacio Madrid. 2009. Feasibility analysis for semiautomatic conversion of text to improve readability. In *In Proceedings of the Second International Conference on Information and Communication Technology and Accessibility*, pages 33–40.

Or Biran, Samuel Brody, and Noemie Elhadad. 2011. Putting it simply: a context-aware approach to lexical simplification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics:shortpapers*, page 496–501, Portland, Oregon. Association for Computational Linguistics.

John Carroll, Guido Minnen, Yvonne Canning, Siobhan Devlin, and John Tait. 1998. Practical simplification of english newspaper text to assist aphasic readers. *Proc. of AAAI-98 Workshop on Integrating Artificial Intelligence and Assistive Technology*.

Mathias Creutz. 2018. Open subtitles paraphrase corpus for six languages.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

Yue Dong, Zichao Li, Mehdi Rezagholizadeh, and Jackie Cheung. 2019. Editnts: An neural programmer-interpreter model for sentence simplification through explicit editing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, page 3393–3402, Florence, Italy,. Association for Computational Linguistics.

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

Angela Fan, David Grangier, and Michael Auli. 2018. Controllable abstractive summarization.

Thibault Fevry and Jason Phang. 2018. Unsupervised sentence compression using denoising auto-encoders.

Sian Gooding and Ekaterina Kochmar. 2019. Complex word identification as a sequence labelling task. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1148–1153, Florence, Italy. Association for Computational Linguistics.

Sian Gooding, Shiva Taslimipoor, and Ekaterina Kochmar. 2020. Incorporating multiword expressions in phrase complexity estimation. In *Proceedings of the 1st Workshop on Tools and Resources to Empower People with REAding DIfficulties (READI)*, pages 14–19, Marseille, France. European Language Resources Association.

Colby Horn, Cathryn Manduca, and David Kauchak. 2014. Learning a lexical simplifier using wikipedia. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Short Papers)*, page 458–463, Baltimore, Maryland, USA.

Chao Jiang, Mounica Maddela, Wuwei Lan, Yang Zhong, and Wei Xu. 2020. Neural crf model for sentence alignment in text simplification.

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

Ekaterina Kochmar, Sian Gooding, and Matthew Shardlow. 2020. Detecting multiword expression type helps lexical complexity assessment.

Vladimir Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet physics doklady*, 10:707—-710.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

Chin-Yew Lin and Franz Josef Och. 2004. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 605–612, Barcelona, Spain.

Yixin Liu, Pengfei Liu, Dragomir Radev, and Graham Neubig. 2022. Brio: Bringing order to abstractive summarization.

Christiane Maaß. 2020. *Easy Language - Plain Language - Easy Language Plus. Balancing Comprehensibility and Acceptability*. Easy – Plain – Accessible. Frank & Timme, Berlin.

Jonathan Mallinson, Rico Sennrich, and Mirella Lapata. 2018. Sentence compression for arbitrary languages via multilingual pivoting. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, page 2453–2464, Brussels, Belgium. Association for Computational Linguistics.

Louis Martin, Eric Clergerie, Benoît Sagot, and Antoine Bordes. 2020a. Controllable sentence simplification.

Louis Martin, Angela Fan, Éric de la Clergerie, Antoine Bordes, and Benoît Sagot. 2020b. Muss: Multilingual unsupervised sentence simplification by mining paraphrases.

Louis Martin, Samuel Humeau, Pierre-Emmanuel Mazaré, Antoine Bordes, Éric Villemonte de La Clergerie, and Benoît Sagot. 2019. Reference-less quality estimation of text simplification systems.

Sergiu Nisioi, Simone Paolo Stajner, Sanja Ponzetto, and Liviu P. Dinu. 2017. Exploring neural text simplification models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Short Papers)*, page 85–91, Vancouver, Canada. Association for Computational Linguistics.

Gustavo H. Paetzold and Lucia Specia. 2016. Unsupervised lexical simplification for non-native speakers. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, page 3761–3767. AAAI Press.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Ellie Pavlick and Chris Callison-Burch. 2016. Simple ppdb: A paraphrase database for simplification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, page 143–148, Berlin, Germany. Association for Computational Linguistics.

Jipeng Qiang, Yun Li, Yi Zhu, Yunhao Yuan, and Xindong Wu. 2020. Lexical simplification with pretrained encoders. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8649–8656.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer.

Luz Rello, Ricardo Baeza-Yates, Stefan Bott, and Horacio Saggion. 2013. Simplify or help? text simplification strategies for people with dyslexia. In *Proceedings of the 10th International Cross-Disciplinary Conference on Web Accessibility*, W4A '13, New York, NY, USA. Association for Computing Machinery.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

Carolina Scarton and Lucia Specia. 2018. Learning simplifications for specific target audiences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Short Papers)*, page 712–718, Melbourne, Australia. Association for Computational Linguistics.

Thomas Scialom, Paul-Alexis Dray, Patrick Gallinari, Sylvain Lamprier, Benjamin Piwowarski, Jacopo Staiano, and Alex Wang. 2021a. Questeval: Summarization asks for fact-based evaluation.

Thomas Scialom, Louis Martin, Jacopo Staiano, Éric Villemonte de la Clergerie, and Benoît Sagot. 2021b. Rethinking automatic evaluation in sentence simplification.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Controlling politeness in neural machine translation via side constraints. In *Proceedings of NAACL-HLT 2016*, page 35–40, San Diego, California. Association for Computational Linguistics.

Matthew Shardlow. 2013. A comparison of techniques to automatically identify complex words. In *Proceedings of the ACL Student Research Workshop*, page 103–109, Sofia, Bulgaria. Association for Computational Linguistics.

Matthew Shardlow, Richard Evans, Gustavo Henrique Paetzold, and Marcos Zampieri. 2021. SemEval-2021 task 1: Lexical complexity prediction. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 1–16, Online. Association for Computational Linguistics.

Kim Cheng Sheang and Horacio Saggion. 2021. Controllable sentence simplification with a unified text-to-text transfer transformer. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 341–352, Aberdeen, Scotland, UK. Association for Computational Linguistics.

Elior Sulem, Omri Abend, and Ari Rappoport. 2018. Semantic structural evaluation for text simplification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 685–696, New Orleans, Louisiana. Association for Computational Linguistics.

Renliang Sun, Hanqi Jin, and Xiaojun Wan. 2021. Document-level text simplification: Dataset, criteria and baseline.

S. Rebecca Thomas and Sven Anderson. 2012. Wordnet-based lexical simplification of a document. In *Proceedings of KONVENS 2012*, pages 80–88.

Sander Wubben, Antal van den Bosch, and Emiel Krahmer. 2012. Sentence simplification by monolingual machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, page 1015–1024, Jeju, Republic of Korea. Association for Computational Linguistics.

Sander Wubben and Emiel Boschand, Antal Krahmer. 2012. A monolingual tree-based translation model for sentence simplification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, page 1353–1361, Jeju, Republic of Korea.

Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. 2016. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415.

Seid Muhie Yimam, Chris Biemann, Shervin Malmasi, Gustavo H. Paetzold, Lucia Specia, Sanja Stajner, Anais Tack, and Marcos Zampieri. 2018. A report on the complex word identification shared task 2018. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*, page 66–78, New Orleans, Louisiana. Association for Computational Linguistics.

Seid Muhie Yimam, Sanja Štajner, Martin Riedl, and Chris Biemann. 2017. CWIG3G2 - complex word identification task across three text genres and two user groups. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 401–407, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert.

Xingxing Zhang and Mirella Lapata. 2017. Sentence simplification with deep reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 584–594, Copenhagen, Denmark. Association for Computational Linguistics.

Sanqiang Zhao, Rui Meng, Daqing , He, Saptono Andi, and Parmanto Bambang. 2018. Integrating transformer and paraphrase rules for sentence simplificationr.

Zhemin Zhu, Delphine Bernhard, and Iryna Gurevych. 2010. A monolingual tree-based translation model for sentence simplification. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, page 1015–1024, Beijing. Association for Computational Linguistics.

George Zipf. 1949. *Human behavior and the principle of least effort: An introduction to human ecology*. Addison-Wesley Press, INC.

A Explaining Kept Complex Words

As a sidenote that didn't get included in our final system: for words that were to be replaced and had no simpler synonym, it could be useful to include their definition in the produced simplification. To this effect, we used PyDictionary, a Python library that relies on WordNet to get word definitions. Including the meaning of the word is done in a post-processing step and has no other effect on the system.

As expected, adding more text (even potentially complex text) produced a poor SARI result. In table 5, we present an example of this definition incorporation procedure.

The Great Dark Spot is thought to represent a hole in the methane cloud deck of Neptune.
The Great Dark Spot is thought to represent a hole in the methane (a colorless odorless gas used as a fuel) cloud deck of Neptune.

Table 5: **Complex word identification and definition example.** The word 'methane' was identified as being complex. Since it does not have a simpler synonym in WordNet, its definition was added to the source sentence.

B Demonstration

Concluding, to go along with our presentation, we built an app using Python and Streamlit in order to illustrate our system's capabilities. This app is currently publicly available on a HuggingFace Spaces page[5] for easy user interaction.

Moreover, both the simplification model and the CWI regressor were made public to the HuggingFace community, available with the names `twigs/bart-text2text-simplifier` and `twigs/cwi-regressor` respectively.

[5] https://huggingface.co/spaces/twigs/simplifier