"Geen makkie": Interpretable Classification and Simplification of Dutch Text Complexity

Eliza Hobo[*]
TNO[+]
eliza.hobo@tno.nl

Charlotte Pouw[*]
University of Amsterdam
c.m.pouw@uva.nl

Lisa Beinborn
Vrije Universiteit Amsterdam
l.beinborn@vu.nl

Abstract

An inclusive society needs to facilitate access to information for all of its members, including citizens with low literacy and with non-native language skills. We present an approach to assess Dutch text complexity on the sentence level and conduct an interpretability analysis to explore the link between neural models and linguistic complexity features.[1] Building on these findings, we develop the first contextual lexical simplification model for Dutch and publish a pilot dataset for evaluation. We go beyond previous work which primarily targeted lexical substitution and propose strategies for adjusting the model's linguistic register to generate simpler candidates. Our results indicate that continual pre-training and multi-task learning with conceptually related tasks are promising directions for ensuring the simplicity of the generated substitutions. Our code repository and the simplification dataset are available on GitHub.[2]

1 Introduction

Reading is a foundational skill for acquiring new information. Many sources of information are only available in written form, including educational material, newspaper articles, and letters from municipalities. Although many people learn how to read as a child, not everyone becomes equally skilled at it. In the Netherlands alone, more than 2.5 out of 14 million people over 16 years old are low-literate, meaning that they experience challenges with reading or writing.[3] As a result, they face obstacles in achieving academic success, seeking employment opportunities, and keeping up-to-date with current events.

One way to address this problem is to reduce text complexity. Texts that contain many infrequent words and complex sentence structures are difficult to read, especially for readers with low literacy and language learners. Automated natural language processing tools for text complexity assessment can help both in assisting editors in the selection of adequate texts and by signaling potential comprehension problems to copywriters. By estimating text complexity, we can select texts that are sufficiently easy for a particular target audience or simplify texts that are too difficult.

Recent neural models for text complexity assessment have obtained good results in classifying texts into discrete categories of complexity (Deutsch et al., 2020; Martinc et al., 2021). The global classification label can be a first indicator but it does not point to specific parts of the input that are complex, leaving it to the human editor to identify the necessary simplifications. In this work, we first explore Dutch complexity prediction on the sentence level (as opposed to full-text classification in previous work) and then zoom in even further.

The complexity of a text is affected by an interplay of various factors, including its structural characteristics, domain, and layout. A crucial component is the choice of the lexical units and their complexity. A system for lexical simplification can support humans in detecting lexical complexity and suggest simpler alternatives. In the sentence *children bear the future, and our resolution to support them determines the world they inherit*, a lexical simplification model could propose to substitute *bear* with simpler words such as *carry*, *hold*, or *shape*. These suggestions can assist human writers in revising and simplifying their text.

Previous approaches to Dutch lexical simplification generated substitution candidates by naively substituting words according to a static alignment

[*]Equal contribution.
[+]The experiments were conducted when all authors were affiliated with Vrije Universiteit Amsterdam.
[1]The colloquial Dutch expression "Geen makkie" in the title can be translated as "not easy" or "not a walk in the park".
[2]https://github.com/clap-lab/makkie/
[3]https://www.lezenenschrijven.nl/reading-and-writing-foundation

of synonyms without considering the context of the sentence. This approach does not account for ambiguous words and synonyms that only maintain semantic coherence in a subset of contexts. In the example above, *resolution* can be interpreted as intention, but in the context of TV screens, it refers to sharpness. In order to ensure meaning preservation, lexical simplification needs to be context-sensitive.

Contributions We fine-tune BERTje (de Vries et al., 2019), a Dutch pre-trained transformer model, to predict sentence-level complexity and use interpretability methods to show that it captures relevant linguistic cues. We visualize the local attribution values of the model's predictions in a demo to point end users to complex parts of the sentence. In order to facilitate the simplification process, we introduce LSBertje, the first contextual model for lexical simplification in Dutch. We explore three approaches to adapt the linguistic register of the model, to re-enforce a preference for simplicity in the generated substitutions.

2 Related Work

We discuss complexity assessment and lexical simplification as separate consecutive stages in line with related work.

2.1 Complexity Assessment

Text complexity is affected by the words we choose and the way we combine them into meaning. The complexity of individual words is determined by features such as length, frequency, morphological complexity, abstractness, and age of acquisition. At the sentence level, syntactic features such as parse tree depth, syntactic ambiguity, and the number of subordinate clauses affect complexity. Features that indicate lexical variety, such as the type-token ratio, can also serve as a proxy for complexity (Schwarm and Ostendorf, 2005; Feng et al., 2009; Vajjala and Meurers, 2012).

Traditional surface-based metrics such as the *Flesch-Kincaid* score are widely used to automatically assess text complexity, but they only consider length characteristics and do not take into account the various intricate factors that influence text complexity. In contrast, feature-based machine learning models leverage numerous features to predict complexity labels, surpassing the capabilities of surface-based metrics (Collins-Thompson and Callan, 2005). Nevertheless, hand-engineering effective features is an expensive and time-consuming process (Filighera et al., 2019).

Neural models for classifying complexity do not rely on hand-engineered features and show marginal improvements over feature-based models (Deutsch et al., 2020; Martinc et al., 2021), but they lack interpretability. In this study, we analyze if neural models leverage relevant linguistic cues when predicting binary complexity labels for Dutch sentences and can therefore reliably detect sentences that qualify for a simplification procedure.

2.2 Lexical Simplification

Lexical simplification characterizes a substitution operation on the lexical level with the goal of reducing the complexity of a sentence and making the text accessible to a wider audience. Lexical simplification of a sentence is typically performed as a pipeline of four consecutive stages: complex word identification, substitution generation, substitution selection and substitution ranking (Sikka and Mago, 2020; Thomas and Anderson, 2012; Paetzold and Specia, 2017b). In this work, we focus on the first two stages.

Complex Word Identification In the initial stage, words with simplification potential need to be identified. Traditional approaches for this subtask use curated lists of complex words (Lee and Yeung, 2018) or word frequency resources to flag words below a certain frequency threshold as complex (Sikka and Mago, 2020). In the most recent shared task for complex word identification (Yimam et al., 2018), feature-based machine learning techniques using length and frequency features obtained the best results. More recent approaches express lexical complexity on a continuous scale (Shardlow et al., 2021) as a binary classification is too simplistic for most educational scenarios. We explore the applicability of gradient-based interpretability techniques for complex word identification (Danilevsky et al., 2020; Sundararajan et al., 2017).

Substitution Generation The generation of substitution candidates has traditionally been performed with lexical resources such as WordNet (Miller, 1995; Carroll et al., 1998). In a more data-driven approach, simple-complex word pairs have been extracted from a parallel corpus that aligns sentences in Wikipedia with their counterparts in Simple Wikipedia (Kauchak, 2013; Paetzold and Specia, 2017a). These static approaches are unable to generate substitution candidates for words that do not occur in the resources or that are spelled differently. In addition, they are prone to generate semantically incoherent candidates since the substitutions are not context-sensitive.

Context-Aware Substitution Generation For meaning-preserving simplification, it is important to consider the context of the complex word. Paetzold and Specia (2016b) propose to use the part of speech of a word to narrow down its meaning. Their approach relies on proximity in a static embedding space to find simplifications, which are then disambiguated with respect to their part of speech. As a result, the relatively simple noun *bear* is represented by a different vector than the rather complex verb *bear*. This syntactically informed approach leads to improvements over non-contextualized models, but it still falls short in capturing more fine-grained differences in meaning; even the verb *bear* can be used in a semantic spectrum ranging from *bearing/delivering a child* to *bearing/having a resemblance*.

To capture such subtle distinctions, recent approaches use contextualized language models such as BERT (Devlin et al., 2019) to generate substitutions tailored to the specific context. Alarcón et al. (2021) search the contextual embedding space of a complex word to find context-aware simplification candidates. They find antonyms of the complex word among the generated candidates, which is detrimental to the goal of preserving the meaning of the complex sentence. Qiang et al. (2020) introduce *LSBert*, which uses a prompting strategy based on BERT's masked language modeling objective to generate context-aware lexical simplification candidates for English sentences. They generate simplifications by masking the complex word. In order to enforce semantic coherence of the masked word, Qiang et al. (2020) feed the input sentences as a duplicated pair and apply the masking operation only on the second sentence. In the recent shared task on multi-lingual lexical simplification (Saggion et al., 2022), approaches that use pre-trained language models produced very competitive results. In all three languages covered in the shared task, English, Spanish, and Portuguese, state-of-the-art results were obtained. In this work, we evaluate the LSBert lexical simplification approach and adapt it to Dutch.

2.3 Complexity Assessment and Simplification for Dutch

Work on complexity and simplification for Dutch is sparse. Vandeghinste and Bulte (2019) analyze complexity classification at the document level using feature-based classifiers, but there is currently no known work on neural sentence-level complexity classification for Dutch. Regarding lexical simplification, Bulté et al. (2018) develop a pipeline using various resources. However, systematically evaluating the pipeline is challenging as there is no existing benchmark dataset for lexical simplification in Dutch.

3 Complexity Classification

We train a neural classifier for determining binary labels of Dutch sentence complexity and compare its performance to several feature-based classifiers. We then analyze if the neural model captures relevant complexity cues.

3.1 Experimental Setup

Data We contrast articles from the Dutch newspapers *De Standaard* and *Wablieft* in line with Vandeghinste and Bulte (2019). The two newspapers cover similar topics and events. As Wablieft targets an audience that prefers simpler language, the articles are significantly shorter (on average, there are 164 words in Wablieft articles vs 383 words in De Standaard articles). The source of an article (Wablieft vs De Standaard) can therefore be easily determined by its length.[4] However, identifying the source is just a proxy for identifying the linguistic characteristics that determine complexity. To go beyond this superficial approach, we instead train our models to predict the complexity of individual sentences.

The corpus contains 12,683 articles from Wablieft and 31,140 articles from De Standaard.[5] We create a balanced dataset by randomly selecting 12,000 articles from each newspaper and preprocessing them using the same steps as Vandeghinste and Bulte (2019). We split the articles into individual sentences and only keep the first sentence of each article to keep the dataset balanced. We label all sentences from Wablieft articles as *easy*

[4]Our BERTje model could distinguish the two types of articles with 99% accuracy when fine-tuned to predict complexity labels for the entire articles.

[5]The data does not include any meta information such as author names and time stamps of publication, which could reveal the source of the article.

and all sentences from De Standaard as *complex*. We use 80% of the data for training, 10% for validation, and 10% for testing. The validation set was used for checking model accuracy at each epoch. Statistics regarding the length and frequency of the words in both types of sentences are shown in Table 1.

	Easy	Complex
#Sentences	12,000	12,000
Word length	**4.33** (2.14–8.60)	**5.10** (2.08–11.80)
Word freq.	**4.95** (1.95–6.38)	**4.78** (1.39–6.44)

Table 1: Descriptive statistics of the easy and complex sentences that are used to train and evaluate our models. Averages are in bold, ranges are between brackets. Frequencies are measured as standardized Zipf frequencies using the Python package wordfreq.

Models We fine-tune a pre-trained transformer model for Dutch sequence classification (`BERTje`, de Vries et al. (2019)) available from Huggingface and add a linear output layer with ReLU activation and dropout (0.5). The model is optimized using `ADAM` with a learning rate of 1e-6 and cross-entropy loss.

We use Support Vector Machines (SVM) as our feature-based classification models. We employ the scikit-learn implementation with all default parameters (Pedregosa et al., 2011).

Complexity Features Our complexity features can be grouped into three categories: length characteristics, frequency effects, and morpho-syntactic properties. Word frequencies are obtained as standardized Zipf frequencies using the Python package wordfreq (Speer et al., 2018). The package combines several frequency resources, including SUBTLEX lists, e.g. Brysbaert and New (2009), and OpenSubtitles (Lison and Tiedemann, 2016). The morpho-syntactic features are computed using the Profiling-UD tool (Brunato et al., 2020). We calculate all features on the sentence level and train our feature-based models on different combinations of these features. An overview of the features is given in Table 3.

3.2 Results

Table 2 shows the prediction accuracy of the fine-tuned BERTje model and several feature-based SVM classifiers for sentence-level complexity classification. We see that the neural model outperforms all feature-based models by 10 percent or more. For the feature-based classifiers, the best results can be obtained by all types of features (frequency + length + morpho-syntactic), but the morpho-syntactic features only improve the frequency and length-based classifiers with 1 percent accuracy. This might be caused by the fact that the morpho-syntactic features are correlated with length (e.g., parse tree depth naturally increases as the sentence length increases). We conclude that frequency and length are the most predictive features for Dutch sentence-level complexity classification, which is in line with previous work for English (Vajjala Balakrishna, 2015).

Model	**Accuracy**
Frequency	.72
Frequency + Morpho-Syntactic	.73
Length	.78
Length + Morpho-Syntactic	.79
Frequency + Length	.79
Frequency + Length + Morpho-Syntactic	.80
Neural Model (fine-tuned BERTje)	**.90**

Table 2: Prediction accuracy of several feature-based SVM models and the fine-tuned BERTje model for sentence-level complexity classification.

Prediction Confidence To gain more insight in the linguistic cues that the neural model relies on, we analyze model confidence with respect to the complexity features that our feature-based models were trained on. Table 3 shows the Spearman correlation between complexity features and model confidence for the *complex* class. We see that the model allocates higher probability values to the *complex* class when word length, sentence length, dependency link length, or the number of low-frequency words increases. As the classification is binary, the inverse relationship can be observed for the *easy* class.

Since the correlation values in Table 3 are relatively low, we analyze the corresponding scatter plots. Figure 1 depicts the correlation between model confidence for the *complex* class and the maximum dependency link of the input sentences. We see that low to medium values for the maximum dependency link length do not clearly affect model confidence, but that high dependency link values always lead to high confidence. We observe the same pattern for the other complexity features. This suggests that the model considers relevant complexity features when making its predictions, but that the evidence needs to be strong enough

Category	Linguistic Feature	ρ
Length	Avg. word length (# chars)	.41
	Sentence length (# tokens)	.40
Morph-Synt.	Max. dependency link length	.43
	Avg. dependency link length	.40
	# Verbal heads	.37
	Parse tree depth	.35
	Lexical density	.12
Freq	# Low frequency words (Zipf<4)	.37
	Avg word frequency	-.04

Table 3: Spearman correlations between sentence-level complexity features and confidence for the *complex* class of the BERTje model, fine-tuned for sentence-level complexity classification. All positive correlations are significant ($p < 0.0001$). The negative correlation between token frequency and model confidence is not significant ($p = 0.03$).

(i.e., the sentence should be sufficiently complex).

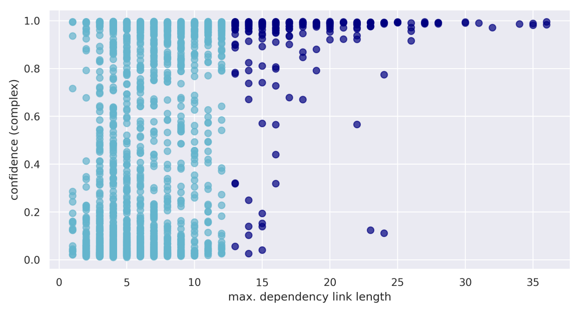


Figure 1: Correlation between BERTje's confidence for the *complex* class and the maximum dependency link length of the input sentences.

3.3 Complex Word Identification

Our results indicate that the fine-tuned BERTje model is a reliable tool for sentence-level complexity classification. It can show an editor which sentences qualify for simplification. Nevertheless, binary complexity classification is an overly simplified operationalization that lacks educational usability. We go one step further and combine the model with feature attribution methods and analyze its utility for the first component of the lexical simplification pipeline: complex word identification.

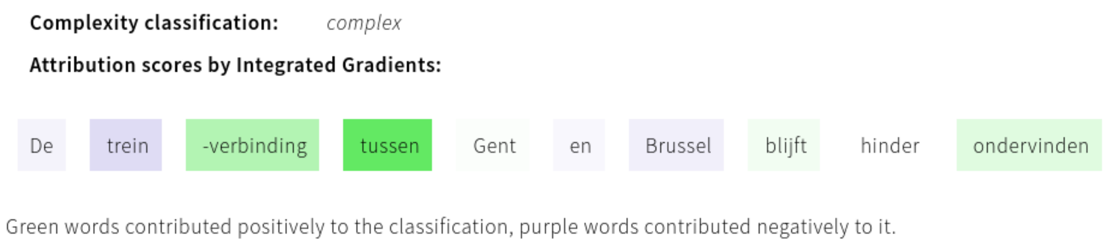
We implement a demo that explains the predictions of our neural complexity classifier. Users can type Dutch input sentences, which are classified as either *easy* or *complex*. Words that contributed positively or negatively to the model's prediction are highlighted, as shown in Figure 2. We use Captum (Kokhlikyan et al., 2020) for extracting token-level attributions. Additionally, the sentence-level complexity features from Table 3 are calculated and shown to the user, which give a more fine-grained perspective on the complexity of the input sentence (see Appendix Figure 4).

Attribution Methods Selecting the right attribution method is not straightforward. Different attribution methods produce varying, sometimes even contrasting explanations for model predictions (Bastings et al., 2022). Atanasova et al. (2020) find that gradient-based techniques produce the best explanations across different model architectures and text classification tasks. We therefore include three gradient-based attribution methods in our demo: Gradient, InputXGradient, and Integrated Gradients. The vanilla Gradient method estimates feature importance by calculating the gradient (i.e. the rate of change) of a model's output with respect to a given input feature (Danilevsky et al., 2020). InputXGradient additionally multiplies the gradients with the input, and Integrated Gradients integrates the gradient of the model's output with respect to the input features along a chosen path between a feature x and a baseline x' (Sundararajan et al., 2017). We use the [PAD] token as our baseline.

Linguistic Plausibility of Attributions Explanations of the complexity predictions are most useful for end-users of the demo (e.g. teachers) if the attribution scores are linguistically plausible. This means that the scores should match our expectations of what makes a sentence complex or easy to understand. Given the intended use of the demo for complex word identification, we analyze the linguistic plausibility of the attributions with respect to lexical complexity. We expect short and frequent words to receive high attributions when the model predicts that a sentence is easy to understand, while longer and less frequent words should receive high attributions when the model predicts that the sentence is complex.

To better understand the differences between our selected attribution methods and to analyze the linguistic plausibility of the observed patterns, we calculate the Spearman correlation between lexical complexity features and attribution scores. Since our model uses subword tokenization, both attribution scores and complexity features are calculated on the subword level. We exclude the special tokens [CLS] and [SEP] from our analyses.

Table 4 shows that Integrated Gradients is the only method for which the correlations have the ex-


Complexity classification: *complex*

Attribution scores by Integrated Gradients:

De trein -verbinding tussen Gent en Brussel blijft hinder ondervinden

Green words contributed positively to the classification, purple words contributed negatively to it.

Figure 2: Complexity classification and attributions scores for the sentence *De treinverbinding tussen Gent en Brussel blijft hinder ondervinden*, taken from the newspaper De Standaard (translation: the train connection between Ghent and Brussels continues to be affected.) The sentence is classified as complex by the fine-tuned BERTje model. Attributions are calculated by Integrated Gradients.

Class	Method	Len	Freq
Easy	Gradient	.61	-.44
	InputXGradient	.07	.18
	Integrated Gradients	-.10	.19
Complex	Gradient	.54	-.48
	InputXGradient	-.09	.04
	Integrated Gradients	.11	-.14

Table 4: Spearman correlation between subword-level complexity features and subword-level attributions. All correlations are significant ($p < 0.0001$.)

pected directionality, i.e. when the model predicts the *easy* class, high attributions are assigned to short/frequent words, and when the model predicts the *complex* class, high attributions are assigned to long/infrequent words. For InputXGradient, we see the opposite pattern, and for Gradient, the directionality of the correlations is the same for both the *easy* and *complex* class. The inconsistency of the three attribution methods is surprising but in line with previous findings (Bastings et al., 2022). More user-centered analyses are required to identify their practical benefits.

To further explore the linguistic plausibility of the attribution scores, we calculate average attribution scores with respect to part-of-speech tags. We again find that the most plausible attributions are generated by the Integrated Gradients approach. In Figure 3, we see that nouns, adverbs, and adjectives are assigned relatively high importance scores when the model predicts the *easy* class. Prepositions, conjunctions, and complementizers receive higher importance when the model predicts the *complex* class. This is plausible since function words often signal a complex sentence structure, while easier sentences typically contain more content words. Additionally, we observe that subwords, which indicate the presence of compound words, receive higher scores when the model predicts the *complex* class. This is helpful for lexical simplification, as compound words are often challenging to read. Finally, we observe that determiners receive high scores when the model predicts the *easy* class, which aligns with lexical complexity since determiners are short and frequent.

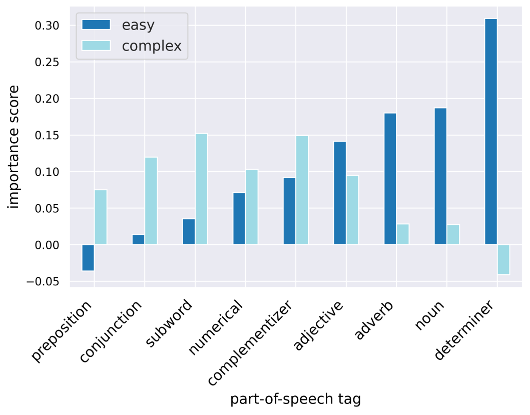


Figure 3: Average attribution scores per part-of-speech tag, generated by Integrated Gradients.

4 Context-Aware Simplification

In the second step of the simplification pipeline, we generate context-aware simplifications for Dutch.

LSBertje We present *LSBertje*, the first model for contextualized lexical simplification in Dutch. We base LSBertje on LSBert (Qiang et al., 2019, 2020) by altering its language-specific components to Dutch. We replace the language model that generates simplifications with the Dutch BERT model, BERTje. We also replace the stemmer used in filtering with the snowball stemmer.[6]

[6]nltk.org/api/nltk.stem.snowball.html

4.1 Dutch Evaluation Data

Dutch evaluation data for lexical simplification does not yet exist. To evaluate our approach, we develop a pilot benchmark dataset using authentic municipal data. We select sentences from a collection of 15,334 sentences from 48 municipal documents based on the presence of a complex word from a list curated by domain experts and based on their word count (less than 20 words). We exclude incomplete sentences such as headers, sentences without verbs, or with less than four words. From the remaining 6,084 sentences, we randomly sample 250 of complex words from the list and find a sentence for the dataset for 108 of the complex words. Eight sentences where simplification was not possible were removed because: 1) they were part of a named entity, 2) the sentence was incomplete or 3) a simple sense of the word was used. This resulted in 100 sentences.

The sentences were simplified by 23 native speakers of Dutch who pursued or obtained an academic degree. They were shown a sentence with the highlighted complex word and five simplification options that LSBertje generated. The annotators could select from these options and propose additional simplifications. For five sentences, no annotator could come up with a lexical simplification candidate. The remaining 95 sentences contained an average of 2.9 simplification candidates, with a maximum of 7.

4.2 Results and Analysis

Table 5 shows that the LSBertje model yields good simplification performance for our dataset. The potential metric shows that the model was able to predict at least one correct simplification candidate in 85% of the sentences. It should be noted that the English benchmark datasets come with a greater variety. In our dataset, a sentence is annotated with 2.9 simplifications on average, whereas BenchLS lists 7.4 substitutions. These size differences can explain the slightly lower potential score and the higher recall for Dutch.

To evaluate the simplicity of the generated substitutions, we assess their frequency using the SUBTLEX-NL corpus (Keuleers et al., 2010) and find that 517 out of 650 generated words occur with higher frequency than the original word. This indicates that the generated simplifications are indeed simpler.

5 Register Adaptation Techniques

LSBertje relies on a base model that was pre-trained for masked language modeling and captures aspects of text complexity only as an incidental byproduct. It uses a masked language modeling mechanism that induces semantic preservation by repeating the input sentence. The goal of generating simpler substitutions is only implicitly targeted by restricting the generation to tokens consisting of a single subtoken. This effectively prevents the model from generating infrequent or morphologically more complex words, but the model is not explicitly optimized for capturing different levels of text complexity. We explore three strategies to adapt the linguistic register of the model so that it generates simpler substitutions: conceptual fine-tuning, continual pre-training, and multi-task learning.

Conceptual Fine-tuning We aim at adapting the linguistic register of the model by fine-tuning LSBert to predict the linguistic complexity of sentences before applying it for generating substitution candidates. The model is fed a pair of sentences and is trained to predict whether the first sentence is simpler or more complex than the second example. We use sentence pairs from the sentence-aligned simple-complex Wikipedia corpus (Kauchak, 2013). The sentences are balanced with respect to the simplification order condition, and we experiment with the number of sentences.[7]

Continual Pre-Training For the second strategy, we adapt the linguistic register by exposing the model to simpler texts using continual pre-training. We continue the pre-training combination of masked language modeling and next-sentence prediction using only sentences from simple Wikipedia.[8] We pair each sentence either with the directly following sentence or with a randomly selected sentence from another Wikipedia article.

Multi-Task Learning We then combine the two ideas and train a model on two tasks simultaneously. We use the same training method but replace next-sentence prediction with complexity prediction.

5.1 Experimental Setup

As the Dutch dataset is too small for representative evaluation, we first explore the register adaptation

[7]cs.pomona.edu/ dkauchak/simplification/

[8]github.com/LGDoor/Dump-of-Simple-English-Wiki

strategies using English evaluation data and the English LSBert model.

Evaluation Data We evaluate the models on three commonly used benchmarking datasets. They consist of sentences from Wikipedia with the complex word highlighted and a list of human-generated simplifications. LexMTurk (Horn et al., 2014), BenchLS (Paetzold and Specia, 2016a) and NNSEval (Paetzold and Specia, 2016b) contain respectively 500, 929, and 239 sentences.

Implementation Details We base our implementation on the Huggingface documentation `Bert.for_Pretraining` and the same model as LSBert.[9] [10] For the masked language modeling components, we mask 15% of the tokens in the input sentences. Optimization is performed using an `ADAM` optimizer and a batch size of two. The continual pre-training is run for two epochs, the multi-task learning for four epochs. We varied the learning rate (5e-5, 5e-6, 5e-7) and the number of sentences (1000, 10.000, 50.000).

5.2 Results

We find that the model adapted with conceptual fine-tuning lost its ability to perform masked language modeling. Its predictions for *bear* in *children bear the future* were: *swallowed, if, knicks, cats, nichol*. These predictions clearly indicate a case of catastrophic forgetting (Liu et al., 2020). In learning a new task, the model forgot its original capabilities.

Both continual pre-training and multi-task learning lead to improved performance on the simplification task in two and three configurations respectively. We find that the configuration of LR 5e-6 and 10.000 sentences is the best for both fine-tuning methods as shown in Table 5. See the Appendix for all scores.

The multi-task learning strategy seems to be the most promising approach. We test the robustness of our findings by training the model using 26 different random seeds. The model outperforms LSBert in 20 cases, see Table 8 of the Appendix for a detailed overview. Overall, we see an increase in precision, recall, and F_1-score. While the model's performance is highly sensitive to task-specific components (the learning rate and the number of sentences), the performance remains robust for variation in the task-independent random seed. The results indicate that multi-task learning is a promising strategy for adapting the model's linguistic register.

[9] `bert-large-uncased-whole-word-masking`

[10] `https://huggingface.co/transformers/v3.0.2/model_doc/bert.html#bertforpretraining`

5.3 Analysis

We analyze the effect of the register adaptation techniques by comparing the frequency of the generated substitutions using the same resources as Qiang et al. (2019) that contains word frequency counts for Wikipedia articles and a children's book corpus. We see that the fine-tuned model generates simplifications that occur more frequently compared to the substitutions generated by LSBert (13,030 vs 20,000 occurrences on average). When we zoom in on the generations, we find that the fine-tuned model correctly generates 356 words that were not captured by LSBert and that these words have a high average frequency of 27,000. These findings indicate that the fine-tuning process indeed leads to the generation of simpler words.

5.4 Register Adaptation Results for Dutch

Due to the absence of a sentence-aligned simplification corpus for Dutch, we only test the continual pre-training strategy on the Dutch data. The results show that the improvements obtained for English cannot yet be observed for Dutch. In the future, we plan to extend our experiments to a larger dataset and to the multi-task learning strategy.

6 Conclusion

In this work, we have introduced two state-of-the-art components for complexity prediction and simplification in Dutch. It can support teachers and text editors in making texts more accessible for people who face reading challenges.

We developed a demo that predicts binary complexity labels for Dutch sentences and highlights words that contributed positively or negatively to the prediction. Additionally, the demo interface provides scales for different aspects of sentence-level complexity to enable a more fine-grained interpretation by the user.

We introduced LSBertje, which is the first model for contextualized lexical simplification in Dutch (to the best of our knowledge). We show that the model can generate adequate simplifications without additional fine-tuning. This base setup can serve as a reasonable starting scenario for context-

	LexMTurk				NNSEval				BenchLS			
Model	Pot.	P	R	$F1$	Pot.	P	R	$F1$	Pot.	P	R	$F1$
LSBert	98.20	29.58	23.01	25.88	90.79	19.04	25.40	21.77	92.36	23.64	32.08	27.22
Cont. Pre-training	98.40	33.46	26.02	29.28	90.79	20.33	27.14	23.25	92.14	25.68	34.84	29.56
MTL	**98.80**	**33.48**	**26.04**	**29.29**	**92.89**	**21.55**	**28.75**	**24.64**	**93.54**	**25.93**	**35.17**	**29.85**
	Dutch Benchmark											
LSBertje	85.26	17.74	65.68	29.16								
Cont. Pre-training	83.16	16.95	59.41	26.37								

Table 5: Simplification performance of the register adaptation techniques as potential (Pot.), precision (P), recall (R), and F_1 for the configuration with a learning rate of 5e-6 and 10,000 fine-tuning sentences.

aware simplification generation for resource-poor languages. We developed a pilot evaluation dataset for Dutch that allowed us to perform initial comparisons. For a more elaborate analysis, a larger Dutch dataset needs to be curated in future work.

We explored strategies to adapt the linguistic register of the model to ensure the simplicity of the generated substitutions and find that both multi-task learning and continual pre-training show considerable potential. We further analyzed the model's robustness and discovered a strong sensitivity to task-specific hyperparameters but little variation across random seeds.

Acknowledgements

Eliza Hobo's simplification experiments were initiated during an internship at the Gemeente of Amsterdam. Iva Gornishka has been a valuable source of insight and support in this process. Charlotte Pouw's experiments on readability were initiated in a joint project with Florian Kunneman and Bruna Guedes supported by the Network Institute (VU Amsterdam) through the Academy Assistants Program. Lisa Beinborn's work was supported by the Dutch National Science Organisation (NWO) through the projects CLARIAHPLUS (CP-W6-19-005) and VENI (Vl.Veni.211C.039).

References

Rodrigo Alarcón, Lourdes Moreno, and Paloma Martínez. 2021. Exploration of spanish word embeddings for lexical simplification. In *CTTS@SEPLN*.

Pepa Atanasova, Jakob Grue Simonsen, Christina Lioma, and Isabelle Augenstein. 2020. A diagnostic study of explainability techniques for text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3256–3274, Online. Association for Computational Linguistics.

Jasmijn Bastings, Sebastian Ebert, Polina Zablotskaia, Anders Sandholm, and Katja Filippova. 2022. "will you find these shortcuts?" a protocol for evaluating the faithfulness of input salience methods for text classification. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 976–991, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Dominique Brunato, Andrea Cimino, Felice Dell'Orletta, Giulia Venturi, and Simonetta Montemagni. 2020. Profiling-UD: a tool for linguistic profiling of texts. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 7145–7151, Marseille, France. European Language Resources Association.

Marc Brysbaert and Boris New. 2009. Moving beyond Kucera and Francis: A critical evaluation of current word frequency norms and the introduction of a new and improved word frequency measure for American English. *Behavior research methods*, 41:977–90.

Bram Bulté, Leen Sevens, and Vincent Vandeghinste. 2018. Automating lexical simplification in dutch. *Computational Linguistics in the Netherlands Journal*, 8:24–48.

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*.

Kevyn Collins-Thompson and Jamie Callan. 2005. Predicting reading difficulty with statistical language models. *Journal of the American Society for Information Science and Technology*, 56(13):1448–1462.

Marina Danilevsky, Kun Qian, Ranit Aharonov, Yannis Katsis, Ban Kawas, and Prithviraj Sen. 2020. A survey of the state of explainable AI for natural language processing. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 447–459, Suzhou, China. Association for Computational Linguistics.

Wietse de Vries, Andreas van Cranenburgh, Arianna Bisazza, Tommaso Caselli, Gertjan van Noord, and Malvina Nissim. 2019. BERTje: A Dutch BERT Model. arXiv:1912.09582.

Tovly Deutsch, Masoud Jasbi, and Stuart Shieber. 2020. Linguistic features for readability assessment. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 1–17, Seattle, WA, USA → Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Lijun Feng, Noémie Elhadad, and Matt Huenerfauth. 2009. Cognitively motivated features for readability assessment. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 229–237, Athens, Greece. Association for Computational Linguistics.

Anna Filighera, Tim Steuer, and Christoph Rensing. 2019. Automatic text difficulty estimation using embeddings and neural networks. In *Transforming Learning with Meaningful Technologies: 14th European Conference on Technology Enhanced Learning, EC-TEL 2019, Delft, The Netherlands, September 16–19, 2019, Proceedings 14*, pages 335–348. Springer.

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 (Volume 2: Short Papers)*, pages 458–463, Baltimore, Maryland. Association for Computational Linguistics.

David Kauchak. 2013. Improving text simplification language modeling using unsimplified text data. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1537–1546, Sofia, Bulgaria. Association for Computational Linguistics.

Emmanuel Keuleers, Marc Brysbaert, and Boris New. 2010. Subtlex-nl: A new measure for dutch word frequency based on film subtitles. *Behavior research methods*, 42(3):643–650.

Narine Kokhlikyan, Vivek Miglani, Miguel Martin, Edward Wang, Bilal Alsallakh, Jonathan Reynolds, Alexander Melnikov, Natalia Kliushkina, Carlos Araya, Siqi Yan, et al. 2020. Captum: A unified and generic model interpretability library for pytorch. *arXiv preprint arXiv:2009.07896*.

John Lee and Chak Yan Yeung. 2018. Personalizing lexical simplification. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 224–232, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Pierre Lison and Jörg Tiedemann. 2016. OpenSubtitles2016: Extracting large parallel corpora from movie and TV subtitles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 923–929, Portorož, Slovenia. European Language Resources Association (ELRA).

Qi Liu, Matt J. Kusner, and Phil Blunsom. 2020. A survey on contextual embeddings. *CoRR*.

Matej Martinc, Senja Pollak, and Marko Robnik-Šikonja. 2021. Supervised and unsupervised neural approaches to text readability. *Computational Linguistics*, 47(1):141–179.

George A. Miller. 1995. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41.

Gustavo Paetzold and Lucia Specia. 2016a. Benchmarking lexical simplification systems. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3074–3080, Portorož, Slovenia. European Language Resources Association (ELRA).

Gustavo Paetzold and Lucia Specia. 2017a. Lexical simplification with neural ranking. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 34–40, Valencia, Spain. Association for Computational Linguistics.

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

Gustavo H Paetzold and Lucia Specia. 2017b. A survey on lexical simplification. *Journal of Artificial Intelligence Research*, 60:549–593.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Jipeng Qiang, Yun Li, Yi Zhu, Yunhao Yuan, and Xindong Wu. 2019. Lexical simplification with pretrained encoders. In *AAAI Conference on Artificial Intelligence*.

Jipeng Qiang, Yun Li, Yi Zhu, Yunhao Yuan, and Xindong Wu. 2020. Lsbert: A simple framework for lexical simplification. *ArXiv*, abs/2006.14939.

Horacio Saggion, Sanja Štajner, Daniel Ferrés, Kim Cheng Sheang, Matthew Shardlow, Kai North, and Marcos Zampieri. 2022. Findings of the tsar-2022 shared task on multilingual lexical simplification. In *Proceedings of the Workshop on Text Simplification, Accessibility, and Readability (TSAR-2022)*, pages 271–283, Abu Dhabi, United Arab Emirates (Virtual). Association for Computational Linguistics.

Sarah Schwarm and Mari Ostendorf. 2005. Reading level assessment using support vector machines and statistical language models. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 523–530, Ann Arbor, Michigan. 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.

Punardeep Sikka and Vijay Mago. 2020. A survey on text simplification.

Robyn Speer, Joshua Chin, Andrew Lin, Sara Jewett, and Lance Nathan. 2018. Luminosoinsight/wordfreq: v2.2.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 3319–3328. JMLR.org.

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

Sowmya Vajjala and Detmar Meurers. 2012. On improving the accuracy of readability classification using insights from second language acquisition. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pages 163–173, Montréal, Canada. Association for Computational Linguistics.

Sowmya Vajjala Balakrishna. 2015. *Analyzing Text Complexity and Text Simplification: Connecting Linguistics, Processing and Educational Applications*. Ph.D. thesis, Universität Tübingen.

Vincent Vandeghinste and Bram Bulte. 2019. Linguistic proxies of readability: Comparing easy-to-read and regular newspaper dutch. *Computational Linguistics in the Netherlands*, 9:81–100.

Seid Muhie Yimam, Chris Biemann, Shervin Malmasi, Gustavo Paetzold, Lucia Specia, Sanja Štajner, Anaïs 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*, pages 66–78, New Orleans, Louisiana. Association for Computational Linguistics.

A Appendix

LR	No. Sents	Potential	Precision	Recall	F_1
		NNSEval			
LSBert		90.79	19.04	25.40	21.77
5e-6	1,000	87.03	17.03	22.72	19.47
5e-6	10,000	91.63	**20.17**	**26.91**	**23.06**
5e-6	50,000	90.79	**20.33**	**27.14**	**23.25**
5e-7	1,000	88.70	17.95	23.95	20.52
5e-7	10,000	88.28	17.24	23.00	19.71
5e-7	50,000	88.28	17.53	23.39	20.04
		LexMTurk			
LSBert		98.20	29.58	23.01	25.88
5e-6	1,000	95.80	25.64	19.94	22.43
5e-6	10,000	**98.60**	**32.16**	**25.01**	**28.14**
5e-6	50,000	**98.40**	**33.46**	**26.02**	**29.28**
5e-7	1,000	97.20	26.76	20.81	23.41
5e-7	10,000	96.00	25.89	20.13	22.65
5e-7	50,000	97.80	26.62	2070	23.29
		BenchLS			
LSBert		92.36	23.64	32.08	27.22
5e-6	1000	88.37	20.13	27.32	23.18
5e-6	10,000	**92.68**	**24.74**	**33.57**	**28.48**
5e-6	50,000	92.14	**25.68**	**34.84**	**29.56**
5e-7	1,000	90.42	21.33	28.95	24.57
5e-7	10,000	89.34	20.56	27.90	23.68
5e-7	50,000	91.50	21.42	29.07	24.67

Table 6: Performance of the Continual Pre-training Setup on the Benchmarking Datasets for Different Experimental Conditions

LR	Num Sents	Potential	Precision	Recall	F_1
		BenchLS			
LSBert		92.36	23.64	32.08	27.22
5e-5	1,000	87.19	21.77	29.54	25.06
5e-5	10,000	91.17	**24.92**	**33.82**	**28.69**
5e-6	1,000	89.99	21.07	28.59	24.26
5e-6	**10,000**	**93.54**	**25.93**	**35.17**	**29.85**
5e-6	50,000	92.03	**24.19**	**32.82**	**27.85**
5e-7	1,000	84.61	18.62	25.26	21.43
5e-7	10,000	88.91	20.23	27.45	23.29
5e-7	50,000	90.74	22.37	30.35	25.76
		LexMTurk			
LSBert		98.20	29.58	23.01	25.88
5e-5	1,000	97.00	28.54	22.20	24.97
5e-5	10,000	98.00	**32.22**	**25.06**	**28.19**
5e-6	1,000	96.60	26.76	20.81	23.41
5e-6	**10,000**	**98.80**	**33.48**	**26.04**	**29.29**
5e-6	50,000	98.20	**30.92**	**24.05**	**27.05**
5e-7	1,000	94.20	24.55	19.09	21.48
5e-7	10,000	96.00	25.98	20.21	22.73
5e-7	50,000	97.00	28.16	21.90	24.64
		NNSEval			
LSBert		90.79	19.04	25.40	21.77
5e-5	1,000	84.52	17.07	22.78	19.52
5e-5	10,000	**91.21**	**19.29**	**25.74**	**22.05**
5e-6	1,000	87.45	18.20	24.29	20.81
5e-6	**10,000**	**92.89**	**21.55**	**28.75**	**24.64**
5e-6	50,000	**92.05**	**19.71**	**26.30**	**22.53**
5e-7	1,000	81.59	14.81	19.77	16.93
5e-7	10,000	86.61	17.36	23.17	19.85
5e-7	50,000	87.45	18.74	25.01	21.43

Table 7: Performance of the Multi-Task Learning Setup on the Benchmarking Datasets for Different Experimental Conditions

Random Seed	Potential	Precision	Recall	$F1$
1	84.94	18.12	24.18	2.71
2	88.28	18.08	24.12	20.66
3	**92.89**	**21.55**	**28.75**	**24.64**
4	88.28	19.29	25.74	22.05
5	90.79	20.33	27.14	23.25
6	86.61	17.62	23.51	20.14
7	91.63	19.87	26.52	22.72
8	90.79	20.75	27.69	23.73
9	87.03	19.08	25.46	21.81
10	88.28	19.67	26.24	22.48
11	90.79	20.17	26.91	23.06
12	89.54	20.59	27.47	23.54
13	92.89	21.00	28.03	24.01
14	90.79	20.96	27.97	23.97
15	85.77	18.45	24.62	21.10
16	88.70	18.70	24.96	21.38
17	89.54	18.83	25.13	21.53
18	91.63	20.67	27.58	23.63
19	89.54	20.00	26.69	22.87
20	88.28	16.86	22.50	19.28
21	90.38	19.21	25.63	21.96
22	88.28	17.91	23.90	20.47
23	92.05	20.88	27.86	23.87
24	87.45	18.20	24.29	20.81
25	93.31	20.92	27.92	23.92
26	90.79	19.96	26.63	22.82
mean	89.59	19.53	26.06	22.32

Table 8: Multi-task learning results for NNSEval with varying random seeds. The learning rate is fixed at 5e-6 and fine-tuning is conducted on 10,000 sentences.

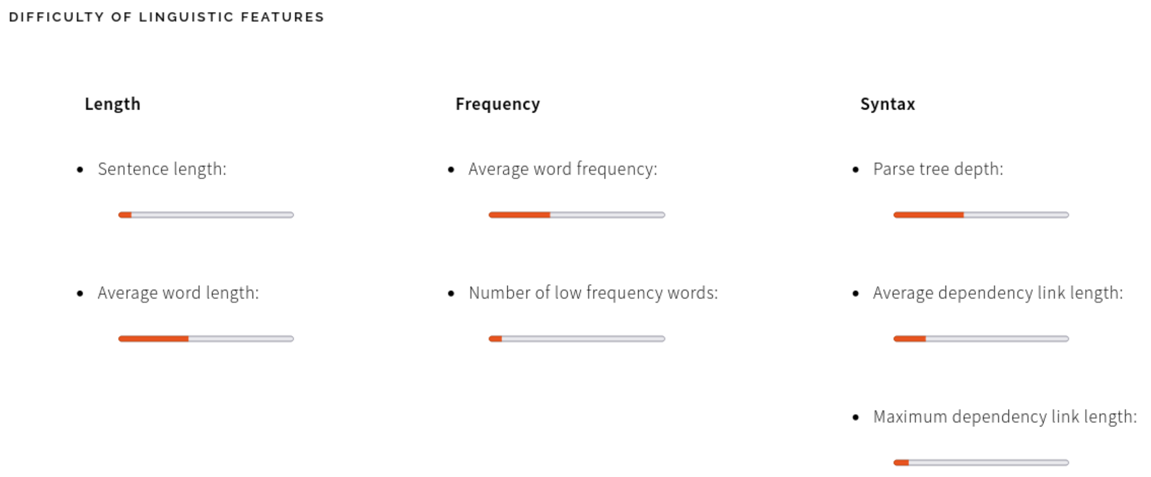

DIFFICULTY OF LINGUISTIC FEATURES
Length
Sentence length:
Average word length:
Frequency
Average word frequency:
Number of low frequency words:
Syntax
Parse tree depth:
Average dependency link length:
Maximum dependency link length:

Figure 4: Complexity features for the sentence *De treinverbinding tussen Gent en Brussel blijft hinder ondervinden*, taken from the newspaper De Standaard (translation: the train connection between Ghent and Brussels continues to be affected.) The sentence is classified as complex.