Entwicklung eines Korpus und einer Metrik zum Übersetzen von deutscher Sprache in Leichte Sprache

Fiete Scheel

28. Januar 2026

Inhaltsverzeichnis

1 Einleitung

1.1 Code und Daten

Der Code der in diesem Projekt verwendet wurde ist unter https://github.com/mcfiet/genai-project zu finden.

1.2 Motivation

> **Für die meisten ein ganz normaler Satz – für andere das Ende der Teilhabe.**

Beim Thema Barrierefreiheit denken viele an Rampen, Aufzüge und automatische Türen. Doch durch die Digitalisierung und die zunehmende Verlagerung von Informationen ins Internet reicht das längst nicht mehr aus.

Auch Sprache selbst kann eine Barriere darstellen. Menschen werden durch komplexe Texte, Fachbegriffe und lange Satzkonstruktionen vom Zugang zu Informationen ausgeschlossen. Dies betrifft nicht nur Personen mit geistigen Einschränkungen, sondern auch ältere Menschen, Menschen mit geringen Deutschkenntnissen oder Situationen, in denen schnelle und klare Information entscheidend ist. Leichte Sprache nützt daher weit mehr Menschen, als die die darauf angewiesen sind.

1.3 Problemstellung

Leichte Sprache ist entscheidend für barrierefreie Kommunikation, ihre Erstellung ist jedoch aufwendig und überwiegend manuell. Für das Deutsche fehlen zugleich robuste, datengetriebene Verfahren, die zuverlässig zwischen Leichter Sprache und Standardsprache unterscheiden oder die Qualität automatisierter Vereinfachung bewerten. Die Anforderungen sind hoch: Bereits einzelne schwer verständliche Wörter können den Leseprozess abbrechen.

1.4 Ziel dieser Arbeit

Ziel dieser Arbeit ist es, eine zuverlässige Metrik zu entwickeln, mit der bewertet werden kann, ob ein Satz oder eine Textsequenz Leichte Sprache ist oder nicht. Diese Metrik ist insbesondere für das spätere Training automatischer Übersetzer wichtig, da sie die Qualität der Vereinfachung objektiv messbar macht. Dazu wird ein reproduzierbarer Datensatz aufgebaut, eine Modellpipeline zur Klassifikation entwickelt und die Metrik auf synthetischen und realitätsnahen Daten getestet.

2 Stand der Forschung

Forschung zur automatischen Bewertung oder Metriken für Leichte Sprache ist bislang kaum vorhanden. Das macht das Thema zugleich spannender und ist ein zentraler Antrieb dieser Arbeit: Es gibt derzeit keine etablierte, datengetriebene Metrik für Leichte Sprache im Deutschen, die als Training- oder Bewertungssignal genutzt

wird. Der größte Teil der Literatur konzentriert sich stattdessen auf die Erzeugung und Ausrichtung von Korpora sowie auf Benchmarking in der Textvereinfachung. Einen wichtigen Beitrag für deutsche Ressourcen liefern Toborek et al. (2023) mit einem alignierten Korpus für Einfaches Deutsch. Auf breiterer Ebene liefert Klöser et al. (2024) eine aktuelle Übersicht zu multilingualer Textvereinfachung, Benchmarks und Methoden. Abseits dieser Arbeiten zur Datengrundlage gibt es bisher kaum Forschung, die explizit Metriken oder Bewertungssignale für Leichte Sprache entwickelt und evaluiert. Genau hier setzt diese Arbeit an.

3 Methodik

3.1 Datengrundlage

Als Ausgangspunkt dienten mehrere bestehende Quellen als Inspiration und erste Versuche: Der ADL-Datensatz mit parallel ausgerichteten Paaren (Standarddeutsch vs. Leichte Sprache) und Qualitätsmerkmalen, das Simple-German-Corpus sowie einfache Sätze aus dem Korpus, die aus den Docx-Dateien der Lebenshilfe kamen. In der praktischen Umsetzung wurde jedoch ein synthetischer Datensatz aufgebaut. Dafür werden mit `run_from_easy_url_list.py` aus einer eigenen URL-Liste Leichte-Sprache-Sätze extrahiert und anschließend, mit regelbasierten Prompts und LLM-gestützter Generierung (lokales Modell) synthetisch zu Paaren erweitert. Die synthetischen Daten liegen in Varianten vor (mehrere Prompt-Konfigurationen, JSON-Antwortformat, Wiederholungslogik bei Parsing-Fehlern), um Robustheit und Datenvielfalt zu erhöhen. Alle Daten werden in `data/` verwaltet und über Skripte reproduzierbar erzeugt.

3.2 Modellarchitekturen

Als Baselines dienen klassische Textmodelle (z. B. TF-IDF + Logistische Regression) sowie eine BiLSTM mit statischen Einbettungen (Hochreiter und Schmidhuber, 1997). Der Hauptfokus liegt auf Sentence-Transformer-Architekturen, die Paare von Sätzen in semantische Embeddings überführen und darauf einen Klassifikationskopf trainieren (Reimers und Gurevych, 2019). Untersucht werden mehrere Varianten: eingefrorener Encoder mit linearem bzw. größerem MLP-Kopf, feingetunter Encoder (vollständig oder nur letzte Schicht), sowie eine parameter-effiziente LoRA-Variante (Hu et al., 2021). Zusätzlich wird ein größerer Encoder (MPNet) als Vergleich herangezogen (Song et al., 2020).

3.3 Trainingssetup

Das Training erfolgt mit einem 80/10/10 Split und frühem Abbruch (Early Stopping) auf dem Validierungsset. Als Optimizer wird AdamW genutzt (Loshchilov und Hutter, 2019), die Lernraten werden je nach Modellvariante getrennt konfiguriert (Kopf vs. Encoder/Adapter). Die Bewertung erfolgt über Balanced Accuracy, um die Klassenverteilung der synthetischen Daten robust abzubilden. Laufzeiten

und Konvergenzverhalten werden mitgeloggt, um die Kosten-Nutzen-Abwägung der Varianten transparent zu machen.

4 Experimente

4.1 Versuchsaufbau

Die Experimente nutzen den synthetischen binären Datensatz `synthetic_normal_2_labeled.csv` mit einem 80/10/10 Split (20 % Test, 10 % der verbleibenden Daten als Validierung). Die Klassen sind balanciert und alle Sätze werden vor der Aufteilung zu Strings gecastet. Die Datengenerierung basiert auf regelbasierten Prompts und LLM-gestützten Paaren (einfach vs. normal) und wurde in mehreren Varianten erweitert (z. B. JSON-Antwortformat, Retry-Logik, mehrere Prompt-Konfigurationen). Dadurch entstehen mehrere TSV-Dateien, die fürs Training konsistent in ein gemeinsames, binär gelabeltes Format überführt werden.

Datenerzeugung und Pipeline Die Datenerzeugung ist in Skripten gekapselt und reproduzierbar. Zentrale Schritte sind das Sammeln einfacher Sätze aus vorhandenen Korpora, das Erzeugen synthetischer Paare und das anschließende Labeling. Der Fokus liegt auf einer robusten Pipeline, die Parser-Fehler abfängt und statistische Laufzeitdaten erfasst, um die Effizienz der Generierung zu bewerten.

Vorverarbeitung Für die LSTM-Baseline wird ein Whitespace-Tokenizer verwendet. Er erstellt ein Vokabular mit `min_freq=2` und einer Obergrenze von `20k` Tokens, plus `<pad>` und `<unk>`. Sequenzen werden auf `64` Tokens gekürzt bzw. aufgefüllt. Alle SBERT-Varianten nutzen den integrierten Tokenizer von `paraphrase-multilingual-MiniLM-L12-v2` (Wang et al., 2020).

Modelle

- **LSTM-Baseline**: Embedding-Dimension 128, Hidden-Dimension 128, eine bidirektionale Schicht, Dropout 0,3, maximale Sequenzlänge 64. Optimizer AdamW mit Lernrate 1
  $times10^{-3}$, Batch-Größe 64, Geduld 4, maximal 30 Epochen (Hochreiter und Schmidhuber, 1997; Loshchilov und Hutter, 2019).

- **SBERT eingefroren**: SentenceTransformer-Encoder eingefroren, nur linearer Klassifikationskopf trainiert (Reimers und Gurevych, 2019). Batch-Größe 32, Lernrate 5
  $times10^{-5}$, Weight Decay 0,01, Geduld 3, bis zu 100 Epochen.

- **SBERT vollständig feinabgestimmt**: Gleiches Setup, aber Encoder nicht eingefroren (vollständiges Fine-Tuning). Identische Optimizer-Parameter.

- **SBERT + LoRA**: Encoder bleibt eingefroren, wird aber mit LoRA-Adaptern ergänzt (Hu et al., 2021) (`r=50`, `alpha=32`, kein Dropout) auf Attention- und

Projektionsmodulen. Lernrate 5
$times10^{-5}$ für Kopf und Adapter, Batch-Größe 32, Geduld 3, bis zu 100 Epochen.

- **SBERT letzte Schicht frei**: Encoder eingefroren mit Ausnahme der letzten Transformerschicht (`UNFREEZE_LAST_N=1`); separate Encoder-Lernrate 5 $times10^{-5}$, Batch-Größe 32, Geduld 3, bis zu 100 Epochen.

Vergleichsbasis Neben den neuronalen Modellen werden klassische Baselines (TF-IDF + Logistische Regression) und Learning-Curve-Analysen in separaten Notebooks gepflegt, um Datenbedarf und Skalierungseffekte sichtbar zu machen. Diese dienen als Referenz, werden aber in der Hauptauswertung auf die wichtigsten Varianten reduziert.

5 Ergebnisse und Diskussion

5.1 Ergebnisse und Diskussion

Tabelle 1 fasst die deutschen Läufe zusammen. Angegeben ist die Balanced Accuracy (BAcc) auf dem Testsplit.

Modell	Epoche	BAcc
SBERT eingefroren (linearer Kopf)	100 (kein Early Stop)	0,719
SBERT eingefroren (größerer Kopf)	23	0,763
SBERT letzte Schicht frei (1)	11	0,834
LSTM-Baseline	7	0,863
SBERT + LoRA	10	0,906
SBERT voll feinabgestimmt (v.f.)	5	**0,913**
SBERT v.f. FlensGenGPTOSS Train Set	4	**0,926**
SBERT v.f. 1 simple to 5 normal (bigger) Train Set	5	**0,947**

Tabelle 1: Ergebnisse der binären Klassifikation.

Encoder-Training Das Einfrieren des SBERT-Encoders limitiert die Lernfähigkeit: mit einem linearen Kopf bleibt die BAcc bei 0,719, mit einem größeren MLP-Kopf steigt sie auf 0,763. Schon das Freigeben der letzten Schicht hebt die Leistung auf 0,834. Volles Fine-Tuning erreicht mit 0,913 den Bestwert nach nur fünf Epochen, was zeigt, dass der Datensatz in der SBERT-Repräsentation linear trennbar ist, aber von End-to-End-Anpassung profitiert.

Kopfvergleich Der größere MLP-Kopf verkürzt die Zeit bis zur besten Val-Accuracy und erhöht die Güte gegenüber dem linearen Kopf, bleibt aber deutlich hinter jeder Variante mit Encoder-Anpassung. Für einen eingefrorenen Encoder lohnt sich ein kräftigerer Kopf also, ersetzt aber kein Fine-Tuning.

Parameter-effizientes LoRA Die LoRA-Variante (r=50, alpha=32) erreicht 0,906 und liegt damit dicht hinter dem vollständigen Fine-Tuning, stoppt aber nach zehn Epochen (Hu et al., 2021). Sie bietet ein günstiges Verhältnis von Qualität zu Rechenaufwand, da nur Adapter und Kopf trainiert werden.

Nicht-Transformer-Baseline Die einfache BiLSTM mit statischen Einbettungen erzielt 0,863 BAcc (Hochreiter und Schmidhuber, 1997). Sie übertrifft den eingefrorenen SBERT-Kopf deutlich, bleibt aber hinter allen Varianten mit Encoder-Anpassung zurück. Dies zeigt, dass ein leichtgewichtiges Sequenzmodell auf dem synthetischen Korpus konkurrenzfähig sein kann, Transformer-Feinabstimmung jedoch weiterhin überlegen ist (Vaswani et al., 2017).

Trainingsdynamik Alle adaptiven SBERT-Modelle lösten Early Stopping zwischen Epoche 5 und 11 aus, trotz 100 möglicher Epochen. Das deutet auf schnelle Konvergenz und begrenztes Overfitting bei der gewählten Geduld hin. Der eingefrorene Encoder lief mit linearem Kopf alle 100 Epochen, mit größerem Kopf stoppte er nach 23 Epochen, blieb aber klar hinter den Varianten mit freigegebenem Encoder zurück.

Rechenaufwand Tabelle 2 zeigt die Laufzeit bis zur besten Validierungs-Accuracy. Volles Fine-Tuning erreicht seine beste Validierungsleistung nach nur rund elf Minuten und ist damit trotz größerer Parameterzahl effizienter als die eingefrorenen Varianten: der lineare Kopf braucht über eine Stunde und bleibt schwach, der größere Kopf stoppt nach 15,7 Minuten und holt etwas mehr heraus. Die LoRA-Variante liegt mit etwa 20 Minuten dazwischen und bietet damit einen guten Kompromiss aus Qualität und Laufzeit. Die einfache LSTM-Baseline konvergiert in unter einer Minute, erreicht aber nicht die maximale SBERT-Qualität.

Einordnung der Ergebnisse Insgesamt zeigt sich ein klares Bild: Ein eingefrorener Encoder limitiert die Leistung deutlich, während bereits partielle Freigaben (letzte Schicht) starke Zugewinne bringen. LoRA erreicht nahezu die Qualität des vollständigen Fine-Tunings bei reduziertem Rechenaufwand, was sie als pragmatische Option für ressourcenbegrenzte Settings auszeichnet (Hu et al., 2021). Die LSTM-Baseline belegt, dass das synthetische Signal stark genug ist, um einfache Sequenzmodelle konkurrenzfähig zu machen, bleibt aber unter dem Optimum der Transformer-Varianten (Vaswani et al., 2017).

Größere synthetische Datensätze Zwei zusätzliche Runs mit voll feinabgestimmtem SBERT zeigen den Effekt größerer und variablerer Trainingsdaten. Das auf dem FlensGenGPTOSS-basierten Datensatz trainierte Modell (balanciert, 0,5/0,5; $n = 63\,510$ mit $31\,755/31\,755$) erreicht eine BAcc von 0,926 nach vier Epochen. Der größere 1:5-Datensatz (ein einfacher Satz, fünf Normalvarianten; deutliche Klassen-Unausgewogenheit von ca. 1/6 zu 5/6; $n = 191\,538$ mit $31\,923/159\,615$) wurde mit Minority-Oversampling trainiert und erzielt eine BAcc von 0,947 auf dem Testsplit (Accuracy 0,966; Recall: 0,920). Damit steigen die Kennzahlen mit wachsender

Datenmenge weiter, allerdings wird der Nutzen durch die Klassenbalance und den Sampling-Ansatz beeinflusst.

Grenzen der Uebertragbarkeit Die Auswertung basiert auf synthetisch erzeugten Paaren. Damit ist zwar ein kontrolliertes, balanciertes Training möglich, die Uebertragbarkeit auf echte Uebersetzungen muss jedoch separat validiert werden. Insbesondere Stil- und Domäneneffekte aus realen Quellen können die Metrik beeinflussen und sollten in einer erweiterten Evaluation adressiert werden.

Modell	Minuten bis beste Val-Acc	Val-Acc
SBERT eingefroren (linearer Kopf)	82,70	0,707
SBERT eingefroren (größerer Kopf)	15,71	0,760
SBERT letzte Schicht frei (1)	9,35	0,827
LSTM-Baseline	0,27	0,863
SBERT + LoRA	20,24	0,902
SBERT voll feinabgestimmt	11,01	0,903

Tabelle 2: Zeit bis zur besten Validierungs-Accuracy (Minuten) und erreichter Val-Accuracy.

6 Fazit und Ausblick

6.1 Fazit und Ausblick

Die Arbeit zeigt, dass eine datengetriebene Metrik für Leichte Sprache erfolgreich auf synthetischen Paaren trainiert werden kann und dass Transformer-Modelle mit Encoder-Anpassung die beste Balanced Accuracy erreichen (Vaswani et al., 2017). Gleichzeitig bleibt die Pipeline nachvollziehbar und reproduzierbar, sodass sowohl Datenerzeugung als auch Modelltraining systematisch erweitert werden können. Die zusätzlichen Runs auf größeren synthetischen Datensätzen (HTTP/FlensGenGPTOSS sowie 1:5-Varianten mit Oversampling) bestätigen den Trend: Mehr Daten steigern die Testleistung, allerdings wird der Zugewinn durch Klassenbalance und Sampling-Strategie beeinflusst. Die Ergebnisse legen nahe, dass die Metrik als Reward-Funktion oder Bewertungssignal in Übersetzungssystemen einsetzbar ist, sofern eine externe Validierung bestätigt, dass das synthetische Signal auf reale Daten übertragbar ist.

Kritik an der Arbeit

Die Evaluierung basiert stark auf synthetischen Daten, die zwar kontrolliert und umfangreich sind, aber reale Übersetzungsfehler und stilistische Varianz nur begrenzt abbilden. Die Generalisierung auf manuell alignierte oder annotierte Korpora wurde noch nicht systematisch gemessen. Zudem kann die starke Abhängigkeit von Prompt-Design und Generierungsregeln zu verzerrten Lernsignalen führen.

Learnings

Die Experimente zeigen, dass die Modelldynamik stark vom Encoder-Training abhängt: Ein eingefrorener Encoder limitiert die Leistung deutlich, während partielle Freigaben und LoRA große Zugewinne bei moderatem Aufwand liefern (Hu et al., 2021). Synthetische Daten können den Prozess beschleunigen, sollten aber früh durch reale Beispiele validiert werden, um Fehlerbilder nicht zu verfestigen. Insgesamt ist das ein sehr guter Schritt, trotzdem war die Zeit für Datensatzerstellung und für den Aufbau der Metrik zu knapp, sodass man trotz der Ergebnisse das Gefühl behält, erst an der Oberfläche gekratzt zu haben. Vielleicht hätte man sich noch mehr auf die Datenerstellung oder auf den Aufbau der Metrik konzentrieren sollen, anstatt zu viele Experimente durchzuführen.

Ausblick

Als nächste Schritte sollten die synthetischen Generatoren weiter ausgebessert werden um die Qualität der Datensätze zu erhöhen. Darauf aufbauend ist eine Validierung auf hand-alignierten und annotierten Datensätzen notwendig, um die Übertragbarkeit zu testen. Methodisch bietet sich eine kombinierte Metrik an, die semantische Übereinstimmung mit Lesbarkeits- und Regelmerkmalen verbindet. Abschließend kann die Metrik als Reward-Funktion in einer Übersetzungspipeline getestet werden, um den praktischen Nutzen direkt zu evaluieren.

Literatur

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., & Chen, W. (2021). LoRA: Low-Rank Adaptation of Large Language Models. *arXiv preprint*. https://arxiv.org/abs/2106.09685

Klöser, J., Mallinson, J., Štajner, S., Shardlow, M., & Saggion, H. (2024). Multilingual Text Simplification: Benchmarks and Methods. *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL 2024)*. https://arxiv.org/pdf/2402.10675

Loshchilov, I., & Hutter, F. (2019). Decoupled Weight Decay Regularization. https://arxiv.org/abs/1711.05101

Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. https://arxiv.org/abs/1908.10084

Song, K., Tan, X., Qin, T., Lu, J., & Liu, T.-Y. (2020). MPNet: Masked and Permuted Pre-training for Language Understanding. https://arxiv.org/abs/2004.09297

Toborek, V., Busch, M., Boßert, M., Bauckhage, C., & Welke, P. (2023). A New Aligned Simple German Corpus. https://aclanthology.org/2023.acl-long.638

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention Is All You Need. https://arxiv.org/abs/1706.03762

Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., & Zhou, M. (2020). MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. https://arxiv.org/abs/2002.10957