Heinrich-Heine-Universität Düsseldorf
Philosophische Fakultät
Computerlinguistik - Institut Linguistik V
Dr. Yulia Zinova

Deep Learning in NLP
WiSe24/25
Dr. Yulia Zinova

Automatic Prediction of Language Proficiency (CEFR Levels) on German Data with Linguistic Features

Name: Nurhayat Altunok
Matrikelnummer: 3083370
nualt100@uni-duesseldorf.de
Computerlinguistik, 7. Semester
BA, PO 2018

Abschlussprüfung für: CL3 Mathematische Linguistik, angemeldet in LSF am 08.01.2025

Datum der Abgabe: 13. Januar 2025

Automatic Prediction of Language Proficiency (CEFR Levels) on German Data with Linguistic Features

Nurhayat Altunok
Heinrich-Heine-Universität Düsseldorf
nualt100@uni-duesseldorf.de

Abstract

This project aims to develop a machine learning model for classifying text into levels of the Common European Framework of Reference for Languages (CEFR). Two machine learning architectures —an LSTM-based model and a feedforward neural network model— are evaluated on a dataset annotated with CEFR levels. The primary objective is to determine the model that achieves the best performance in terms of accuracy, precision, recall, and F1-score. The accurate classification of text into these levels can be beneficial for language learning applications, educational assessments, and personalized learning experiences.

1 Introduction

This study compares two machine learning models for classifying text data from various corpora (*MERLIN, Falko, DISKO*) and personally generated dataset by *Ahlers*, covering proficiency levels from A1 to C2. The goal is to evaluate the models' performance in classifying text into CEFR levels using preprocessed and enriched data.

The first model is a feedforward neural network, which is suitable for tasks where the input features are not sequential. It consists of fully connected layers and uses ReLU activation functions and dropout for regularization. The second model leverages Long Short-Term Memory (LSTM) networks, a type of recurrent neural network (RNN) designed to handle sequential data. The LSTM model is effective in capturing temporal dependencies and patterns in the text data. Both models are implemented using PyTorch and are evaluated using metrics such as accuracy, precision, recall, and F1-score to assess their performance in classifying text into CEFR levels.

The paper is structured around three key objectives:

1. Identifying the model that demonstrates superior performance in classifying text into CEFR levels.
2. Investigating the relationship between linguistic features and classification accuracy.
3. Highlighting the challenges and opportunities associated with automated CEFR classification.

By addressing these goals, the study provides insight into the effectiveness of machine learning approaches for assessing language proficiency.

2 Dataset

The dataset used in this study is a collection of essays written in German, labeled with varying proficiency levels based on the CEFR, which classifies language proficiency into six levels: A1, A2, B1, B2, C1, and C2. In this project, I grouped these levels into three broader categories: NCEFR (A, B, C). Consequently, I analyzed the data using both the NCEFR and CEFR categorizations.

The final combined dataset, named *DataCombined*, originally consisted of 2,092 essays. To optimize the dataset for the machine learning model, I removed 44 essays from the B2 level, as there was an abundance of data from this level compared to the others. This reduced the dataset to 2,048 essays, which named as *DataCleanUP*. The distribution of texts across CEFR levels can be seen in Table 1.

Dataset	A1	A2	B1	B2	C1	C2
Merlin	57 texts	306 texts	331 texts	293 texts	42 texts	4 texts
FalkoC2	0 texts	0 texts	0 texts	0 texts	0 texts	95 texts
FalkoMixed	0 texts	0 texts	0 texts	87 texts	0 texts	70 texts
AhlersA1	122 texts	0 texts	0 texts	0 texts	0 texts	0 texts
DISKO	0 texts	0 texts	27 texts	216 texts	294 texts	58 texts
My Data						
DataCombined	179 texts	306 texts	358 texts	596 texts	426 texts	227 texts
DataCleanUP	179 texts	306 texts	358 texts	552 texts	426 texts	227 texts

Table 1: Text Distribution Across CEFR Proficiency Levels

2.1 Features and Annotations

This project involved two key datasets: *DataFeaturesDetUP* and *DataCleanUP*. These datasets were derived from a raw text dataset DataCombined and were carefully designed to serve specific purposes. While *DataFeaturesDetUP* provided a detailed breakdown of linguistic and structural features extracted from the text, *DataCleanUP* was the result of selecting necessary features to build the final dataset for training the machine learning model. Below, I describe the structure of these datasets and the role of each feature in the project, followed by an explanation of how the final dataset was created.

Details of the Features

DataFeaturesDetUP is a comprehensive dataset created through detailed feature extraction and linguistic analysis of the raw text data. The aim of this dataset was to provide interpretability and insights into the linguistic, grammatical, readability, syntax-based, and structural properties of the text. Each column corresponds to a specific feature that was extracted and processed during the exploratory phase of the project. Below is a detailed description of the 21 features included in this dataset:

Grammar Related Features

These features reflect the overall linguistic quality of the text and were particularly useful for analyzing text complexity and the need for preprocessing. The total number of grammar-related errors in the text, as detected by the *language tool python* library and calculated by taking into account the grammatical errors below:

- **Style Errors:** Instances where the text violated stylistic conventions for clarity or appropriateness.
- **Misspelling Errors:** The count of misspellings detected in the text.
- **Duplication Errors:** The count of repeated words or phrases identified in the text, helping to detect redundancy and improve clarity.
- **Uncategorized Errors:** Grammar issues that do not fit into specific predefined categories, offering a broader overview of potential errors in the text.
- **Whitespace Errors:** Issues in the text involving improper use of spacing.
- **Typographical Errors:** Mistakes related to typographical inconsistencies or errors, also including improper use of spacing.

Whitespace and typographical errors were not considered as error types in this project and were extracted from the *Total Errors* column. These errors cannot be regarded as valid since they result from the dataset's inclusion of unnecessary whitespaces. Consequently, they do not reflect genuine errors made by language learners. Additionally, duplication errors were infrequent and were therefore removed from the dataframe.

Lexical Features

These features helped quantify the richness and diversity of the vocabulary used in the text.

- **Type-Token Ratio (TTR):** A measure of linguistic diversity, calculated as the number of unique words (types) divided by the total number of words (tokens) in the text. Higher values indicate a more varied vocabulary.
- **Lexical Density:** The proportion of content words (nouns, verbs, adjectives, adverbs) out of the total number of words. It is used to assess the complexity and informativeness of the text.

Readability Metrics

These metrics provided insights into the complexity and accessibility of the text.

- **Flesch Reading Ease:** A measure of how easy a text is to read, where higher scores correspond to simpler texts. Scores range from 0 (very difficult) to 100 (very easy).
- **Wiener Sachtextformel 1-4:** Four variations of the Wiener Sachtextformel readability scores that indicate the difficulty of understanding a text. Values range from 4 (easy) to 15 (difficult).

Syntactic and Structural Features

- **Average Sentence Length:** Represents the average number of words per sentence in the text. Longer sentences often indicate more complex syntax.
- **Verb Conjugations:** Counts for different tense and mood categories (e.g., present, past, future, conditional, subjunctive) extracted using SpaCy's morphological analysis.
- **Unique Tenses:** The number of distinct tenses used in the verbs of a text.
- **Dependency Counts:** The frequency of different syntactic dependency relations (e.g., subjects, objects, modifiers). Dependency relations considered for the model: Noun Kernel, Coordinating Conjunction, Accusative Object, Subject, Conjunctions, Predicative, Modifier of Noun Phrase, Morphological Particle, Root, Complementizer, Modifier, Object Clausal. Details about dependency relations, based on the TIGER annotation schema, are provided in the Appendix.
- **Tree Depth:** The maximum depth of the dependency tree in the text, revealing the complexity of sentence structures.
- **Average Dependency Distance:** The average distance between tokens and their syntactic heads in the dependency tree, which reflects syntactic arrangement and coherence.

3 Methodology

This research explores multi-level CEFR classification tasks using machine learning and deep learning techniques. The overall methodology involves preprocessing the dataset, feature extraction for both text and numerical data, model development and implementation using neural networks in PyTorch, and evaluation using established classification metrics. Two specific architectures have been implemented: a feed-forward neural network and an LSTM-based network. Key experiments are conducted to evaluate the performance of both models for classification tasks at 3 CEFR levels (A, B, C) and 6 CEFR levels (A1, A2, B1, B2, C1, C2).

3.1 Data Preparation

The dataset used in this study contains both textual and numerical features. The target variable represents CEFR classification levels, grouped into both 3 levels (A, B, C) and 6 levels (A1, A2, B1, B2, C1, C2).

The following preprocessing steps were applied to ensure the data's readiness for training the models:

1. **Handling Missing Values:** Missing values in the dataset were replaced with empty strings for consistency.
2. **Text Features:** The text column was extracted, and term frequency-inverse document frequency (TF-IDF) was applied for feature extraction. The *TfidfVectorizer* was configured to

generate a maximum of 5,000 features, and the output was converted to an array for model integration.

3. **Numerical Features:** The columns containing numerical data were normalized using *StandardScaler* to ensure consistency in feature scaling. This resulted in scaled numerical features ready for combination with textual features.

The textual features from TF-IDF and the normalized numerical features were combined into a single feature matrix (*combined features*) to be used as input for the models.

3.2 Implementation of Models

The implementation focused on two model architectures: a feed-forward neural network and an LSTM-based approach. Experiments were conducted to evaluate and compare their performance on text classification tasks.

Feed-forward Neural Network (FFNN)

The feedforward neural network was implemented using PyTorch and consists of the following components:

1. **Architecture Layout**
   - **Input layer:** Accepts the combined features (TF-IDF and normalized numerical data).
   - **Two hidden layers:** The first layer consists of 8 neurons, followed by a second layer with 4 neurons, with *ReLU* activation applied after each layer.
   - **Dropout regularization:** A dropout rate of 0.25 was applied after each hidden layer to prevent overfitting.
   - **Output layer:** For 3 CEFR levels: 3 neurons with a *Softmax* activation function. For 6 CEFR levels: 6 neurons with a *Softmax* activation function.
2. **Loss Function and Optimizer**
   CrossEntropyLoss was utilized as the loss function, while the *Adam* optimizer was applied with a learning rate of 0.001.

Training was performed for 100 epochs with early stopping to prevent overfitting by monitoring validation loss. The model with the lowest validation loss was saved for evaluation.

Long Short-Term Memory Based Model (LSTM)

The second approach utilized a LSTM-based architecture in PyTorch, designed to capture sequential dependencies present in the data:

1. **Architecture Layout**
   - **Input layer:** Processes the combined feature matrix. Each feature vector is treated as one sequence in a batch.
   - **LSTM Layer:** This layer contained 8 hidden units with 2 stacked layers and a dropout rate of 0.25.
   - **Fully Connected Layer:** A single fully connected (dense) layer maps the final LSTM output to the target size (3 neurons for 3 CEFR levels, or 6 neurons for 6 levels).
   - **Output Activation:** A *Softmax* activation function was applied to generate probability distributions over the CEFR levels.
2. **Loss Function and Optimizer**
   As with the FFNN, the *CrossEntropyLoss* and *Adam* optimizer (with a learning rate of 0.001) were employed here as well.

The LSTM model was also trained for 100 epochs, with early stopping implemented based on validation loss.

3.3 CEFR Mapping and Dataset Splitting

The target variable in the dataset was encoded both for 3 CEFR levels and 6 CEFR levels.

The mapping was done as Table 2:

Levels	Mapping
3 Levels	A $\rightarrow$ 1, B $\rightarrow$ 2, C $\rightarrow$ 3
6 Levels	A1 $\rightarrow$ 1, A2 $\rightarrow$ 2, B1 $\rightarrow$ 3, B2 $\rightarrow$ 4, C1 $\rightarrow$ 5, C2 $\rightarrow$ 6

Table 2: CEFR Mapping

After preprocessing and feature extraction, the dataset was split as follows:

- **Training Set:** 60% of the data for model training.
- **Validation Set:** 20% of the data for validation during training to monitor overfitting.
- **Test Set:** 20% of the data for the final evaluation of model performance.

The splits ensured a balanced distribution of classes across all subsets. Further, the data was converted to PyTorch tensors and wrapped into *DataLoader* objects for efficient batch loading during training.

4 Experiments

The experiments were conducted to classify texts into 3 CEFR levels (A, B, C) and 6 CEFR levels (A1, A2, B1, B2, C1, C2). Two main architectures were utilized: a feedforward neural network (FFNN) and a Long Short-Term Memory (LSTM)-based network. The purpose of the experiments was to evaluate the effectiveness of the two approaches under several configurations.

4.1 Hardware and Software Environment

All experiments were conducted on a standard computing setup equipped with a CPU, without the use of specialized hardware accelerators such as GPUs. The software environment consisted of Python version 3.12.0 as the primary programming language. Key libraries used for preprocessing, training, and evaluation included *PyTorch, scikit-learn, pandas, matplotlib, seaborn, numpy, textstat, and spaCy.* These tools facilitated the implementation and analysis of the experimental framework.

4.2 Hyperparameters

The following hyperparameters were optimized and applied during model training:

1. **Feedforward Neural Network (FFNN)**
   - Learning rate: 0.001
   - Batch size: 16, 32, and 64 (tested across different configurations)
   - Number of hidden neurons: 8 in the first layer and 4 in the second layer
   - Dropout rate: 0.25
   - Total epochs: 100
2. **Long Short-Term Memory Based Model (LSTM)**
   - Learning rate: 0.001
   - Batch size: 32
   - Number of hidden units: 8
   - Number of LSTM layers: 2
   - Dropout rate: 0.25
   - Total epochs: 100

For both models, early stopping was applied with a patience of 3 epochs, based on validation loss.

4.3 Input Configurations

Key hyperparameters were optimized during model training to enhance performance. The feature matrix combined TF-IDF vectors (up to 5,000 dimensions) with normalized numerical features. Two target configurations were tested: Classification into three CEFR levels and into six CEFR levels, enabling a detailed evaluation of the model's proficiency prediction capabilities.

5 Results and Analysis: Evaluation Metrics and Confusion Matrix

The performance of the models was evaluated using commonly used metrics to ensure robust assessment across multiple dimensions. These metrics include accuracy, precision, recall, and F1-score, as well as detailed analysis through confusion matrices. All metrics were computed for both the 3-level classification task (A, B, C) and the 6-level classification task (A1, A2, B1, B2, C1, C2).

The confusion matrices, shown in the Figure 1, Figure 2, Figure 3, Figure 4, provide further insight into how often the models mislabeled data into neighboring or unrelated classes, highlighting areas where the models excelled or struggled.

5.1 Feedforward Neural Network (FFNN)

The 3-level task involves classification into A (beginner), B (intermediate), and C (advanced) categories.

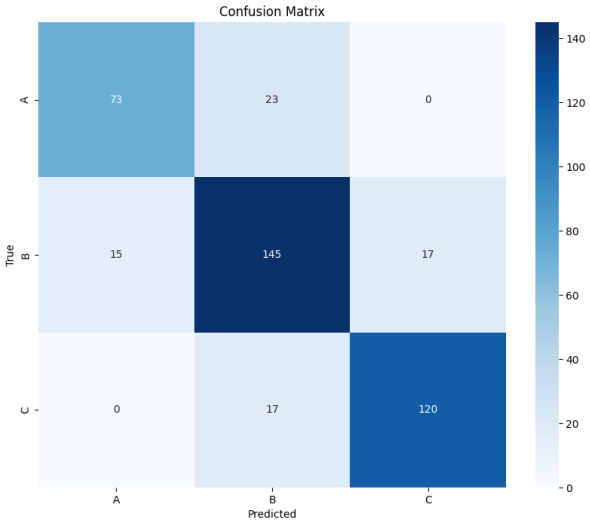

Figure 1: CM for 3-Level

The model performed consistently well across all metrics for the **3-level classification task**. The classification report and confusion matrix revealed the following insights.

Accuracy:	82.4%
Precision:	82.5%
Recall:	82.4%
F1-score:	82.4%

- **Class A:** 73 were correctly classified as A, with a recall of 76%. 23 were misclassified as B, indicating overlap between A and B. No A samples were misclassified as C.
- **Class B:** 145 samples were correctly classified as B, with a recall of 82%. 15 were misclassified as A, and 17 as C, showing moderate confusion with both beginner and advanced levels.
- **Class C:** 120 samples were correctly classified as C, with a recall of 88%. 17 were misclassified as B, but none were misclassified as A, reflecting strong separation of C from lower levels.

The 6-level task provides more detailed categorization into A1, A2, B1, B2, C1, and C2 levels, increasing the complexity of the classification.

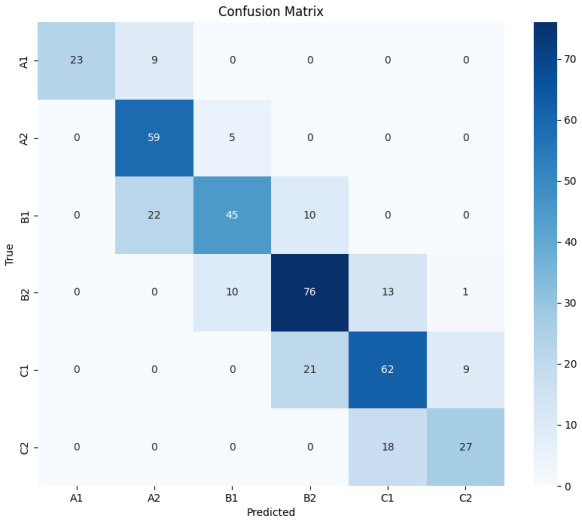

Figure 2: CM for 6-Level

The FFNN struggled slightly more with the 6-level classification. The classification report and confusion matrix revealed the following insights.

Accuracy:	71.2%
Precision:	72.4%
Recall:	71.2%
F1-score:	71.0%

- **Class A1:** The FFNN achieved excellent precision (100%) but had a lower recall (72%). The model correctly classified 23 samples but misclassified 9 as A2, indicating overlap between A1 and A2.
- **Class A2:** Precision for A2 was 66%, with a higher recall of 92%, meaning the model was effective at identifying most A2 instances but occasionally misclassified other levels as A2: 59 were correctly classified, with 5 misclassified as B1, reflecting some confusion with more complex texts.
- **Class B1:** For B1, both precision and recall were moderate, with an F1-score of 66%, revealing overlap with nearby classes. 45 were correctly identified, but

22 as A2 and 10 as B2, showing difficulty distinguishing intermediate levels.

- **Class B2:** The FFNN handled B2 reasonably well, achieving a precision of 71% and a recall of 76%, leading to a balanced F1-score of 73%: 76 were classified correctly, though 10 were predicted as B1, and 13 as C1, highlighting confusion with neighboring higher levels.

- **Class C1:** Precision and recall for C1 were 67%, indicating reasonable performance in identifying this class, though slightly less consistent than B2. 62 were correctly classified, but 21 were misclassified as B2, presenting challenges in distinguishing advanced texts with intermediate overlaps.

- **Class C2:** It was the weakest-performing class for the FFNN, with both a precision and recall below 60%. This indicates challenges in separating C2: Only 27 were accurately classified, while 18 were misclassified as C1, reflecting significant overlap in the most advanced levels.

5.2 LSTM-Based Neural Network

The LSTM model performed slightly better than the FFNN for the **3-level task**.

The model performed better than the FFNN on A-level classification while maintaining similar performance for C-levels. A slightly lower recall was observed for the B-level classification compared to FFNN.

Accuracy:	83.4%
Precision:	83.5%
Recall:	83.4%
F1-score:	83.2%

- **Class A:** The LSTM performed exceptionally well for A, with 88 correctly classified as A, corresponding to a recall of 92%. 8 instances were misclassified as B, showing significantly less confusion with B compared to the FFNN.

- **Class B:** Out of 177 B samples, the model correctly classified 133, achieving a recall of 75%. 21 were misclassified as A, and 23 as C, showing similar issues as the FFNN but to a lesser extent.

- **Class C:** 121 C-level samples were correctly identified, yielding a recall of 88%. 16 were misclassified as B, indicating slight confusion similar to the FFNN. None were misclassified as A.

The LSTM struggled more in the **6-level task**, slightly underperforming compared to the FFNN.

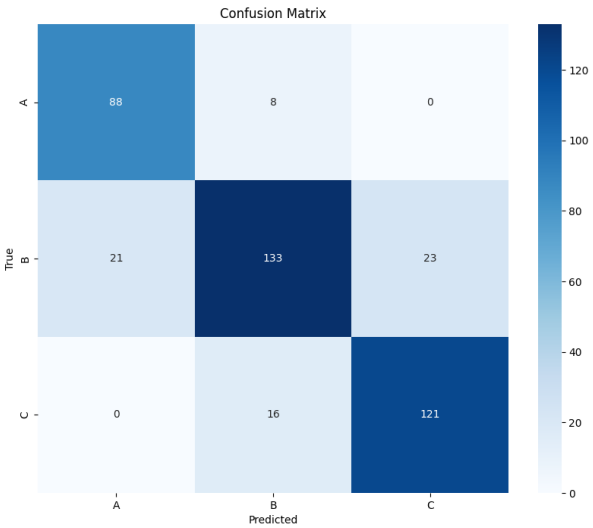

Figure 3: CM for 3-Level

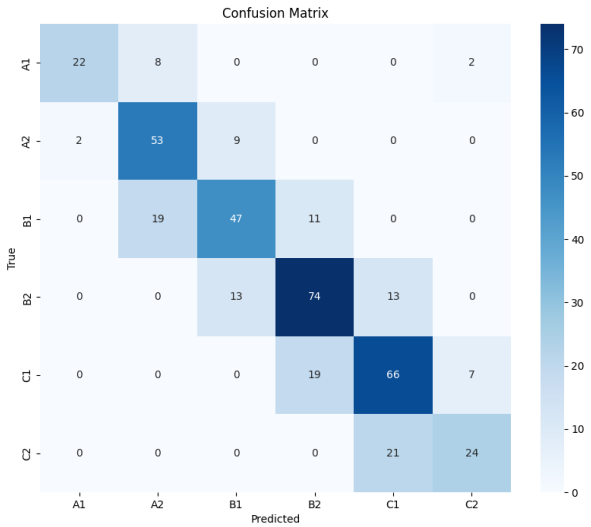

Figure 4: CM for 6-Level

It performed slightly worse in terms of overall accuracy for the 6-level task but showed some interesting class-wise behaviors.

Accuracy: 69.7%
Precision: 70.4%
Recall: 69.7%
F1-score: 69.5%

- **Class A1:** The LSTM achieved a high precision (92%) but had a low recall (69%). The model correctly classified 22 samples, but 8 were misclassified as A2 and 2 were misclassified as C2, due to feature overlaps at these levels.
- **Class A2:** It had a precision of 66% and a recall of 83% , which is consistent with the FFNN. However, A2 instances were often confused with nearby levels such as B1 and A1: 53 were correctly classified, with 9 misclassified as B1, suggesting difficulty distinguishing intermediate cases.
- **Class B1:** This class saw a notable decline in performance compared to the FFNN, with a recall of only 61% and an F1-score of 64% . This suggests B1 was the most challenging class for the LSTM to identify correctly: 47 were correctly identified, though 19 were predicted as A2, and 11 as B2.
- **Class B2:** The LSTM achieved a precision score of 71% and a recall of 74% , performing comparably to the FFNN for this level: 74 were classified correctly, with 13 were misclassified as B1 and 13 were misclassified as C1.
- **Class C1:** It maintained consistent performance with a precision of 66% and a recall of 72%: 66 were correctly classified, but 19 were misclassified as B2 and 7 were misclassified as C2, indicating difficulty distinguishing finer nuances in advanced texts.
- **Class C2:** It considered to be a difficult class to classify, with a recall of only 53%, though precision improved slightly to 73%, indicating a better balance than the FFNN: Only 24 were correctly classified, while 21 were misclassified as C1.

6 Model Comparison

The performance of the models varied based on the classification task. As shown in Table 3, the LSTM model outperformed the FFNN model for the 3-level classification task, achieving higher accuracy, precision, recall, and F1-score. In contrast, the FFNN model excelled in the 6-level classification task across all metrics. The FFNN is better for detailed classifications with overlapping features, while the LSTM is more effective for broader distinctions, leveraging sequential dependencies in tasks like the 3-level classification.

Task	Model	Accuracy	Precision	Recall	F1-score
3 Levels	FFNN	82.4%	82.5%	82.4%	82.4%
	LSTM	83.4%	83.6%	83.4%	83.2%
6 Levels	FFNN	71.2%	72.4%	71.2%	71.0%
	LSTM	69.7%	70.4%	69.7%	69.5%

Table 3: Performance Metrics for 3-Level and 6-Level Classification Tasks

Both models demonstrated strong performance in classifying beginner (A-level) and advanced (C-level) texts. However, misclassifications were common among neighboring levels, such as A1 with A2, B1 with B2, and C1 with C2, highlighting the challenge posed by overlapping linguistic features. Notably, the LSTM model exhibited slightly better clarity in separating A-level and C-level texts compared to the FFNN model. A comprehensive and detailed explanation of the information summarized here is provided in the Appendix.

3-Level Classification Task: The LSTM model demonstrated notable strengths in separating beginner (A-level) texts from intermediate (B-level) texts and achieved high recall for advanced (C-level) texts. However, it faced moderate challenges in

distinguishing intermediate (B-level) texts from adjacent classes. In contrast, the FFNN model achieved results comparable to the LSTM for advanced (C-level) texts but struggled more in differentiating beginner (A-level) texts from intermediates.

6-Level Classification Task: In the more granular 6-level classification task, the LSTM model excelled in capturing sequential relationships for beginner-level texts (A1 and A2). Despite this, it encountered difficulties with intermediate and advanced levels (B1, B2, C1, and C2), largely due to overlapping linguistic features. Misclassifications frequently occurred between neighboring classes, such as A2 being classified as B1 or C1 as C2. The FFNN model, on the other hand, performed consistently better across intermediate and advanced levels, demonstrating greater stability in separating classes with overlapping features. However, its limited ability to model sequential relationships impacted its effectiveness in classifying beginner-level texts.

7 Challenges and Limitations

The inclusion of specific features played a crucial role in optimizing the performance of the model. A significant example from this project involves the dependency relations utilized: *Noun Kernel, Coordinating Conjunction, Accusative Object, Subject, Conjunctions, Predicative, Modifier of Noun Phrase, Morphological Particle, Root, Complementizer, Modifier, Object Clausal.*

During feature selection, I reduced the initial set of 56 features to 21, guided by feature importance derived using the *Random Forest algorithm*. Although features like Whitespace Errors and Typographical Errors were found to be important, they had to be excluded due to inaccurate data. This was necessary to avoid using incorrect information, but it caused a noticeable drop in the model's accuracy. The five most important features identified were *Modifier, Noun Kernel, Object Clausal, Subject, and Average Dependency Distance,* all of which were included in the training process. However, *Punctuation*, which ranked sixth in importance, and similar features were omitted for the same reason as Whitespace Errors, thereby limiting the model's ability to account for certain syntactic nuances.

Another limitation lies in the dataset itself. The total number of texts available was 2,092, and while this is sufficient for baseline experimentation, a larger, cleaner dataset would likely yield better results. For instance, proper punctuation usage plays a critical role in demonstrating linguistic proficiency but was not included in the model training due to inconsistencies similar to those in Whitespace Errors. This shows the need for better-quality data to use features more effectively and improve model performance.

Despite these limitations, the results still highlight the effectiveness of feature selection in enhancing model performance. Specifically, for the FFNN model, incorporating language dependency relationships using the refined dataset *DataCleanUP* (instead of *DataClean*) increased accuracy from 0.78 to 0.82 for three-level classification and from 0.54 to 0.71 for six-level classification. Similarly, for the LSTM model, accuracy improved from 0.80 to 0.83 for three levels and from 0.61 to 0.69 for six levels. These outcomes affirm the importance of well-selected linguistic features, even in the presence of data quality and quantity constraints. The confusion matrix results obtained without these features can be found in the Table 4.

Task	Model	Accuracy	Precision	Recall	F1-score
3 Levels	FFNN	78.53%	79.67%	75.59%	81.20%
	LSTM	80.48%	80.44%	80.48%	80.39%
6 Levels	FFNN	54.14%	43.49%	54.41%	46.47%
	LSTM	61.70%	57.26%	61.70%	58.79%

Table 4: Confusion Matrices without selected features

8 Discussions

This study evaluates the effectiveness of two machine learning architectures,

Feedforward Neural Networks (FFNN) and Long Short-Term Memory (LSTM) networks, in classifying German language proficiency levels as defined by the CEFR. The findings of my study are compared with previous works in the field, focusing on feature selection, methodological differences, and dataset composition. These comparisons highlight how this research builds on and diverges from earlier studies in automated language proficiency classification.

- **Feature Utilization and Accuracy**
  Szügyi et al. (2019) utilized a comprehensive feature set, including traditional, lexical, morphological, syntactic, and error features. Despite employing 40 features, their classification accuracy (82%) aligns closely with the FFNN model's performance in my study (82.4%) and is slightly below the LSTM's accuracy (83.4%) for the three-level classification task. Interestingly, this study achieved comparable or better accuracy using only 21 features, highlighting the efficiency of the feature selection process in this work. This underscores the potential of deep learning approaches to simplify feature extraction while maintaining high performance.
- **Methodological Differences**
  Szügyi et al. (2019) relied on a balanced dataset and implemented an SVM-based model. In contrast, my study used neural network architectures optimized through regularization and feature scaling. The ability of neural networks to capture non-linear relationships likely contributed to the higher accuracy.
- **Corpus Considerations**
  While Szügyi et al. (2019) worked with 1,836 texts, my study used a more extensive dataset of 2,048 texts. The inclusion of the Disko and Ahlers corpora further enriched the data, which may have influenced the models' robustness in my study.

Szügyi et al. (2019) referenced Hancke (2013), which achieved 72.5% accuracy using a MERLIN dataset for A1–C1 classification. Although my study demonstrates an improvement in three-level classification tasks, the six-level classification accuracy of both the FFNN (71.2%) and LSTM (69.7%) models is slightly lower than Hancke's five-level classification accuracy. This difference may be attributed to the increased complexity of distinguishing between six levels compared to five, as well as differences in the datasets and features used. For example, Hancke's work focused on a smaller number of classes (A1–C1), while this study incorporates additional advanced-level texts (C2) and a broader range of corpora.

Similarly, the work of Weiss and Meurers (2018) on readability models aligns with the findings in my study: morphological features and syntactic complexity significantly influence classification accuracy. However, my study also emphasizes the predictive power of lexical diversity and dependency relationships, consistent with observations from Treffers-Daller et al. (2018).

9 Conclusion

This study explored the automatic classification of German text into CEFR proficiency levels using machine learning models. By comparing a feedforward neural network (FFNN) and a Long Short-Term Memory (LSTM)-based model, the research demonstrated the potential of neural networks in achieving high accuracy for language proficiency assessment. The FFNN performed better for detailed (six-level) classifications, while the LSTM excelled in broader (three-level) tasks, effectively leveraging sequential dependencies in text.

The study underscored the importance of feature selection, showing that a carefully curated subset of linguistic features can rival or even outperform larger feature sets used in prior researches. Among the key insights, the challenges in distinguishing neighboring proficiency levels, such as B1 and B2, emerged due to overlapping linguistic features. This issue parallels observations by Szügyi

et al. (2019), who reported misclassification around intermediate levels. Additionally, the analysis highlighted the predictive strength of features like readability scores and dependency relations, emphasizing their crucial role in enhancing classification performance.

10 Future Directions

While the results are promising, there is still room for improvement in several areas. Future research could delve into advanced model architectures, such as transformer-based models like BERT or XLM-RoBERTa, which excel at capturing complex language patterns and contextual dependencies. Expanding the linguistic features to include elements like discourse markers, pragmatic cues, and error analysis could improve the model's interpretability and performance.

This research not only confirms the value of deep learning for language proficiency classification but also lays the groundwork for future advancements. By achieving comparable or superior accuracy with fewer features, it provides a more scalable approach for automated language proficiency assessment. The advanced version of this work could potentially be utilized by examination centers in the future for determining language proficiency levels more efficiently.

References

Julia Hancke. 2013. Automatic prediction of cefr proficiency levels based on linguistic features of learner language. Master's thesis, University of Tübingen.

Edit Szügyi, Sören Etler, Andrew Beaton, and Manfred Stede. 2019. Automated assessment of language proficiency on german data. In *Proceedings of the 15th Conference on Natural Language Processing (KONVENS)*, pages 30–37. University of Potsdam.

Jeanine Treffers-Daller, Phil Parslow, and Sara Williams. 2018. Back to basics: How measures of lexical diversity can help discriminate between cefr levels. *Applied Linguistics*, 39(3):302–327.

Zarah Weiss and Detmar Meurers. 2018. Modeling the readability of german targeting adults and children. In *27th International Conference on Computational Linguistics (Coling 2018)*, pages 303–313.

A Appendix

Annotation	Description
Dep_da	Dative
Dep_ag	Attribute, Genitive
Dep_nk	Noun Kernel
Dep_ng	Negation
Dep_ams	Measure Argument of Adjective
Dep_ac	Adpositional Case marker
Dep_op	Prepositional Objects
Dep_cvc	Collocational Verb Construction
Dep_app	Apposition
Dep_pg	Phrasaler Genitive
Dep_re	Repeated Element
Dep_adc	Adjective Component
Dep_rs	Reported Speech
Dep_punct	Punctuation
Dep_cm	Comparative Conjunction
Dep_rc	Relative Clause
Dep_cd	Coordinating Conjunction
Dep_pnc	Proper Noun Component
Dep_ju	Junctor, Function label
Dep_oa	Accusative Object
Dep_uc	Unit Component
Dep_dm	Discourse markers
Dep_ep	Expletives
Dep_vo	Vocative
Dep_sb	Subject
Dep_svp	Separable Verb Prefix
Dep_avc	Adverbial Phrase Component
Dep_ph	Placeholder
Dep_cj	Conjunctions
Dep_pd	Predicative
Dep_mnr	Modifier of Np to the Right
Dep_pm	Morphological Particle
Dep_ROOT	Root
Dep_cc	Comparative Complement
Dep_cp	Complementizer
Dep_nmc	Number Component
Dep_mo	Modifier
Dep_oc	Object Clausal
Dep_dep	Unclassified Dependent
Dep_og	Object Genitive
Dep_sbp	Subject Passivised
Dep_par	Parenthesis

Table 5: The TIGER Annotation Scheme
https://www.ims.uni-stuttgart.de/documents/ressourcen/korpora/tiger-corpus/annotation/tiger_scheme-syntax.pdf

Class	Model	Correct Predictions	Key Misclassifications	Observations
A (3-Level)	FFNN	73	23 misclassified as B	FFNN struggled more than LSTM in separating beginner (A) texts from intermediates (B). No errors with C.
	LSTM	**88**	**8 misclassified as B**	LSTM handled A-level classification significantly better, achieving a recall of **92%**.
B (3-Level)	FFNN	**145**	15 as A; 17 as C	Strong performance, but moderate confusion with both beginner and advanced levels.
	LSTM	133	**21 as A; 23 as C**	More errors compared to FFNN in identifying intermediate B-level texts.
C (3-Level)	FFNN	120	17 as B	FFNN had good performance in separating Class C from lower levels.
	LSTM	**121**	**16 as B**	Slightly better performance than FFNN but maintained similar confusion with B.
A1 (6-Level)	FFNN	**23**	9 as A2	Good separation between A1 and other levels but some confusion with A2.
	LSTM	22	**8 as A2; 2 as C2**	Close performance, with LSTM also misclassifying samples as advanced (C2), reflecting potential sequential relevance missed in simpler patterns.
A2 (6-Level)	FFNN	**59**	5 as B1	FFNN generally performed better in distinguishing A2 from intermediate texts.
	LSTM	53	**9 as B1; 2 as A1**	LSTM struggled slightly more than FFNN in separating A2 and B1 categories.
B1 (6-Level)	FFNN	45	**22 as A2; 10 as B2**	B1 was one of the more challenging classes for FFNN, with samples misclassified evenly into both neighboring levels (A2 and B2).
	LSTM	**47**	19 as A2; 11 as B2	LSTM handled B1 slightly better overall but exhibited similar confusion.
B2 (6-Level)	FFNN	**76**	10 as B1; 13 as C1; 1 as C2	Strong performance, with most errors related to confusion between intermediate B1 and advanced C1.
	LSTM	74	**13 as B1; 13 as C1**	Comparable performance to FFNN with similar misclassification patterns.
C1 (6-Level)	FFNN	62	**21 as B2; 9 as C2**	Strong overall accuracy, though confusion with both intermediate B2 and advanced C2 classes affected performance.
	LSTM	**66**	19 as B2; 7 as C2	LSTM performed better in separating C1 from C2, but similar confusion with B2 remained a common factor.
C2 (6-Level)	FFNN	**27**	**18 as C1**	C2 was the weakest class for FFNN, with high confusion with C1, reflecting difficulty in distinguishing advanced texts.
	LSTM	24	21 as C1	The LSTM performed slightly worse for C2, struggling even more with separating the most advanced texts from C1.

Table 6: Confusion Matrix Class-Wise Comparison