Natural Language Processing Journal 14 (2026) 100202

Contents lists available at ScienceDirect

Natural Language Processing Journal

journal homepage: www.elsevier.com/locate/nlp

Enhancing language models with boosting and targeted fine-tuning for real-word error detection

Corina Masanti [a,b,*], Hans-Friedrich Witschel [b], Kaspar Riesen [a]

[a] Institute of Computer Science, University of Bern, Bern, 3012, Switzerland
[b] Institute for Information Systems, University of Applied Sciences and Arts Northwestern Switzerland, Olten, 4600, Switzerland

ARTICLE INFO

Keywords:
Error detection
Real-word errors
Synthetic data
Boosting
Ensemble learning

ABSTRACT

Over the past years, extensive research has led to significant advancements in tools for the automatic detection and correction of errors in documents. Despite this progress, several challenges remain unresolved. In particular, the identification of real-word errors – errors involving words that are grammatically valid but contextually inappropriate within a given sentence – continues to pose a considerable difficulty. Addressing such errors requires models with a sophisticated understanding of linguistic context. Transformer-based language models are particularly well-suited for this task due to their contextual modeling capabilities. To further enhance their performance, we propose a boosting-based training approach in conjunction with a synthetically generated data set created via pattern-based noise injection. We evaluate this method across three transformer-based architectures, viz. mBERT, LLaMA 3, and Mistral 7B. Our experimental results show that the boosting-based strategy consistently improves real-word error detection across all models. A subsequent in-depth error analysis reveals limitations in the synthetic training data, prompting the development of a targeted fine-tuning procedure designed to address these shortcomings and further optimize model performance. A comparison with prompt-based inference using a large language model demonstrates that specialized, fine-tuned models yield more reliable performance for this task. Finally, an evaluation under realistic class imbalance highlights practical trade-offs between ranking quality and threshold-based detection, particularly for rare error types.

1. Introduction

The automatic detection and correction of errors in text has long been a persistent challenge in *Natural Language Processing* (NLP), and research in this area has evolved from rule-based approaches (Damerau, 1964) to modern data-driven methods (Omelianchuk et al., 2020). These advances have led to increasingly effective proofreading systems, with transformer-based models (Vaswani et al., 2017) dominating the field today. However, even after the introduction of these models, there are still several challenges and open research questions. Particularly, the detection and correction of real-word errors continues to pose a significant difficulty. Roughly speaking, real-word errors occur when the erroneous word can be found in the underlying dictionary but is inappropriate within its sentence context. One major obstacle in effectively addressing such errors lies in the lack of high-quality training data, especially for languages beyond English, which continues to dominate the focus of current research efforts. This issue is further worsened by language-specific challenges, which are not only linguistically complex but also frequently violated in real-world texts. As a result, transformer models trained on such noisy data may learn to prefer incorrect patterns simply because they occur more frequently.

To address error detection in languages other than English, we have developed and proposed a number of solutions in recent years. For example, we present a novel benchmark data set for automatic error detection and correction in text documents (Masanti et al., 2023). This data set includes a wide range of document types, primarily written by native speakers of German, French, Italian, and English, all professionally proofread. Next, we refine the data set and conduct an in-depth error analysis (Masanti et al., 2024), and propose a preliminary version of a boosting-based training algorithm that sequentially trains language models to improve the detection of errors (Masanti et al., 2025).

The present paper combines, consolidates and substantially extends our previous research in various dimensions – both methodologically and in terms of experimental evaluation. In summary, the contributions of this paper are as follows.

* Corresponding author.
E-mail address: corina.masanti@unibe.ch (C. Masanti).

https://doi.org/10.1016/j.nlp.2026.100202
Received 4 July 2025; Received in revised form 9 January 2026; Accepted 7 February 2026
Available online 11 February 2026

- We substantially extend the boosting-based approach (Masanti et al., 2025) by using two additional models of diverse architectures to assess the robustness of our ensemble approach.
- We thoroughly evaluate the effectiveness of our novel synthetic data generation pipeline through experiments on a large real-world data set and conduct a detailed error analysis to identify gaps and opportunities for further refinements of the synthetic training data.
- Based on the error analysis, we generate additional targeted synthetic training data to address identified weaknesses. We then apply targeted fine-tuning to further improve model performance and assess the impact of this strategy.

We are not the first to investigate ensemble-based methods in the context of language models. For instance, a common ensemble method used with language models is stacking, where the predictions of multiple base models are combined to create a more accurate final prediction. One specific approach of this category is majority voting, where the most frequently predicted label is selected as the final label (Stahlberg and Kumar, 2021). Another technique is probability averaging, where the output probabilities from multiple base models are averaged to determine the final prediction. Both techniques have demonstrated strong performance, outperforming single-model systems (Omelianchuk et al., 2020). Although there are several promising approaches, to our knowledge, the present work is the first to leverage the ensemble learning strategy of boosting in the specific configuration of automatic error detection with language models.

Importantly, the goal of this work is not to introduce a new ensemble or meta-learning architecture in the traditional sense, but to study how targeted data selection during fine-tuning affects the learning dynamics of *Large Language Models* (LLMs). Consequently, the central empirical question is not whether boosting outperforms alternative ensemble combination rules, but whether targeted error-driven data augmentation yields measurable gains over non-targeted augmentation.

The remainder of this paper is structured as follows. Section 2 reviews related work and describes the current state of the art. Section 3 introduces the data set used in this paper and formally defines the underlying task. In Section 4, we describe the models used in the experimental analysis and present the novel boosting-based method for training language models in detail. Section 5 reports, discusses, and interprets the experimental results across both synthetic and real-world data sets. This includes a thorough error analysis, an evaluation of the targeted fine-tuning procedure, and a performance comparison with prompt-based systems, alongside a discussion on practical applications. Finally, Section 6 concludes the paper and outlines potential directions for future research.

2. Related work

Both error detection and error correction in text is a well-established research area in NLP (Bryant et al., 2023). This section provides a comprehensive overview of the current state of the art in the field.

While certain error types demand sophisticated approaches, others can be effectively addressed using rule-based methods. For example, non-word errors – cases where the erroneous word does not exist in the language's dictionary – can often be identified and corrected using dictionary-based methods. A traditional approach involves checking each word in a sequence against a lexicon. If a word cannot be found, it is flagged as incorrect. The system then searches for the most likely intended word based on minimal edit distance computations (Damerau, 1964). However, this approach fails to handle out-of-dictionary words, foreign language terms, and real-word errors. To address these types of errors, contextual models are typically employed. These models estimate the likelihood of a word's occurrence based on its surrounding context, enabling the detection of semantically or syntactically implausible word choices. One technique involves the use of trigram language models, which consider the probability of a word given the two preceding words (Wilcox-O'Hearn et al., 2008).

The introduction of the transformer architecture (Vaswani et al., 2017) led to the development of language models capable of capturing contextual dependencies far more effectively than earlier architectures. Empirical comparisons across various benchmarks have consistently shown that these neural architectures outperform traditional models by a significant margin (Bryant et al., 2019; Alikaniotis et al., 2019; Patra et al., 2023). Consequently, recent research has shifted almost exclusively toward optimizing these high-capacity models. Specifically, transformer-based language models such as BERT (Devlin et al., 2018), T5 (Raffel et al., 2020), or GPT (Brown et al., 2020) (e.g., in prompt-based settings (Loem et al., 2023)) represent the current state of the art.

However, even with the recently introduced language models, some of which have demonstrated remarkable capabilities, one major challenge remains, viz. the need for high-quality, labelled training data. In the context of error detection and correction, this typically requires sentences annotated by professional proofreaders, where each original erroneous sentence is paired with a corrected version. To support research in this area and mitigate this limitation, shared tasks have been introduced. These initiatives contribute new publicly available data sets and provide a standardized platform for evaluating and comparing different systems using benchmark data sets.

One well-known example is the CoNLL-2014 shared task (Ng et al., 2014), which involves detecting and correcting essays written by learners of English as a second language. One limitation of this data set is that the learner's English exhibits a smaller vocabulary and simpler syntactic structures compared to the writing of native speakers. With the goal of re-evaluating the field five years after the introduction of CoNLL-2014, the BEA-2019 shared task was introduced (Bryant et al., 2019). With it, new annotated data sets were released, namely the English Write & Improve (W&I) and LOCNESS corpus. These data sets are designed to represent a much wider range of language levels and abilities than previous corpora. Despite these advances, the focus on the English language remains a major limitation in the field, resulting in limited resources and representation for other languages. One initiative addressing this limitation is the MultiGED-2023 shared task (Volodina et al., 2023), which includes annotated data for five languages (Czech, English, German, Italian, and Swedish). While this shared task represents a significant step towards greater linguistic diversity, the scarcity of high-quality resources for the many underrepresented languages remains a continous challenge (Etoori et al., 2018).

One potential approach to address data scarcity in underrepresented languages is the use of synthetic data. Two common techniques for generating such data are back-translation and noise injection (Kiyono et al., 2019). Back-translation involves training a model to produce ungrammatical sentences from grammatical ones (e.g., Rei et al., 2017). This reverse model can then be used to create large-scale synthetic data sets. With noise injection, artificial errors are introduced into grammatically correct sentences to simulate human mistakes. Both techniques rely on understanding the types of errors humans actually make – a necessity when dealing with complex phenomena that even native speakers often struggle with, such as context-sensitive orthographic or grammatical rules. Without such insights, synthetic data may fail to reflect real-world patterns, thereby limiting model performance. This issue is compounded by the fact that shared task data sets often lack these nuanced errors, particularly for languages beyond English. High-quality annotations and thorough error analysis are thus essential to guide realistic data generation and improve the reliability of NLP systems.

This objective can be achieved using rule-based or pattern-based strategies. For pattern-based approaches, error patterns are extracted from existing annotated data sets and applied inversely to falsify sentences and simulate human behaviour (Yuan and Felice, 2013). The advantage of this method is that since the patterns are derived from

Table 1
Example sentences (**S**) from the data set illustrating each real-word error type, along with their corresponding correction (**C**). The corrected words are highlighted in boldface.

Type	Original Sentence (S) and its Correction (C)
Case	**S:** Alle externen Anlässe und schlussendlich auch der Tag der offenen `Türe` wurden abgesagt. **C:** Alle externen Anlässe und schliesslich auch der Tag der offenen `Tür` wurden abgesagt.
Verb	**S:** Auf die Teilnehmenden `wartete` ein spannendes Programm und angeregte Diskussionen. **C:** Auf die Teilnehmenden `warteten` ein spannendes Programm und angeregte Diskussionen.
Capitalization	**S:** Wir sind der Frage nachgegangen, was die Gesundheitstrends von `Morgen` beeinflusst. **C:** Wir sind der Frage nachgegangen, was die Gesundheitstrends von `morgen` beeinflusst.

real-world data, the generated errors tend to resemble naturally occurring ones. The drawback, however, is that a certain amount of annotated data is required to extract meaningful patterns.

3. Data set and task

For the research described in this paper, we use a recently introduced data set which contains text from approximately 50,000 documents of a Swiss proofreading agency (Masanti et al., 2023). This data set includes both the original versions and the annotated corrections made by expert proofreaders. The underlying documents cover a broad range of document types, such as annual reports, business correspondence, technical manuals, legal documents, marketing materials, presentation slides, social media posts, newsletters, websites, and magazine articles. Moreover, the documents originate from clients across various industries, including pharmaceuticals, finance, insurance, retail, and telecommunications. The texts are written in German, French, Italian, and English, reflecting Switzerland's multilingual nature, while German is most prevalent.

The majority of the documents are authored by native speakers, resulting in high-quality texts in terms of vocabulary and syntax. Moreover, they contain more complex error types than typical learner corpora. This makes the data set particularly well-suited for the development and evaluation of models detecting real-word errors.

In the present study, we focus on the analysis of documents in German, a language whose linguistic complexity makes it particularly suitable for investigating real-word errors. To identify real-word errors, we filter for cases where the erroneous word in the original sentence is present in a German dictionary. These instances are then classified into the following three main error types (Table 1 shows an example sentence including the human-made correction for each error type).

- *Case errors*: A *case error* is defined as the use of an incorrect grammatical case (nominative, genitive, dative, or accusative), including mismatches in number, such as using singular instead of plural. Case errors are frequent even among native speakers, making them challenging for automated systems to detect.
- *Verb errors*: *Verb errors* involve verbs that are incorrectly conjugated or used in the wrong tense. These can be particularly subtle when subject-verb agreement is violated, as in the example shown in Table 1, where the plural subject *die Teilnehmenden* incorrectly triggers the singular verb form.
- *Capitalization errors*: *Capitalization errors* refer to the incorrect use of uppercase or lowercase letters. For instance, in Table 1, the word *Morgen* is incorrectly capitalized, suggesting a noun meaning 'morning', whereas the intended meaning was the adverb *morgen* ('tomorrow').

Unfortunately, the number of available real-world documents, and, as a result, the number of real-world errors of the three error types is too small to effectively train error detection models. Fig. 1 illustrates the distribution of error types in the data set. In the corpus, approximately 24% of sentences contain at least one error, with real-word errors accounting for approximately one quarter of the total error count. Within this group, capitalization and case errors dominate, while verb errors are comparatively rare.[1] The category *Other real-word errors* is excluded from further analysis, as it primarily comprises stylistic substitutions or lexical paraphrases that are difficult to detect reliably and fall outside the scope of this work.

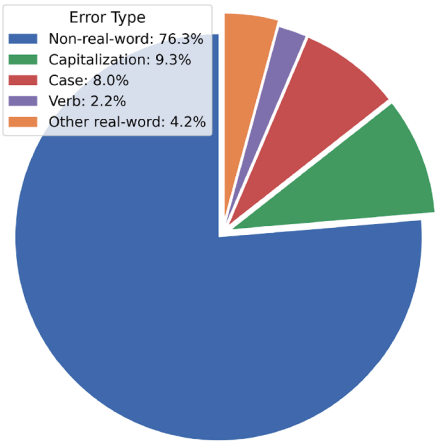

Fig. 1. Distribution of error types in the real-world data set. Only about one quarter of the observed errors are genuine real-word errors.

To enable a systematic evaluation on a sufficiently large number of challenging instances, we generate additional training data using pattern-based noise injection, which is described in detail in the remainder of this section (Fig. 2 illustrates the complete synthetic data generation pipeline). For this purpose, we construct balanced data sets with a 50% / 50% split between erroneous and correct sentences. While this setup does not reflect real-world error frequencies, it prioritizes high recall to ensure the system remains sensitive to subtle errors. From a professional proofreader's perspective, prioritizing high recall is essential, as undetected errors require mandatory manual review, whereas false positives can be quickly dismissed during the verification process. The errors selected for this setup are the result of extensive filtering and manual verification and represent the most challenging real-word error cases observed in the data.

The process begins with a real-world data set of roughly 50,000 documents (**Artefact A**). In **Step 1** (termed Filter), we identify sentences containing real-word errors and classify them into the three error types case, verb, and capitalization. Now, each data point consists of a sentence pair, which includes the original erroneous sentence and its corrected version as shown in **Artefact B**. Thus, the final data sets include twice the number of individual sentences shown in the figure.

In **Step 2** (termed Extract), we use these sentence pairs to extract word-level error patterns (**Artefact C**). Specifically, we extract pairs of

[1] Note that a single sentence may contain multiple error types (e.g., both a capitalization and a case error); such sentences are counted once for each applicable category when computing the error-type distribution.

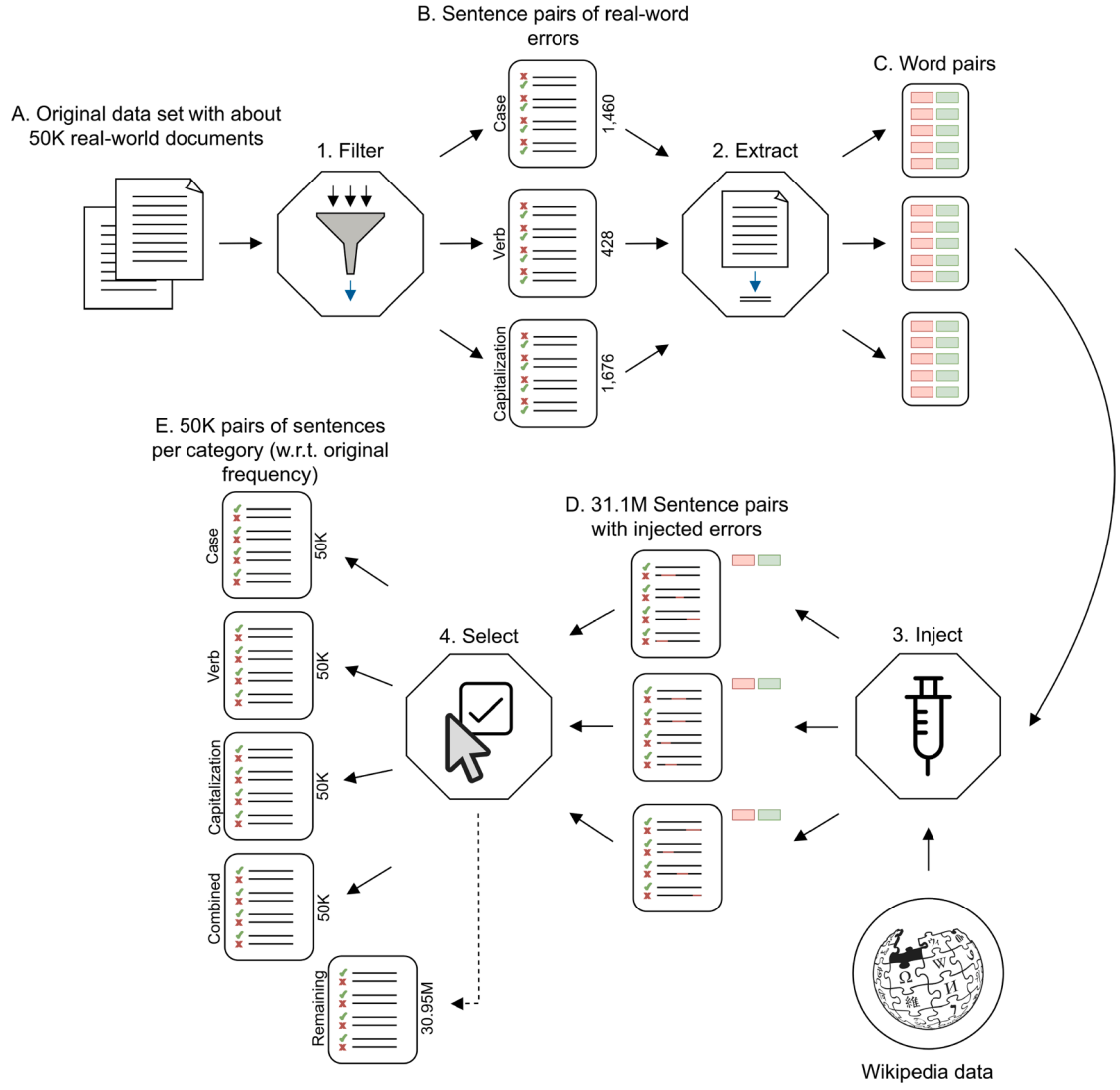

Fig. 2. Illustration of the four main steps (1. Filter, 2. Extract, 3. Inject, 4. Select) and the five resulting Artefacts A-E of the complete pipeline to generate synthetic data.

erroneous and correct words ($w_{(e)}$, $w_{(c)}$). For instance, the capitalization error in Table 1 yields the pair ('Morgen', 'morgen').

Next, in **Step 3** (termed Inject), we apply a well-known strategy for noise injection (Yuan and Felice, 2013) by reversing the correction process. That is, we search for the correct word $w_{(c)}$ (e.g., 'morgen') in a separate corpus of grammatically correct sentences and replace it with its erroneous counterpart $w_{(e)}$ (e.g., 'Morgen'). As a source for the separate corpus of correct sentences, we use Wikipedia and a specific text extractor (Attardi, 2015). This results in a large corpus of 31.1 million sentence pairs (**Artefact D**). Table 2 presents one example sentence from Wikipedia alongside the corresponding injected error for each error type.

Finally, in **Step 4** (termed Select), we sample 50,000 instances per error type to construct three baseline data sets. Additionally, we create a combined data set by sampling 50,000 instances that include a balanced mix of all three error types. The remaining 30.95 million sentences are reserved for strategic augmentation through boosting, as described in Section 4.

To reflect the distribution of the errors in the real-world data set, we preserve the relative frequency of each error pair in the synthetic data as accurately as possible. For instance, if the pair ('Morgen', 'morgen') accounts for 5% of the real-world error instances, we target a similar frequency in the synthetic data set. In practice, however, this alignment can only be approximated, as certain terms, particularly Swiss-specific expressions, are underrepresented or entirely missing in the Wikipedia corpus used to retrieve the grammatically correct sentences.

For training and evaluation, each of the four data sets (**Artefact E**) is balanced such that 50% of the sentences contain an error from the relevant error type and the remaining 50% of the sentences refer to their error-free counterparts. As a result, we create data sets of 100,000 labelled examples per error type, where each sentence is annotated as correct (0) or incorrect (1). The motivation behind this approach is that we adopt an oversampling strategy to ensure adequate representation.

Moreover, as each of the four data sets is balanced, the downstream task can be interpreted as binary classification, meaning that models trained on this data learn to assign a binary label to each sentence, indicating the presence (label = 1) or absence (label = 0) of an error.

While the synthetic data may in general not fully reflect the complexity and variability of naturally occurring language errors, we mitigate this limitation by using the patterns observed in real-world data. Moreover, by aligning the error distribution in the synthetic data with that of the original data set, we ensure that the generated examples remain closely tied to authentic error phenomena. However, we acknowl-

Table 2
Examples sentences from the separate corpus (Wikipedia), with the injected errors highlighted in boldface (for each error type).

Type	Original Sentence (from Wikipedia)	Sentence with Injected Error
Case	Während des Studiums bildeten Studienseminare **zur** Fotografie […]	Während des Studiums bildeten Studienseminare **zu** Fotografie […]
Verb	Die Vereinigung begann damals mit 20 Personen und **hatte** am 31. März 2010 bundesweit 381 Mitglieder.	Die Vereinigung begann damals mit 20 Personen und **hat** am 31. März 2010 bundesweit 381 Mitglieder.
Capitalization	Beim **Überprüfen** der Geschenke stellen die Pigeoncotes fest, dass sie nun tatsächlich acht Kännchen haben.	Beim **überprüfen** der Geschenke stellen die Pigeoncotes fest, dass sie nun tatsächlich acht Kännchen haben.

Table 3
Overview of the architecture and pre-training procedure for the models mBERT, LLaMA 3, and Mistral 7B.

	mBERT	LLaMA 3	Mistral 7B
Architecture	Encoder-only	Decoder-only	Decoder-only
Number of Parameters	110 million	8 billion	7 billion
Number of Layers	12	32	32
Hidden Size	768	4096	4096
Vocabulary Size	30,522	128,000	32,000
Architecture Choices	Multi-head attention	Grouped-query attention	Grouped-query attention, sliding window attention
Pre-training Data	BookCorpus, English Wikipedia	Original: Webpages, GitHub, Wikipedia, Project Gutenberg, ArXiv; Additional: Occiglot Fineweb v0.5	Original: Not disclosed; Additional: Wikipedia, OSCAR-2301, Tagesschau (2018–2023)
Pre-training objectives	Masked language modeling, next sentence prediction	Next-token prediction	Not disclosed

edge that some error types require more context than our current strategy provides. For instance, consider the rule for capitalization after a colon in German – a phenomenon where even Wikipedia may contain frequent deviations from the norm. This makes it possible that, during data generation, we might inadvertently produce a correct form instead of an error (or vice versa), particularly when converting between lowercase and uppercase forms. To uncover and address such issues, we conduct a detailed error analysis in the experimental evaluation provided in Section 5.

4. Boosting language models

In this section, we present our novel method in detail. We begin by briefly reviewing the selected transformer-based models[2] (Section 4.1) that serve as the foundation for our boosting approach (detailed in 4.2).

4.1. Basic language models

The boosting approach proposed in this paper is based on the use of language models. In principle, any available model can be used – in our work and experimental evaluation, we focus on three models built upon the encoder-decoder components introduced in the transformer architecture (Vaswani et al., 2017). Table 3 shows an overview of the characteristics of each of the three models.

The first model is mBERT, short for *Multilingual Bidirectional Encoder Representation from Transformers* (Devlin et al., 2019). The model utilises a 12-layer bidirectional transformer encoder with a dimension of 768, a vocabulary size of 30,522 and a total of 110 million parameters. Pre-training is done using a multilingual corpus of 104 languages and included two unsupervised tasks, namely masked language modeling and next sentence prediction. This pre-training procedure allows the model to capture left and right context at every layer. Although smaller than many recently published language models, its parameter efficiency makes it well-suited for real-world deployment. We select the hyperparameters for each of the three models based on preliminary experiments. For mBERT, we use the best parameters identified by means of a grid search (Masanti et al., 2024). We use a learning rate of 1.1e-5, a batch size of 64 and train the model for 20 epochs.

The second model is a large language model based on Meta's LLaMA 3 (Touvron et al., 2023). This family of powerful language models is pre-trained via next-token prediction on 15 trillion tokens, which results in strong language modeling and world knowledge. However, with fewer than one trillion multilingual tokens used during pre-training, LLaMA 3 shows suboptimal performance in German. Thus, we use a German-adapted variant further trained on a large German corpus (DiscoResearch, 2024). LLaMA 3 has an architecture similar to its predecessors with a decoder-only structure, 32 layers, a dimension of 4,096, and 8 billion parameters. Performance gains derive from higher-quality and more diverse pre-training data along with architectural tweaks such as grouped query attention – dividing query heads into groups resulting in faster processing time – and a 128,000 token vocabulary. We employ Optuna (Akiba et al., 2019) to efficiently manage the hyperparameter optimization process. Optuna uses the *Tree-structured Parzen Estimator* (TPE), a Bayesian optimization algorithm that models high-performing versus low-performing trials to propose hyperparameters with the greatest expected improvement. To fine-tune the model, we apply LoRA (*Low-Rank Adaptation of Large Language Models*) (Hu et al., 2022) with a rank of 16 and α of 32. The other hyperparameters include a learning rate of 4e-5, a batch size of 32, and three training epochs.

The third and final model used in the evaluation is Mistral 7B (Jiang et al., 2023). We use a model variant, which is further trained on a large corpus of German data to improve performance on German inputs (Björn et al., 2024). Mistral 7B is a decoder-only transformer language model with 7 billion parameters, a dimension of 4,096, and a vocabulary size of

[2] Note that our task of error detection is framed as a classification problem rather than text generation. Therefore, encoder-only models like mBERT and decoder-only models like LLaMA and Mistral are more appropriate and efficient than encoder-decoder architectures. Hence, these models are not included in this work.

32,000. It incorporates grouped query and sliding window attention to efficiently process long sequences with reduced inference cost. To find the optimal hyperparameters, we again employ Optuna. We apply LoRA with a rank of 16 and α of 32. We train the model for three epochs with a batch size of 32 and a learning rate of 4e-5.

4.2. Boosting procedure

The research presented in this paper is motivated by the needs of professional proofreaders, with the goal of supporting them in their daily work. Therefore, rather than developing a standalone system, we aim to create a tool that integrates into their workflow and addresses their specific requirements. In practice, this has the following implications. While false positives are relatively easy for experts to dismiss, undetected errors pose a greater risk. Therefore, achieving high recall is essential to ensure that as many errors as possible are flagged by the model, even if it comes at the cost of slightly lower precision.

To meet these requirements, we propose to progressively incorporate synthetic data from the synthetic data pool (described in Section 3) into the training set. Rather than adding data at random, we propose a boosting-inspired strategy. Boosting (Freund, 1990; Schapire, 1990) is a widely used ensemble technique in machine learning applications. Boosting sequentially trains models, with each new model focusing on correcting the errors of its predecessor. Due to the complexity of linguistic data, especially in the context of error detection and correction, boosting-based methods are less commonly applied in NLP. Nevertheless, adaptations inspired by these techniques exist. For example, *boosted prompting* iteratively augments the prompt set with new prompts that better generalize to regions of the target problem space where prior prompts underperform (Pitis et al., 2023).

In contrast, our novel procedure – outlined in Algorithm 1 – introduces boosting at the data level rather than at the prompt level. We iteratively identify the model's weaknesses and selectively inject synthetic examples targeting these weaknesses into the training set. This ensures that each training iteration focuses on the most challenging and underrepresented errors.

Algorithm 1 Step-wise integration of synthetic data into model fine-tuning via a boosting-inspired approach.

```
1  for each model in model_list do
2      Initialize added_errors ← ∅;
3      for epoch = 1 to n do
4          Fine-tune model on training data;
5          Evaluate on validation set;
6          newly_added ← ∅;
7          for each misclassified (incorrect-word, correct-word) pair do
8              if pair not in added_errors then
9                  Select up to N synthetic samples for this pair;
10                 Add samples to training set;
11                 Add pair to added_errors;
12                 Add pair to newly_added;
13         if newly_added = ∅ then
14             break
15 Test model on test set;
```

Algorithm 1 begins by selecting one of the three transformer-based models described in Section 4.1 (mBERT, LLaMA 3, Mistral 7B) and initializing *added_errors* to the empty set (line 1 and 2). In each epoch (line 3 to 14), the model is first fine-tuned on the current training set (line 4), followed by an evaluation on the validation set to assess its performance (line 5). Then, we identify the misclassified sentences of the validation set. Specifically, those containing errors (true label = 1) that the model incorrectly classifies as error-free (predicted label = 0). From these examples, we extract the associated error pairs, consisting of the incorrect word and its corrected version. For each error pair that has not been previously added (lines 6 to 8), we retrieve up to N synthetic training examples, if available. These samples are then incorporated into the training data for the next epoch (lines 9 to 12). Each batch of N samples is balanced, containing $N/2$ samples with the incorrect word and $N/2$ samples with the correct word. The algorithm keeps track of all newly added error pairs in the current epoch (using the set *newly_added*). If no samples can be added, the process terminates early to avoid redundant computation (line 13 and 14).[3]

This step-wise data augmentation and training loop continues for up to n epochs or until no further synthetic data can be injected. After the final iteration, the model is evaluated on an independent test set to assess overall performance (line 15).

Theoretically, the values for the number of epochs n and the number of synthetic samples N introduced at each iteration can be arbitrarily chosen. In our experiments, we set $n = 5$ and $N = 500$, as larger training sets significantly increase time for fine-tuning, and model accuracy tends to plateau by the fifth epoch.

Our new approach has two advantages over a naïve brute force idea of simply integrating all available synthetic data into the fine-tuning process at once.

- First, with the proposed approach, we start with a smaller subset of the entire data set (namely 50,000 documents instead of 31.8 million), which makes the overall handling of the models substantially more efficient. We then selectively add only those additional samples that contribute to improving the model, thereby keeping a reasonable computation time. This strategy is particularly effective for large data sets that contain redundant or noisy samples, as well as for highly imbalanced data sets where rare or difficult cases require greater emphasis.
- Second, when combining real-world and synthetic data, the proposed method enables initial fine-tuning on real-world samples, followed by the gradual and targeted inclusion of synthetic samples, thereby reducing the likelihood of overfitting to artificial error patterns.

Moreover, we emphasize that the proposed approach differs fundamentally from classical ensemble and meta-learning techniques such as stacking, voting, or model-level boosting. These methods combine multiple predictors at inference time and therefore conflate improvements due to aggregation with those due to data exposure. In contrast, the novel boosting method operates exclusively at the data level, keeping the model architecture, inference procedure, and number of predictors fixed. Although this shifts computational complexity to the training phase, it avoids the high latency and memory overhead associated with classic ensemble methods.

5. Experimental evaluation

The experimental evaluation presented in this section is divided into four parts. In the first part, we focus on assessing the effectiveness of the boosting-based approach. To this end, we train and test the models exclusively on synthetic data. In the second part, we examine how well these models transfer to real-world data. This allows us to identify potential gaps in the synthetic data creation pipeline and evaluate the model's generalization ability. This second part also includes an in-depth error analysis and the results of a targeted fine-tuning to further improve model performance. In the third part, we compare our approach to prompt-based grammatical error detection with GPT-4o (Hurst et al., 2024), which serves as a strong state-of-the-art reference. In the final part, we evaluate our model under a more realistic error distribution

[3] Remember that the alignment with the actual error distribution is only partially achievable as some errors, especially those involving rare expressions or Swiss-specific vocabulary, are underrepresented in the Wikipedia corpus. See Section 3 for further details.

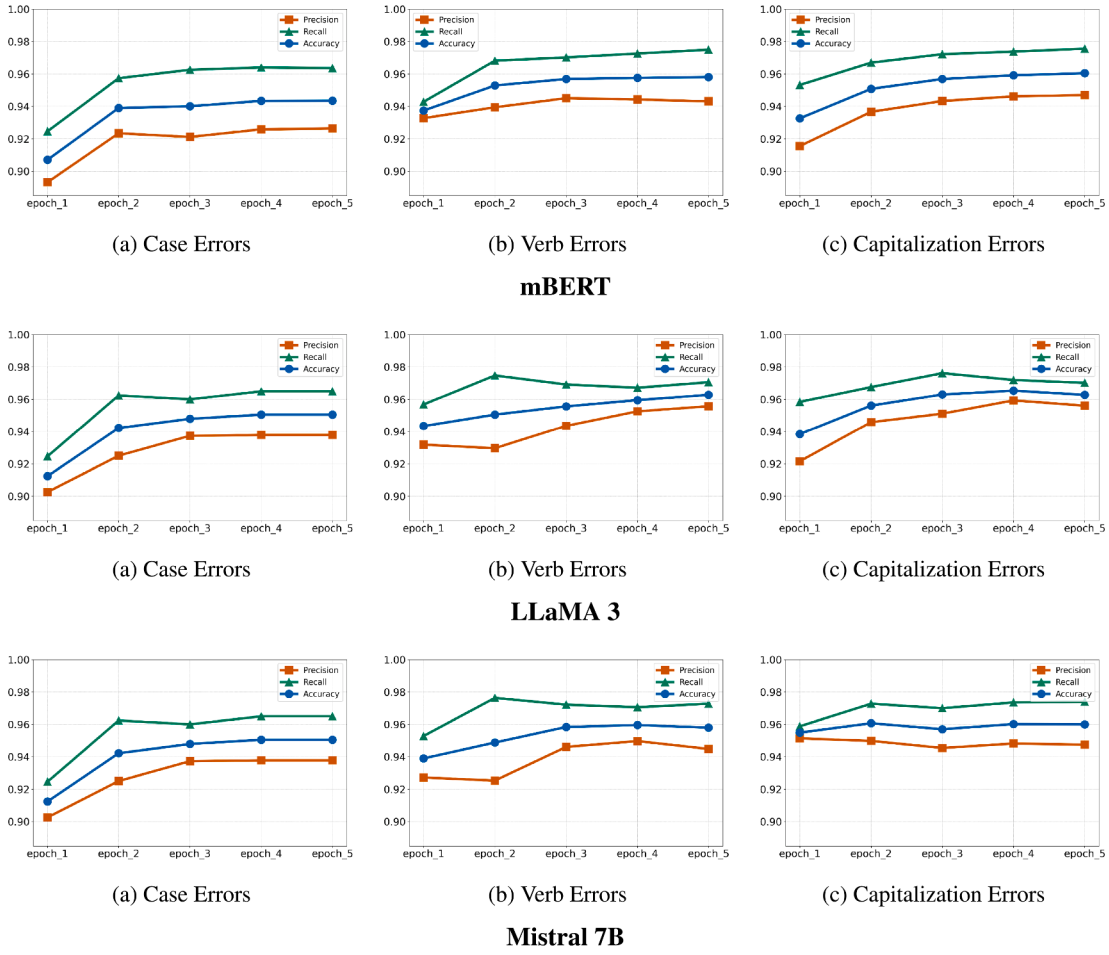

Fig. 3. Boosting results on the validation set for each error type across three models. Each image shows Precision, Recall, and Accuracy over all epochs for a specific model and error type.

to assess its robustness and to identify potential limitations in practical settings.

5.1. Evaluation with synthetic data

As outlined in Section 3, our focus is on detecting real-word errors across three error types. For each error type, we perform experimental evaluations in three setups (in all three setups, we use the same test set to ensure comparability of results):

1. **Baseline:** In this setup, the base models (mBERT, LLaMA 3, Mistral 7B) are fine-tuned on the synthetic data set only. We split the data set into 50% for training, 10% for validation, and 40% for testing.
2. **Boosting:** In this setup, we start from the same training set as the baseline. Yet, now we apply the novel boosting strategy detailed in Section 4. That is, we incrementally augment the training set with targeted synthetic samples to improve model performance.
3. **Random Selection:** For this third experimental setup, we augment the baseline training set with the same number of synthetic samples that are used in the boosting setup, but select them randomly. This experiment – which can be interpreted as an ablation study – isolates the effect of the boosting strategy by comparing it to simple data augmentation without targeted sample selection.

The random selection setup represents a deliberately strong control condition. By augmenting the training data with the same number of synthetic samples as in the boosting setup – while removing any form of informed selection – it isolates the effect of targeted, error-driven data integration. Consequently, any consistent performance gains of boosting over random selection can be attributed exclusively to the proposed selection strategy, rather than to increased data volume or incidental regularization effects.

Fig. 3 shows the validation results of the boosting procedure over $n = 5$ epochs for mBERT, LLaMA 3, and Mistral 7B. Each subfigure visualizes the model's performance for a specific error type (case, verb, capitalization). Across all error types, we observe increased accuracy and recall over the epochs achieved by all models. The most significant improvement occurs after the initial epoch of synthetic data integration, especially for case errors.

Tables 4, 5, and 6 summarize the final results achieved on the test set in the three experimental setups (Baseline, Boosting, Random Selection) for each of the three models (mBERT, LLaMA 3, Mistral 7B) as well as for all error types. The highest scores for each error type and evaluation metric are highlighted in boldface.

Overall, the three models achieve comparable scores. This means that we cannot identify a clear winner between mBERT, LLaMA, and Mistral 7B in our experiment (which makes these models virtually interchangeable). However, what we can clearly and fundamentally conclude is that all three models benefit greatly from boosting (in 25 out of 36 metric scores across all experiments, this approach yields the best performance). That is, across all three error types, we observe that the boosting technique consistently yields the best overall performance. In particular, taking all three metrics together (precision, recall, and accu-

Table 4
Results for mBERT: Precision (P), Recall (R), and Accuracy (A) on the test set for each error type individually, as well as for all error types combined. The highest scores for each error type are highlighted in boldface.

	Reference Systems		Ours
	Baseline	Random Selection	Boosting
Case	P = 0.9143, R = 0.9266, A = 0.9199	P = 0.9200, R = 0.9308, A = 0.9249,	P = **0.9306**, R = **0.9535**, A = **0.9412**
Verb	P = **0.9457**, R = 0.9331, A = 0.9398	P = 0.9248, R = 0.9425 , A = 0.9330	P = 0.9418, R = **0.9718**, A = **0.9559**
Capitalization	P = 0.9413 , R = 0.9431, A = 0.9421	P = 0.9133, R = 0.9448 , A = 0.9276	P = **0.9500**, R = **0.9594**, A = **0.9545**
Combined	P = 0.8897, R = 0.8084, A = 0.8541	P = **0.9073** R = 0.9112, A = **0.9090**	P = 0.9012 , R = **0.9179**, A = 0.9086

Table 5
Results for LLaMA 3: Precision (P), Recall (R), and Accuracy (A) on the test set for each error type individually, as well as for all error types combined. The highest scores for each error type are highlighted in boldface.

	Reference Systems		Ours
	Baseline	Random Selection	Boosting
Case	P = 0.9317, R = 0.9245, A = 0.9284	P = 0.9363, R = 0.9302, A = 0.9335	P = **0.9371**, R = **0.9530**, A = **0.9445**
Verb	P = 0.9437, R = 0.9421, A = 0.9429	P = 0.9333, R = 0.9456, A = 0.9390	P = **0.9538**, R = **0.9650**, A = **0.9592**
Capitalization	P = 0.9519, R = 0.9498, A = 0.9509	P = 0.8739, R = **0.9602**, A = 0.9108	P = **0.9592**, R = 0.9587, A = **0.9590**
Combined	P = 0.8877, R = 0.8988, A = 0.8926	P = **0.9201**, R = 0.9178, A = **0.9191**	P = 0.9027, R = **0.9258**, A = 0.9130

Table 6
Results for Mistral 7B: Precision (P), Recall (R), and Accuracy (A) on the test set for each error type individually, as well as for all error types combined. The highest scores for each error type are highlighted in boldface.

	Reference Systems		Ours
	Baseline	Random Selection	Boosting
Case	P = 0.9374, R = 0.9130, A = 0.9260	P = **0.9419** , R = 0.9091, A = 0.9265	P = 0.9405, R = **0.9525**, A = **0.9461**
Verb	P = **0.9536**, R = 0.9322, A = 0.9434	P = 0.9302, R = 0.9408 , A = 0.9351	P = 0.9440, R = **0.9667**, A = **0.9547**
Capitalization	P = **0.9520**, R = 0.9548, A = 0.9533	P = 0.9326, R = 0.9487, A = 0.9401	P = 0.9478, R = **0.9677**, A = **0.9572**
Combined	P = 0.8893, R = 0.8965, A = 0.8925	P = **0.9013**, R = 0.9241, A = **0.9115**	P = 0.8994, R = **0.9257**, A = 0.9111

racy), we observe that the novel boosting approach significantly outperforms the baseline with very few exceptions. In these few exceptions, the two systems are also almost equal (e.g., for verb errors with the basic model mBERT), or the baseline method pays for the better result with a significant deterioration in the other metrics (e.g., Mistral 7B achieves slightly better precision than our boosting approach for verb errors in the baseline setup, but achieves a significantly worse recall value).

Regarding the ablation study, our results show that the random selection strategy slightly improves recall compared to the baseline setups. However, random selection also results in slight to significant decreases in precision in general (except for case errors for mBERT and LLaMA 3 as well as the combined setup, where precision improves slightly). These results indicate that while random data augmentation can enhance recall, it often comes at the cost of precision, unless the injected data aligns well (by chance) with the model's existing capabilities. For our novel method, this means that its strength lies not only in the simple addition of further training data, but in the targeted boosting approach we propose in this paper.

Clearly the classification task becomes more challenging when all error types are combined into a single data set. All three models (mBERT, LLaMA 3, Mistral 7B) produce lower scores across all evaluation metrics for this combined data set (regardless the experimental setup). In this combined setup, the random selection strategy shows slightly better performance than boosting in terms of precision – yet, the boosting approach is still the best model according to recall, which is the more important metric.

We visualize the learning process using the extracted embeddings from mBERT in Fig. 4.

For the illustration, we apply UMAP (McInnes and Healy, 2018) as a dimensionality reduction technique and project the high-dimensional embeddings into two dimensions. More specifically, we illustrate how the representations of the validation set evolve over the course of training. We compare three stages: (a) the pre-trained mBERT model without any fine-tuning, (b) the model after the first epoch of our boosting algorithm, and (c) the final model after completing the full boosting procedure. These visualizations highlight how the model's representations become increasingly separable as training progresses. Initially, the embeddings extracted from the pre-trained model show no separability between the two classes. After the first epoch, a clear distinction between the classes can be observed. Finally, at the end of the boosting procedure, the separation becomes more refined, reflecting the model's capabilities to distinguish between correct and incorrect sentences.

5.2. Evaluation with real-world data

As observed in the previous section, fine-tuning the models with the proposed boosting technique produces in general the best results in terms of recall, while preserving high precision. However, the experiments presented above are conducted on synthetic data only. Therefore, in this subsection we now aim to evaluate the performance of the novel boosting models on real-world data described in Section 3.

The experiments on real-world data are divided into two parts. In the first part, we perform a comprehensive error analysis on a subset of the predictions (for each error type) to identify strategies for improving recall. In the second part, we use the results of this error analysis to further improve the models with targeted fine-tuning. For these experiments, we only use one basic model, namely mBERT, for two reasons. First, we can see from the experiments above that the differences between the individual language models (mBERT, LLaMA 3, Mistral 7B) are only marginal. Second, although we can assume that the experiments and analyses that follow can be readily transferred to the other language models, no major gains in knowledge are to be expected from this expansion of the experimental volume.

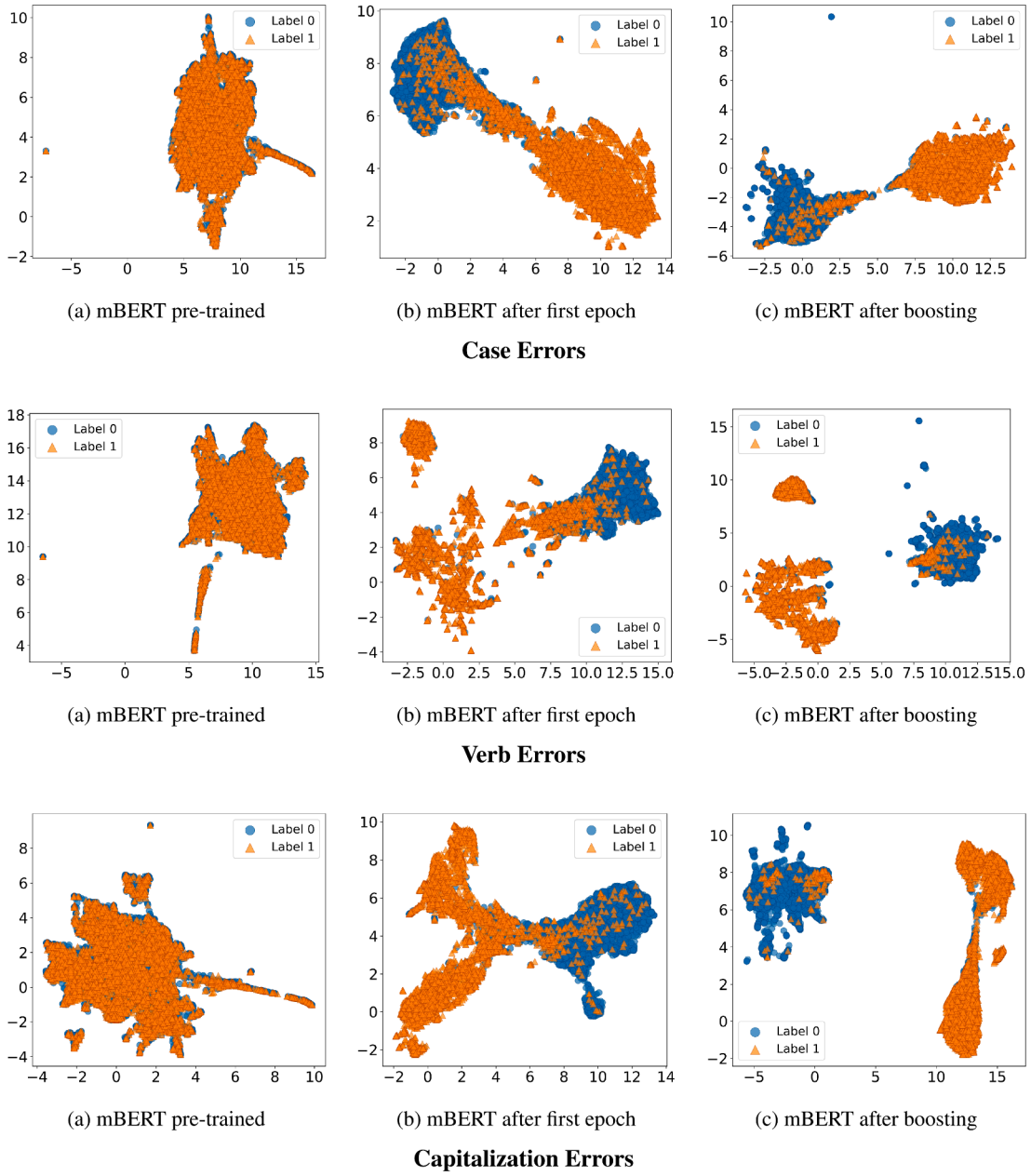

Fig. 4. Two-dimensional UMAP projections of the embeddings from the validation set at different stages of training with mBERT. We compare (a) the pre-trained model without fine-tuning, (b) the model after one epoch of boosting, and (c) the final model after completing the boosting procedure.

5.2.1. Error analysis

The error analysis described in this subsection is performed for each of the three error types individually with the goal of identifying patterns that indicate the potential to improve the model's error detection abilities. In particular, for each error type, we manually analyze 50 false negative predictions to identify recurring error patterns that the baseline model fails to detect (in our case mBERT).

Each of the identified error patterns is classified into two categories, viz. *feasible* and *infeasible*. The category *feasible* includes error patterns that can be synthetically generated with high reliability and low complexity. The *infeasible* category includes patterns for which synthetic data generation is impractical or unjustified due to their rarity. By structuring our analysis this way, we are able to prioritize the most impactful and feasible error patterns.

Fig. 5 shows the results of our error analysis for one error type (case errors). Feasible categories are represented by green bars, while infeasible ones are shown in red. Similar plots are available for the other two error types in Appendix A.

In the 50 erroneous sentences we identify 11 error patterns in total. Three of those patterns are classified as feasible to generate synthetic data (accounting for 27 out of 50 occurrences). That is, creating meaningful synthetic data for further identification appears entirely feasible for around half of the errors. The most prevalent error pattern in this category is the occurrence of an incorrect case after a preposition. For example, the German preposition 'auf' can require either the accusative or dative case, depending on whether a direction ('Ich lege es auf *den* Tisch') or a location ('Das Buch liegt auf *dem* Tisch') is expressed. This dual behavior makes such cases prone to error and underscores the value

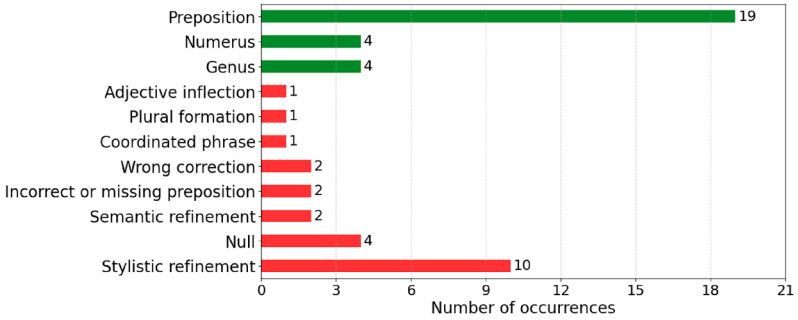

Fig. 5. Results of the error analysis for case errors. Each of the 50 errors can be assigned to one of 11 error patterns for this error type. The diagram shows the absolute frequency of the errors split into these 11 patterns.

Table 7
Overview of error patterns from the category feasible for each error type, along with the corresponding explanation.

Error Type	Error Pattern	Explanation
Case	Preposition	Prepositions govern the grammatical case of subsequent nouns.
	Numerus	Nouns must match the singular or plural form based on context.
	Genus	Nouns have a fixed grammatical gender.
Verb	Coordinated Subjects	Multiple singular subjects joined together require a plural verb.
	Singular Subject	A singular subject requires a verb in sinuglar form.
	1 von X	After the expressions *1 von X*, the verb needs to be singular.
	Du und X	The phrase *Du und X* requires the verb to be conjugated in third-person plural.
	Percentage	Percentage-based subjects require a plural verb form.
Capitalization	Colon	After a colon if the following phrase is not a full sentence the first letter needs to be in lowercase.
	Adjective	Adjectives need to be in lowercase.

of detailed analyses for guiding more nuanced synthetic data generation. The other two patterns are named *Numerus* and *Genus*. *Numerus* errors occur when a word is incorrectly formed in the singular or plural form. Finally, *Genus* errors arise when determiners are replaced with incorrect gender forms.

In terms of verb errors, we find that five out of 13 patterns, with a total of 25 out of 50 occurrences, fall into the feasible category. One notable verb error pattern arises when two or more singular subjects are joined (coordinated subjects) and the verb erroneously appears in the singular instead of the required plural form. The other verb error patterns are as follows. The *1 von X* pattern refers to cases where the phrase *1 von X* (with *X* representing an arbitrary noun) functions as the subject and the verb is incorrectly conjugated in the plural instead of the correct singular form. In the *Du und X* pattern, the subject consists of *Du und X* and the verb is wrongly conjugated in the second-person plural instead of the third-person plural. *Percentage* errors occur when the verb is mistakenly conjugated in singular form, although the subject refers to a percentage of a plural entity.

In the case of capitalization errors, we find that two out of seven patterns, with a total of 31 out of 50 occurrences, fall into the feasible category. The most noticeable pattern in capitalization errors is the one concerning colon rules. If a colon is followed by a complete sentence, the first word must be capitalized; otherwise, it should be written in lowercase. This rule is often misunderstood, even by native German speakers, and therefore, they are likely to appear in pre-training data without correction. This makes it harder for the models to learn the correct pattern, and highlights the need for targeted synthetic data based on detailed analysis. The second feasible pattern is the *adjective* pattern, an adjective is mistakenly capitalized, even though it should appear in lowercase.

Table 7 summarizes all feasible patterns for each error type along with an explanation.

5.2.2. Targeted fine-tuning

The error analysis presented in the previous section reveals some prominent gaps in the synthetic data generation process. In particular, it turns out that some nuances of real-world data could not be accurately replicated with the proposed method to generate the synthetic data. One possibility to address this issue is to introduce hand-crafted rules that capture these errors in combination with the language model. However, relying on rules is tedious and inflexible as writing rules for each error pattern is not only time-consuming but also difficult to scale and maintain. Moreover, broadly defined rules often lead to many false positives, while narrowly defined ones risk missing valid cases.

Instead, we propose using a data-driven approach. The goal is to generate targeted synthetic data reflecting undetected errors and fine-tune the model accordingly. For the proposed approach, we again use Wikipedia data as the basis for error injections to replicate the patterns categorized as feasible (shown in Table 7). With the ten error patterns identified as feasible, we develop ten injection strategies. Table 8 gives both an overview of these ten injection strategies for each error pattern and one example sentence for each pattern.

For the experimental evaluation, we determine the optimal number of additional samples for each error type. Therby, our goal is to maximize recall while keeping a precision of at least 70% or higher. The best results using mBERT are obtained with 30,000 additional samples for case errors, and 6000 added samples each for both verb and capitalization errors.

Fig. 6 shows a comparison between the initial inference results and the results after additional fine-tuning[4] We observe that recall improves across all error types, though this comes at a slight deterioration of the precision. The most significant improvement for the individual error types can be observed for the capitalization errors, where recall increases from 0.71 to 0.81. We attribute this to the colon error pattern, which is likely underrepresented in the original training data. However, at the same time precision decreases from 0.78 to 0.71, indicating a higher rate of false positives. In the verb error category, we already have a high recall of 0.92 before further fine-tuning (which remains

[4] Remember that this evaluation is conducted on the real-world test set, excluding the samples used for error analysis.

Table 8
Injection rules and representative synthetic examples for each error pattern categorized as feasible in the error analysis. Injected errors are visualized in boldface.

Injection Rule	Original Correct Sentence	Sentence with Injected Error
If a preposition is detected, we alter the case of the noun it governs.	Der Bahnhof wird von `der` Linie U1 bedient.	Der Bahnhof wird von `den` Linie U1 bedient.
We change the numerus of a noun (singular ↔ plural).	Das führte zu jahrelanger `Fehde` zwischen den beiden Geschlechtern.	Das führte zu jahrelanger `Fehden` zwischen den beiden Geschlechtern.
We detect a noun's gender and substitute it with an incorrect one.	`Der` Sitz der Countyverwaltung (County Seat) befindet sich in Newport.	`Die` Sitz der Countyverwaltung (County Seat) befindet sich in Newport.
When *1 von X* appears, we replace the singular verb with its plural form.	1 von 25 Patienten `gab` eine Antwort.	1 von 25 Patienten `gaben` eine Antwort.
If *Du und X* appears, we replace the verb with an incorrect form.	Du und dein Bruder `fandet` einen Weg.	Du und dein Bruder `fanden` einen Weg.
If the subject is a percentage, we change the verb into its singular form.	In Island `sind` ungefähr 10% der Höfen nach Frauen benannt.	In Island `ist` ungefähr 10% der Höfen nach Frauen benannt.
If coordinated singular subjects are detected, we replace the verb with its singular form.	Anna und Max `spielen` Fussball.	Anna und Max `spielt` Fussball.
If a singular subject is detected, we replace the verb with its plural form.	Mittlerweile `ist` die Zahl der Absolventen auf über 4000 gestiegen.	Mittlerweile `sind` die Zahl der Absolventen auf über 4000 gestiegen.
If the word follwing a colon is in lowercase, we change it into uppercase.	Sieg: `zwei` Punkte; Unentschieden: `ein` Punkt; Niederlage: `kein` Punkt	Sieg: `Zwei` Punkte; Unentschieden: `Ein` Punkt; Niederlage: `Kein` Punkt
If we detect an adjective, we capitalize it.	Sie weist ein `doppeltes` Peristom auf.	Sie weist ein `Doppeltes` Peristom auf.

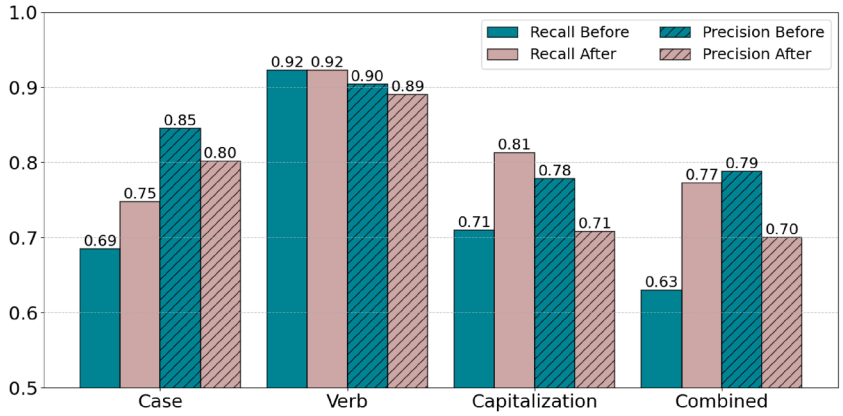

Fig. 6. Comparison of inference results on the real-world test set between mBERT trained with the boosting algorithm and the same model after targeted fine-tuning.

stable after fine-tuning). Precision also remains relatively stable, dropping marginally from 0.90 to 0.89. Case errors turn out to be the most difficult error type. The implementation of injecting rules is particularly complex, as correct case assignment depends on the context and the ability to detect syntactic dependencies within the sentence. Despite these challenges, the targeted fine-tuning improves recall from 0.69 to 0.75, which can be interpreted as a considerable improvement. Precision, however, declines from 0.85 to 0.80.

When considering all error types together (termed 'Combined' in Fig. 6), recall improves from 0.63 to 0.77, demonstrating the overall effectiveness of the additional fine-tuning. However, this comes with a decrease in precision from 0.79 to 0.70, highlighting the trade-off between recall and precision. For instance, a closer inspection reveals that the model starts to overgeneralize certain patterns, such as wrongly flagging correct capitalization after colons. That is, while recall improves through targeted fine-tuning, maintaining precision becomes more difficult.

5.3. Comparison with prompt-based systems

Recent work has demonstrated the strong performance of LLMs in grammatical error detection and correction, outperforming many traditional systems in zero-shot and few-shot settings through prompt-based inference (Loem et al., 2023). Given their widespread adoption and competitive performance, we compare our approach against prompt-based grammatical error detection using GPT-4o (Hurst et al., 2024) as a state-of-the-art reference. The temperature is fixed to 0.5 for all evaluations, selected based on preliminary experiments.

We evaluate three prompt variants of increasing structure:

(i) a simple classification prompt,
(ii) a *chain-of-thought (CoT)* prompt that encourages step-by-step reasoning,
(iii) an enhanced CoT prompt augmented with few-shot examples and an internal verification framework to minimize over-corrections.

All prompts instruct the model to produce a binary decision indicating whether an error is present, together with a textual justification inside predefined tags. We explicitly showcase the three prompts for capitalization errors in Appendix B, translated into English (the original prompts are in German). For all other error types, we use the same prompt structure, only modifying the description of the error type.

Table 9 shows precision, recall, and accuracy achieved on the synthetic test set for each prompt variant. We compare these results against our best-performing model derived in Section 5.1.

Transitioning from the simple prompt to CoT increases precision at the cost of recall for verb and capitalization errors, indicating more conservative predictions. For case errors, the introduction of reasoning slightly improves recall while degrading precision. This suggests that while reasoning helps the model identify more potential case errors, it also introduces more false positives. The enhanced CoT prompt (including examples) achieves the highest prompt-based accuracy across all error types. For capitalization, it significantly recovers the recall lost in the

Table 9
Results of GPT-4o compared to our best model (LLaMA 3, Boosting): Precision (P), Recall (R), and Accuracy (A) on the test set for each error type individually. The highest scores for each error type are highlighted in boldface.

	GPT-4o			Ours
	Simple Prompt	CoT	Enhanced CoT + Examples	Boosting
Case	P = 0.8144, R = 0.8808, A = 0.8400	P = 0.8033, R = 0.8827, A = 0.8333	P = 0.8262, R = 0.8916, A = 0.8520	P = **0.9371**, R = **0.9530**, A = **0.9445**
Verb	P = 0.8166, R = 0.8429, A = 0.8268	P = 0.8955, R = 0.7783, A = 0.8437	P = 0.8606, R = 0.8333, A = 0.8492	P = **0.9538**, R = **0.9650**, A = **0.9592**
Capitalization	P = 0.7220, R = 0.8047, A = 0.7474	P = 0.8374, R = 0.6426, A = 0.7588	P = 0.7706, R = 0.8134, A = 0.7856	P = **0.9592**, R = **0.9587**, A = **0.9590**

Table 10
Representative examples of false positive and false negative predictions generated by GPT-4o. Each example includes the input sentence (S), the answer of the model translated into English (A), and an explanation of the underlying grammatical structure (E).

Case	Sentence (S), Answer (A), and Explanation (E)
False Positives	**S**: Das Spiel in Griechenland fand vor einer griechisch-national gesinnten Kulisse statt, was nach dem Ende zu Verstimmungen inoffiziellen türkischen Kreisen führte. **A**: The error lies in "inoffiziellen türkischen Kreisen". Since this is a reaction after the end of the game, the dative case should be used: "inoffiziellen türkischen Kreisen". **E**: The model marks the sentence as incorrect based on an invented rule that links case choice to temporal context, and then output a "correction" identical to the input.
	S: Die Interne Revision bietet Sicherheit durch unabhängige und objektive Prüfungen und Beratung in Bezug auf die Angemessenheit und Wirksamkeit der Governance und des Risikomanagements. **A**: The case error lies in "der Governance und des Risikomanagements". Both terms should be in the same case. The correct forms would be "der Governance und des Risikomanagements" in the genitive case or "die Governance und das Risikomanagement" in the accusative case. **E**: The model incorrectly assumes that the coordinated nouns are not already in the same case. It first suggests a "correction" identical to the input and then proposes an alternative that is ungrammatical in this context since the genitive is required.
False Negatives	**S**: […] Nach der Machtergreifung mussten Iversen alle Ämter niederlegen. **A**: The text contains no verb errors regarding person, number, or tense that alter the context. All verbs are used correctly within the context. **E**: The subject *Iversen* refers to a single person and is therefore singular. The model likely misinterprets it as plural due to the frequent plural suffix *-en* in German.
	S: Fahrer wuchs in einer Musikerfamilie auf: ihr Vater ist der Schlagzeuger Thomas Fahrer, ihre Mutter die Sängerin Githe Christensen. **A**: This text contains no real-word capitalization errors. All words are correctly capitalized or lowercase, according to their function in the sentence. **E**: The model fails to apply capitalization rules following a colon that introduces a full sentence. In this context, *ihr* should be capitalized as *Ihr*.

standard CoT variant (from 0.6426 to 0.8134) while maintaining competitive precision. Across all prompt configurations, GPT-4o performs worst on capitalization errors. This difficulty likely stems from the fact that German requires nouns and substantivized words to be capitalized, which is a distinct feature that contrasts with the English-heavy training data typical of general-purpose LLMs.

Across all error types, our boosting approach substantially outperforms prompt-based inference by approximately 10 percentage points in accuracy. This suggests that specialized models fine-tuned for error detection in German are more reliable than zero-shot or few-shot prompting with general-purpose LLMs.

To better understand these results, we manually inspect ten false positives and ten false negatives per error type produced by the CoT prompt. Table 10 shows four representative examples. The key takeaways from this analysis are as follows:

- Reasoning-related errors: Approximately 40% of false positives across all error types stems from invented or incorrect grammatical rules, internally inconsistent reasoning, and stylistic "corrections" that go beyond the targeted error categories.
- Case errors: Many case-related errors result from incorrect identification of the grammatical case.
- Verb errors: False positives often arise from failures to correctly identify the grammatical subject and its number, while false negatives are mainly observed in syntactically complex sentences.
- Capitalization errors: The model frequently misinterprets multi-word proper names and substantivized adjectives and struggles with required capitalization after punctuation, such as following colons.

5.4. Limitations in practical applications

In the experiments presented so far, we evaluate our models using balanced data sets with a 50% / 50% distribution of erroneous and correct sentences, as described in Section 3. This setting is intentionally chosen to support controlled experimentation and direct method comparison, allowing the effects of the proposed boosting strategy to be observed without being obscured by extreme class imbalance. However, we acknowledge that this setup does not reflect real-world conditions, where grammatical errors – and real-word errors in particular – occur far less frequently.

To assess the practical applicability of our approach under realistic conditions, we therefore adapt the test sets to match the empirical error distributions observed in real-world data. Starting from the original test sets, we subsample sentences such that the proportion of erroneous and correct instances corresponds to the real-world distribution for each error type. As a compromise between computational feasibility and the need for a sufficient number of positive samples to ensure reliable evaluation, we fix the resulting test set size to 100,000 sentences. Additional correct sentences are drawn from the Wikipedia corpus and inference

Table 11
Results on adapted test sets reflecting real-world class imbalance with mBERT: We report F1, ROC-AUC, and Average Precision (AP). The highest scores for each error type are highlighted in boldface..

	Reference Systems				Ours	
	Baseline		Random Selection		Boosting	
Case	F1	= 0.4451	F1	= 0.4535	F1	= **0.4538**
	ROC-AUC	= 0.9786	ROC-AUC	= 0.9837	ROC-AUC	= **0.9865**
	AP	= 0.7495	AP	= **0.7958**	AP	= 0.7920
Verb	F1	= **0.2218**	F1	= 0.1763	F1	= 0.1749
	ROC-AUC	= 0.9859	ROC-AUC	= 0.9872	ROC-AUC	= **0.9898**
	AP	= **0.6576**	AP	= 0.6543	AP	= 0.6354
Capitalization	F1	= 0.6114	F1	= 0.4634	F1	= **0.6246**
	ROC-AUC	= 0.9870	ROC-AUC	= 0.9815	ROC-AUC	= **0.9907**
	AP	= **0.8233**	AP	= 0.7901	AP	= 0.8216

is performed using the models obtained in Section 5.1, without further retraining.[5]

Table 11 reports the results under this realistic class imbalance.

In addition to the F1 score, we report *ROC-AUC* and *Average Precision (AP)*, which are more informative in highly imbalanced settings. The results reveal substantial differences between these metrics, indicating that while the models are generally able to rank erroneous sentences reliably (as reflected by high ROC-AUC and AP values), they also produce a non-negligible number of false positives, leading to reduced precision and consequently lower F1 scores. This effect is most pronounced for verb errors, which constitute the rarest error type in the data set.

From a practical perspective, the strong ranking performance suggests that these models can still be useful in downstream applications such as human-assisted proofreading. In such a setting, sentences could be reviewed in descending order of predicted error likelihood, and the process could be stopped once the proportion of false positives becomes too high. Moreover, the gap between ranking-based metrics and F1 indicates that adjusting the decision threshold, for example, by requiring higher model confidence before predicting an error, may improve precision and F1 scores.

Across all error types, the boosting approach achieves the highest ROC-AUC scores and yields the best F1 scores for two out of three error categories. For case and capitalization errors, this suggests that boosting increases the number of true positives while maintaining a level of false positives comparable to the baseline and random selection strategies. In contrast, for verb errors, both boosting and random selection increase the number of false positives without a corresponding gain in true positives, resulting in lower F1 scores than the baseline despite higher ROC-AUC values. This divergence illustrates an inherent trade-off of the boosting strategy: while it improves sensitivity and ranking performance, it may be ill-suited for extremely rare error types unless combined with more conservative decision thresholds or downstream filtering mechanisms.

6. Conclusions and future work

Transformer-based language models currently represent the state of the art in natural language processing, largely due to their ability to capture complex contextual dependencies and linguistic patterns. Over time, numerous architectural enhancements have improved the performance of these models resulting in a variety of model variants. These systems demonstrate remarkable accuracy across diverse data sets and languages. However, some challenges in automatic detection and correction of textual errors remain. In particular, handling low-resource languages and subtle error types remains a difficulty for current models. In this paper, we discuss the challenge of detecting real-word errors, which are particularly difficult due to their subtle and context-dependent nature.

We propose a novel boosting-based approach that incrementally improves real-word error detection by fine-tuning language models with synthetic data. The approach begins with standard fine-tuning on the training set, followed by evaluating performance on the validation set. Errors that remain undetected in the validation set are then addressed by adding additional synthetic samples to the training set that encompass these errors. This process is repeated for several iterations, each time addressing the model's remaining weaknesses. We demonstrate the effectiveness of this approach across three different model architectures (mBERT, LLaMA 3, Mistral 7B). Additionally, we visualize the model's embeddings of the validation set after different stages of our training algorithm using UMAP projections. These plots show increasing class separability, highlighting the effectiveness of the proposed training process.

While our boosting approach improves recall with stable precision, an in-depth error analysis reveals that synthetic data struggles to capture the full complexity of real-world data. To address this, we identify error patterns that the model fails to detect and generate targeted synthetic data for further fine-tuning. However, certain persistent errors, such as incorrect capitalization after colons, remain difficult to detect. This likely stems from the fact that LLMs are pre-trained on vast web-scale corpora that contain frequent human mistakes, which may be internalized as acceptable patterns. Consequently, fine-tuning must overcome deeply embedded biases introduced during pre-training. The targeted fine-tuning step nonetheless enhances performance, demonstrating the value of recall-driven synthetic data augmentation.

A comparison with prompt-based grammatical error detection using GPT-4o further shows that general-purpose LLMs, even when augmented with chain-of-thought reasoning and few-shot examples, remain less reliable than task-specific models for real-word error detection in German. In particular, prompt-based inference exhibits systematic weaknesses for capitalization errors and syntactically complex constructions, highlighting the limitations of relying solely on zero-shot or few-shot prompting in professional proofreading scenarios.

Moreover, the evaluation under realistic class imbalance reveals important practical constraints. While the proposed boosting strategy consistently yields strong ranking performance, as reflected by high ROC-AUC and AP values, this does not always translate into improved threshold-based detection for very rare error types. For such cases, gains in sensitivity are offset by an increase in false positives, indicating that

[5] For this experiment, we report results only for mBERT. As shown in Section 5.1, the three evaluated models exhibit comparable relative trends across all experimental setups. Restricting this analysis to mBERT therefore allows us to assess the impact of realistic class imbalance without introducing additional computational overhead, while preserving the generality of the conclusions.

boosting is most effective for moderately frequent error categories and may require careful threshold calibration in real-world deployments.

For future work, we see several rewarding avenues to pursue. First, the noise injection strategy to generate synthetic data could be refined to better reflect patterns observed in real-world data. The conducted error analysis could be used as a starting point for this purpose. For example, if a capitalization error frequently occurs after a colon, it would be more realistic to inject an error only in that specific syntactic context, rather than replacing the word without regard to its context. Second, it would be interesting to assess the effectiveness of our boosting-based approach when applied to larger models beyond mBERT, LLaMA 3, and Mistral 7B. Finally, the proposed approach could be extended to other domains that involve subtle, context-dependent errors, such as OCR post-correction, where high-quality annotated real-world data is limited, and synthetic augmentation is valuable.

CRediT authorship contribution statement

Corina Masanti: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization; **Hans-Friedrich Witschel:** Writing – review & editing, Supervision, Resources, Project administration, Funding acquisition, Conceptualization; **Kaspar Riesen:** Writing – review & editing, Supervision, Resources, Project administration, Funding acquisition, Conceptualization.

Research Data

Both method and synthetic data will be made available upon request. Some documents contain confidential information and cannot be shared.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Corina Masanti reports financial support was provided by Innosuisse Swiss Innovation Agency. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

Supported by Innosuisse project 101.128 IP-ICT: A PROSE – Advanced PROofreading SErvices and Rotstift AG

Appendix A. Error Analysis

Fig. A.7 shows the error patterns for verb errors, while Fig. A.8 presents the error patterns for capitalization errors.

Appendix B. Prompt Design

The three prompt variants are presented in the boxed examples below.

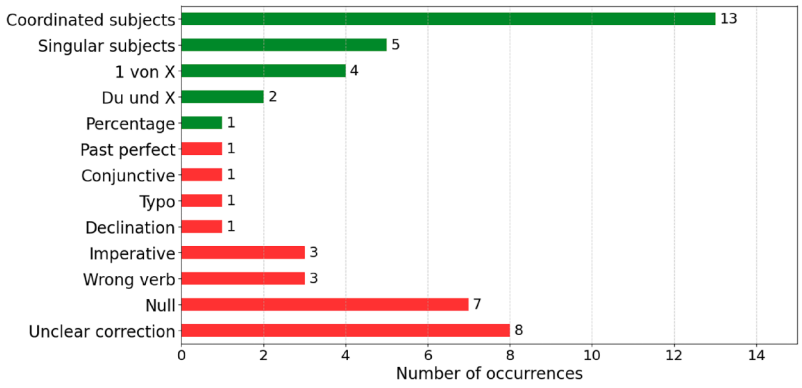

Fig. A.7. Results of the error analysis for verb errors. The y-axis represents the identified error patterns, while the x-axis shows the number of occurrences among the 50 analyzed false negative samples.

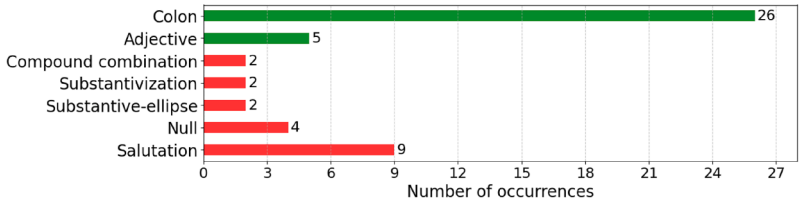

Fig. A.8. Results of the error analysis for capitalization errors. The y-axis represents the identified error patterns, while the x-axis shows the number of occurrences among the 50 analyzed false negative samples.

Simple Prompt (Capitalization):
Is the following text between the <text></text> tags correct or incorrect with respect to capitalization? Output 'correct' inside the <answer></answer> tags if no error is present, and 'incorrect' if an error is present. Explain your decision inside the <why></why> tags.

CoT Prompt (Capitalization):
Your task is to check the text between the <text></text> tags specifically for real-word errors of the capitalization category. Real-word errors are errors in which words are spelled correctly but are used incorrectly in the given context.

Proceed step by step: Examine the text word by word and focus exclusively on whether words are incorrectly capitalized or lowercased such that the intended context no longer fits. Output 'correct' inside the <answer></answer> tags if no errors of this category are present, and 'incorrect' if at least one error of this category is present. Explain the decision inside the <why></why> tags and name the relevant span(s).

Enhanced CoT + Examples Prompt (Capitalization):
Your task is to check the text between the <text></text> tags specifically for real-word errors of the capitalization category. Real-word errors are errors in which words are spelled correctly but are used incorrectly in the given context.

Proceed step by step: Examine the text word by word and focus exclusively on whether words are incorrectly capitalized or lowercased such that the intended context no longer fits. Output 'correct' inside the <answer></answer> tags if no errors of this category are present, and 'incorrect' if at least one error of this category is present. Explain the decision inside the <why></why> tags. The explanation must be limited to a maximum of two sentences and must quote at least one concretely affected word.

Before making your decision, internally answer the following questions:
(1) Can a specific word form in the text be clearly identified that is potentially affected by a grammatical rule violation?
(2) Is there a clearly justifiable and mandatory rule (i.e., not optional or stylistic) of the capitalization category that applies to this word form?
(3) Does the potential rule violation clearly belong to this error category rather than to style, semantics, or another grammatical category?

Examples:
<text>Bei bestehen erhalten die Absolventen das Eidgenössische Fähigkeits-zeugnis.</text>
<answer>incorrect</answer>
<why>'bestehen' is used here as a noun and must therefore be capitalized. The correct form would be 'Bestehen'.</why>

<text>Von 1931 bis 1936 arbeitete sie als Berufsberaterin und Berufsschullehrerin in Glarus.</text>
<answer>correct</answer>
<why>All words are correctly capitalized or lowercased in the given context. There is no capitalization error.</why>

<text>Zudem war sie dreimal für den Literaturnobelpreis nominiert (1927, 1928 und 1930): ihr Gesamtwerk umfasst 48 Romane und Novellen sowie etwa 85 Kurzgeschichten.</text>
<answer>incorrect</answer>
<why>After a colon, the following word must be capitalized if a complete sentence follows. Here, a complete sentence begins after the colon, so 'Ihr' would be correct instead of 'ihr'.</why>

<text>Produziert man sie im Weltraum, bekommen sie eine ideal runde Form.</text>
<answer>correct</answer>
<why>All words are correctly capitalized or lowercased in the given context. There is no capitalization error.</why>

Now analyze the following text:

References

Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M., 2019. Optuna: a next-generation hyperparameter optimization framework. In: Teredesai, A., Kumar, V., Li, Y., Rosales, R., Terzi, E., Karypis, G. (Eds.), Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4–8, 2019. ACM, pp. 2623–2631. https://doi.org/10.1145/3292500.3330701

Alikaniotis, D., Raheja, V., Tetreault, J.R., 2019. The unreasonable effectiveness of transformer language models in grammatical error correction. In: Yannakoudakis, H., Kochmar, E., Leacock, C., Madnani, N., Pilán, I., Zesch, T. (Eds.), Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications, BEA@ACL 2019, Florence, Italy, August 2, 2019. Association for Computational Linguistics, pp. 127–133. https://doi.org/10.18653/V1/W19-4412

Attardi, G., 2015. Wikiextractor. https://github.com/attardi/wikiextractor.

Björn, P., ChristophS., 2024. LAION, Hessianai, laion leolm: linguistically enhanced open language model https://huggingface.co/DiscoResearch/Llama3-German-8B.

Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D.M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., Amodei, D., 2020. Language models are few-shot learners. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (Eds.), Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6–12, 2020, Virtual. https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.HTML.

Bryant, C., Felice, M., Andersen, Ø.E., Briscoe, T., 2019. The BEA-2019 shared task on grammatical error correction. In: Yannakoudakis, H., Kochmar, E., Leacock, C., Madnani, N., Pilán, I., Zesch, T. (Eds.), Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications, BEA@ACL 2019, Florence, Italy, August 2, 2019. Association for Computational Linguistics, pp. 52–75. https://doi.org/10.18653/V1/W19-4406

Bryant, C., Yuan, Z., Qorib, M.R., Cao, H., Ng, H.T., Briscoe, T., 2023. Grammatical error correction: a survey of the state of the art. Comput. Linguistics 49 (3), 643–701. https://doi.org/10.1162/COLI-A-00478

Damerau, F., 1964. A technique for computer detection and correction of spelling errors. Commun. ACM 7 (3), 171–176. https://doi.org/10.1145/363958.363994

Devlin, J., Chang, M., Lee, K., Toutanova, K., 2018. BERT: pre-training of deep bidirectional transformers for language understanding. CoRR abs/1810.04805. http://arxiv.org/abs/1810.04805.

Devlin, J., Chang, M., Lee, K., Toutanova, K., 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In: Burstein, J., Doran, C., Solorio, T. (Eds.), Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2–7, 2019, Volume 1 (Long and Short Papers). Association for Computational Linguistics, pp. 4171–4186. https://doi.org/10.18653/V1/N19-1423

DiscoResearch, 2024. Llama3-german-8b https://huggingface.co/DiscoResearch/Llama3-German-8B.

Etoori, P., Chinnakotla, M., Mamidi, R., 2018. Automatic spelling correction for resource-scarce languages using deep learning. In: Shwartz, V., Tabassum, J., Voigt, R., Che, W., de Marneffe, M., Nissim, M. (Eds.), Proceedings of ACL 2018, Melbourne, Australia, July 15–20, 2018, Student Research Workshop. Association for Computational Linguistics, pp. 146–152. https://doi.org/10.18653/v1/P18-3021

Freund, Y., 1990. Boosting a weak learning algorithm by majority. In: Fulk, M.A., Case, J. (Eds.), Proceedings of the Third Annual Workshop on Computational Learning Theory, COLT 1990, University of Rochester, Rochester, NY, USA, August 6–8, 1990. Morgan Kaufmann, pp. 202–216. http://dl.acm.org/citation.cfm?id=92640.

Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W., 2022. LoRA: low-rank adaptation of large language models. In: The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25–29, 2022. OpenReview.net. https://openreview.net/forum?id=nZeVKeeFYf9.

Hurst, A., Lerer, A., Goucher, A.P., Perelman, A., Ramesh, A., Clark, A., Ostrow, A.J., Welihinda, A., Hayes, A., Radford, A., Madry, A., Baker-Whitcomb, A., Beutel, A., Borzunov, A., Carney, A., Chow, A., Kirillov, A., Nichol, A., Paino, A., Renzin, A., Passos, A.T., Kirillov, A., Christakis, A., Conneau, A., Kamali, A., Jabri, A., Moyer, A., Tam, A., Crookes, A., Tootoonchian, A., Kumar, A., Vallone, A., Karpathy, A., Braunstein, A., Cann, A., Codispoti, A., Galu, A., Kondrich, A., Tulloch, A., Mishchenko, A., Baek, A., Jiang, A., Pelisse, A., Woodford, A., Gosalia, A., Dhar, A., Pantuliano, A., Nayak, A., Oliver, A., Zoph, B., Ghorbani, B., Leimberger, B., Rossen, B., Sokolowsky, B., Wang, B., Zweig, B., Hoover, B., Samic, B., McGrew, B., Spero, B., Giertler, B., Cheng, B., Lightcap, B., Walkin, B., Quinn, B., Guarraci, B., Hsu, B., Kellogg, B., Eastman, B., Lugaresi, C., Wainwright, C.L., Bassin, C., Hudson, C., Chu, C., Nelson, C., Li, C., Shern, C.J., Conger, C., Barette, C., Voss, C., Ding, C., Lu, C., Zhang, C., Beaumont, C., Hallacy, C., Koch, C., Gibson, C., Kim, C., Choi, C., McLeavey, C., Hesse, C., Fischer, C., Winter, C., Czarnecki, C., Jarvis, C., Wei, C., Koumouzelis, C., Sherburn, D., 2024. Gpt-4o system card. CoRR abs/2410.21276. https://doi.org/10.48550/ARXIV.2410.21276

Jiang, A.Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D.S., de Las Casas, D., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Lavaud, L.R., Lachaux, M., Stock, P., Scao, T.L., Lavril, T., Wang, T., Lacroix, T., Sayed, W.E., 2023. Mistral 7b. CoRR abs/2310.06825. https://doi.org/10.48550/ARXIV.2310.06825

Kiyono, S., Suzuki, J., Mita, M., Mizumoto, T., Inui, K., 2019. An empirical study of incorporating pseudo data into grammatical error correction. In: Inui, K., Jiang, J., Ng, V., Wan, X. (Eds.), Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3–7, 2019. Association for Computational Linguistics, pp. 1236–1242. https://doi.org/10.18653/V1/D19-1119

Loem, M., Kaneko, M., Takase, S., Okazaki, N., 2023. Exploring effectiveness of GPT-3 in grammatical error correction: a study on performance and controllability in prompt-based methods. In: Kochmar, E., Burstein, J., Horbach, A., Laarmann-Quante, R., Madnani, N., Tack, A., Yaneva, V., Yuan, Z., Zesch, T. (Eds.), Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications, BEA@ACL 2023, Toronto, Canada, 13 July 2023. Association for Computational Linguistics, pp. 205–219. https://doi.org/10.18653/V1/2023.BEA-1.18

Masanti, C., Witschel, H.F., Riesen, K., 2023. Novel benchmark data set for automatic error detection and correction. In: Métais, E., Meziane, F., Sugumaran, V., Manning, W., Reiff-Marganiec, S. (Eds.), Natural Language Processing and Information Systems - 28th International Conference on Applications of Natural Language to Information Systems, NLDB 2023, Derby, UK, June 21–23, 2023, Proceedings. Springer, pp. 511–521. https://doi.org/10.1007/978-3-031-35320-8-38

Masanti, C., Witschel, H.F., Riesen, K., 2024. Automated error detection through specialized task implementation. In: Wallraven, C., Liu, C., Ross, A. (Eds.), Pattern Recognition and Artificial Intelligence - 4th International Conference, ICPRAI 2024, Jeju Island, South Korea, July 03–06, 2024, Proceedings, Part II. Springer, pp. 182–195. https://doi.org/10.1007/978-981-97-8705-0-12

Masanti, C., Witschel, H.F., Riesen, K., 2025. Boosting language models for real-word error detection. In: Santana, M.C., De Marsico, M., Fred, A. (Eds.), Proceedings of the 14th International Conference on Pattern Recognition Applications and Methods, ICPRAM 2025, Porto, Portugal, February 23–25, 2025. SCITEPRESS, pp. 318–325. https://doi.org/10.5220/0013251500003905

McInnes, L., Healy, J., 2018. UMAP: uniform manifold approximation and projection for dimension reduction. CoRR abs/1802.03426. http://arxiv.org/abs/1802.03426.

Ng, H.T., Wu, S.M., Briscoe, T., Hadiwinoto, C., Susanto, R.H., Bryant, C., 2014. The coNLL-2014 shared task on grammatical error correction. In: Ng, H.T., Wu, S.M., Briscoe, T., Hadiwinoto, C., Susanto, R.H., Bryant, C. (Eds.), Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task, CoNLL 2014, Baltimore, Maryland, USA, June 26–27, 2014. ACL, pp. 1–14. https://doi.org/10.3115/V1/W14-1701

Omelianchuk, K., Atrasevych, V., Chernodub, A.N., Skurzhanskyi, O., 2020. Gector - grammatical error correction: tag, not rewrite. In: Burstein, J., Kochmar, E., Leacock, C., Madnani, N., Pilán, I., Yannakoudakis, H., Zesch, T. (Eds.), Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications, BEA@ACL 2020, Online, July 10, 2020. Association for Computational Linguistics, pp. 163–170. https://doi.org/10.18653/V1/2020.BEA-1.16

Patra, B., Singhal, S., Huang, S., Chi, Z., Dong, L., Wei, F., Chaudhary, V., Song, X., 2023. Beyond english-centric bitexts for better multilingual language representation learning. In: Rogers, A., Boyd-Graber, J.L., Okazaki, N. (Eds.), Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9–14, 2023. Association for Computational Linguistics, pp. 15354–15373. https://doi.org/10.18653/V1/2023.ACL-LONG.856

Pitis, S., Zhang, M.R., Wang, A., Ba, J., 2023. Boosted prompt ensembles for large language models. CoRR abs/2304.05970. https://doi.org/10.48550/ARXIV.2304.05970

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J., 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. J. Mach. Learn. Res. 21, 140:1–140:67. http://jmlr.org/papers/v21/20-074.HTML.

Rei, M., Felice, M., Yuan, Z., Briscoe, T., 2017. Artificial error generation with machine translation and syntactic patterns. In: Tetreault, J.R., Burstein, J., Leacock, C., Yannakoudakis, H. (Eds.), Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications, BEA@EMNLP 2017, Copenhagen, Denmark, September 8, 2017. Association for Computational Linguistics, pp. 287–292. https://doi.org/10.18653/V1/W17-5032

Schapire, R.E., 1990. The strength of weak learnability. Mach. Learn. 5, 197–227. https://doi.org/10.1007/BF00116037

Stahlberg, F., Kumar, S., 2021. Synthetic data generation for grammatical error correction with tagged corruption models. In: Burstein, J., Horbach, A., Kochmar, E., Laarmann-Quante, R., Leacock, C., Madnani, N., Pilán, I., Yannakoudakis, H., Zesch, T. (Eds.), Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications, BEA@EACL, Online, April 20, 2021. Association for Computational Linguistics, pp. 37–47. https://www.aclweb.org/anthology/2021.bea-1.4/.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., Lample, G., 2023. Llama: Open and efficient foundation language models. CoRR abs/2302.13971. https://doi.org/10.48550/ARXIV.2302.13971

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I., 2017. Attention is all you need. In: Guyon, I., von Luxburg, U., Bengio, S., Wallach, H.M., Fergus, R., Vishwanathan, S.V.N., Garnett, R. (Eds.), Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4–9, 2017, Long Beach, CA, USA, pp. 5998–6008. https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.HTML.

Volodina, E., Bryant, C., Caines, A., De Clercq, O., Frey, J.-C., Ershova, E., Rosen, A., Vinogradova, O., 2023. MultiGED-2023 shared task at NLP4CALL: multilingual grammatical error detection. In: Proceedings of the 12th Workshop on NLP for Computer Assisted Language Learning, pp. 1–16.

Wilcox-O'Hearn, L.A., Hirst, G., Budanitsky, A., 2008. Real-word spelling correction with trigrams: a reconsideration of the mays, damerau, and mercer model. In: Gelbukh, A.F. (Ed.), Computational Linguistics and Intelligent Text Processing, 9th International Conference, CICLing 2008, Haifa, Israel, February 17–23, 2008, Proceedings. Springer, pp. 605–616. https://doi.org/10.1007/978-3-540-78135-6-52

Yuan, Z., Felice, M., 2013. Constrained grammatical error correction using statistical machine translation. In: Ng, H.T., Tetreault, J.R., Wu, S.M., Wu, Y., Hadiwinoto, C. (Eds.), Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task, CoNLL 2013, Sofia, Bulgaria, August 8–9, 2013. ACL, pp. 52–61. https://aclanthology.org/W13-3607/.