Universiteit Leiden

From Classroom to Context Window: Pedagogical Principles in LLM Fine-tuning for GEC

Mohammad Hammad Arif (s3668630)

Supervisors:
Tom Kouwenhoven & Sabijn Perdijk

Bachelor Thesis– Informatica

Leiden Institute of Advanced Computer Science (LIACS)
liacs.leidenuniv.nl

2026

Abstract

Language models hold potential for educational applications [15], which raises a natural question: can insights from educational psychology also inform how these models are trained? This thesis tests one such idea. The worked examples principle [23] holds that studying a fully solved example, including all solution steps, leads to better learning than solving equivalent problems independently, particularly for novice learners. We investigate whether translating this principle into a training data format improves the fine-tuning of a small language model on Grammatical Error Correction (GEC).

Two training formats are compared under otherwise identical conditions. A plain fine-tuning condition trains the model on correction pairs alone. A worked examples condition adds a rule-based grammatical explanation to each pair, generated automatically from error-type annotations in the training data. We fine-tune Pythia-410M [4] on a standard learner English GEC corpus and measure performance using ERRANT $F_{0.5}$ [7, 17], a GEC metric that weights precision twice as heavily as recall, across ten random seeds.

The plain fine-tuned condition achieves a mean $F_{0.5}$ of 31.29. Without the explanation at inference time, the Worked Examples condition achieves a mean $F_{0.5}$ of 11.73, a statistically significant difference ($t(9) = 33.96$, $p < .001$). When the explanation is provided at inference time using gold annotations from a held-out test set sampled from the training data, the same Worked Examples model achieves a mean $F_{0.5}$ of 64.17, significantly higher than the fine-tuned condition ($t(9) = -90.62$, $p < .001$). The model learned effectively from the worked examples format. The problem is not a failure to learn, but that the learned behaviour depends on the explanation being present. WinoGrande scores remain near the 50% chance baseline for all conditions (base: 53.59, fine-tuned: 52.72, worked examples: 52.88, seed 41 only), suggesting that fine-tuning produced task-specific learning rather than improved general linguistic competence.

These results point to a practical challenge in training data design: a model can learn effectively from additional information in its training examples, yet fail to apply that learning when the same information is unavailable at inference time. In a realistic GEC deployment setting, gold annotations are not available for new sentences, so the explanation cannot be provided at inference. This finding is relevant for anyone designing fine-tuning setups where training and deployment conditions differ.

Contents

1 Introduction

Educational psychology has identified several principles that reliably improve how humans learn. Among these are curriculum learning, which structures material from simple to complex [3], and retrieval practice, which shows that actively recalling information leads to better long-term retention [19, 20]. This thesis focuses on a third principle: worked examples. According to this principle, studying a fully solved problem, including all solution steps, leads to better learning than solving equivalent problems independently, particularly for novice learners on tasks with well-defined correct answers [23, 22, 2, 11]. Whether this principle, developed to understand human learning, also holds when the learner is a language model being fine-tuned on a specific task is an open question.

There is growing research exploring the relationship between language models and human language learning. Some work asks whether language models can serve as model learners to probe questions about human language acquisition [25, 9], and shared tasks such as the BabyLM [26] challenge investigate whether models trained on developmentally plausible amounts of data can acquire language more efficiently [26]. Such work largely asks what language models can tell us about humans. This thesis asks the reverse: whether a principle from human learning can inform how a language model is trained.

The task used to test this is Grammatical Error Correction (GEC), where a model receives an erroneous sentence and must produce the corrected version. GEC is a good fit for three reasons. First of all, corrections have clearly defined gold-standard answers, which is the condition under which worked examples are most effective in human learning. The second reason is that the training data contains error type annotations that can be used to generate rule-based explanations automatically, without needing a separate teacher model. And lastly, GEC has a standard benchmark and evaluation metric, making it straightforward to compare training conditions fairly.

Despite the strong body of work on GEC systems and on training data format effects in language model fine-tuning, the specific question of whether pedagogical principles from human learning can be translated into GEC training data formats has not been directly studied. Prior GEC work focuses on architectural choices, data augmentation, and evaluation benchmarks. Work on fine-tuning small decoder-only language models for GEC is limited, and no prior study has compared a plain correction training format against one that augments each example with a rule-based grammatical explanation.

This thesis addresses the following research questions:

RQ1 How does a training data format based on the worked examples principle affect the GEC performance of a fine-tuned small language model?

RQ2 Does the direction of this effect align with what the worked examples literature predicts for

novice learners on well-defined tasks?

Based on the worked examples literature, the hypothesis for RQ1 is that the Worked Examples condition will outperform plain fine-tuning. GEC satisfies both conditions that make the principle most effective: the model has no GEC-specific knowledge before fine-tuning, and each correction has a well-defined gold standard [22, 2].

The goal is not to claim that language models learn the way humans do. The goal is to test whether a structural property of worked examples, providing the reasoning alongside the answer, produces the same directional effect in a model as it does in a human learner. Two contributions are made. First, the worked examples principle is operationalised as a concrete GEC training format using rule-based explanations derived from error annotations, and compared against plain fine-tuning across ten random seeds in a controlled experiment. Second, this thesis identifies a boundary condition for training data design: adding information to training examples is only beneficial if that information is also available at inference time, or if the model is explicitly trained to operate without it.

The rest of this thesis is structured as follows. Section 2 covers the background needed to understand this work, introducing Grammatical Error Correction and how it is evaluated, the worked examples principle and the conditions under which it is most effective, and how language model fine-tuning works. Section 3 then discusses related work on prior GEC systems and on training data design for language models. Section 4 describes the full experimental setup, including the dataset, evaluation metrics, training conditions, and pipeline verification. Section 5 presents the results for all three evaluated conditions and examines the model outputs to explain what caused them. Section 6 interprets the findings in relation to RQ1 and RQ2, and draws practical conclusions for training data design. Finally, Section 7 summarises the main findings and outlines directions for future research.

2 Background

This section introduces the three areas of knowledge that underpin this thesis: Grammatical Error Correction and how it is evaluated, the worked examples principle from educational psychology, and how language models are fine-tuned for specific tasks. Together these provide the conceptual foundation for the experimental design described in Section 4.

2.1 Grammatical Error Correction

Grammatical Error Correction is the task of automatically detecting and correcting grammatical errors in written text. The input is a sentence that may contain one or more errors, whereas

the output is the corrected version. For example, the input *"She go to school every day"* should produce the output *"She goes to school every day"*. The model must identify the error, determine the correction, and produce a fluent output without changing parts of the sentence that are already correct.

GEC has a long history in computational linguistics. Early systems relied on hand-crafted rules targeting specific error types such as subject-verb agreement, article usage, and preposition errors [17]. These systems were reliable on the errors they explicitly handled but could not generalise beyond their rule sets. Statistical approaches followed, treating GEC as a translation problem and learning correction patterns from parallel corpora of erroneous and corrected text [13]. These systems generalised better but still required careful feature engineering. Neural sequence-to-sequence models then replaced explicit feature design entirely by learning to map erroneous sentences to corrected ones end-to-end. More recently, tagging-based approaches have shown competitive results by predicting edit operations (insert, delete, or replace) directly on the input tokens, rather than generating the full corrected sentence from scratch [18]. This approach is faster, tends to make fewer unnecessary changes, and its output format aligns naturally with how GEC is evaluated. The BEA-2019 shared task brought these approaches together under a standardised benchmark and evaluation framework [6]. Top systems in the shared task achieved $F_{0.5}$ scores in the range of 60–67, using large pretrained transformer models combined with extensive data augmentation. A score in this range means that for roughly six or seven out of every ten corrections a system proposes, the correction matches the gold standard, and the system catches the majority of real errors present in the text.

The dataset used in this thesis is the BEA-2019 W&I+LOCNESS dataset [6], which combines learner writing from the Cambridge Write and Improve platform with the LOCNESS corpus. Essays are submitted by English language learners across different proficiency levels, identified by CEFR labels A, B, and C, and corrected by trained human annotators. The resulting sentence pairs were processed through ERRANT [7], an automatic error annotation toolkit that extracts the edits between original and corrected sentences and classifies each edit into a linguistic error category. This means the training data already contains not just the original and corrected sentences, but also the position, replacement text, and error type of every correction. For example, `R:VERB:TENSE` for a replaced verb tense, `M:PREP` for a missing preposition, or `U:DET` for an unnecessary determiner.

ERRANT also serves as the evaluation framework. It compares the edits a system produces against the gold standard edits and computes precision, recall, and $F_{0.5}$ [7, 17]. $F_{0.5}$ weights precision twice as heavily as recall, reflecting the preference for conservative corrections in GEC: an incorrect correction can damage an otherwise acceptable sentence, which is generally more harmful than leaving an error uncorrected. The full formula is introduced in Section 4.3 alongside the training objective.

2.2 Language Model Fine-tuning

Language models are first trained on large text corpora in a process called pretraining. During pretraining, the model learns to predict the next token in a sequence, which over billions of examples leads it to acquire broad knowledge of language structure, vocabulary, and common patterns of text [5]. This pretrained model has no knowledge of any specific task. Fine-tuning then continues training on a smaller, task-specific dataset, teaching the model new behaviour tailored to that task while building on what was learned during pretraining. In this thesis, fine-tuning teaches the model to map erroneous sentences to corrected ones using the BEA-2019 training data.

One common form of fine-tuning is instruction tuning, where training examples are formatted as instructions paired with desired responses [27]. Wei et al. showed that fine-tuning on a large collection of instruction-response pairs across many tasks substantially improves a model's ability to follow instructions on tasks it was not directly trained on. Wang et al. extended this by showing that instruction-following data can be generated by the model itself, without human annotation [24]. Both works establish an important point: the format of training examples matters independently of their content, because the structure of examples shapes what the model learns to do.

The training setup in this thesis is inspired by this paradigm. The model is presented with a fixed prompt format that implies a task without stating it explicitly, and learns the expected behaviour from the repeated structure of the training data. The specific format is described in Section 4.

2.3 Pythia-410M

The Pythia model family consists of decoder-only autoregressive language models ranging from 70 million to 12 billion parameters [4]. All models in the family are trained on the same data in the same order, with intermediate checkpoints saved throughout training. This design makes Pythia well-suited to controlled experiments: any differences in behaviour between conditions can be attributed to the experimental manipulation rather than to differences in pretraining.

Pythia-410M has 410 million parameters and was pretrained on The Pile [10], a large English text dataset of approximately 825 gigabytes drawn from 22 different sources. Before any GEC fine-tuning, the model achieves 53.7% on WinoGrande [21], a commonsense reasoning benchmark in which a model selects the most plausible of two options to complete a sentence. A model that simply guessed randomly between the two options would score 50%, so 53.7% means the model performs only marginally better than chance, confirming it has no meaningful linguistic reasoning ability before task-specific training begins.

Pythia-410M was selected for three practical reasons: it fits on a single consumer GPU, its training procedure is fully documented with open weights, and it is large enough to show meaningful

learning from fine-tuning. Whether these properties held in practice is discussed in Section 6.

2.4 Connecting Human Learning to Model Training

Before moving to related work, it is worth being explicit about what the analogy between human learning and model training claims and what it does not, because the discussion in Section 6 returns to this point.

Human learners build knowledge structures called schemas in long-term memory over extended periods of time, retrieving and applying them to new problems [22]. None of these mechanisms have a counterpart in stochastic gradient descent. A language model sees each training example for at most a few passes during training epochs, updates its weights through backpropagation, and has no memory between training runs.

What the Worked Examples condition tests is not whether the mechanisms are the same. It tests whether a specific structural change, adding an explanation alongside the answer, produces a similar effect in a language model as it does in a human learner. Worked examples reduce the information-theoretic demand by providing the reasoning structure directly. Whether that structural property transfers to gradient-based optimisation is an empirical question, and the one this thesis addresses.

3 Related Work

Related work on this thesis spans three areas. The first is prior GEC systems, covering the approaches that have been developed and what they achieved. The second is human corrective feedback research, which places GEC in a broader pedagogical context and motivates the use of explanations in training data. The third is rationale-based and format-aware fine-tuning, which is the machine learning work most directly comparable to the training conditions tested here.

3.1 Prior GEC Systems

Automated GEC has been an active research area, with approaches ranging from statistical methods [17] to modern neural architectures [6, 18]. This section traces that development through the work most relevant to this thesis.

The CoNLL-2014 shared task [17] marked an important step in standardising GEC evaluation. Unlike previous shared tasks that targeted specific error types, CoNLL-2014 required participating systems to correct all grammatical error types present in learner text. The test set consisted of 50 essays written by students at the National University of Singapore, annotated by two expert annotators. Systems were evaluated using the M2 scorer with $F_{0.5}$, weighting precision twice as

heavily as recall. The task also provided the NUCLE corpus as training data, a large collection of learner essays with professional error annotations. Participating systems used a range of approaches including rule-based classifiers and statistical language models, with the best systems achieving $F_{0.5}$ scores around 35–38 on the test set [17]. To put that in concrete terms: a score of 35 means that out of every ten corrections a system proposed, roughly three to four were correct, and the system caught only a fraction of the real errors present in the text.

Following CoNLL-2014, Junczys-Dowmunt and Grundkiewicz [13] showed that treating GEC as a machine translation problem could substantially improve results. Their system used a phrase-based SMT setup with task-specific parameter tuning towards the M2 metric, combining web-scale language models with large-scale error-corrected parallel corpora. This approach achieved an M^2 $F_{0.5}$ of 46.37 on the CoNLL-2014 test set. Meaning the system was now getting close to half its proposed corrections right while catching a substantially larger share of real errors, outperforming all previously published results by a large margin and establishing SMT as the dominant paradigm for GEC at the time.

The BEA-2019 shared task [6] addressed two limitations of CoNLL-2014. First, it introduced a new and more diverse dataset, the W&I+LOCNESS corpus, covering a wider range of learner proficiency levels and native language backgrounds than the single-institution essays used in CoNLL-2014. Second, it replaced the M2 scorer with ERRANT [7] as the standard evaluation framework, enabling finer-grained analysis of corrections by error type. The shared task introduced multiple tracks controlling the amount of annotated data available to participants. Top systems in the restricted track used transformer-based neural machine translation models and achieved $F_{0.5}$ scores in the range of 64–70 on the test set [6]. This range represents the upper end of what automated GEC systems have achieved on this benchmark: a system at 70 $F_{0.5}$ is making correct corrections roughly seven times out of ten while catching the large majority of real errors in learner text.

GECToR [18] introduced a different architectural direction. Rather than generating the corrected sentence token by token, GECToR predicts a sequence of edit operations directly on the input tokens using a transformer encoder. The system is pretrained on nine million synthetically generated parallel sentences, then fine-tuned in two stages: first on errorful corpora, then on a combination of errorful and error-free parallel data. This tagging approach avoids the computational cost of autoregressive decoding and runs up to ten times faster than sequence-to-sequence alternatives. The best single model achieves $F_{0.5} = 65.3$ on CoNLL-2014 and $F_{0.5} = 72.4$ on BEA-2019, with ensemble models reaching 66.5 and 73.6 respectively [18].

3.2 Written Corrective Feedback in Human Learning

Before automated GEC systems existed, the question of how to correct grammatical errors in learner writing was studied extensively in applied linguistics under the term written corrective

feedback (WCF). This body of work is relevant here because it provides empirical evidence from human learning about whether providing explanations alongside corrections changes what learners acquire.

A key distinction in WCF research is between focused and unfocused feedback. Unfocused feedback corrects all error types in a piece of writing without specifically identifying what kind of error each correction addresses. Focused feedback targets specific error types and often includes explicit error codes that tell the learner what grammatical category the error falls into. Deng et al. [8] compared both approaches in a controlled study with 47 ESL students in Hong Kong over an eight-week intensive course. Three groups received either focused WCF with explicit error codes (e.g. `Art` for article errors, `V-T` for verb tense errors, `Prep` for preposition errors), unfocused WCF that marked all errors without categorisation, or a control condition receiving only content and organisation feedback. The study found that the focused WCF group significantly outperformed both the unfocused and control groups on subsequent writing accuracy across all three targeted error types. No significant difference was found between the unfocused and control groups, suggesting that correction alone, without information about the error type, provided little benefit.

Kao et al. [14] examined unfocused WCF specifically, finding that it improved article accuracy with gains holding in delayed posttests, but produced weak effects on preposition and verb tense errors. Together, these findings suggest that providing learners with explicit information about the nature of their errors, not just the correction itself, plays an important role in what is acquired.

3.3 Fine-tuning Decoder-Only Models for GEC

The approaches described above all use either encoder-decoder architectures or tagging-based models. A closer parallel to the setup in this thesis is found in work that fine-tunes small decoder-only language models directly on GEC data.

Lamelas [16] investigated whether small decoder-only language models could serve as an efficient alternative to large LLMs for grammar correction. Three models were fine-tuned on the BEA-2019 W&I+LOCNESS training data [6]: GPT-2 (117M parameters), GPT-2 Medium, and GPT-Neo-125M. All three are autoregressive causal language models that generate the corrected sentence token by token from left to right, conditioned on the erroneous input. The prompt format used was `"Correct this text": [input] | Corrected: [target]`, structurally similar to the `Fix: [input]\nCorrect: [target]` format used in this thesis. Models were fine-tuned for 3 epochs using AdamW with a learning rate of 5e-5 and greedy decoding at inference time. Evaluation was performed on the JFLEG dataset, a fluency-focused GEC benchmark distinct from BEA-2019, using GLEU and M^2 $F_{0.5}$.

The results showed a large gap between small decoder-only models and large LLMs. GPT-3.5 Turbo achieved a GLEU score of approximately 75 and an M^2 $F_{0.5}$ of approximately 51. The best

performing small model (Gemma 2B, tested without fine-tuning) reached only GLEU 25 and M^2 $F_{0.5}$ 16. The average M^2 $F_{0.5}$ across all fine-tuned SLMs was approximately 5, meaning these models were correcting almost nothing correctly, performing only marginally better than a system that makes no corrections at all. Notably, fine-tuning produced almost no improvement in $F_{0.5}$: scores before and after fine-tuning were nearly identical, suggesting the models failed to learn meaningful correction behaviour from the training data. The paper concluded that small decoder-only models struggle with fine-grained grammar correction and that substantial advances in training are needed before they can approach LLM performance.

3.4 Instruction Tuning and Format Effects

A significant line of work in language model research shows that the format in which training examples are presented changes what a model learns, independently of the content of those examples. Wei et al. [27] demonstrated this through instruction tuning. They took a 137B parameter pretrained language model and fine-tuned it on over 60 NLP datasets, each reformatted as a natural language instruction paired with a desired response. The instruction templates describe the task in plain language: for example, a sentiment classification example might be presented as *"Does the following review express a positive or negative opinion? [review]"* followed by the correct label. The resulting model, FLAN, was evaluated zero-shot on task types it had never been trained on. FLAN surpassed zero-shot GPT-3 (175B parameters) on 20 out of 25 evaluation datasets, and even outperformed GPT-3 few-shot on several benchmarks [27]. This is notable because GPT-3 has 175 billion parameters compared to FLAN's 137 billion, yet the instruction-formatted fine-tuning alone was enough for the smaller model to outperform the larger one on most tasks.

Ablation studies in the same work revealed which factors drove the improvement. Increasing the number of task clusters in the fine-tuning mixture consistently raised zero-shot performance on held-out tasks. Model scale mattered: smaller models benefited less from instruction tuning. And replacing natural language instruction templates with no instructions or with classification-style prompts removed most of the benefit. The conclusion is that it is specifically the natural language instruction format that teaches the model to generalise, not the additional data or the diversity of tasks alone. Instruction tuning is most effective on tasks naturally verbalized as instructions, such as natural language inference, question answering, and translation, and less effective on tasks formulated directly as language modelling objectives where instructions would be redundant [27].

Wang et al. [24] extended this finding by showing that instruction-formatted training data does not need to be human-authored. They started with 175 seed instruction-output pairs written by humans, covering a range of tasks. A language model was prompted to generate new instruction-output pairs using these seeds as context, producing 52,000 diverse instruction-output examples covering tasks the seed set did not explicitly include. Quality filtering removed near-duplicate

instructions using ROUGE-L similarity, and a classifier was applied to exclude examples where the model's output did not follow the instruction. Fine-tuning on this self-generated data produced a model that performed comparably to InstructGPT on human preference evaluations, despite using no human-annotated instruction data beyond the initial 175 seed examples [24]. This result reinforces the conclusion from Wei et al.: what matters is that training examples are structured as instructions, not who produced them or what specific tasks they cover. The format itself carries information that the model learns to generalise from.

Ballout et al. [1] investigated the role of explanations in fine-tuning language models directly, finding that adding intermediate reasoning steps to training examples substantially improved performance on smaller models, and enabled models to solve tasks they could not solve from answers alone. Their setup evaluates models with the explanation present at inference time, which is why the benefit materialises. This is the closest prior work to the training condition tested in this thesis, and the contrast is informative: their results confirm that explanation-augmented fine-tuning works when the inference format matches the training format, which is exactly the condition this thesis cannot satisfy in a realistic GEC deployment setting.

4 Methodology

This section describes how the worked examples principle was translated into a concrete experimental setup. The central design choice is the addition of rule-based grammatical explanations to GEC training examples, forming the Worked Examples condition. We describe how these explanations are generated, what dataset and evaluation metrics are used, how the four training conditions differ, and how the results are evaluated and compared.

4.1 Operationalising Worked Examples

The worked examples principle holds that studying a fully solved problem, including all solution steps, leads to better learning than solving equivalent problems independently, particularly for novice learners on tasks with well-defined correct answers [23, 22, 2]. To test whether this principle transfers to language model fine-tuning, it must be translated into a concrete training data format. The challenge is generating explanations automatically, without a separate teacher model or human annotation, using only what is already available in the training data.

The BEA-2019 W&I+LOCNESS dataset provides this automatically. As described in Section 2, every sentence pair in the dataset comes pre-annotated with structured ERRANT edits, each specifying the position of the error, the replacement text, and the grammatical error category. These annotations make it possible to generate a rule-based explanation for every training example

without any additional labelling effort.

Explanations are generated using a template system that operates on the ERRANT edit annotations for each sentence pair. Up to two representative edits are selected per example, skipping uninformative ones such as edits that only change whitespace or punctuation. For each selected edit, a description is produced based on the edit type. An insertion produces a sentence of the form *"The correction adds 'X' to fix [error type]."*. A deletion produces *"The correction removes 'X' because it is unnecessary here."*. A replacement produces *"The correction changes 'X' to 'Y' to fix [error type]."*. When multiple edits are described, their sentences are concatenated. A closing sentence is then appended based on the highest-priority error type found among the selected edits, following a fixed precedence order: verb tense errors take priority over subject-verb agreement, which takes priority over verb form, preposition, and determiner errors, with a generic fallback for other categories.

For the example pair where *"It's difficult answer at the question"* is corrected to *"It's difficult to answer the question"*, the ERRANT annotations produce two edits: an insertion of *to* with error type `M:VERB:FORM`, and a deletion of *at* with error type `U:PREP`. The generated explanation is: *"The correction adds 'to' to fix verb form. The correction removes 'at' because it is unnecessary here. This gives the verb the form that fits the surrounding structure."*

This explanation is inserted between the erroneous sentence and the correction in the training string, producing the three-part format shown in Table 2. The end-of-sequence token (`<|endoftext|>`, token ID 0) is appended to every training string to teach the model when to stop generating. During training, all tokens up to and including the `Correct:` marker receive label -100 and are excluded from the loss. Only the correction tokens and the end-of-sequence token contribute to the gradient. The explanation is present in the input context when the model predicts the correction tokens, but contributes no gradient signal of its own.

4.2 Dataset

All experiments use the BEA-2019 W&I+LOCNESS dataset [6], a learner English corpus combining writing from the Cambridge Write and Improve platform with the LOCNESS corpus. Essays were produced by English language learners across proficiency levels A, B, and C on the CEFR scale and corrected by trained human annotators. The resulting sentence pairs were processed through ERRANT [7] as part of the original dataset release, so each pair already contains structured edit annotations specifying the position, replacement text, and grammatical error category of every correction.

Sentence pairs where the original and corrected sentence are identical are removed before training, since they carry no correction signal. After this step the data is split into three sets. A test set of 2,819 pairs was sampled from the remaining training data before any model training took place,

ensuring no training condition ever saw these examples during fine-tuning. The remaining 19,892 pairs form the training set, which is identical across both fine-tuned conditions: the Fine-tuned condition uses the correction pairs directly, while the Worked Examples condition uses the same pairs with the rule-based explanation field added. The development set consists of 2,819 pairs from the original BEA-2019 development split and is used to evaluate performance after each training epoch. The BEA-2019 blind test set, which requires submission to an external evaluation server, is not used in this study.

Table 1 summarises the dataset splits.

Table 1: Dataset splits used in all experiments.

Split	Pairs	Source
Training	19,892	BEA-2019 training data (after test sampling)
Development	2,819	BEA-2019 development split
Test	2,819	Sampled from BEA-2019 training data

4.3 Evaluation Metrics

Three metrics are used throughout: ERRANT $F_{0.5}$ for measuring GEC performance, cross-entropy loss for monitoring training behaviour, and a paired t-test for comparing conditions statistically.

4.3.1 ERRANT $F_{0.5}$

ERRANT computes GEC performance by comparing predicted edits against gold edits at the token level. For each sentence, it extracts the set of edits from the original to the prediction, and the set of edits from the original to the gold correction. These edit sets are compared to compute three counts:

$$TP = |\text{predicted edits} \cap \text{gold edits}| \tag{1}$$

$$FP = |\text{predicted edits} \setminus \text{gold edits}| \tag{2}$$

$$FN = |\text{gold edits} \setminus \text{predicted edits}| \tag{3}$$

From these, precision and recall are computed:

$$P = \frac{TP}{TP + FP} \tag{4}$$

$$R = \frac{TP}{TP + FN} \tag{5}$$

Precision measures what fraction of proposed corrections were actually correct. Recall measures what fraction of real errors were corrected. These are combined into $F_{0.5}$:

$$F_{0.5} = \frac{1.25 \cdot P \cdot R}{0.25 \cdot P + R} \tag{6}$$

$F_{0.5}$ gives twice as much weight to precision as to recall. The reasoning behind this weighting is specific to GEC: an incorrect correction can damage an otherwise acceptable sentence, which is generally more harmful than leaving an error uncorrected. A system that only corrects what it is confident about is therefore preferred. This asymmetry has a direct consequence for interpreting results: a model that makes almost no corrections will have high precision but very low recall, and will score poorly on $F_{0.5}$ because low recall cannot be compensated by high precision in this weighting scheme.

4.3.2 Cross-Entropy Loss

The cross-entropy loss measures how confidently the model predicts the correct correction tokens. For each correction token t_i, the model assigns a probability $P(t_i \mid t_1, \ldots, t_{i-1})$ given the preceding tokens. The loss averages the negative log-probabilities over all correction tokens:

$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^{N} \log P(t_i \mid t_1, \ldots, t_{i-1}) \tag{7}$$

where N is the number of correction tokens. A lower loss means the model assigns higher probability to the correct correction tokens on average. To give a sense of scale: a loss around 0.9 reflects near-random prediction, where the model has essentially no idea what correction token should come next. A loss below 0.2 means the model is highly confident about what token follows, given the context it has seen. Loss values are on a natural logarithm scale, so they are not percentages and should not be read as fractions of errors corrected.

Loss is not the same as $F_{0.5}$. For the Worked Examples condition, loss is measured with the explanation present in the input context, since the development set is evaluated using the same format as training. The explanation is not part of what the model predicts, it is part of what the model is given, but its presence as context means the prediction task is easier than it would be without it. $F_{0.5}$, by contrast, is always measured using the plain inference format with no

explanation, making it the fair cross-condition comparison metric. These two metrics can therefore point in different directions when training and inference formats differ.

4.3.3 Statistical Testing

To assess whether differences in $F_{0.5}$ between conditions are reliable rather than due to random seed variation, a paired t-test is used. Each condition was run across ten random seeds (41–50), producing ten $F_{0.5}$ scores per condition. The paired t-test compares the mean $F_{0.5}$ of two conditions across these ten paired observations, with degrees of freedom $df = 9$. The test asks: given the differences we observe between conditions across seeds, how likely is it that those differences are just due to chance? A p-value below 0.05 is the standard threshold for concluding that the difference is reliable rather than accidental.. Results are reported as $t(9)$ and p-values. All t-test calculations were performed independently in both Python and Excel and verified to produce identical results to six decimal places.

4.4 Training Conditions

Three experimental conditions are compared. The Baseline is the pretrained Pythia-410M model evaluated without any fine-tuning, serving as a reference point for near-zero GEC performance. In other words, it shows what the model does when it has never been taught the correction task at all. The Fine-tuned condition fine-tunes the model on plain correction pairs. The Worked Examples condition fine-tunes the model on correction pairs augmented with rule-based grammatical explanations, as described in Section 4.1. The Worked Examples condition is evaluated in two ways at inference time: once without the explanation present, and once with the explanation generated from the gold annotations in the test set. These two evaluations use the same trained checkpoint and differ only in the inference prompt, directly isolating the effect of the explanation at inference time.

Table 2 shows the training and inference formats for all conditions. The Baseline condition involves no training and no format; it uses the same inference prompt as the Fine-tuned condition.

All fine-tuning experiments use Pythia-410M [4], a 410 million parameter decoder-only autoregressive language model pretrained on The Pile [10]. The model and its tokeniser are loaded from the pretrained weights without modification before each training run.

4.4.1 Training Pipeline

Each training example is formatted as a single string according to the condition format shown in Table 2. The end-of-sequence token `<|endoftext|>` (token ID = 0) is appended to every training string, teaching the model to stop generating at the end of a correction. The string is then tokenised

Table 2: Training and inference formats for all conditions. **S** is the erroneous sentence, **C** is the gold correction, **E** is the rule-based explanation, and <EOS> denotes the end-of-sequence token (<|endoftext|>, ID 0). The Worked Examples and Worked Examples with Explanation conditions share the same trained checkpoint.

Condition	Training format	Inference format
Baseline	—	`Fix: S\nCorrect:`
Fine-tuned	`Fix: S\nCorrect: C<EOS>`	`Fix: S\nCorrect:`
Worked Examples	`Fix: S\nExplanation: E\nCorrect: C<EOS>`	`Fix: S\nCorrect:`
Worked Examples + Explanation	`Fix: S\nExplanation: E\nCorrect: C<EOS>`	`Fix: S\nExplanation: E\nCorrect:`

into a sequence of integer IDs using the NeoX tokeniser. No special tokens are prepended; the end-of-sequence token is the only special token added, and it is added manually by the training code.

Label masking is applied to ensure the model only learns to predict the correction. All token positions from the beginning of the string up to and including the `Correct:` marker are assigned label -100, which tells the optimiser to skip those positions in the loss calculation. Only the correction tokens and the end-of-sequence token receive their actual token IDs as labels and contribute to gradient updates. This means the model is only trained to produce the correction itself. It never receives any learning signal for generating the prompt, the Fix: marker, or the explanation field. Those parts are there as context for the model to read, not as targets for it to learn to produce. For the Worked Examples condition the masked prefix is longer, extending through the `Explanation:` field as well, so the model is never trained to generate the explanation. It is only trained to predict the correction given the explanation as context.

Sequences are padded dynamically within each batch using a dedicated padding token <|padding|> (token ID = 1), which is distinct from the end-of-sequence token. Padding positions receive an attention mask value of zero so the model ignores them, and label -100 so they contribute nothing to the loss.

Training uses the hyperparameters that are mentioned in Table 3. These are kept identical across the Fine-tuned and Worked Examples conditions. Gradient accumulation means that instead of updating the model weights after every batch of 4 examples, the gradients are accumulated over 8 consecutive batches before an update is applied. This makes training behave as if 32 examples were processed at once, which leads to more stable weight updates, while keeping the actual memory used per step low enough to fit on a single consumer GPU.

Table 3: Training hyperparameters, identical across all fine-tuned conditions.

Setting	Value
Optimiser	AdamW
Learning rate	2×10^{-5}
Weight decay	0.01
Warmup steps	100
Batch size	4
Gradient accumulation	8 steps (effective batch size 32)
Precision	bfloat16
Epochs	10
Max sequence length	663 tokens
Decoding	Greedy (`do_sample=False`)
Max new tokens	100

Each condition is run across ten random seeds (41–50). For each seed, the pretrained weights are reloaded from scratch, ensuring fully independent runs. The checkpoint with the lowest development loss across all epochs is selected and saved for evaluation. The Baseline condition involves no training and is evaluated once on the development and test sets using the pretrained weights directly.

The Worked Examples condition is deliberately evaluated at inference time without the explanation. The inference prompt matches the Fine-tuned condition exactly (`Fix: S\nCorrect:`), even though the model was always trained with the explanation present. This creates an intentional mismatch between the training format and the inference format, reflecting the realistic deployment setting where gold annotations are not available for new sentences. A separate evaluation of the same checkpoint with the explanation present (`Worked Examples + Explanation`) is also performed to determine whether the model learned the correction task when the expected context is provided.

Before running the full ten-seed experiments, all pipeline components were verified using a lightweight test run on a small data subset, confirming correct label masking, padding behaviour, and loss computation.

4.5 Evaluation Procedure

After each training epoch, the development set is evaluated using the plain inference format: `Fix: S\nCorrect:` for all conditions, including Worked Examples. Prompts within each batch are left-padded to the same length so that all sequences end at the same position, which is necessary because autoregressive models generate from the rightmost token position. The model generates up to 100 new tokens using greedy decoding, and the first line of the decoded output is taken as the predicted correction.

ERRANT is then applied to all development pairs, comparing the edits between the original and predicted sentence against the edits between the original and gold correction. True positives, false positives, and false negatives are summed across all 2,819 pairs and used to compute corpus-level precision, recall, and $F_{0.5}$. The checkpoint with the lowest development loss across all ten epochs is selected for final evaluation.

Final evaluation is performed on the sampled test set of 2,819 pairs. Three evaluations are run on this set. The Baseline, Fine-tuned, and Worked Examples conditions are each evaluated using the plain inference format. Additionally, the Worked Examples condition is evaluated a second time with the explanation present in the inference prompt, using gold annotations from the test set to generate the explanation at test time. This produces four sets of test results: Baseline, Fine-tuned, Worked Examples, and Worked Examples with Explanation.

General linguistic competence is assessed using WinoGrande [21], a commonsense reasoning benchmark in which the model is given a sentence with a blank and two candidate words, and must select the more plausible option. For example, given the sentence *"The trophy didn't fit in the suitcase because it was too big"* with candidates *trophy* and *suitcase*, the correct answer is *trophy*. A model with no understanding would score 50% by guessing randomly between the two options. Accuracy is determined by comparing the log-likelihood the model assigns to each candidate completion: the candidate receiving the higher log-likelihood is selected. WinoGrande is evaluated on the pretrained Baseline, the best Fine-tuned checkpoint from seed 41, and the best Worked Examples checkpoint from seed 41. Since only a single seed is used for WinoGrande evaluation, no statistical comparison across seeds is possible.

All $F_{0.5}$ scores are reported as mean and standard deviation across the ten seeds. Pairwise differences between conditions are assessed using a paired t-test with $df = 9$, as described in Section 4.3.

5 Results

This section presents results for all four evaluated conditions: the Baseline (untrained Pythia-410M), the Fine-tuned condition, the Worked Examples condition evaluated without the explanation at inference time, and the Worked Examples condition evaluated with the explanation present. For each condition, the $F_{0.5}$ scores on both the development and test sets are reported, alongside development loss, precision, and recall. Statistical comparisons between conditions use the paired t-test described in Section 4.3. WinoGrande results are reported at the end of the section to assess whether fine-tuning affected performance on a general commonsense language benchmark.

5.1 Baseline: Untrained Pythia-410M

Tables 4 and 5 show the full results across all four conditions on both the development and test sets. The Baseline establishes the performance floor, while the three fine-tuned conditions show what different training formats produce from the same pretrained model.

Table 4: Development set results across all conditions. $F_{0.5}$, precision, and recall are reported as percentages. Values for the fine-tuned conditions are mean ± standard deviation across ten seeds (41–50). The Baseline is a single deterministic run. Development loss is reported on its raw scale. The Worked Examples and WE + Explanation conditions share the same checkpoint and therefore have identical development loss.

Condition	P (dev)	R (dev)	$F_{0.5}$ (dev)	Avg. dev loss
Baseline	0.98	0.96	0.98	0.945
Fine-tuned	33.27 ± 0.54	22.74 ± 0.46	30.44 ± 0.37	0.396 ± 0.003
Worked Examples	33.67 ± 1.98	4.14 ± 1.00	13.72 ± 2.24	0.168 ± 0.002
WE + Explanation	63.96 ± 0.94	55.23 ± 0.43	62.00 ± 0.75	0.168 ± 0.002

Table 5: Test set results across all conditions. $F_{0.5}$, precision, and recall are reported as percentages. Values for the fine-tuned conditions are mean ± standard deviation across ten seeds (41–50). The Baseline is a single deterministic run.

Condition	P (test)	R (test)	$F_{0.5}$ (test)
Baseline	1.04	0.86	1.00
Fine-tuned	34.68 ± 0.60	22.51 ± 0.51	31.29 ± 0.45
Worked Examples	31.90 ± 2.08	3.36 ± 0.71	11.73 ± 1.76
WE + Explanation	66.35 ± 1.25	56.74 ± 0.67	64.17 ± 1.04

Before any fine-tuning, Pythia-410M achieves an $F_{0.5}$ of 1.0 on both the development and test sets, with a development loss of 0.945. An $F_{0.5}$ of 1.0 is essentially zero performance. This means the model is making almost no correct corrections at all. The development loss of 0.945 confirms this: the model has no idea what correction token should come next, and is predicting near-randomly. The model has never seen the `Fix/Correct` prompt format and has no understanding of the GEC task, making these near-zero scores expected.

The actual outputs reveal three distinct failure patterns, each traceable to the composition of The Pile [10], the pretraining corpus Pythia was trained on. The Pile spans 22 diverse text sources including academic papers, web text, books, code, legal documents, and question-answering forums. This diversity means the model has learned many text continuation patterns, but none of them correspond to grammar correction.

The first pattern is reinterpretation of the prompt as a general question. The Pile includes 15.6 million Stack Exchange documents in question-and-answer format [10], and the model appears to

recognise the GEC prompt as a question requiring a Stack Exchange-style answer. For Dev 01, given the input *"It's difficult answer at the question 'what are you going to do in the future?'"*, the model outputs:

> *"It is not a question of what you will do in the future, but what you will do in the present. A: I think the answer is 'I don't know'. I don't know what I'm going to do in the future..."*

The *"A:"* prefix is characteristic of Stack Exchange answer formatting. The model has parsed the GEC prompt as a question about the future and responded from pretraining knowledge rather than attempting any correction.

The second pattern is looping of the prompt structure. When the model does not recognise the input as a question, it continues the structured format it sees in the prompt. For Dev 05, given *"I think that the public transport will always be in the future."*, the model loops the `Fix/Correct` structure indefinitely:

> *"I think that the public transport will always be in the future. \nCorrect: I think that the public transport will always be in the future. \nCorrect: ..."*

This behaviour reflects the model treating `Correct:` as a structural separator it has seen in some form during pretraining, without any understanding of what should follow it.

The third pattern is generation of entirely unrelated content. For Dev 09, given the input *"In my experience when I didn't have a car I used to use the bus to go to the school and go back to my house."*, the model outputs:

> *"I have a car and I use it to go to the school and go back to my house. I have a car and I use it to go to the school and go back to my house..."*

The model has inverted the meaning of the sentence, generating the opposite of what was described in the input, and then loops that generated phrase indefinitely. Rather than engaging with the error correction task, it has followed a semantic association from its pretraining data, producing fluent but entirely incorrect output.

All three patterns confirm that the Baseline has zero GEC capability. The $F_{0.5}$ of 1.0 establishes the performance floor: any score above this is a direct consequence of fine-tuning.

5.2 Fine-tuned Results

Fine-tuning on plain correction pairs produces a substantial improvement over the Baseline. The Fine-tuned condition achieves a mean $F_{0.5}$ of 30.44 on the development set and 31.29 on the test set

(Table 4 and Table 5). To put these numbers in context, top systems at the BEA-2019 shared task achieved $F_{0.5}$ scores in the range of 66–69 [6], and GECToR reaches 72.4 using a large pretrained transformer with nine million synthetic training sentences [18]. The Fine-tuned condition uses none of that: a 410 million parameter model trained directly on 19,892 sentence pairs without any data augmentation. A score of 31.29 on the test set reflects a model that has learned to make genuine grammatical corrections, but operates in a specific performance regime: it identifies and corrects errors that are local and structurally unambiguous, while systematically avoiding anything that requires multi-word restructuring or substantive lexical substitution. The output examples make this visible. The model correctly inserts a comma, drops an incorrect article, and removes a preposition error across different development sentences, but returns a sentence unchanged when the required correction involves significant rephrasing, and leaves *"study it easier"* uncorrected when the gold standard requires *"study it more easily"*. A score in the low thirties is precisely what that operational profile looks like numerically.

Development loss drops from 0.945 to 0.396, meaning the model is now predicting correction tokens with roughly twice the confidence it had before training. This does not mean it makes no mistakes, but it is no longer guessing nearly at random.. Figure 1 shows the training and development loss per epoch for seed 41. Training loss decreases steadily across all ten epochs, indicating the model continues to fit the training data throughout. Development loss, however, stabilises around 0.36 after the fourth epoch and does not improve further, implying the model has learned as much as it can about the correction task from this training setup. More training time would not help it generalise better to new sentences. This divergence between training and development loss suggests the model has reached the limit of what it can generalise to unseen sentences given this training setup. Running for more epochs would reduce training loss further but would not improve GEC performance.

The precision and recall breakdown clarifies how the model applies its corrections. Precision (33.27 on dev, 34.68 on test) is notably higher than recall (22.74 on dev, 22.51 on test). This means the model makes roughly two correct edits for every three it attempts, but catches only around one in four or five of the real errors present in the gold standard. In practice, the model is conservative: it only intervenes when it is confident, and leaves many errors untouched. This behaviour is consistent with training toward $F_{0.5}$, which weights precision twice as heavily as recall. A model that learns to avoid uncertain corrections and makes no edit when in doubt will be penalised less by $F_{0.5}$ than by a metric that weighs recall equally. The result is a system that behaves like a cautious editor: reliable when it acts, but selective about when it acts at all.

The actual outputs confirm the model is functioning as a genuine GEC system. Several types of real corrections appear across the development examples. For Dev 02, given *"When I was younger I used to say..."*, the model correctly inserts a comma after *younger* and removes the double period,

producing output that matches the gold reference exactly. For Dev 05, given *"I think that the public transport will always be in the future."*, the model correctly drops the article:

> *"I think that public transport will always be in the future."*

For Dev 08, the model correctly fixes a preposition error, changing *"ask to you something"* to *"ask you something"*. For Dev 10, it correctly changes *"realized"* to *"realize"* and removes the unnecessary article before *public transport*. These are genuine corrections across different error types (punctuation, determiners, prepositions, and verb tense) showing that the training format has successfully taught the model to identify and fix a range of grammatical errors.

The model is not without errors. For Dev 03, given a sentence requiring substantial restructuring, the model returns the sentence unchanged. For Dev 04, it leaves *"study it easier"* uncorrected despite the gold requiring *"study it more easily"*. These cases involve significant lexical substitution or multi-word restructuring that the model consistently avoids, which explains the relatively low recall. For Dev 06, the model makes an incorrect correction, removing *will* from *"The rich people will buy a car"*, introducing an error that was not present in the input. This illustrates that while precision is higher than recall, the model is not perfectly precise either.

The low standard deviation across seeds (± 0.37 on dev $F_{0.5}$, ± 0.45 on test $F_{0.5}$) shows that these characteristics are stable across different random seeds. The score the model achieves, the balance between precision and recall, and the types of errors it makes and misses are consistent properties of the training setup rather than artefacts of a particular initialisation.

The full raw outputs for all ten development examples are provided in Appendix C. Loss curves for all ten seeds are provided in Appendix A.

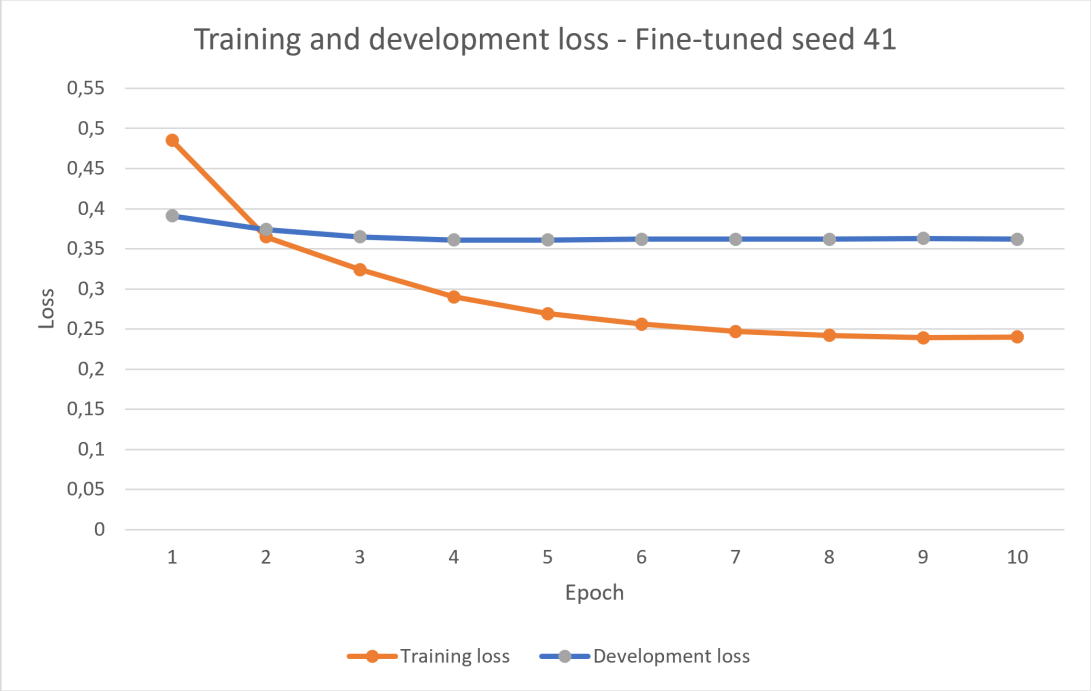

Figure 1: Training and development loss per epoch for the Fine-tuned condition, seed 41. Training loss decreases steadily across all ten epochs. Development loss decreases gradually over the first four epochs and then stabilises around 0.36, while training loss continues to decline. The widening gap between the two curves suggests the model reaches its ceiling on the development set early in training.

5.3 Worked Examples Results

The Worked Examples condition, evaluated without the explanation at inference time, achieves a mean $F_{0.5}$ of 13.72 on the development set and 11.73 on the test set (Tables 4 and 5). To understand what these numbers mean, it helps to look at precision and recall separately.

Precision is 33.67 on the development set and 31.90 on the test set. These values are close to the Fine-tuned condition's 33.27 and 34.68. This means that when the Worked Examples model does make a correction, it is right at roughly the same rate as the Fine-tuned model. The corrections it chooses to make are not degraded in quality.

Recall tells a completely different story. It is 4.14 on the development set and 3.36 on the test set, compared to 22.74 and 22.51 for the Fine-tuned condition. A recall of around 3–4 means the model corrects only about one in twenty-five real errors. For the vast majority of erroneous sentences, it makes no attempt at all. This is what drives the low $F_{0.5}$: the metric cannot be high when recall is this close to zero, even if precision is acceptable.

What makes this precision-recall dissociation diagnostically useful is what it reveals about the nature of the learning failure. The model has not lost the ability to produce corrections, it has lost

the ability to decide when to produce them. The training format paired every erroneous sentence with an explanation that described exactly what needed changing and why, and the correction followed from that explanation. During training, the decision to intervene was never left to the model: the presence of the explanation settled it. At inference time, when only the erroneous sentence is present, the model has no equivalent signal telling it that a correction is warranted. The result is a model that has learned the form of corrections well enough to execute them with competitive precision on the rare occasions it acts, but has no reliable internal trigger for when to act at all.

The outputs confirm this directly. Of the ten development examples, nine are returned completely unchanged. For a learner essay sentence containing two clear grammatical errors, *"It's difficult answer at the question 'what are you going to do in the future?'"*, the model produces the same sentence back word for word. The same is true for sentences with missing commas, unnecessary articles, and incorrect verb forms, all errors the Fine-tuned condition handles correctly. The only exception across the ten examples is a sentence where the model removes a preposition error, changing *"ask to you"* to *"ask you"*, a correct but partial correction.

The full raw outputs for all ten development examples are provided in Appendix D. Loss curves for all ten seeds are provided in Appendix B.

The development loss of 0.168 appears to contradict this picture, since it is less than half the Fine-tuned condition's 0.396. Lower loss normally indicates better learning. Here, however, the loss is measured on the development set using the training format, which includes the explanation in the input context. The model is highly confident at predicting correction tokens when given an explanation telling it exactly what to change and where. This confidence is entirely conditional on the explanation being present. At inference time, where only the erroneous sentence is given, that confidence has nothing to attach to, and the model defaults to copying the input. The loss measures performance on the training prompt format, not on the inference format, which is why it points in the opposite direction from $F_{0.5}$.

Figure 2 shows the training and development loss per epoch for seed 41. Training loss decreases steadily across all ten epochs, reaching 0.090 by epoch 10. This is a very low value, indicating the model has nearly memorised the training format and is assigning very high confidence to the expected correction tokens when the explanation is present in the input. Development loss reaches its minimum at epoch 4 (0.177) and then rises slightly to stabilise around 0.182, while training loss keeps decreasing. This small divergence after epoch 4 indicates the model is fitting more tightly to the training format in later epochs without generalising further to unseen sentences. The best checkpoint is therefore selected from epoch 4.

The high standard deviation across seeds (± 2.24 on development $F_{0.5}$, ± 1.76 on test $F_{0.5}$) is notably larger than the Fine-tuned condition's ± 0.37 and ± 0.45. The explanation lies in the recall

collapse: when a model corrects fewer than one sentence in ten, small differences between seeds in which sentences they happen to act on produce proportionally larger swings in $F_{0.5}$ than the same variation would in a model that intervenes regularly. The core pattern, however, holds across all ten seeds: near-zero recall and substantially lower $F_{0.5}$ than the Fine-tuned condition.

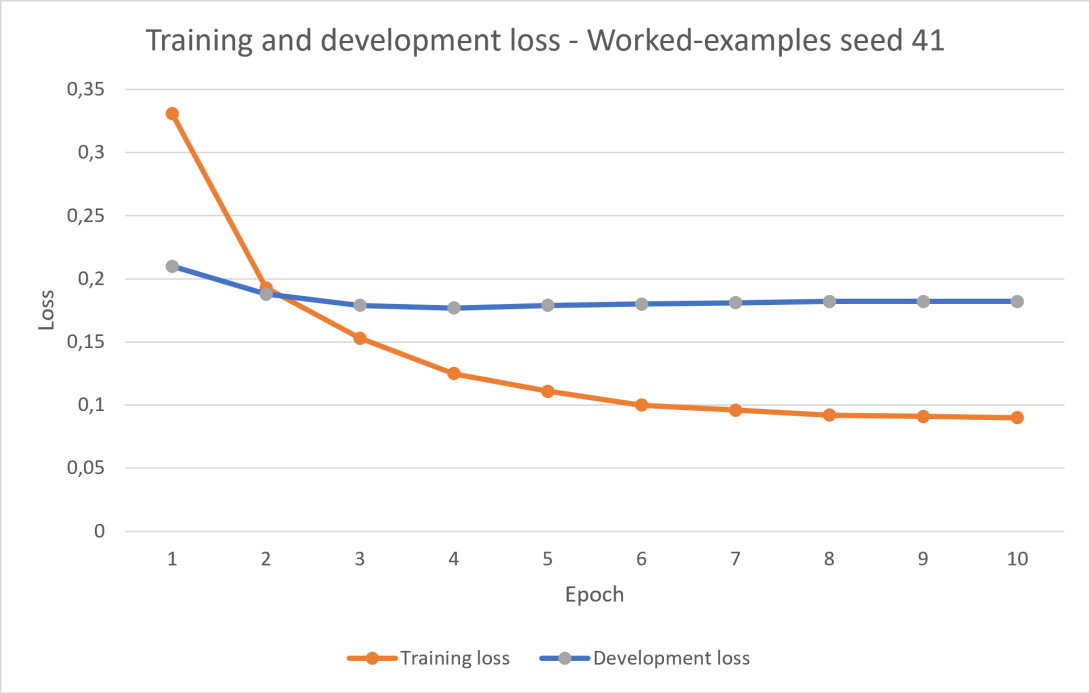

Figure 2: Training and development loss per epoch for the Worked Examples condition, seed 41. Training loss decreases steadily across all ten epochs. Development loss reaches its minimum at epoch 4 and then rises slightly, suggesting the model begins to overfit to the training format. The best checkpoint is selected from epoch 4.

5.4 Statistical Comparison

To formally compare conditions, a paired t-test is applied to the $F_{0.5}$ scores across the ten seeds. Each seed produces one $F_{0.5}$ score per condition, giving ten paired observations. The paired design accounts for seed-level variation by comparing each condition within the same seed rather than across different seeds, making the test sensitive to the true difference between training formats.

The comparison between the Fine-tuned condition (mean 31.29 on test) and the Worked Examples without explanation (mean 11.73 on test) gives $t(9) = 33.96$, $p = 8.21 \times 10^{-11}$. The t-statistic of 33.96 means the difference in $F_{0.5}$ scores between the Fine-tuned and Worked Examples conditions is nearly 34 standard errors wide. To put that simply, across all ten random seeds the Fine-tuned condition scored higher than Worked Examples every single time, by a consistent margin, making it essentially impossible to attribute the difference to luck. A value this large

leaves essentially no room for the result to be explained by chance variation across seeds. The p-value of 8.21×10^{-11} expresses the same thing numerically: if the two training formats produced identical underlying performance, under the null hypothesis of no difference, observing a gap this large across ten seeds is vanishingly unlikely.

The comparison between the Fine-tuned condition and the Worked Examples with explanation (mean 64.17 on test) gives $t(9) = -90.62$, $p = 1.23 \times 10^{-14}$. The negative sign reflects the direction of the difference: the Worked Examples with explanation condition scores higher than the Fine-tuned condition. The t-statistic of 90.62 is the largest effect observed in the experiment, and it reflects a result that goes beyond a statistical finding, it describes a model that functions qualitatively differently when the explanation is present. Precision rises from 34.68 to 66.35, and recall rises from 22.51 to 56.74. The recall shift is the more telling of the two: it moves from one error caught in roughly four or five to more than one in two. A model that previously declined to act on the majority of erroneous sentences now intervenes on the majority of them, and does so with substantially higher accuracy. This is not the same model performing marginally better, it is the same checkpoint operating in an entirely different regime, distinguished only by whether the explanation is present in the input at inference time.

Together these two tests establish that both differences are reliable and not an artefact of which random seeds were chosen. The Worked Examples model without the explanation is reliably worse than the Fine-tuned model, and the same model with the explanation is reliably far better. The contrast between the two Worked Examples variants, the same trained checkpoint differing only in inference format, is what makes the finding legible: the model learned the correction task effectively, but that learning is bound to the prompt format used during training and cannot be accessed without it.

The full raw outputs for the Worked Examples condition evaluated with the explanation present are provided in Appendix E.

5.5 WinoGrande Results

WinoGrande is used to assess whether fine-tuning on GEC data changed the model's performance on a general commonsense language benchmark, or whether the improvements in $F_{0.5}$ reflect purely task-specific learning. Table 6 shows the results for all three evaluated models. Because WinoGrande is evaluated on a single seed (seed 41), no statistical comparison across seeds is possible, and the standard error reflects uncertainty within the benchmark itself rather than across training runs.

All three models score near the 50% random chance baseline. The untrained model achieves 53.59%, just above chance. After fine-tuning on GEC data, the Fine-tuned condition scores 52.72% and the Worked Examples condition scores 52.88%. The differences between all three are within the measurement standard error of ±1.4%. This means fine-tuning on GEC data produced no

Table 6: WinoGrande accuracy for the untrained Baseline, the Fine-tuned condition (seed 41), and the Worked Examples condition (seed 41). All models are evaluated in zero-shot mode using log-likelihood selection. Standard error is the benchmark-level measurement uncertainty.

Model	Accuracy	SE
Baseline (untrained)	53.59%	±1.4%
Fine-tuned (seed 41)	52.72%	±1.4%
Worked Examples (seed 41)	52.88%	±1.4%

measurable improvement in general commonsense reasoning ability, and if anything produced a very slight decline that falls within noise.

This result matters for interpreting the $F_{0.5}$ findings. The substantial improvement in GEC performance from fine-tuning is not accompanied by any improvement in performance on a general commonsense language benchmark. The model has learned something specific to the correction task and the prompt format, not broader language understanding. This is consistent with the expectation for a 410 million parameter model fine-tuned on a small task-specific dataset: it adapts to the task without acquiring new general capabilities.

6 Discussion

The main finding of this experiment is that the Worked Examples model acquired correction behaviour that is substantially more accurate when the explanation is present at inference time, but cannot demonstrate that learning under standard inference conditions where the explanation is absent. This section interprets the findings in relation to both research questions, places the results in the context of prior GEC work, and draws practical conclusions for training data design.

6.1 Does the Worked Examples Effect Transfer?

The first research question asked whether training on worked examples improves GEC performance in Pythia-410M compared to training on plain correction pairs. The hypothesis was that the worked examples format would benefit the model as a novice learner on a well-defined task, following the prediction of the worked examples literature [23, 2]. The result is negative when evaluated in the standard setting: the Worked Examples condition without explanation achieves a mean test $F_{0.5}$ of 11.73, compared to 31.29 for the Fine-tuned condition, a difference of $t(9) = 33.96$, $p = 8.21 \times 10^{-11}$. This implies that the model trained with explanations performed roughly three times worse than the model trained without them, and this gap was completely consistent across all ten random seeds. The hypothesis is not supported under standard inference conditions.

The second research question asked whether any benefit depends on the explanation being present at inference time. The answer is yes. The same checkpoint evaluated with the explanation achieves a mean test $F_{0.5}$ of 64.17, compared to 31.29 for the Fine-tuned condition, a difference of $t(9) = -90.62$, $p = 1.23 \times 10^{-14}$. The model trained on worked examples substantially outperforms the model trained on plain correction pairs, but only when the inference prompt matches the training format. This result is important precisely because it separates two questions that the standard evaluation conflates: whether the model failed to learn, and whether the model can demonstrate that learning under the inference conditions used. The WE + Explanation result answers the first question directly. The model did learn from the worked examples format. What it cannot do is access that learning without the explanation present.

This does not mean the worked examples principle cannot transfer to language model training. It means this specific implementation creates a condition that prevents the benefit from materialising in the standard evaluation setting. The analogy to human learning is instructive here, though it has limits. In human learning research, a worked example is a study-phase scaffold that learners eventually internalise and no longer need [22]. A language model is not a human learner, and the mechanisms of gradient-based optimisation do not map cleanly onto human memory consolidation or schema formation. What this experiment tests is not whether those mechanisms are equivalent, but whether a specific structural property of worked examples, providing the reasoning alongside the answer, produces a comparable directional effect. The WE + Explanation result suggests it does, conditionally. The model has acquired correction behaviour that is substantially more accurate when the expected context is present, which is consistent with the worked examples prediction. The difficulty is that this behaviour is bound to a prompt format that requires gold annotations not available for new sentences.

This connects to the human corrective feedback literature. Deng et al. [8] found that focused written corrective feedback, which provides explicit error codes telling the learner what type of error they made, led to significantly better accuracy gains than unfocused feedback that marked errors without categorisation. Kao et al. [14] found that even unfocused feedback produced modest gains on article errors, but weak and inconsistent effects on preposition and verb tense errors. These studies suggest that explicit information about the nature of an error plays an important role in what is learned. The worked examples format used here serves a similar function: it provides an explicit account of what is wrong and why the correction follows. The difference is that human learners in WCF studies are assessed on independent writing tasks, not on reformatting the same corrective signal. The evaluation challenge in this experiment is therefore not fully analogous to the human learning setting.

The comparison with Hsieh et al. [12] is also relevant, though the parallel is not direct. Hsieh et al. trained a student model on rationale-augmented examples produced by a larger teacher model,

and crucially evaluated the student with the rationale present at inference time. This is a different setup from what is tested here: their rationales are generated by a teacher model and serve a knowledge distillation function, whereas the explanations in this thesis are rule-based template outputs derived from ERRANT annotations. The more relevant comparison is at the level of the evaluation design. Their approach consistently provides the rationale at inference while this experiment deliberately withholds it to reflect a realistic deployment condition. The difference in results between WE and WE + Explanation in this thesis suggests that the presence or absence of explanatory context at inference time is the operative factor, not the quality or source of the explanation itself.

Ballout et al. [1] similarly found that explanation-augmented fine-tuning improved performance substantially, but their evaluation kept the explanation present at inference time, which is precisely the condition that makes the benefit accessible.

6.2 Comparison with Prior GEC Work

The Fine-tuned condition achieves a mean test $F_{0.5}$ of 31.29. To place this in context, the BEA-2019 shared task reported top system scores in the range of 64–70 [6], and GECToR reaches 72.4 on the same benchmark using a tagging-based transformer pretrained on nine million synthetic sentences [18]. Earlier, Junczys-Dowmunt and Grundkiewicz [13] achieved an M^2 $F_{0.5}$ of 46.37 on CoNLL-2014 using a phrase-based SMT system with large-scale parallel corpora and web-scale language models. All three represent substantially different setups from this experiment: larger architectures, orders of magnitude more training data, and extensive augmentation pipelines. The gap between 31.29 and these figures is therefore expected rather than surprising.

The more informative comparison is with Lamelas [16], whose setup is the closest to the one used here. Lamelas fine-tuned small decoder-only models (GPT-2, GPT-2 Medium, and GPT-Neo-125M) directly on the BEA-2019 W&I+LOCNESS training data without synthetic augmentation, using a prompt format structurally similar to the one in this thesis. Across all fine-tuned small models, the average M^2 $F_{0.5}$ was approximately 5, and fine-tuning produced almost no improvement over untrained baselines, suggesting those models failed to acquire meaningful correction behaviour from the training data [16]. The Fine-tuned condition here achieves 31.29, substantially above that range. The differences in training setup between this experiment and Lamelas, explicit label masking that restricts gradient updates to correction tokens only, ten training epochs rather than three, and a larger base model, are the most plausible explanations for why meaningful correction behaviour emerged here when it did not in the comparable prior work.

The WE + Explanation condition achieves 64.17, placing it within the range of top BEA-2019 shared task systems. This comparison is not straightforward: WE + Explanation uses gold annotations at inference time to generate the explanation, a condition that real deployment does

not provide. It does, however, establish that the trained checkpoint contains sufficient correction ability to approach competitive performance levels when the expected context is present. The gap between 11.73 (Worked Examples without explanation) and 64.17 (with explanation) from the same checkpoint is therefore not a gap in what the model learned, but a gap in what the evaluation conditions allow it to demonstrate.

The scores reported here are based on a held-out test set sampled from the BEA-2019 training data, so they reflect performance under the same training distribution rather than the official blind test setting. The comparisons to prior work are therefore intended as context for scale, not as direct benchmark competition.

6.3 The Low Loss Paradox

The Worked Examples condition achieves a development loss of 0.168, compared to 0.396 for the Fine-tuned condition. Under normal circumstances, lower loss on a held-out set indicates better generalisation. Here it indicates the opposite problem. Loss is measured using the training format, which includes the explanation in the input context. The model is highly confident at predicting correction tokens when an explanation tells it exactly what to change. That confidence collapses at inference time when no explanation is present. The loss metric and the $F_{0.5}$ metric are measuring performance in different conditions, which is why they point in opposite directions.

This points to a general principle in supervised fine-tuning: if the training format includes features that will not be present at inference time, development loss computed on the training format is not a reliable indicator of inference performance. $F_{0.5}$ evaluated under inference conditions is the metric that captures what the model can actually do when deployed. For future experiments involving augmented training formats, evaluating development performance in the inference format, not the training format, would give a more accurate picture of learning progress throughout training.

6.4 Implications for Training Data Design

The practical conclusion is specific. Extra structure in training examples only helps if that structure is available at inference time, or if the model is explicitly trained to operate without it. For GEC, adding explanations to training data in the format used here does not improve performance in the standard evaluation setting where explanations are absent at inference.

Two alternative approaches follow directly from this analysis and represent natural next steps. A self-explanation approach would train the model to first generate its own explanation before producing the correction, so that both training and inference use the same three-part format without requiring gold annotations. Training on both formats simultaneously, with and without the explanation, could teach the model to handle inference without the explanation field while still

benefiting from explanation-guided training examples. Both alternatives would test whether the correction behaviour acquired in the WE + Explanation condition can be made accessible under standard inference conditions, which is the key open question this experiment leaves unresolved.

6.5 Limitations

All experiments use a single model, Pythia-410M. Larger models might handle the training-inference mismatch differently, potentially by generalising more effectively from explanation-guided training to explanation-free inference. Whether the model size is the limiting factor cannot be determined from this experiment alone. The selection of Pythia-410M was motivated by practical constraints rather than a systematic comparison across scales.

The explanations used here are generated from rule-based templates and are formulaic in structure. More naturalistic explanations generated by a stronger language model might produce different training dynamics and could form a closer analogue to the kind of explanatory feedback studied in the WCF literature.

The WinoGrande results suggest that fine-tuning produced no measurable change in general linguistic competence. This is consistent with task-specific learning, but it also means that whether the Worked Examples format conveys any richer understanding of grammatical structure than plain correction pairs remains an open question. Probing the internal representations of the fine-tuned models, rather than only measuring their outputs under a specific prompt format, would give a more direct answer to what grammatical knowledge the model has actually acquired.

7 Conclusions

This thesis investigated whether the worked examples principle from educational psychology transfers to language model fine-tuning for Grammatical Error Correction. Two research questions were posed. The first asked how a training data format based on worked examples affects the GEC performance of a fine-tuned small language model. The second asked whether the direction of any effect aligns with what the worked examples literature predicts for novice learners on well-defined tasks.

The answer to the first research question is that the worked examples format, as implemented here, reduces GEC performance in the standard evaluation setting. The Worked Examples condition without explanation achieves a mean test $F_{0.5}$ of 11.73, compared to 31.29 for the Fine-tuned condition, a difference of $t(9) = 33.96$, $p = 8.21 \times 10^{-11}$. Under standard inference conditions, the hypothesis that worked examples would benefit the model is not supported.

The answer to the second research question is more nuanced. When the same Worked Examples

checkpoint is evaluated with the explanation present at inference time, it achieves a mean test $F_{0.5}$ of 64.17, compared to 31.29 for the Fine-tuned condition, a difference of $t(9) = -90.62$, $p = 1.23 \times 10^{-14}$. The model trained on worked examples substantially outperforms the model trained on plain correction pairs when the inference prompt matches the training format. This result is consistent with the directional prediction of the worked examples literature: providing the reasoning alongside the answer does produce a better-learned correction behaviour. The problem is not a failure to learn but a failure of the evaluation conditions to match the training format.

WinoGrande scores remain near the 50% chance baseline for all conditions, with differences falling within the benchmark's standard error of ±1.4%. Fine-tuning on GEC data produced task-specific learning without any measurable change in performance on a general commonsense language benchmark, which is consistent with the expectation for a small model fine-tuned on a narrow task-specific dataset.

Taken together, these results identify a specific boundary condition for training data design. Adding explanatory structure to training examples can produce more effective learning, but that learning is only accessible at inference time if the same structure is present, or if the model is explicitly trained to operate without it. In a realistic GEC deployment setting, gold annotations are not available for new sentences, so the explanation cannot be provided at inference. This limits the practical utility of the worked examples format as implemented here, while leaving open the question of whether modified implementations could recover the benefit under standard evaluation conditions.

7.1 Further Research

Several directions follow directly from the findings and limitations of this experiment. The most immediate is resolving the training-inference mismatch that this experiment deliberately introduced. A self-explanation approach, where the model is trained to generate its own explanation before producing the correction, would align training and inference formats without requiring gold annotations. Training on both formats simultaneously, with and without the explanation field, could also teach the model to handle inference without the explanation while still benefiting from explanation-guided training examples. Either approach would test whether the correction behaviour demonstrated in the WE + Explanation condition can be made accessible under standard deployment conditions.

A second direction concerns what the Worked Examples model has actually learned about grammatical structure, independently of the prompt format used to elicit corrections. The WinoGrande results suggest that general linguistic competence was not affected by fine-tuning, but they do not speak to whether the worked examples format instilled richer grammatical representations than plain correction training did. Probing the internal representations of both fine-tuned models, for

example through targeted linguistic acceptability tasks or structured grammatical probes, would give a more direct answer to whether the explanation format leaves any trace in the model beyond the correction behaviour observable at inference time.

A third direction is scale. All experiments here use Pythia-410M. Larger models may handle the training-inference mismatch differently, and the relationship between model scale and the ability to generalise from explanation-guided training to explanation-free inference is not known. Running the same experimental conditions on a range of model sizes would establish whether the findings here are specific to small models or reflect a more general property of the worked examples training format.

A fourth direction is the application of related pedagogical principles that were not tested in this experiment. Curriculum learning, which structures training examples from simple to complex [3], and retrieval practice, which shows that actively recalling information improves long-term retention [19, 20], are both candidates for translation into GEC training data formats. Whether either principle produces the same kind of format-dependent learning observed here, or whether they interact differently with the training-inference mismatch, is an open question. Comparing all three pedagogical principles under the same controlled setup would give a broader picture of how insights from educational psychology can and cannot transfer to language model fine-tuning.

References

[1] Mohamad Ballout, Ulf Krumnack, Gunther Heidemann, and Kai-Uwe Kühnberger. Show me how it's done: The role of explanations in fine-tuning language models. In *Proceedings of the 15th Asian Conference on Machine Learning*, volume 222 of *Proceedings of Machine Learning Research*, pages 90–105. PMLR, 2024.

[2] Christina A. Barbieri, Dana Miller-Cotto, Sarah N. Clerjuste, and Kamal Chawla. A meta-analysis of the worked examples effect on mathematics performance. *Educational Psychology Review*, 35(1), 2023.

[3] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML)*, pages 41–48, 2009.

[4] Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. Pythia: A suite for analyzing large language models across training and scaling. *arXiv preprint arXiv:2304.01373*, 2023.

[5] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, 2020.

[6] Christopher Bryant, Mariano Felice, Øistein E. Andersen, and Ted Briscoe. The BEA-2019 shared task on grammatical error correction. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 52–75, 2019.

[7] Christopher Bryant, Mariano Felice, and Ted Briscoe. Automatic annotation and evaluation of error types for grammatical error correction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, pages 793–805, 2017.

[8] Chengchen Deng, Xiao Wang, Siyu Lin, Wei Xuan, and Qi Xie. The effect of coded focused and unfocused corrective feedback on ESL student writing accuracy. *Journal of Language and Education*, 8(4):36–57, 2022.

[9] Richard Futrell and Kyle Mahowald. How linguistics learned to stop worrying and love the language models. *arXiv preprint arXiv:2501.17047*, 2025.

[10] Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The Pile: An 800GB dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2021.

[11] John Hattie. *Visible Learning: A Synthesis of Over 800 Meta-Analyses Relating to Achievement*. Routledge, 2009.

[12] Cheng-Yu Hsieh, Chen-Yu Li, Chih-Hung Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Chen, and Tomas Pfister. Distilling step-by-step! Outperforming larger language models with less training data and smaller model sizes. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8003–8017, 2023.

[13] Marcin Junczys-Dowmunt and Roman Grundkiewicz. Phrase-based machine translation is state-of-the-art for automatic grammatical error correction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*, pages 1546–1556, 2016.

[14] Mei-Ling Kao et al. The effectiveness of unfocused corrective feedback on second language student writers' acquisition of english article, prepositional and verb tense usages. *Humanities and Social Sciences Communications*, 2025.

[15] Enkelejda Kasneci, Kathrin Seßler, Stefan Küchemann, Maria Bannert, Daryna Dementieva, Frank Fischer, Urs Gasser, Georg Groh, Stephan Günnemann, Eyke Hüllermeier, et al. ChatGPT for good? On opportunities and challenges of large language models for education. *Learning and Individual Differences*, 103:102274, 2023.

[16] Anthony Lamelas. Evaluating small decoder-only language models for grammar correction and text simplification. *arXiv preprint arXiv:2601.03874*, 2026.

[17] Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. The CoNLL-2014 shared task on grammatical error correction. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–14, 2014.

[18] Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhanskyi. GECtoR – grammatical error correction: Tag, not rewrite. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 163–170, 2020.

[19] Henry L. Roediger and Jeffrey D. Karpicke. Test-enhanced learning: Taking memory tests improves long-term retention. *Psychological Science*, 17(3):249–255, 2006.

[20] Christopher A. Rowland. The effect of testing versus restudy on retention: A meta-analytic review of the testing effect. *Psychological Bulletin*, 140(6):1432–1463, 2014.

[21] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial winograd schema challenge at scale. In *Communications of the ACM*, volume 64, pages 99–106, 2021.

[22] John Sweller. Cognitive load during problem solving: Effects on learning. *Cognitive Science*, 12(2):257–285, 1988.

[23] John Sweller and Graham A. Cooper. The use of worked examples as a substitute for problem solving in learning algebra. *Cognition and Instruction*, 2(1):59–89, 1985.

[24] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL 2023)*, pages 13484–13508, 2023.

[25] Alex Warstadt and Samuel R. Bowman. What artificial neural networks can tell us about human language acquisition. In *Algebraic Structures in Natural Language*, pages 17–60. CRC Press, 2022.

[26] Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34, 2023.

[27] Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations (ICLR 2022)*, 2022.

A Per-Seed Loss Plots: Fine-tuned Condition

Figures 3–12 show training and development loss per epoch for each of the ten random seeds in the Fine-tuned condition. In all seeds, development loss stabilises after approximately four epochs while training loss continues to decrease, consistent with the pattern described in Section 5.2 for seed 41.

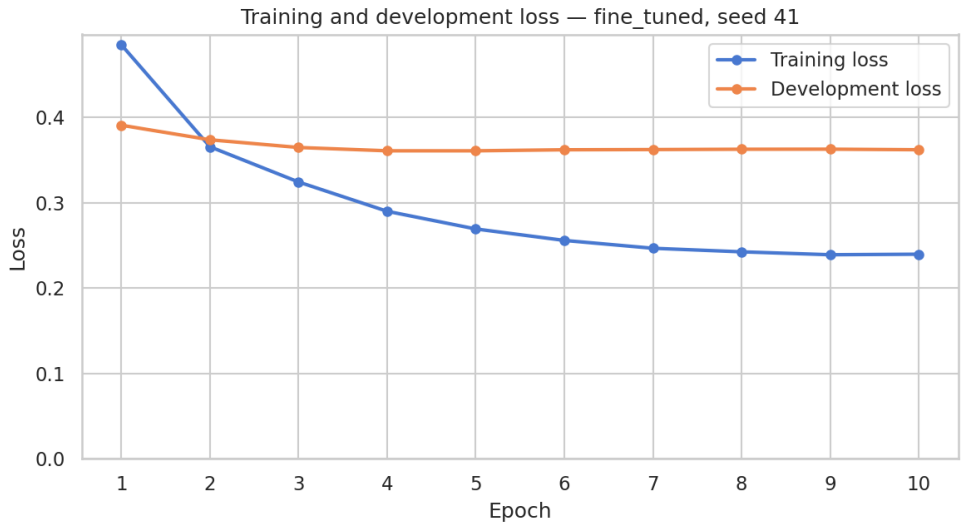

Figure 3: Fine-tuned condition, seed 41.

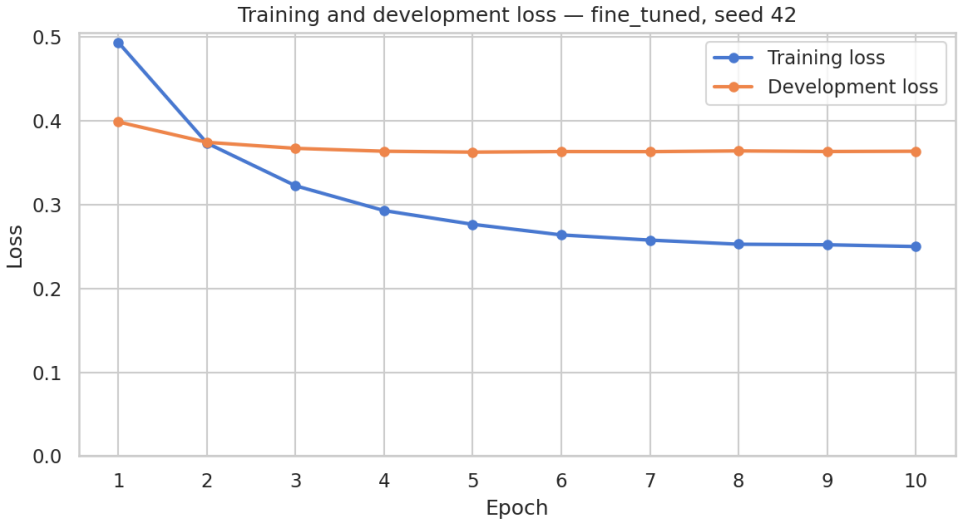

Figure 4: Fine-tuned condition, seed 42.

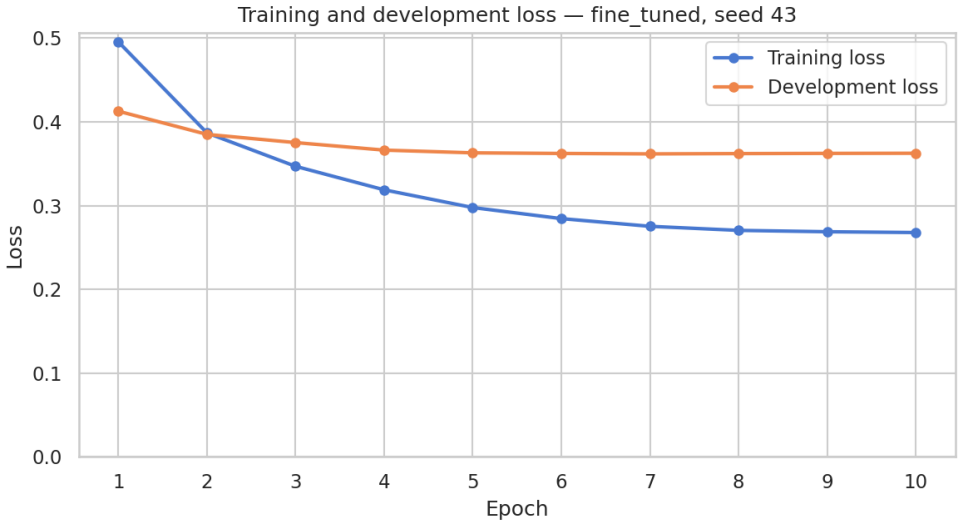

Figure 5: Fine-tuned condition, seed 43.

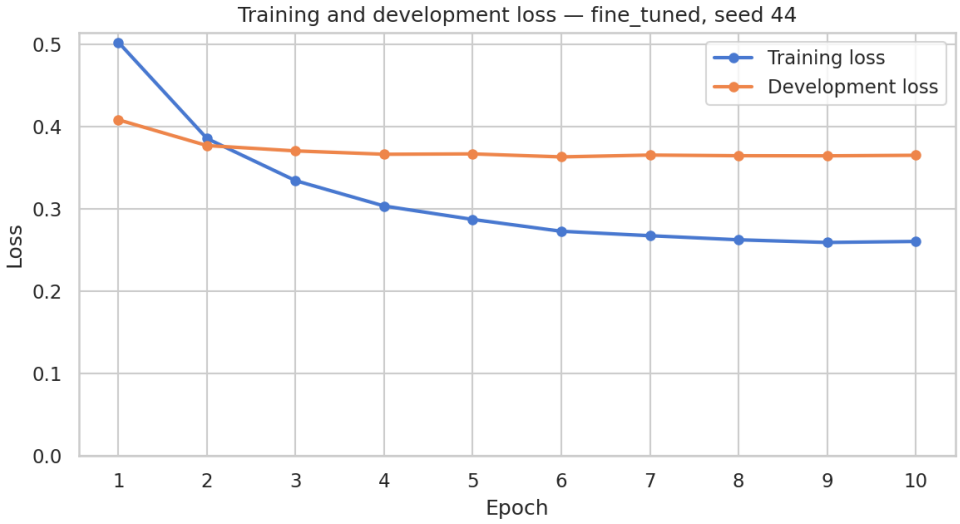

Figure 6: Fine-tuned condition, seed 44.

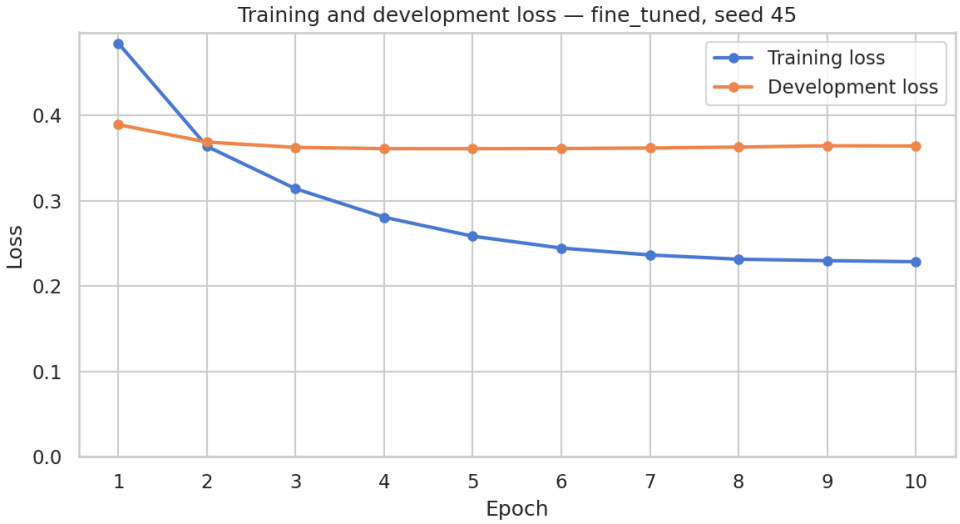

Figure 7: Fine-tuned condition, seed 45.

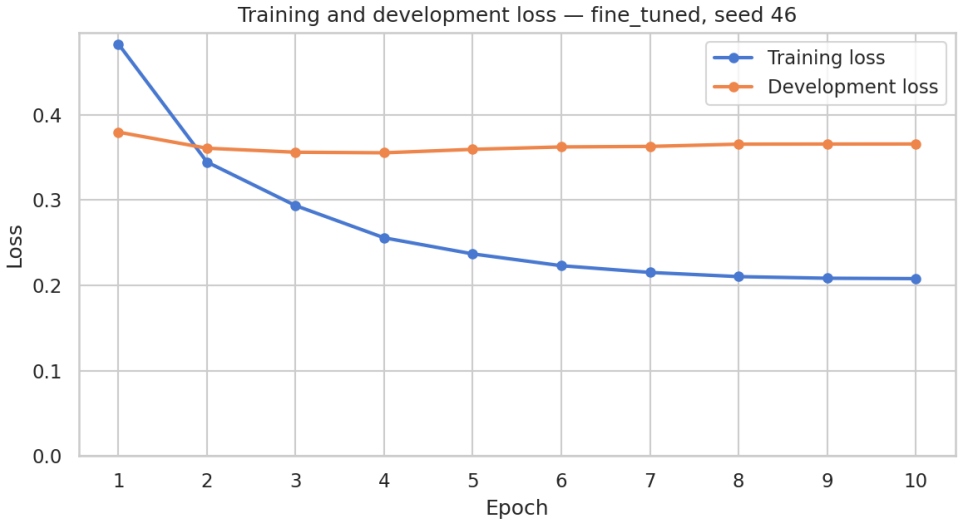

Figure 8: Fine-tuned condition, seed 46.

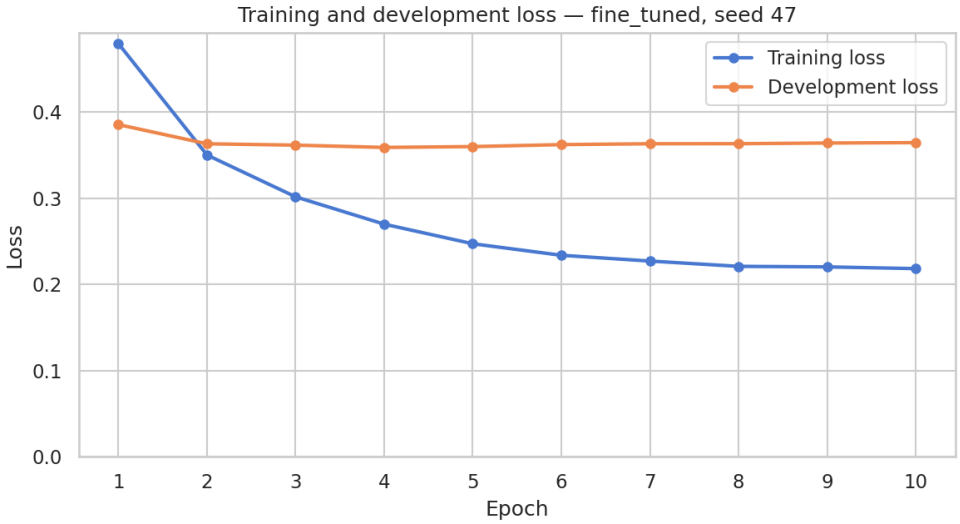

Figure 9: Fine-tuned condition, seed 47.

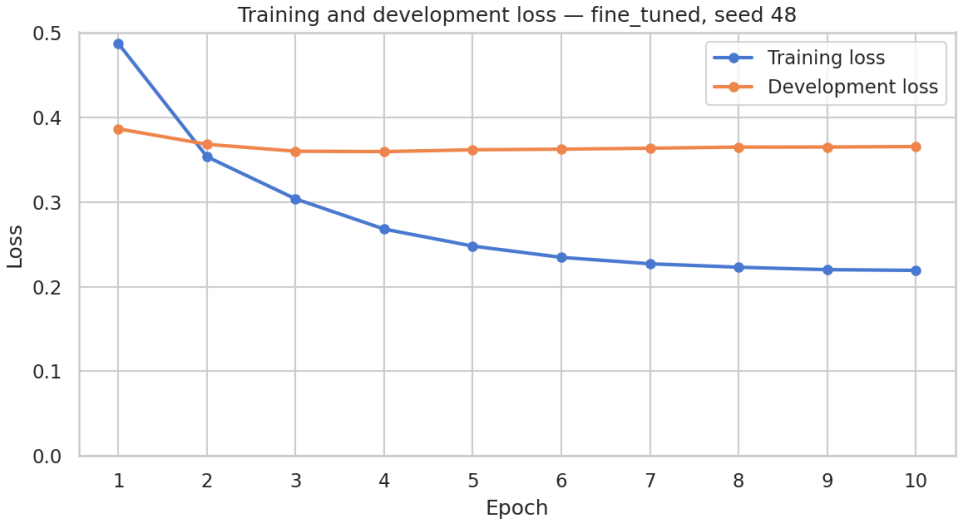

Figure 10: Fine-tuned condition, seed 48.

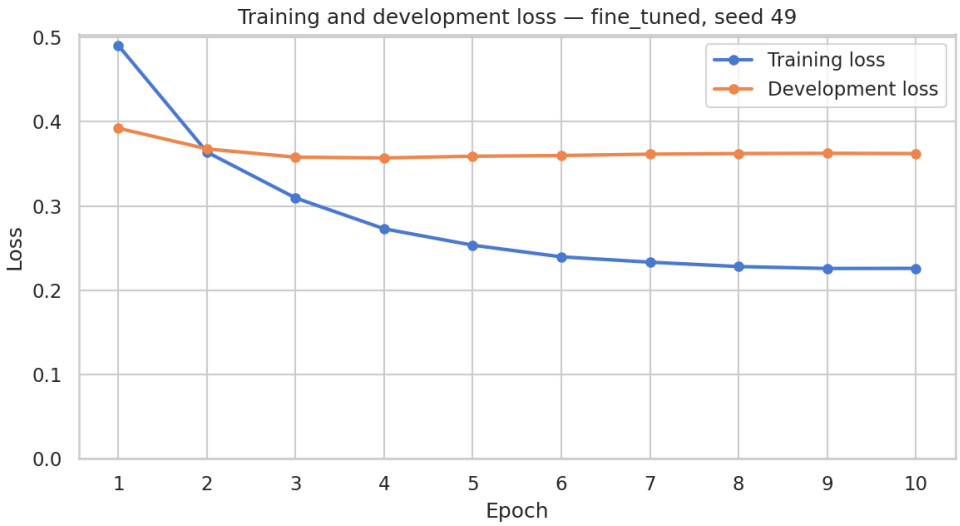

Figure 11: Fine-tuned condition, seed 49.

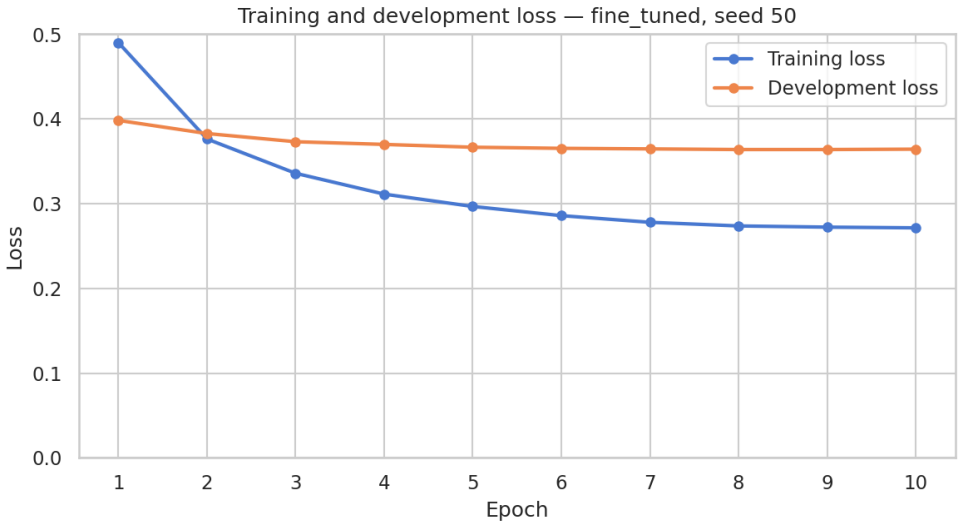

Figure 12: Fine-tuned condition, seed 50.

B Per-Seed Loss Plots: Worked Examples Condition

Figures 13–22 show training and development loss per epoch for each of the ten random seeds in the Worked Examples condition. In all seeds, development loss reaches its minimum around epoch 4 and then rises slightly while training loss continues to decrease, consistent with the pattern described in Section 5.3 for seed 41. The best checkpoint is selected from the epoch with the lowest development loss in each seed.

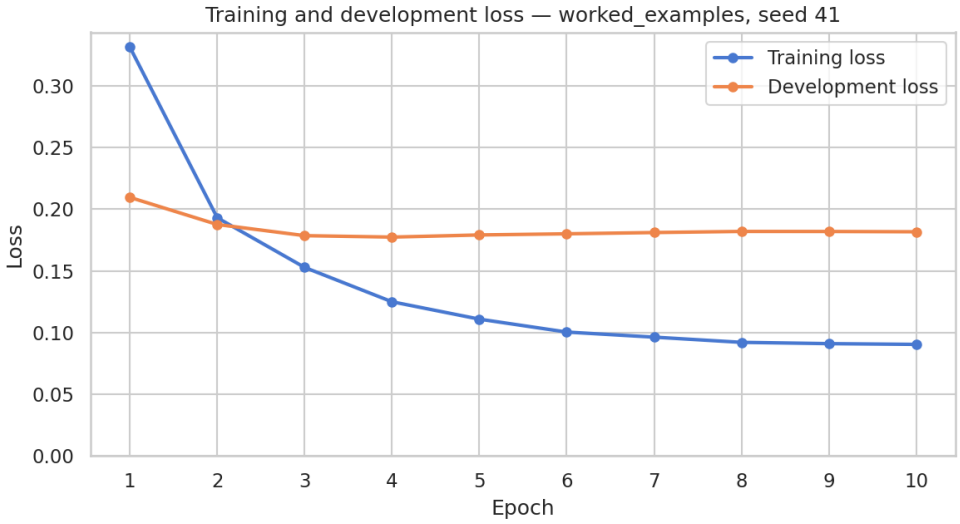

Figure 13: Worked Examples condition, seed 41.

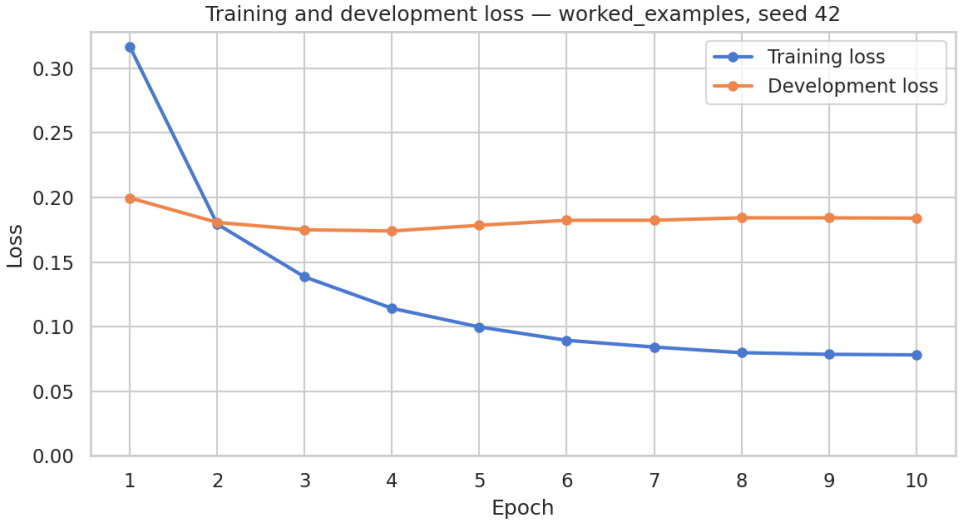

Figure 14: Worked Examples condition, seed 42.

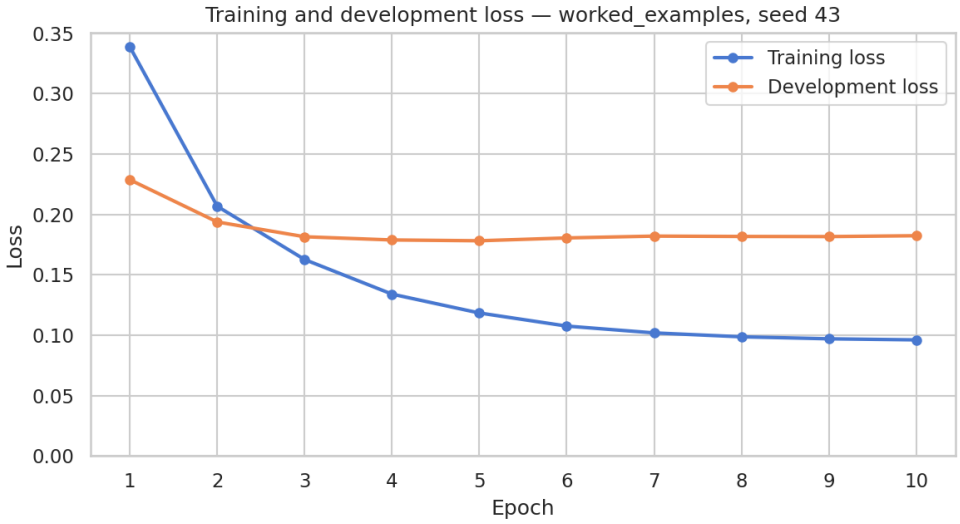

Figure 15: Worked Examples condition, seed 43.

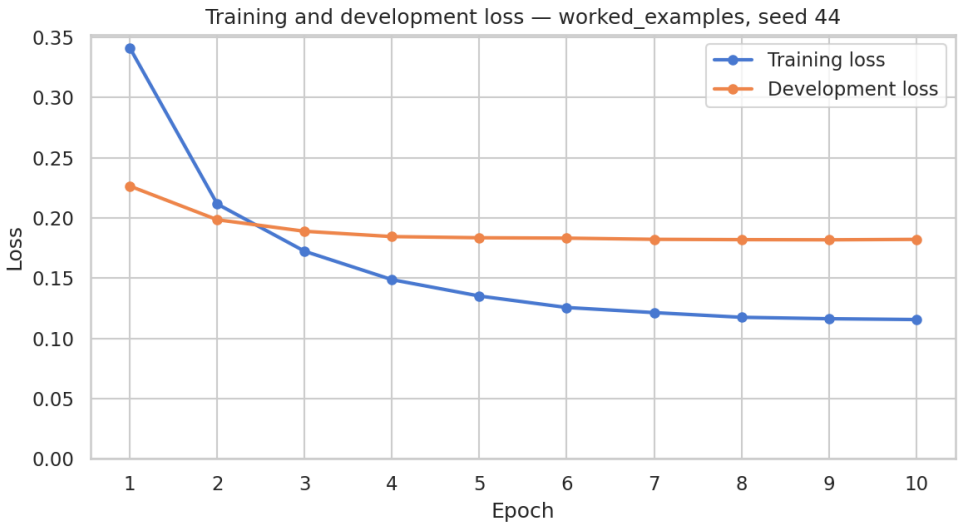

Figure 16: Worked Examples condition, seed 44.

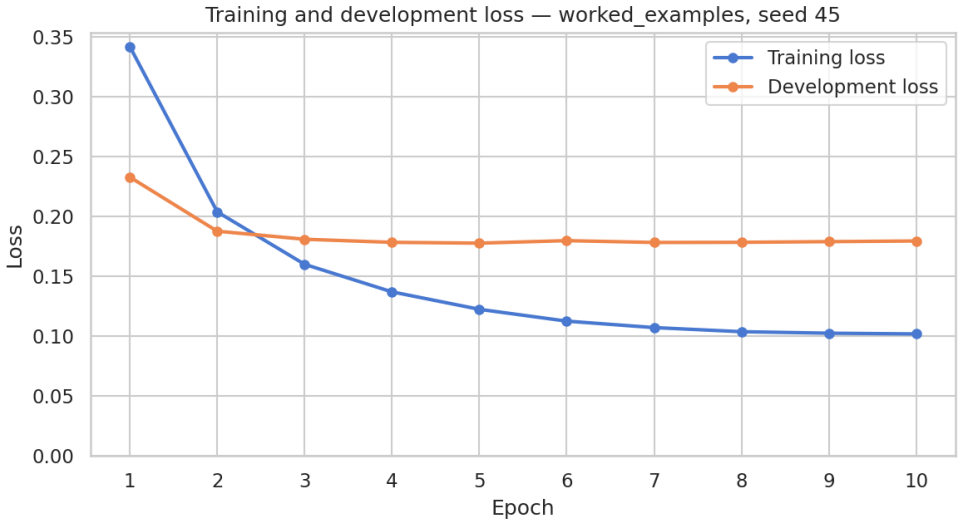

Figure 17: Worked Examples condition, seed 45.

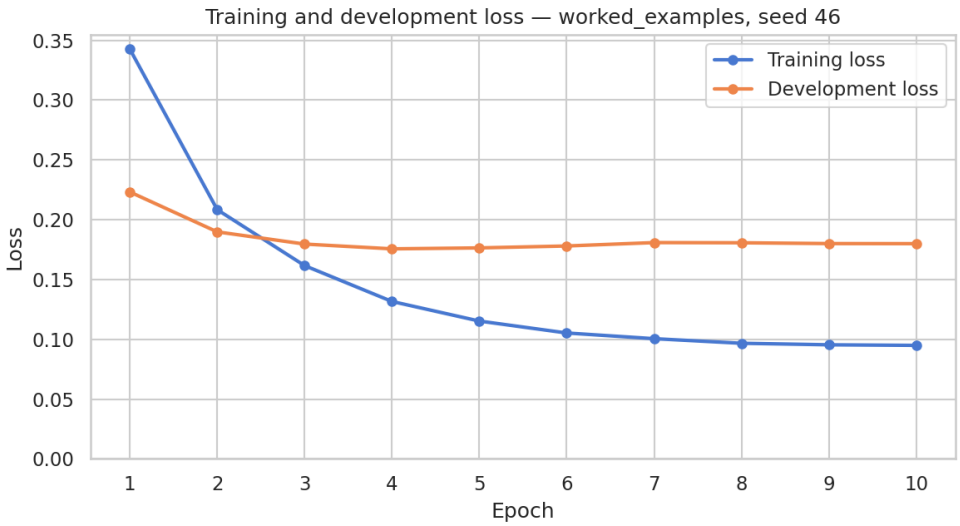

Figure 18: Worked Examples condition, seed 46.

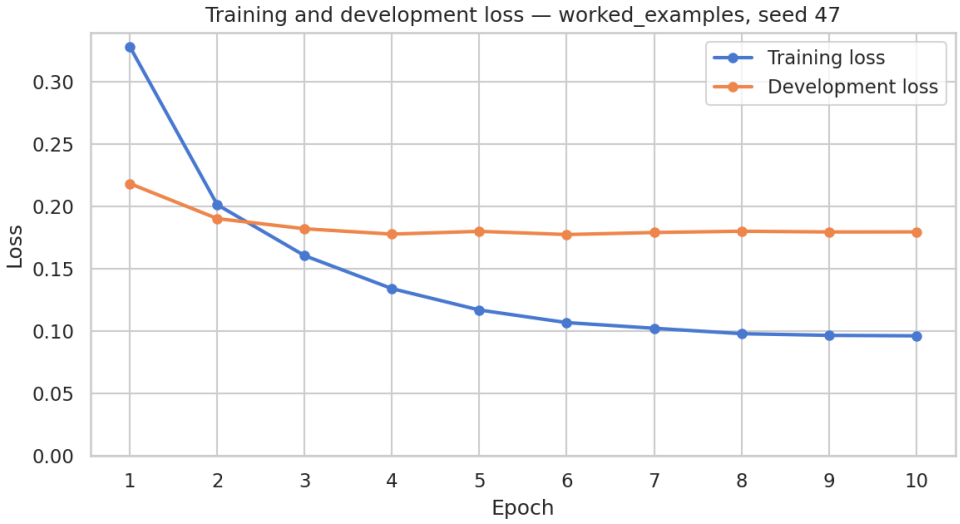

Figure 19: Worked Examples condition, seed 47.

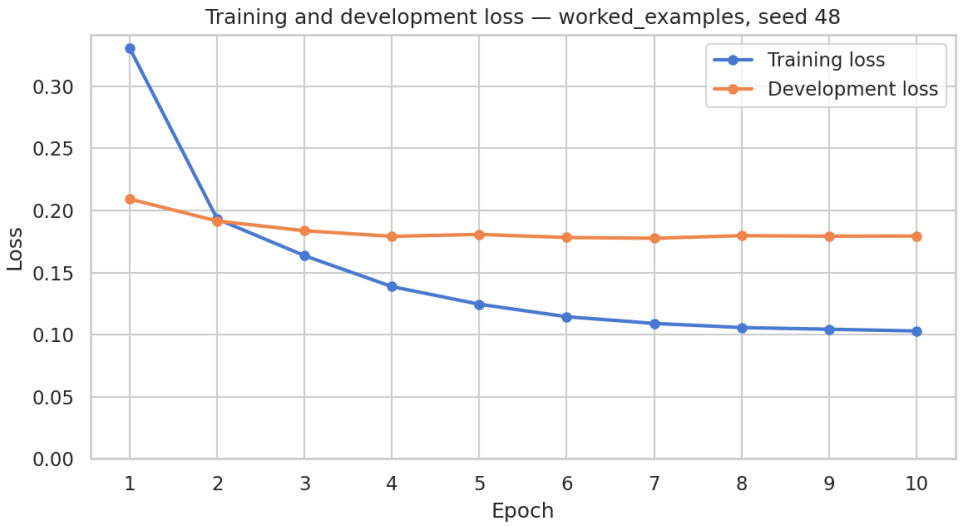

Figure 20: Worked Examples condition, seed 48.

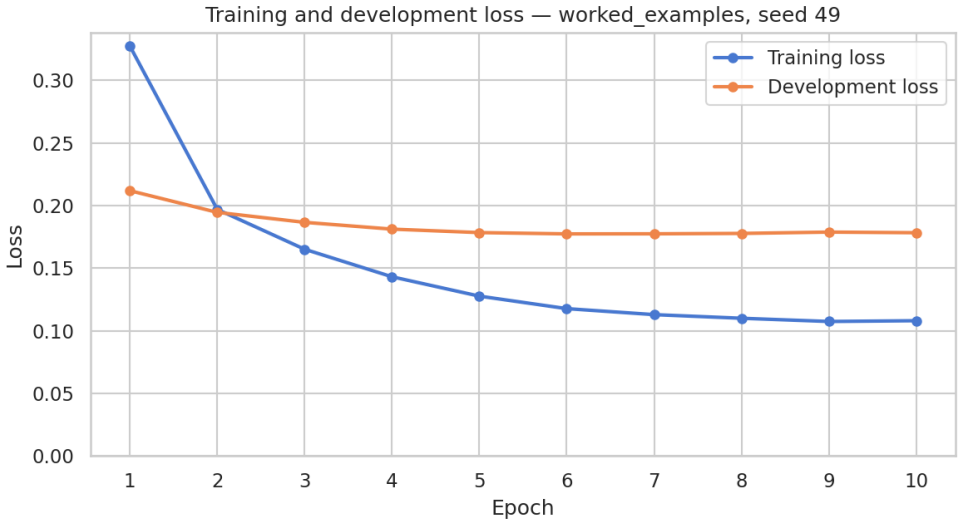

Figure 21: Worked Examples condition, seed 49.

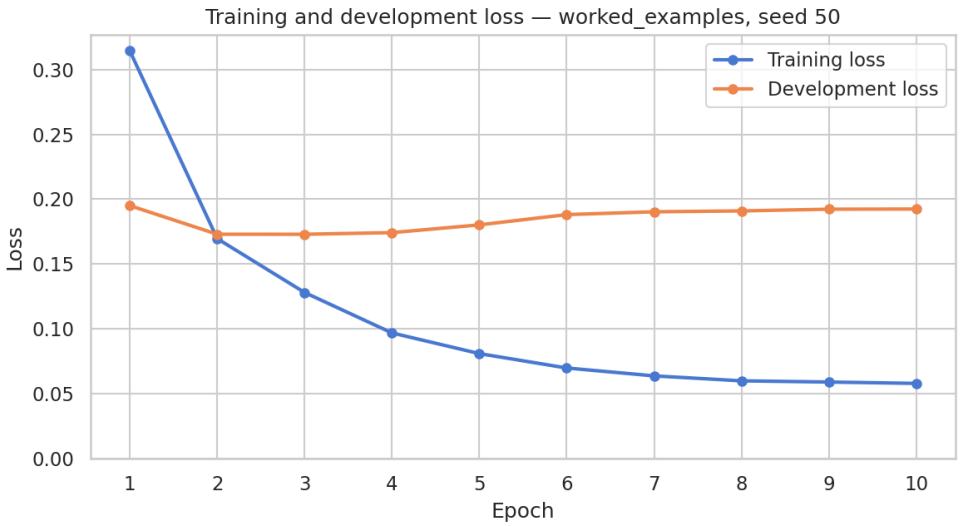

Figure 22: Worked Examples condition, seed 50.

C Dev Inference Outputs: Fine-tuned Condition (Seed 41)

The following are the raw development set outputs for the Fine-tuned condition, seed 41. These outputs are the basis for the qualitative analysis in Section 5.2.

```
============================================================
Run: 2026-06-16 14:21:15
============================================================
Notebook : fine_tuned_trained
Condition: fine_tuned
Seed     : 41
Model    : EleutherAI/pythia-410m  (fine-tuned)
Train loss: 0.2946
Dev loss : 0.3959

Dev inference -- first 10 examples
============================================================

[Dev 01]
  1. Prompt repr   : 'Fix: It \'s difficult answer at the question " what are you going to do
    in the future ? " if the only one who has to know it is in two minds .\nCorrect:'
  2. Prompt IDs    : [27281, 27, 733, 686, 84, 2834, 3662, 387, 253, 1953, 346, 752, 403, 368,
    1469, 281, 513, 275, 253, 2852, 3736, 346, 604, 253, 760, 581, 665, 556, 281, 871, 352, 310,
     275, 767, 13846, 964, 187, 47390, 27]
  3. Generated IDs : [733, 686, 84, 247, 2834, 3662, 281, 253, 1953, 346, 752, 403, 368, 1469,
    281, 513, 275, 253, 2852, 3736, 346, 604, 253, 760, 581, 665, 556, 281, 871, 352, 310, 275,
     767, 13846, 964, 0]
  4. Decoded repr  : 'It \'s a difficult answer to the question " what are you going to do in
    the future ? " if the only one who has to know it is in two minds .'
  5. Clean output  : 'It \'s a difficult answer to the question " what are you going to do in
    the future ? " if the only one who has to know it is in two minds .'
  6. Gold reference: 'It \'s difficult to answer the question " what are you going to do in the
     future ? " if the only one who has to know it is in two minds .'

[Dev 02]
  1. Prompt repr   : "Fix: When I was younger I used to say that I wanted to be a teacher , a
    saleswoman and even a butcher .. I do n't know why .\nCorrect:"
  2. Prompt IDs    : [27281, 27, 2091, 309, 369, 9243, 309, 908, 281, 1333, 326, 309, 3078,
    281, 320, 247, 9732, 1157, 247, 6224, 17217, 285, 1014, 247, 45975, 10712, 309, 513, 295,
    626, 871, 2139, 964, 187, 47390, 27]
  3. Generated IDs : [2091, 309, 369, 9243, 1157, 309, 908, 281, 1333, 326, 309, 3078, 281,
    320, 247, 9732, 1157, 247, 6224, 17217, 285, 1014, 247, 45975, 964, 309, 513, 295, 626, 871,
     2139, 964, 0]
  4. Decoded repr  : "When I was younger , I used to say that I wanted to be a teacher , a
```

```
    saleswoman and even a butcher . I do n't know why ."
  5. Clean output  : "When I was younger , I used to say that I wanted to be a teacher , a
    saleswoman and even a butcher . I do n't know why ."
  6. Gold reference: "When I was younger , I used to say that I wanted to be a teacher , a
    saleswoman and even a butcher . I do n't know why ."

[Dev 03]
  1. Prompt repr   : 'Fix: I would like to study Psychology because one day I would open my own
     psychology office and help people .\nCorrect:'
  2. Prompt IDs    : [27281, 27, 309, 651, 751, 281, 1263, 29624, 984, 581, 1388, 309, 651,
    1527, 619, 1211, 20162, 3906, 285, 1361, 952, 964, 187, 47390, 27]
  3. Generated IDs : [309, 651, 751, 281, 1263, 29624, 984, 581, 1388, 309, 651, 1527, 619,
    1211, 20162, 3906, 285, 1361, 952, 964, 0]
  4. Decoded repr  : 'I would like to study Psychology because one day I would open my own
    psychology office and help people .'
  5. Clean output  : 'I would like to study Psychology because one day I would open my own
    psychology office and help people .'
  6. Gold reference: 'I would like to study Psychology , because one day I would like to open
    my own psychology clinic and help people .'

[Dev 04]
  1. Prompt repr   : "Fix: It 's difficult because I 'll have to study hard and a lot , but I
    think that if you like a subject , you 'll study it easier .\nCorrect:"
  2. Prompt IDs    : [27281, 27, 733, 686, 84, 2834, 984, 309, 686, 620, 452, 281, 1263, 1892,
    285, 247, 2257, 1157, 533, 309, 1158, 326, 604, 368, 751, 247, 2256, 1157, 368, 686, 620,
    1263, 352, 6927, 964, 187, 47390, 27]
  3. Generated IDs : [733, 686, 84, 2834, 984, 309, 686, 620, 452, 281, 1263, 1892, 285, 247,
    2257, 1157, 533, 309, 1158, 326, 604, 368, 751, 247, 2256, 1157, 368, 686, 620, 1263, 352,
    6927, 964, 0]
  4. Decoded repr  : "It 's difficult because I 'll have to study hard and a lot , but I think
    that if you like a subject , you 'll study it easier ."
  5. Clean output  : "It 's difficult because I 'll have to study hard and a lot , but I think
    that if you like a subject , you 'll study it easier ."
  6. Gold reference: "It 's difficult because I 'll have to study hard and a lot , but I think
    that if you like a subject , you 'll study it more easily ."

[Dev 05]
  1. Prompt repr   : 'Fix: I think that the public transport will always be in the future .\
    nCorrect:'
  2. Prompt IDs    : [27281, 27, 309, 1158, 326, 253, 1345, 4616, 588, 1900, 320, 275, 253,
    2852, 964, 187, 47390, 27]
  3. Generated IDs : [309, 1158, 326, 1345, 4616, 588, 1900, 320, 275, 253, 2852, 964, 0]
  4. Decoded repr  : 'I think that public transport will always be in the future .'
  5. Clean output  : 'I think that public transport will always be in the future .'
```

```
  6. Gold reference: 'I think that public transport will always exist in the future .'

[Dev 06]
  1. Prompt repr   : 'Fix: The rich people will buy a car but the poor people always need to
     use a bus or taxi .\nCorrect:'
  2. Prompt IDs    : [27281, 27, 380, 6793, 952, 588, 4489, 247, 1113, 533, 253, 4105, 952,
     1900, 878, 281, 897, 247, 1685, 390, 24928, 964, 187, 47390, 27]
  3. Generated IDs : [380, 6793, 952, 4489, 247, 1113, 533, 253, 4105, 952, 1900, 878, 281,
     897, 247, 1685, 390, 24928, 964, 0]
  4. Decoded repr  : 'The rich people buy a car but the poor people always need to use a bus or
      taxi .'
  5. Clean output  : 'The rich people buy a car but the poor people always need to use a bus or
      taxi .'
  6. Gold reference: 'Rich people will buy a car , but poor people always need to use a bus or
     taxi .'

[Dev 07]
  1. Prompt repr   : 'Fix: I consider that is more convenient to drive a car because you carry
     on more things in your own car than travelling by car .\nCorrect:'
  2. Prompt IDs    : [27281, 27, 309, 1908, 326, 310, 625, 11638, 281, 4446, 247, 1113, 984,
     368, 4459, 327, 625, 1841, 275, 634, 1211, 1113, 685, 22710, 407, 1113, 964, 187, 47390,
     27]
  3. Generated IDs : [309, 1908, 326, 352, 310, 625, 11638, 281, 4446, 247, 1113, 984, 368,
     4459, 327, 625, 1841, 275, 634, 1211, 1113, 685, 22710, 407, 1113, 964, 0]
  4. Decoded repr  : 'I consider that it is more convenient to drive a car because you carry on
      more things in your own car than travelling by car .'
  5. Clean output  : 'I consider that it is more convenient to drive a car because you carry on
      more things in your own car than travelling by car .'
  6. Gold reference: 'I consider it more convenient to drive a car , because you carry more
     things in your own car than when travelling by car .'

[Dev 08]
  1. Prompt repr   : "Fix: Also , you 'll meet friendly people who usually ask to you something
      to be friends and change your telephone number .\nCorrect:"
  2. Prompt IDs    : [27281, 27, 5220, 1157, 368, 686, 620, 2525, 11453, 952, 665, 3798, 1642,
     281, 368, 1633, 281, 320, 3858, 285, 1818, 634, 11265, 1180, 964, 187, 47390, 27]
  3. Generated IDs : [5220, 1157, 368, 686, 620, 2525, 11453, 952, 665, 3798, 1642, 368, 1633,
     281, 320, 3858, 285, 1818, 634, 11265, 1180, 964, 0]
  4. Decoded repr  : "Also , you 'll meet friendly people who usually ask you something to be
     friends and change your telephone number ."
  5. Clean output  : "Also , you 'll meet friendly people who usually ask you something to be
     friends and change your telephone number ."
  6. Gold reference: "Also , you 'll meet friendly people who usually ask you to be friends and
      exchange telephone numbers ."
```

```
[Dev 09]
  1. Prompt repr   : "Fix: In my experience when I did n't have a car I used to use the bus to
     go to the school and go back to my house .\nCorrect:"
  2. Prompt IDs    : [27281, 27, 496, 619, 2793, 672, 309, 858, 295, 626, 452, 247, 1113, 309,
     908, 281, 897, 253, 1685, 281, 564, 281, 253, 2143, 285, 564, 896, 281, 619, 2419, 964, 187,
      47390, 27]
  3. Generated IDs : [496, 619, 2793, 1157, 672, 309, 858, 295, 626, 452, 247, 1113, 1157, 309,
      908, 281, 897, 253, 1685, 281, 564, 281, 2143, 285, 564, 896, 281, 619, 2419, 964, 0]
  4. Decoded repr  : "In my experience , when I did n't have a car , I used to use the bus to
     go to school and go back to my house ."
  5. Clean output  : "In my experience , when I did n't have a car , I used to use the bus to
     go to school and go back to my house ."
  6. Gold reference: "In my experience , when I did n't have a car I used to use the bus to go
     to school and go back to my house ."

[Dev 10]
  1. Prompt repr   : "Fix: In my opinion , the car is n't necessary when you have crashed in
     the street , in that moment you realized the importance of a public transport .\nCorrect:"
  2. Prompt IDs    : [27281, 27, 496, 619, 4743, 1157, 253, 1113, 310, 295, 626, 3309, 672,
     368, 452, 25753, 275, 253, 6406, 1157, 275, 326, 2774, 368, 8156, 253, 6349, 273, 247, 1345,
      4616, 964, 187, 47390, 27]
  3. Generated IDs : [496, 619, 4743, 1157, 253, 1113, 310, 295, 626, 3309, 672, 368, 452,
     25753, 275, 253, 6406, 1157, 275, 326, 2774, 368, 8968, 253, 6349, 273, 1345, 4616, 964, 0]
  4. Decoded repr  : "In my opinion , the car is n't necessary when you have crashed in the
     street , in that moment you realize the importance of public transport ."
  5. Clean output  : "In my opinion , the car is n't necessary when you have crashed in the
     street , in that moment you realize the importance of public transport ."
  6. Gold reference: "In my opinion , a car is n't necessary when you have crashed in the
     street . At that moment , you realize the importance of public transport ."
```

D Dev Inference Outputs: Worked Examples Condition (Seed 41)

The following are the raw development set outputs for the Worked Examples condition evaluated without the explanation at inference time, seed 41. The inference prompt matches the Fine-tuned condition exactly (`Fix: S\nCorrect:`). These outputs are the basis for the qualitative analysis in Section 5.3.

```
============================================================
Run: 2026-06-13 18:07:11
============================================================
Notebook : worked_examples_trained
Condition: worked_examples
Seed     : 41
Model    : EleutherAI/pythia-410m  (fine-tuned)
Train loss: 0.1299
Dev loss : 0.1599

Dev inference -- first 10 examples
(Inference prompt: Fix:/Correct: only -- no explanation at test time)
============================================================

[Dev 01]
  1. Prompt repr   : 'Fix: It \'s difficult answer at the question " what are you going to do
    in the future ? " if the only one who has to know it is in two minds .\nCorrect:'
  2. Prompt IDs    : [27281, 27, 733, 686, 84, 2834, 3662, 387, 253, 1953, 346, 752, 403, 368,
    1469, 281, 513, 275, 253, 2852, 3736, 346, 604, 253, 760, 581, 665, 556, 281, 871, 352, 310,
     275, 767, 13846, 964, 187, 47390, 27]
  3. Generated IDs : [733, 686, 84, 2834, 3662, 387, 253, 1953, 346, 752, 403, 368, 1469, 281,
    513, 275, 253, 2852, 3736, 346, 604, 253, 760, 581, 665, 556, 281, 871, 352, 310, 275, 767,
     13846, 964, 0]
  4. Decoded repr  : 'It \'s difficult answer at the question " what are you going to do in the
     future ? " if the only one who has to know it is in two minds .'
  5. Clean output  : 'It \'s difficult answer at the question " what are you going to do in the
     future ? " if the only one who has to know it is in two minds .'
  6. Gold reference: 'It \'s difficult to answer the question " what are you going to do in the
     future ? " if the only one who has to know it is in two minds .'

[Dev 02]
  1. Prompt repr   : "Fix: When I was younger I used to say that I wanted to be a teacher , a
    saleswoman and even a butcher .. I do n't know why .\nCorrect:"
  2. Prompt IDs    : [27281, 27, 2091, 309, 369, 9243, 309, 908, 281, 1333, 326, 309, 3078,
    281, 320, 247, 9732, 1157, 247, 6224, 17217, 285, 1014, 247, 45975, 10712, 309, 513, 295,
```

```
   626, 871, 2139, 964, 187, 47390, 27]
  3. Generated IDs : [2091, 309, 369, 9243, 309, 908, 281, 1333, 326, 309, 3078, 281, 320, 247,
     9732, 1157, 247, 6224, 17217, 285, 1014, 247, 45975, 10712, 309, 513, 295, 626, 871, 2139,
     964, 0]
  4. Decoded repr  : "When I was younger I used to say that I wanted to be a teacher , a
    saleswoman and even a butcher .. I do n't know why ."
  5. Clean output  : "When I was younger I used to say that I wanted to be a teacher , a
    saleswoman and even a butcher .. I do n't know why ."
  6. Gold reference: "When I was younger , I used to say that I wanted to be a teacher , a
    saleswoman and even a butcher . I do n't know why ."

[Dev 03]
  1. Prompt repr   : 'Fix: I would like to study Psychology because one day I would open my own
     psychology office and help people .\nCorrect:'
  2. Prompt IDs    : [27281, 27, 309, 651, 751, 281, 1263, 29624, 984, 581, 1388, 309, 651,
    1527, 619, 1211, 20162, 3906, 285, 1361, 952, 964, 187, 47390, 27]
  3. Generated IDs : [309, 651, 751, 281, 1263, 29624, 984, 581, 1388, 309, 651, 1527, 619,
    1211, 20162, 3906, 285, 1361, 952, 964, 0]
  4. Decoded repr  : 'I would like to study Psychology because one day I would open my own
    psychology office and help people .'
  5. Clean output  : 'I would like to study Psychology because one day I would open my own
    psychology office and help people .'
  6. Gold reference: 'I would like to study Psychology , because one day I would like to open
    my own psychology clinic and help people .'

[Dev 04]
  1. Prompt repr   : "Fix: It 's difficult because I 'll have to study hard and a lot , but I
    think that if you like a subject , you 'll study it easier .\nCorrect:"
  2. Prompt IDs    : [27281, 27, 733, 686, 84, 2834, 984, 309, 686, 620, 452, 281, 1263, 1892,
    285, 247, 2257, 1157, 533, 309, 1158, 326, 604, 368, 751, 247, 2256, 1157, 368, 686, 620,
    1263, 352, 6927, 964, 187, 47390, 27]
  3. Generated IDs : [733, 686, 84, 2834, 984, 309, 686, 620, 452, 281, 1263, 1892, 285, 247,
    2257, 1157, 533, 309, 1158, 326, 604, 368, 751, 247, 2256, 1157, 368, 686, 620, 1263, 352,
    6927, 964, 0]
  4. Decoded repr  : "It 's difficult because I 'll have to study hard and a lot , but I think
    that if you like a subject , you 'll study it easier ."
  5. Clean output  : "It 's difficult because I 'll have to study hard and a lot , but I think
    that if you like a subject , you 'll study it easier ."
  6. Gold reference: "It 's difficult because I 'll have to study hard and a lot , but I think
    that if you like a subject , you 'll study it more easily ."

[Dev 05]
  1. Prompt repr   : 'Fix: I think that the public transport will always be in the future .\
    nCorrect:'
```

```
  2. Prompt IDs    : [27281, 27, 309, 1158, 326, 253, 1345, 4616, 588, 1900, 320, 275, 253,
    2852, 964, 187, 47390, 27]
  3. Generated IDs : [309, 1158, 326, 253, 1345, 4616, 588, 1900, 320, 275, 253, 2852, 964, 0]
  4. Decoded repr  : 'I think that the public transport will always be in the future .'
  5. Clean output  : 'I think that the public transport will always be in the future .'
  6. Gold reference: 'I think that public transport will always exist in the future .'

[Dev 06]
  1. Prompt repr   : 'Fix: The rich people will buy a car but the poor people always need to
    use a bus or taxi .\nCorrect:'
  2. Prompt IDs    : [27281, 27, 380, 6793, 952, 588, 4489, 247, 1113, 533, 253, 4105, 952,
    1900, 878, 281, 897, 247, 1685, 390, 24928, 964, 187, 47390, 27]
  3. Generated IDs : [380, 6793, 952, 588, 4489, 247, 1113, 533, 253, 4105, 952, 1900, 878,
    281, 897, 247, 1685, 390, 24928, 964, 0]
  4. Decoded repr  : 'The rich people will buy a car but the poor people always need to use a
    bus or taxi .'
  5. Clean output  : 'The rich people will buy a car but the poor people always need to use a
    bus or taxi .'
  6. Gold reference: 'Rich people will buy a car , but poor people always need to use a bus or
    taxi .'

[Dev 07]
  1. Prompt repr   : 'Fix: I consider that is more convenient to drive a car because you carry
    on more things in your own car than travelling by car .\nCorrect:'
  2. Prompt IDs    : [27281, 27, 309, 1908, 326, 310, 625, 11638, 281, 4446, 247, 1113, 984,
    368, 4459, 327, 625, 1841, 275, 634, 1211, 1113, 685, 22710, 407, 1113, 964, 187, 47390,
    27]
  3. Generated IDs : [309, 1908, 326, 310, 625, 11638, 281, 4446, 247, 1113, 984, 368, 4459,
    327, 625, 1841, 275, 634, 1211, 1113, 685, 22710, 407, 1113, 964, 0]
  4. Decoded repr  : 'I consider that is more convenient to drive a car because you carry on
    more things in your own car than travelling by car .'
  5. Clean output  : 'I consider that is more convenient to drive a car because you carry on
    more things in your own car than travelling by car .'
  6. Gold reference: 'I consider it more convenient to drive a car , because you carry more
    things in your own car than when travelling by car .'

[Dev 08]
  1. Prompt repr   : "Fix: Also , you 'll meet friendly people who usually ask to you something
     to be friends and change your telephone number .\nCorrect:"
  2. Prompt IDs    : [27281, 27, 5220, 1157, 368, 686, 620, 2525, 11453, 952, 665, 3798, 1642,
    281, 368, 1633, 281, 320, 3858, 285, 1818, 634, 11265, 1180, 964, 187, 47390, 27]
  3. Generated IDs : [5220, 1157, 368, 686, 620, 2525, 11453, 952, 665, 3798, 1642, 368, 1633,
    281, 320, 3858, 285, 1818, 634, 11265, 1180, 964, 0]
  4. Decoded repr  : "Also , you 'll meet friendly people who usually ask you something to be
```

```
    friends and change your telephone number ."
  5. Clean output  : "Also , you 'll meet friendly people who usually ask you something to be
    friends and change your telephone number ."
  6. Gold reference: "Also , you 'll meet friendly people who usually ask you to be friends and
     exchange telephone numbers ."

[Dev 09]
  1. Prompt repr   : "Fix: In my experience when I did n't have a car I used to use the bus to
    go to the school and go back to my house .\nCorrect:"
  2. Prompt IDs    : [27281, 27, 496, 619, 2793, 672, 309, 858, 295, 626, 452, 247, 1113, 309,
    908, 281, 897, 253, 1685, 281, 564, 281, 253, 2143, 285, 564, 896, 281, 619, 2419, 964, 187,
     47390, 27]
  3. Generated IDs : [496, 619, 2793, 672, 309, 858, 295, 626, 452, 247, 1113, 309, 908, 281,
    897, 253, 1685, 281, 564, 281, 253, 2143, 285, 564, 896, 281, 619, 2419, 964, 0]
  4. Decoded repr  : "In my experience when I did n't have a car I used to use the bus to go to
     the school and go back to my house ."
  5. Clean output  : "In my experience when I did n't have a car I used to use the bus to go to
     the school and go back to my house ."
  6. Gold reference: "In my experience , when I did n't have a car I used to use the bus to go
    to school and go back to my house ."

[Dev 10]
  1. Prompt repr   : "Fix: In my opinion , the car is n't necessary when you have crashed in
    the street , in that moment you realized the importance of a public transport .\nCorrect:"
  2. Prompt IDs    : [27281, 27, 496, 619, 4743, 1157, 253, 1113, 310, 295, 626, 3309, 672,
    368, 452, 25753, 275, 253, 6406, 1157, 275, 326, 2774, 368, 8156, 253, 6349, 273, 247, 1345,
     4616, 964, 187, 47390, 27]
  3. Generated IDs : [496, 619, 4743, 1157, 253, 1113, 310, 295, 626, 3309, 672, 368, 452,
    25753, 275, 253, 6406, 1157, 275, 326, 2774, 368, 8156, 253, 6349, 273, 247, 1345, 4616,
    964, 0]
  4. Decoded repr  : "In my opinion , the car is n't necessary when you have crashed in the
    street , in that moment you realized the importance of a public transport ."
  5. Clean output  : "In my opinion , the car is n't necessary when you have crashed in the
    street , in that moment you realized the importance of a public transport ."
  6. Gold reference: "In my opinion , a car is n't necessary when you have crashed in the
    street . At that moment , you realize the importance of public transport ."
```

E Dev Inference Outputs: Worked Examples with Explanation (Seed 41)

The following are the raw development set outputs for the Worked Examples condition evaluated with the explanation present at inference time, seed 41. The inference prompt includes the explanation field generated from gold annotations (`Fix: S\nExplanation: E\nCorrect:`). These outputs are the basis for the qualitative analysis in Section 5.4.

```
============================================================
Run: 2026-06-07 18:07:39
============================================================
Notebook : worked_examples_trained_with_explanation
Condition: worked_examples_with_explanation
Seed     : 41
Model    : EleutherAI/pythia-410m  (fine-tuned)
Train loss: 0.2201
Dev loss : 0.1843

Dev inference -- first 10 examples
(Inference prompt: Fix:/Explanation:/Correct: -- matches training format)
============================================================

[Dev 01]
  1. Prompt repr   : 'Fix: It \'s difficult answer at the question " what are you going to do
     in the future ? " if the only one who has to know it is in two minds .\nExplanation: The
     correction adds \'to\' to fix verb form. The correction removes \'at\' because it is
     unnecessary here. This gives the verb the form that fits the surrounding structure.\
     nCorrect:'
  2. Prompt IDs    : [27281, 27, 733, 686, 84, 2834, 3662, 387, 253, 1953, 346, 752, 403, 368,
     1469, 281, 513, 275, 253, 2852, 3736, 346, 604, 253, 760, 581, 665, 556, 281, 871, 352, 310,
      275, 767, 13846, 964, 187, 1672, 45525, 27, 380, 10618, 11323, 686, 936, 8, 281, 4993,
     17257, 830, 15, 380, 10618, 26586, 686, 255, 8, 984, 352, 310, 15279, 1060, 15, 831, 4245,
     253, 17257, 253, 830, 326, 13840, 253, 8704, 2605, 15, 187, 47390, 27]
  3. Generated IDs : [733, 686, 84, 2834, 3662, 281, 253, 1953, 346, 752, 403, 368, 1469, 281,
     513, 275, 253, 2852, 3736, 346, 604, 253, 760, 581, 665, 556, 281, 871, 352, 310, 275, 767,
      13846, 964, 0]
  4. Decoded repr  : 'It \'s difficult answer to the question " what are you going to do in the
     future ? " if the only one who has to know it is in two minds .'
  5. Clean output  : 'It \'s difficult answer to the question " what are you going to do in the
     future ? " if the only one who has to know it is in two minds .'
  6. Gold reference: 'It \'s difficult to answer the question " what are you going to do in the
     future ? " if the only one who has to know it is in two minds .'
```

```
[Dev 02]
  1. Prompt repr   : "Fix: When I was younger I used to say that I wanted to be a teacher , a
    saleswoman and even a butcher .. I do n't know why .\nExplanation: The correction adds ','
    to fix punctuation. The correction changes '..' to '.' to fix punctuation. This punctuation
     change makes the sentence structure easier to read.\nCorrect:"
  2. Prompt IDs    : [27281, 27, 2091, 309, 369, 9243, 309, 908, 281, 1333, 326, 309, 3078,
    281, 320, 247, 9732, 1157, 247, 6224, 17217, 285, 1014, 247, 45975, 10712, 309, 513, 295,
    626, 871, 2139, 964, 187, 1672, 45525, 27, 380, 10618, 11323, 686, 4117, 281, 4993, 17256,
    2368, 15, 380, 10618, 2544, 686, 537, 8, 281, 686, 2464, 281, 4993, 17256, 2368, 15, 831,
    17256, 2368, 1818, 2789, 253, 6197, 2605, 6927, 281, 1239, 15, 187, 47390, 27]
  3. Generated IDs : [2091, 309, 369, 9243, 1157, 309, 908, 281, 1333, 326, 309, 3078, 281,
    320, 247, 9732, 1157, 247, 6224, 17217, 285, 1014, 247, 45975, 964, 309, 513, 295, 626, 871,
     2139, 964, 0]
  4. Decoded repr  : "When I was younger , I used to say that I wanted to be a teacher , a
    saleswoman and even a butcher . I do n't know why ."
  5. Clean output  : "When I was younger , I used to say that I wanted to be a teacher , a
    saleswoman and even a butcher . I do n't know why ."
  6. Gold reference: "When I was younger , I used to say that I wanted to be a teacher , a
    saleswoman and even a butcher . I do n't know why ."

[Dev 03]
  1. Prompt repr   : "Fix: I would like to study Psychology because one day I would open my own
     psychology office and help people .\nExplanation: The correction adds ',' to fix
    punctuation. The correction adds 'like to' to fix verb choice. This punctuation change
    makes the sentence structure easier to read, and the remaining 1 additional edit complete
    the final correction.\nCorrect:"
  2. Prompt IDs    : [27281, 27, 309, 651, 751, 281, 1263, 29624, 984, 581, 1388, 309, 651,
    1527, 619, 1211, 20162, 3906, 285, 1361, 952, 964, 187, 1672, 45525, 27, 380, 10618, 11323,
     686, 4117, 281, 4993, 17256, 2368, 15, 380, 10618, 11323, 686, 3022, 281, 8, 281, 4993,
    17257, 4327, 15, 831, 17256, 2368, 1818, 2789, 253, 6197, 2605, 6927, 281, 1239, 13, 285,
    253, 5780, 337, 3081, 12921, 3426, 253, 2457, 10618, 15, 187, 47390, 27]
  3. Generated IDs : [309, 651, 751, 281, 1263, 29624, 1157, 984, 581, 1388, 309, 651, 1527,
    619, 1211, 20162, 3906, 285, 1361, 952, 751, 281, 964, 0]
  4. Decoded repr  : 'I would like to study Psychology , because one day I would open my own
    psychology office and help people like to .'
  5. Clean output  : 'I would like to study Psychology , because one day I would open my own
    psychology office and help people like to .'
  6. Gold reference: 'I would like to study Psychology , because one day I would like to open
    my own psychology clinic and help people .'

[Dev 04]
  1. Prompt repr   : "Fix: It 's difficult because I 'll have to study hard and a lot , but I
    think that if you like a subject , you 'll study it easier .\nExplanation: The correction
    changes 'easier' to 'more easily' to fix adverb choice. This gives the sentence the right
```

```
    adverb choice for this context.\nCorrect:"
  2. Prompt IDs    : [27281, 27, 733, 686, 84, 2834, 984, 309, 686, 620, 452, 281, 1263, 1892,
    285, 247, 2257, 1157, 533, 309, 1158, 326, 604, 368, 751, 247, 2256, 1157, 368, 686, 620,
    1263, 352, 6927, 964, 187, 1672, 45525, 27, 380, 10618, 2544, 686, 19010, 1321, 8, 281, 686,
     3062, 4354, 8, 281, 4993, 519, 25340, 4327, 15, 831, 4245, 253, 6197, 253, 987, 519, 25340,
     4327, 323, 436, 3634, 15, 187, 47390, 27]
  3. Generated IDs : [733, 686, 84, 2834, 984, 309, 686, 620, 452, 281, 1263, 1892, 285, 247,
    2257, 1157, 533, 309, 1158, 326, 604, 368, 751, 247, 2256, 1157, 368, 686, 620, 1263, 352,
    625, 4354, 964, 0]
  4. Decoded repr  : "It 's difficult because I 'll have to study hard and a lot , but I think
    that if you like a subject , you 'll study it more easily ."
  5. Clean output  : "It 's difficult because I 'll have to study hard and a lot , but I think
    that if you like a subject , you 'll study it more easily ."
  6. Gold reference: "It 's difficult because I 'll have to study hard and a lot , but I think
    that if you like a subject , you 'll study it more easily ."

[Dev 05]
  1. Prompt repr   : "Fix: I think that the public transport will always be in the future .\
    nExplanation: The correction removes 'the' because it is unnecessary here. The correction
    changes 'be' to 'exist' to fix verb choice. This fixes the article or determiner so the
    noun phrase is complete.\nCorrect:"
  2. Prompt IDs    : [27281, 27, 309, 1158, 326, 253, 1345, 4616, 588, 1900, 320, 275, 253,
    2852, 964, 187, 1672, 45525, 27, 380, 10618, 26586, 686, 783, 8, 984, 352, 310, 15279, 1060,
     15, 380, 10618, 2544, 686, 1257, 8, 281, 686, 31477, 8, 281, 4993, 17257, 4327, 15, 831,
    26019, 253, 3929, 390, 1571, 7068, 594, 253, 28407, 12616, 310, 3426, 15, 187, 47390, 27]
  3. Generated IDs : [309, 1158, 326, 1345, 4616, 588, 1900, 2226, 275, 253, 2852, 964, 0]
  4. Decoded repr  : 'I think that public transport will always exist in the future .'
  5. Clean output  : 'I think that public transport will always exist in the future .'
  6. Gold reference: 'I think that public transport will always exist in the future .'

[Dev 06]
  1. Prompt repr   : "Fix: The rich people will buy a car but the poor people always need to
    use a bus or taxi .\nExplanation: The correction changes 'The rich' to 'Rich' to fix
    determiner or article use. The correction adds ',' to fix punctuation. This fixes the
    article or determiner so the noun phrase is complete, and the remaining 1 additional edit
    complete the final correction.\nCorrect:"
  2. Prompt IDs    : [27281, 27, 380, 6793, 952, 588, 4489, 247, 1113, 533, 253, 4105, 952,
    1900, 878, 281, 897, 247, 1685, 390, 24928, 964, 187, 1672, 45525, 27, 380, 10618, 2544,
    686, 510, 6793, 8, 281, 686, 17389, 8, 281, 4993, 1571, 7068, 390, 3929, 897, 15, 380,
    10618, 11323, 686, 4117, 281, 4993, 17256, 2368, 15, 831, 26019, 253, 3929, 390, 1571, 7068,
     594, 253, 28407, 12616, 310, 3426, 13, 285, 253, 5780, 337, 3081, 12921, 3426, 253, 2457,
    10618, 15, 187, 47390, 27]
  3. Generated IDs : [5612, 952, 588, 4489, 247, 1113, 1157, 533, 253, 4105, 952, 1900, 878,
    281, 897, 247, 1685, 390, 24928, 964, 0]
```

```
  4. Decoded repr  : 'Rich people will buy a car , but the poor people always need to use a bus
     or taxi .'
  5. Clean output  : 'Rich people will buy a car , but the poor people always need to use a bus
     or taxi .'
  6. Gold reference: 'Rich people will buy a car , but poor people always need to use a bus or
    taxi .'

[Dev 07]
  1. Prompt repr   : "Fix: I consider that is more convenient to drive a car because you carry
    on more things in your own car than travelling by car .\nExplanation: The correction
    changes 'that' to 'it' to fix pronoun choice. The correction removes 'is' because it is
    unnecessary here. Together these edits produce the corrected sentence: 'I consider it more
    convenient to drive a car , because you carry more things in your own car than when
    travelling by car .', and the remaining 3 additional edits complete the final correction.\
    nCorrect:"
  2. Prompt IDs    : [27281, 27, 309, 1908, 326, 310, 625, 11638, 281, 4446, 247, 1113, 984,
    368, 4459, 327, 625, 1841, 275, 634, 1211, 1113, 685, 22710, 407, 1113, 964, 187, 1672,
    45525, 27, 380, 10618, 2544, 686, 3529, 8, 281, 686, 262, 8, 281, 4993, 43980, 4327, 15,
    380, 10618, 26586, 686, 261, 8, 984, 352, 310, 15279, 1060, 15, 20058, 841, 1407, 953, 4711,
     253, 15045, 6197, 27, 686, 42, 1908, 352, 625, 11638, 281, 4446, 247, 1113, 1157, 984, 368,
     4459, 625, 1841, 275, 634, 1211, 1113, 685, 672, 22710, 407, 1113, 964, 1383, 285, 253,
    5780, 495, 3081, 1407, 953, 3426, 253, 2457, 10618, 15, 187, 47390, 27]
  3. Generated IDs : [309, 1908, 352, 625, 11638, 281, 4446, 247, 1113, 1157, 984, 368, 4459,
    625, 1841, 275, 634, 1211, 1113, 685, 672, 22710, 407, 1113, 964, 0]
  4. Decoded repr  : 'I consider it more convenient to drive a car , because you carry more
    things in your own car than when travelling by car .'
  5. Clean output  : 'I consider it more convenient to drive a car , because you carry more
    things in your own car than when travelling by car .'
  6. Gold reference: 'I consider it more convenient to drive a car , because you carry more
    things in your own car than when travelling by car .'

[Dev 08]
  1. Prompt repr   : "Fix: Also , you 'll meet friendly people who usually ask to you something
     to be friends and change your telephone number .\nExplanation: The correction removes 'to'
     because it is unnecessary here. The correction removes 'something' because it is
    unnecessary here. This selects the preposition that fits the phrase naturally, and the
    remaining 3 additional edits complete the final correction.\nCorrect:"
  2. Prompt IDs    : [27281, 27, 5220, 1157, 368, 686, 620, 2525, 11453, 952, 665, 3798, 1642,
    281, 368, 1633, 281, 320, 3858, 285, 1818, 634, 11265, 1180, 964, 187, 1672, 45525, 27, 380,
     10618, 26586, 686, 936, 8, 984, 352, 310, 15279, 1060, 15, 380, 10618, 26586, 686, 17873,
    8, 984, 352, 310, 15279, 1060, 15, 831, 34899, 253, 638, 3321, 326, 13840, 253, 12616,
    10748, 13, 285, 253, 5780, 495, 3081, 1407, 953, 3426, 253, 2457, 10618, 15, 187, 47390,
    27]
  3. Generated IDs : [5220, 1157, 368, 686, 620, 2525, 11453, 952, 665, 3798, 1642, 368, 1633,
```

```
   281, 320, 3858, 285, 1818, 634, 11265, 1180, 964, 0]
  4. Decoded repr  : "Also , you 'll meet friendly people who usually ask you something to be
   friends and change your telephone number ."
  5. Clean output  : "Also , you 'll meet friendly people who usually ask you something to be
   friends and change your telephone number ."
  6. Gold reference: "Also , you 'll meet friendly people who usually ask you to be friends and
    exchange telephone numbers ."

[Dev 09]
  1. Prompt repr   : "Fix: In my experience when I did n't have a car I used to use the bus to
   go to the school and go back to my house .\nExplanation: The correction adds ',' to fix
   punctuation. The correction removes 'the' because it is unnecessary here. This fixes the
   article or determiner so the noun phrase is complete.\nCorrect:"
  2. Prompt IDs    : [27281, 27, 496, 619, 2793, 672, 309, 858, 295, 626, 452, 247, 1113, 309,
   908, 281, 897, 253, 1685, 281, 564, 281, 253, 2143, 285, 564, 896, 281, 619, 2419, 964, 187,
    1672, 45525, 27, 380, 10618, 11323, 686, 4117, 281, 4993, 17256, 2368, 15, 380, 10618,
   26586, 686, 783, 8, 984, 352, 310, 15279, 1060, 15, 831, 26019, 253, 3929, 390, 1571, 7068,
    594, 253, 28407, 12616, 310, 3426, 15, 187, 47390, 27]
  3. Generated IDs : [1157, 275, 619, 2793, 1157, 672, 309, 858, 295, 626, 452, 247, 1113,
   1157, 309, 908, 281, 897, 253, 1685, 281, 564, 281, 2143, 285, 564, 896, 1728, 964, 0]
  4. Decoded repr  : ", in my experience , when I did n't have a car , I used to use the bus to
    go to school and go back home ."
  5. Clean output  : ", in my experience , when I did n't have a car , I used to use the bus to
    go to school and go back home ."
  6. Gold reference: "In my experience , when I did n't have a car I used to use the bus to go
   to school and go back to my house ."

[Dev 10]
  1. Prompt repr   : "Fix: In my opinion , the car is n't necessary when you have crashed in
   the street , in that moment you realized the importance of a public transport .\
   nExplanation: The correction changes 'the' to 'a' to fix determiner or article use. The
   correction changes ', in' to '. At' to fix phrasing. This fixes the article or determiner
   so the noun phrase is complete, and the remaining 3 additional edits complete the final
   correction.\nCorrect:"
  2. Prompt IDs    : [27281, 27, 496, 619, 4743, 1157, 253, 1113, 310, 295, 626, 3309, 672,
   368, 452, 25753, 275, 253, 6406, 1157, 275, 326, 2774, 368, 8156, 253, 6349, 273, 247, 1345,
    4616, 964, 187, 1672, 45525, 27, 380, 10618, 2544, 686, 783, 8, 281, 686, 66, 8, 281, 4993,
    1571, 7068, 390, 3929, 897, 15, 380, 10618, 2544, 26848, 275, 8, 281, 29996, 2058, 8, 281,
    4993, 9839, 2355, 15, 831, 26019, 253, 3929, 390, 1571, 7068, 594, 253, 28407, 12616, 310,
    3426, 13, 285, 253, 5780, 495, 3081, 1407, 953, 3426, 253, 2457, 10618, 15, 187, 47390,
   27]
  3. Generated IDs : [496, 619, 4743, 1157, 247, 1113, 310, 295, 626, 3309, 672, 368, 452,
   25753, 275, 253, 6406, 964, 2058, 326, 2774, 1157, 368, 8156, 253, 6349, 273, 1345, 4616,
   964, 0]
```

```
4. Decoded repr  : "In my opinion , a car is n’t necessary when you have crashed in the
  street . At that moment , you realized the importance of public transport ."
5. Clean output  : "In my opinion , a car is n’t necessary when you have crashed in the
  street . At that moment , you realized the importance of public transport ."
6. Gold reference: "In my opinion , a car is n’t necessary when you have crashed in the
  street . At that moment , you realize the importance of public transport ."
```