MIXDETECT: WORD-LEVEL LOCALIZATION AND QUANTIFICATION OF AI EDITING

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models are increasingly used to edit human-written text rather than generate entire texts from scratch. Conventional AI-text detectors mainly distinguish human-written from fully AI-generated text, while recent methods for AI-edited text typically provide only a text-level label or editing-degree score. We introduce MIXDETECT, a word-level framework for localizing and quantifying AI editing. MIXDETECT separately predicts whether each word has been edited and, conditional on editing, how substantial the edit is, allowing editing scope and editing intensity to be estimated separately. During training, source–edited pairs are aligned to construct word-level supervision, while inference requires only the input text. Experiments show that MIXDETECT accurately localizes AI-edited words, reflects differences in editing intensity, and reveals different scope–intensity patterns across editing degrees and operations. The overall AI editing magnitude increases under additional AI editing, decreases when AI-generated text is edited by humans, and remains nearly unchanged under ordinary human-to-human editing. The aggregated text-level predictions also perform well on binary and ternary AI-text classification and remain effective under domain and generator shifts. These results show that AI editing can be analyzed beyond a single authorship label or editing-degree score by identifying both where AI editing occurs and how substantial the edits are.

1 INTRODUCTION

The strong text generation capabilities of large language models (LLMs) have made them widely used for editing, polishing, and rewriting human-written text. The degree of editing can range from minor grammar correction and fluency polishing to paraphrasing and substantial rewriting (Thai et al., 2026; Saha & Feizi, 2025). Such AI-assisted writing is common in scientific and academic writing (Salvagno et al., 2023), education (Cotton et al., 2024), professional and workplace writing (Noy & Zhang, 2023), and everyday writing tasks (Chatterji et al., 2025). Conventional AI-text detectors such as DetectGPT (Mitchell et al., 2023), Fast-DetectGPT (Bao et al., 2024), Binoculars (Hans et al., 2024), LastDE++ (Xu et al., 2025), and EchoPrompt (Bao et al., 2026a) are designed to distinguish human-written from AI-generated text. They cannot detect partially AI-edited text or quantify different degrees of AI editing. For AI-edited text, the task is therefore not simply to determine whether a text is human-written or AI-generated, but to measure the degree of AI editing.

Recent studies have extended AI-text detection to AI-edited and human–AI collaborative text. Several works formulate the problem as three-class classification over human-written, AI-generated, and AI-edited or mixed text, including MixSet (Zhang et al., 2024), EditLens (Thai et al., 2026), GPTZero (Adam et al., 2026), and EVIL-Detect (Bao et al., 2026b). More recently, RACE (Li et al., 2026) further distinguishes four classes according to the creator and editor of the text. However, these methods still output a text-level class label or editing score rather than quantify the degree of AI involvement within different parts of the text.

To quantify the degree of AI involvement within a text, we define the *overall AI editing magnitude* as S, as illustrated in Figure 1. The overall AI editing magnitude consists of two components: *editing scope* Q, which measures the proportion of words affected by AI editing, and *editing intensity* M, which measures the average magnitude of the edits among the affected words. Their product gives

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

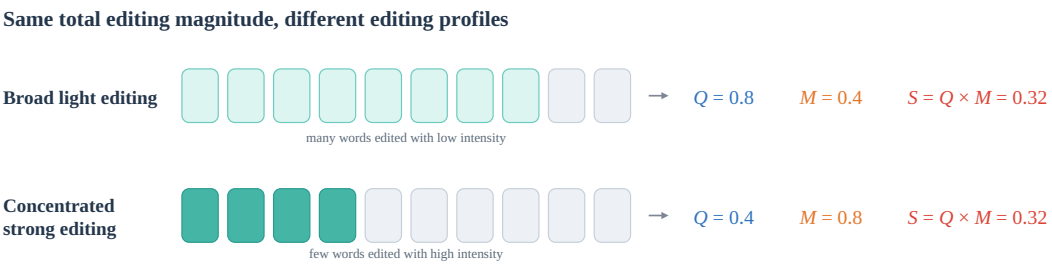

Figure 1: The same overall AI editing magnitude S can arise from different editing patterns. Broad, light editing has high scope Q but low intensity M, whereas concentrated rewriting has lower scope but higher intensity. Word-level predictions preserve this distinction.

the overall AI editing magnitude,

$$S = QM. \tag{1}$$

A text with broad but light editing can have a similar S to a text with fewer but more substantial edits. However, a similar overall AI editing magnitude does not imply the same AI editing behavior, because the two texts can differ substantially in both editing scope and editing intensity. Therefore, quantifying AI editing requires not only measuring its overall magnitude but also characterizing how the editing differs within the text.

To characterize how AI editing varies within a text, we introduce MIXDETECT, a word-level framework for localizing and quantifying AI editing. For each word, MIXDETECT predicts *edit occurrence*, indicating whether the word is affected by AI editing, and *conditional editing intensity*, estimating how substantial the edit is once editing occurs. Their product defines the corresponding word-level AI-editing score. These word-level predictions constitute the primary outputs of MIXDETECT, and, when text-level analysis is needed, they can be aggregated to derive editing scope Q, editing intensity M, and the overall AI editing magnitude S. We train MIXDETECT to estimate edit occurrence and intensity using word-level supervision derived from aligned source–edited pairs. Once trained, the model predicts word-level AI-editing scores from the input text, which can be aggregated for text-level analysis or classification.

Experiments show that MIXDETECT accurately localizes AI-edited words, achieving an AUROC of 0.885, while its predicted intensity increases with edit magnitude. The aggregated overall AI editing magnitude is also highly consistent with EditLens, with Spearman correlations of 0.913 for Edit-Only and 0.939 for Full. The overall AI editing magnitude increases after additional AI editing, decreases when AI-generated text is edited by humans, and remains nearly unchanged under ordinary human-to-human editing. For standard AI-text detection, MIXDETECT achieves a macro-F1 of 0.916 on ternary human-written, AI-edited, and AI-generated text classification and remains effective under domain and generator shifts.

Our contributions are threefold:

- We formulate the detection of AI editing as a word-level problem that separates *where* editing occurs from *how substantial* the edits are, providing separate estimates of editing scope and intensity rather than only a single overall score.
- We introduce source-aligned word-level supervision for learning edit occurrence and conditional intensity from source–edited pairs, while requiring only a single input text at inference time.
- We conduct extensive evaluations across editing degrees, editing operations, and different editing directions, showing that the predictions reflect diverse AI-editing patterns while retaining strong performance on conventional AI-text classification.

2 RELATED WORK

AI-generated text detection. Most AI-text detectors are designed to distinguish human-written from AI-generated text. Existing methods use token likelihood and rank (Gehrmann et al., 2019), perturbation-based curvature (Mitchell et al., 2023), conditional probability curvature (Bao et al., 2024), cross-model perplexity ratios (Hans et al., 2024), local–global probability statistics (Xu et al.,

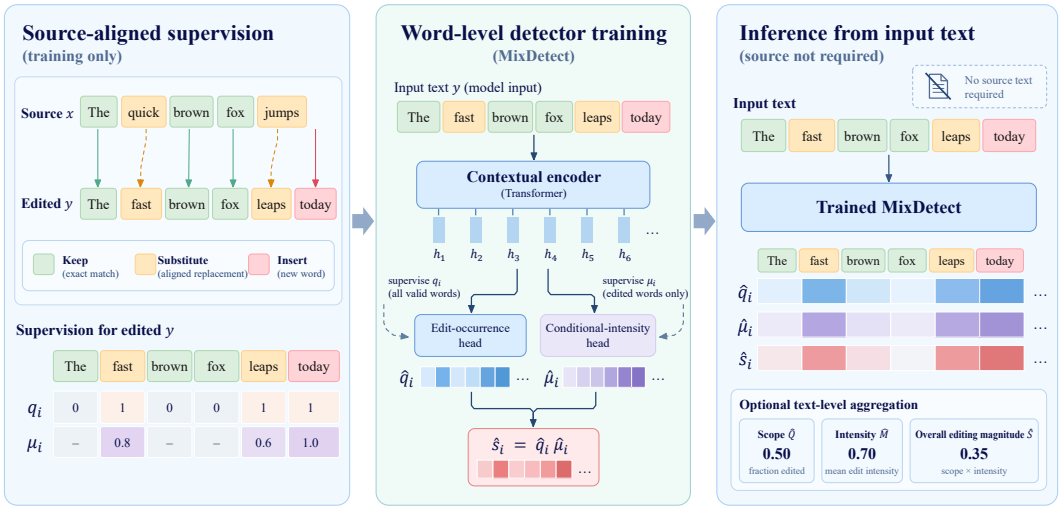

Figure 2: Overview of MIXDETECT. During training, the source–edited pair is aligned to construct word-level supervision for edit occurrence q_i and conditional intensity μ_i. At inference time, the detector receives only the input text and predicts $\hat{q}_i$ and $\hat{\mu}_i$ for each word, with $\hat{s}_i = \hat{q}_i \hat{\mu}_i$.

2025), or latent prompt restoration (Bao et al., 2026a). These methods typically output a detection score or label for the entire text and are not designed for AI-edited text detection or word-level edit localization.

AI-edited and mixed-authorship text. Recent studies extend AI-text detection to AI-edited and mixed human–AI text. SeqXGPT (Wang et al., 2023) and PaLD (Lei et al., 2025) localize human- and AI-generated regions within mixed-authorship text. At the text level, EditLens (Thai et al., 2026) and EVIL-Detect (Bao et al., 2026b) distinguish human-written, AI-generated, and AI-edited text, with EditLens (Thai et al., 2026) additionally estimating a continuous editing-degree score. RACE (Li et al., 2026) further distinguishes four classes according to the creator and editor of the text. These methods focus on authorship localization or text-level classification and editing-degree estimation rather than quantifying the degree of AI editing within different parts of the text.

AI-editing benchmarks. Several recent datasets provide source–edited pairs for studying AI editing. MixSet (Zhang et al., 2024) includes mixed human–AI text under different editing and co-authoring settings. EditLens (Thai et al., 2026) contains human-written source texts together with AI-edited and fully AI-generated variants, supporting evaluation across different degrees of AI involvement. APT-Eval (Saha & Feizi, 2025) varies the requested degree and proportion of AI polishing. BEEMO (Artemova et al., 2025) contains AI responses followed by both model and expert-human edits, while LAMP (Chakrabarty et al., 2025) provides AI-generated texts edited by professional writers together with human-annotated edit regions. These benchmarks cover complementary aspects of AI-assisted editing, including editing degree, editing proportion, editor type, and localized edits.

3 METHOD

Overview. Figure 2 gives an overview of MIXDETECT. MIXDETECT is designed to detect AI editing at the word level. For AI-edited training examples, a source–edited pair (x, y) is aligned to construct word-level supervision for edit occurrence q_i and conditional editing intensity μ_i. The detector takes a single input text and predicts $\hat{q}_i$ and $\hat{\mu}_i$ for each word, with $\hat{s}_i = \hat{q}_i \hat{\mu}_i$ representing the corresponding word-level AI-editing score. These word-level predictions are the primary outputs of MIXDETECT. When text-level analysis is required, they can be aggregated to obtain editing scope, editing intensity, and the overall AI editing magnitude. At inference time, the detector requires only the input text and does not require its source.

3.1 SOURCE-ALIGNED WORD SUPERVISION

Let $x = (u_1, \ldots, u_m)$ denote a human-written source text and $y = (w_1, \ldots, w_n)$ its AI-edited text, where w_i indexes the non-punctuation words considered by the model. For each word w_i, we define an edit-occurrence label $q_i \in \{0, 1\}$ indicating whether the word is affected by AI editing. For an edited word with $q_i = 1$, we further define a conditional editing intensity $\mu_i \in [0, 1]$ that measures the magnitude of the edit relative to its corresponding source content. The word-level AI-editing score is then

$$s_i = q_i \mu_i. \tag{2}$$

Thus, $s_i = 0$ for an unchanged word, while larger values indicate stronger edits at edited words. The conditional intensity μ_i is undefined when $q_i = 0$ and is not supervised for unchanged words.

To construct these labels, we align the source text x with the edited text y. Each word w_i is assigned one of three alignment types: KEEP if it is preserved from the source, SUBSTITUTE if it corresponds to a different source word, and INSERT if it has no aligned source word. AI editing is not strictly word-by-word: words can be inserted, deleted, or replaced, so source and edited words cannot be matched reliably by position alone. Exact lexical matching is also insufficient because an edited word may differ lexically from its source word while preserving similar meaning. We therefore align the source and edited texts in two stages.

First, we identify normalized exact lexical matches between x and y and use contiguous matched spans as anchors. These anchors identify clearly preserved content and divide the remaining unmatched words into smaller intervals. Second, within each interval, we match source and edited words using contextual semantic similarity while preserving their order. If source words u_j and $u_{j'}$ are aligned with edited words w_i and $w_{i'}$, respectively, we require

$$j < j' \quad \Longrightarrow \quad i < i'.$$

This constraint prevents crossing matches and reduces spurious correspondences between distant words.

For a source word u_j and an edited word w_i, we compute their contextual similarity as

$$c_{ji} = \mathrm{clip}\left(\cos(\mathbf{e}_j^x, \mathbf{e}_i^y), 0, 1\right), \tag{3}$$

where $\mathbf{e}_j^x$ and $\mathbf{e}_i^y$ are their contextual word representations. Within each anchor-delimited interval, dynamic programming finds the order-preserving one-to-one alignment with the largest cumulative similarity. The procedure for obtaining contextual word representations and the dynamic-programming alignment algorithm are provided in Appendix A.

The resulting alignment directly determines the word-level supervision:

$$(q_i, \mu_i, s_i) = \begin{cases} (0, -, 0), & w_i \text{ is KEEP}, \\ (1, 1 - c_{ji}, 1 - c_{ji}), & w_i \text{ is SUBSTITUTE}, \\ (1, 1, 1), & w_i \text{ is INSERT}. \end{cases} \tag{4}$$

For a SUBSTITUTE, $1 - c_{ji}$ measures the contextual dissimilarity between the edited word and its aligned source word, so semantically similar substitutions receive lower intensity values and larger changes receive higher values. An INSERT has no aligned source word and is assigned the maximum intensity value $\mu_i = 1$. For a KEEP word, μ_i is undefined and is not included in the intensity loss.

Punctuation is excluded from supervision. Unmatched source words correspond to deletions; because they have no corresponding position in the edited text, they do not receive word-level labels. The complete alignment and label-construction procedure is given in Algorithm 2.

3.2 TRAINING THE WORD-LEVEL DETECTOR

The source-aligned labels provide an edit-occurrence target q_i for each valid word and a conditional editing-intensity target μ_i for edited words. We train a word-level detector to predict both quantities from a single input text.

Given an input text, a contextual encoder produces a word representation $\mathbf{h}_i$ for each word w_i. Two prediction heads operate on the shared representation:

$$\hat{q}_i = \sigma\left(\mathbf{a}_q^\top \mathbf{h}_i + b_q\right), \qquad \hat{\mu}_i = \sigma\left(\mathbf{a}_\mu^\top \mathbf{h}_i + b_\mu\right), \tag{5}$$

where $\hat{q}_i$ predicts edit occurrence and $\hat{\mu}_i$ predicts conditional editing intensity. The predicted word-level AI-editing score is

$$\hat{s}_i = \hat{q}_i \hat{\mu}_i. \tag{6}$$

Training labels. For human-written texts, every valid word is assigned $(q_i, \mu_i, s_i) = (0, -, 0)$. AI-edited texts use the source-aligned labels in Equation 4.

We use **Edit-Only** as the main setting, trained only on human-written and AI-edited texts. Its purpose is to learn the AI-editing signal relative to human-written text. If the model correctly learns this signal, the predicted scores for fully AI-generated text should also be higher. We additionally train a **Full** setting that includes fully AI-generated texts, allowing us to evaluate the effect of these examples on conventional AI-text classification and on fine-grained edit localization and intensity estimation.

Because fully AI-generated texts do not have a corresponding human-written source for source–edited alignment, we assign

$$(q_i, \mu_i, s_i) = (1, 1, 1) \tag{7}$$

to every valid word in these texts. The two settings use the same model architecture and training objective and differ only in the training data.

Conditional training objective. Let $m_i \in \{0, 1\}$ indicate whether w_i is a valid non-punctuation word, and define $m_i^+ = m_i q_i$ so that the intensity loss is applied only to edited words. The model is trained with

$$\mathcal{L} = \frac{\sum_i m_i \ \mathrm{BCE}(\hat{q}_i, q_i) + \sum_i m_i^+ \ \mathrm{BCE}(\hat{\mu}_i, \mu_i)}{\sum_i m_i}. \tag{8}$$

The occurrence loss is computed for every valid word, whereas the intensity loss is computed only where $q_i = 1$. Assigning zero intensity to unchanged words would make the intensity head learn whether an edit occurs in addition to how substantial it is. Masking unchanged words keeps the two prediction targets separate.

Further implementation details are provided in Appendix C.1.

3.3 Inference and Text-Level Aggregation

At inference time, MixDetect takes only the input text and does not require its source or the alignment procedure. For each word w_i, the detector outputs edit occurrence $\hat{q}_i$, conditional editing intensity $\hat{\mu}_i$, and word-level AI-editing score $\hat{s}_i = \hat{q}_i \hat{\mu}_i$. These word-level predictions are the primary outputs of MixDetect and are used to localize and characterize AI editing within the text.

For text-level analysis, the word-level predictions are aggregated into editing scope, editing intensity, and overall AI editing magnitude:

$$\hat{Q} = \frac{1}{n} \sum_{i=1}^{n} \hat{q}_i, \qquad \hat{M} = \frac{\sum_{i=1}^{n} \hat{q}_i \hat{\mu}_i}{\sum_{i=1}^{n} \hat{q}_i}, \qquad \hat{S} = \frac{1}{n} \sum_{i=1}^{n} \hat{q}_i \hat{\mu}_i = \hat{Q} \hat{M}. \tag{9}$$

Here, $\hat{Q}$ denotes the predicted editing scope, $\hat{M}$ denotes the predicted editing intensity, and $\hat{S}$ denotes the predicted overall AI editing magnitude. These text-level predictions are obtained by aggregating the word-level predictions rather than by additional prediction heads. They are used for text-level editing analysis and evaluation, while the primary outputs of MixDetect remain the word-level predictions. Appendix B shows an example of the predicted word-level editing scores.

4 Results

We evaluate MixDetect on both fine-grained AI-editing analysis and standard AI-text detection. The experiments cover word-level edit localization and intensity estimation, different editing degrees and operations, changes across editing directions, as well as text-level classification and ablation studies. Unless otherwise specified, all MixDetect models are trained on the EditLens training split (Thai et al., 2026) and evaluated on each target dataset without further tuning. Additional details on datasets, implementation, baselines, evaluation metrics, and experimental protocols are provided in Appendix C.

Table 1: Word-level localization and text-level evaluation on the EditLens test split. AUROC and AUPRC evaluate edit localization over valid words, while $\mathrm{AP}_{\mathrm{text}}$ computes average precision within each text and then averages across texts. ρ_Q and ρ_M are partial Spearman correlations for editing scope and intensity, respectively, and ρ_S is the Spearman correlation for overall AI editing magnitude.

Training	AUROC	AUPRC	$\mathrm{AP}_{\mathrm{text}}$	ρ_Q	ρ_M	ρ_S
Edit-Only	0.885	**0.835**	0.679	**0.878**	**0.640**	**0.861**
Full	0.885	0.817	**0.682**	0.873	0.617	0.852

4.1 Word-Level Localization and Editing Intensity

We first evaluate whether MixDetect can localize AI-edited words, estimate the intensity of individual edits, and reflect the overall AI editing magnitude through its aggregated word-level predictions consistently with EditLens.

Word-level localization. We compare the predicted edit-occurrence score $\hat{q}_i$ with the word-level labels constructed from source–edited alignment. Table 1 shows that both training settings perform well on edit localization. Edit-Only reaches an AUROC of 0.885 and an AUPRC of 0.835, indicating that the detector can accurately localize edited words from the input text.

Editing intensity. We next examine whether $\hat{\mu}_i$ varies with the magnitude of an edit. We divide Substitute words in the EditLens test split into small, medium, and large changes according to their source–edited similarity and report Insert words separately. Figure 3 shows a clear increasing trend for Edit-Only: the mean predicted intensity rises from 0.587 for small changes to 0.730 for large changes and reaches 0.817 for insertions. Across substituted words, $\hat{\mu}_i$ also correlates with the reference intensity ($\rho = 0.384$). Full follows the same increasing trend. At the text level, the aggregated predictions also correlate strongly with the corresponding reference values, with Edit-Only reaching $\rho_Q = 0.878$, $\rho_M = 0.640$, and $\rho_S = 0.861$ (Table 1).

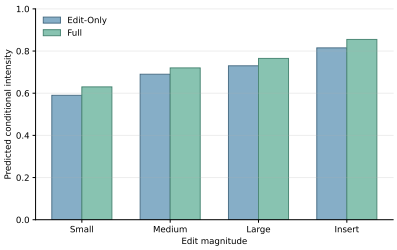

Figure 3: Predicted intensity increases with edit magnitude.

Consistency with EditLens. We also compare the overall AI editing magnitude S predicted by MixDetect with the official EditLens editing-degree score on the same test split. The two measures have Spearman correlations of 0.913 for Edit-Only and 0.939 for Full. Among texts derived from the same source, their pairwise ordering agrees with EditLens in 97.9% of comparisons for Edit-Only and 99.2% for Full. These results show that the overall AI editing magnitude estimated by MixDetect is highly consistent with EditLens, while MixDetect additionally provides word-level edit localization and editing-intensity estimates.

4.2 Human-to-AI Editing

We next evaluate how MixDetect behaves when human-written text undergoes different forms of AI editing. APT-Eval (Saha & Feizi, 2025) provides different requested editing degrees and proportions, while the Grammarly data (Thai et al., 2026) apply different editing operations to the same human sources.

Editing degree and proportion. APT-Eval varies both the requested degree of AI polishing and the requested proportion of the text to be edited. Figure 4 shows that the overall AI editing magnitude S increases with both settings. For Edit-Only, S rises from 0.085 to 0.317 across editing degrees, with a Spearman correlation of 0.578, and from 0.040 to 0.183 as the requested editing proportion increases, with a correlation of 0.507. Full follows the same trends. The separate scope and intensity predictions show that these changes are mainly reflected in editing scope: Q increases steadily, whereas M remains comparatively stable. This indicates that stronger editing requests primarily affect a larger portion of the text rather than uniformly increasing the magnitude of each individual edit. On the editing-degree setting, Edit-Only also achieves a higher Spearman correlation than EditLens (0.578 vs. 0.470).

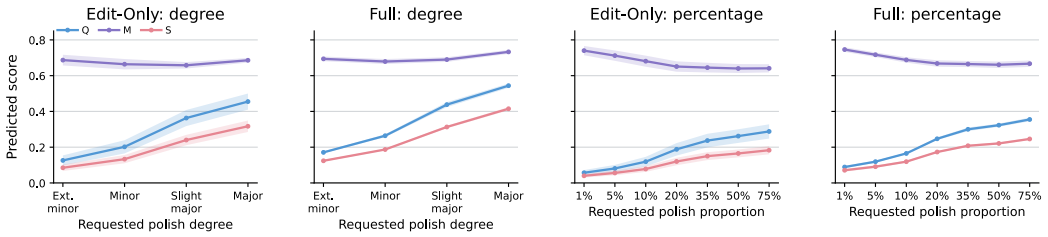

Figure 4: Predicted editing scope, intensity, and overall AI editing magnitude under different requested editing degrees and proportions on APT-Eval.

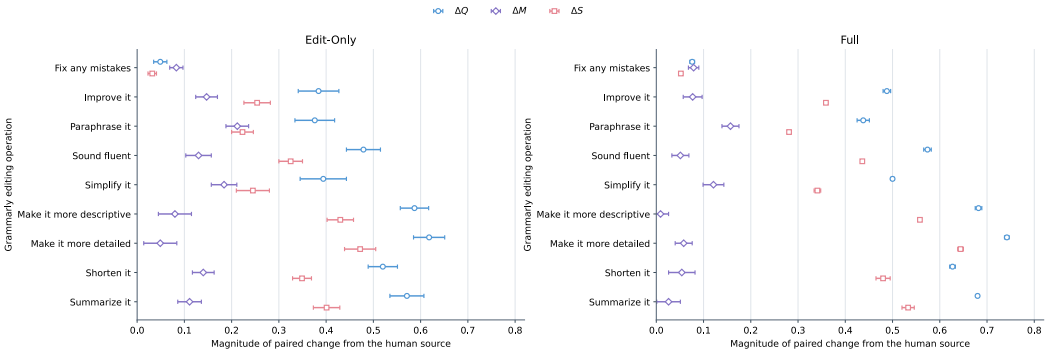

Figure 5: Changes in editing scope, intensity, and overall AI editing magnitude across different Grammarly editing operations. ΔQ, ΔM, and ΔS denote the magnitudes of the paired score changes relative to the corresponding human source.

Different AI editing operations. We further evaluate nine Grammarly editing operations applied independently to the same human sources, covering correction, paraphrasing, rewriting, expansion, shortening, and summarization. Figure 5 reports the magnitudes of the paired changes in Q, M, and S relative to the corresponding human source. The operations produce different scope–intensity patterns. Light correction, such as *Fix any mistakes*, changes all three scores only slightly. Expansion-oriented operations, especially *Make it more detailed* and *Make it more descriptive*, produce the largest changes in Q, consistent with their addition of new content across a larger portion of the text. In contrast, *Paraphrase it* produces a relatively large change in M, reflecting stronger changes among the affected words without the largest increase in scope. The overall AI editing magnitude S correlates with the realized word-level edit distance, with Spearman correlations of 0.719 for Edit-Only and 0.683 for Full, and increases after AI editing in 99.5% of the pairs. The different changes in Q and M across operations further show that similar overall AI editing magnitudes can arise from different combinations of scope and intensity.

4.3 Generalization Across Editing Directions

The previous section focuses on human-to-AI editing, which matches the setting used to construct the source-aligned supervision. We next examine how the overall AI editing magnitude changes in the expected direction under three other settings: additional AI editing of AI-generated text, human editing of AI-generated text, and human-to-human editing.

Table 2: Paired changes in the overall AI editing magnitude S on BEEMO under three AI editing prompts (P1–P3).

Detector	Editor	ΔS (P1)	ΔS (P2)	ΔS (P3)
Edit-Only	Llama-3.1-70B	+0.097	+0.150	+0.158
Edit-Only	GPT-4o	+0.063	+0.118	+0.077
Full	Llama-3.1-70B	+0.048	+0.061	+0.114
Full	GPT-4o	+0.030	+0.073	+0.041

Additional AI editing of AI-generated text. Further AI editing of an already AI-generated text should increase S if the score reflects additional AI editing. BEEMO (Artemova et al., 2025) provides

this setting, where AI responses are further edited by Llama-3.1-70B (Grattafiori et al., 2024) or GPT-4o (OpenAI, 2024) under three editing prompts. As shown in Table 2, $\Delta S > 0$ after additional AI editing in all settings. Thus, S increases under additional AI editing even when the source text is already AI-generated.

Human editing of AI-generated text. Human editing of AI-generated text should reduce the overall AI editing magnitude S. BEEMO contains expert-human edits of AI responses, while LAMP (Chakrabarty et al., 2025) contains AI-generated texts edited by professional writers together with human-annotated edit regions. On BEEMO, the average ΔS is -0.070 for Edit-Only and -0.168 for Full. S decreases in 66.5% of the pairs for Edit-Only and in 70.4% for Full. The trend is stronger on LAMP: the average ΔS is -0.192 for Edit-Only and -0.116 for Full. S decreases in 96.8% of the pairs for Edit-Only and in 78.0% for Full. For Edit-Only, larger human edits are associated with larger decreases in S. The Spearman correlation between ΔS and token edit distance is -0.750. The decrease is also concentrated in the edited regions: the local score decreases by 0.309 inside human-annotated edit regions, compared with 0.106 in unchanged regions. These local score changes distinguish edited from unchanged regions with an AUROC of 0.769. Overall, human editing tends to reduce S.

Human-to-human editing. Finally, we examine whether ordinary human editing changes S. ArgRewrite (Kashefi et al., 2022) contains human Draft1–Draft2 editing pairs together with source-matched AI-edited alternatives. For Edit-Only, the mean S scores of the original and human-edited texts are nearly identical (0.01087 and 0.01089, respectively). In contrast, the source-matched AI-edited texts obtain a mean score of 0.594. In all 86 source families, the AI-edited version receives a higher score than both human versions. Thus, the model does not simply respond to textual change, but distinguishes AI editing from ordinary human editing.

4.4 Standard AI-Text Classification

Although MixDetect is designed for word-level analysis of AI editing, its aggregated text-level predictions can also be used for standard AI-text classification. We evaluate two binary tasks on the EditLens test split and ternary classification on EditLens, Enron, and Unseen Llama. The Any AI task distinguishes human-written text from AI-edited or AI-generated text, while Full AI distinguishes fully AI-generated text from human-written and AI-edited text. The ternary task distinguishes human-written, AI-edited, and fully AI-generated text. We compare against the official EditLens RoBERTa-large model (Thai et al., 2026; Liu et al., 2019) and four representative AI-text detectors: EchoPrompt (Bao et al., 2026a), Fast-DetectGPT (Bao et al., 2024), Binoculars (Hans et al., 2024), and LastDE++ (Xu et al., 2025). Qwen2.5-3B Base and Instruct (Yang et al., 2024) are used as proxy models where required by the corresponding detector. For each method, binary thresholds and ternary threshold pairs are selected on the EditLens validation set to maximize macro-F1 and then fixed for all test and OOD evaluations.

Table 3 summarizes the classification results. On EditLens, Edit-Only achieves the best Any-AI macro-F1 among the compared methods at 0.929. Full performs better on tasks that explicitly distinguish fully AI-generated text, reaching 0.980 on Full AI and 0.916 on ternary classification. This difference is consistent with the training data: Full includes fully AI-generated examples, whereas Edit-Only is trained only on human-written and AI-edited text.

Out-of-distribution generalization. We further evaluate ternary classification under domain and generator shifts. Enron (Klimt & Yang, 2004) introduces an unseen email domain, while Unseen Llama contains text generated by a model family not observed during training. As shown in Table 3, Full reaches macro-F1 scores of 0.865 on Enron and 0.893 on Unseen Llama, outperforming the compared methods on both datasets. Edit-Only achieves 0.730 and 0.828, respectively, despite not using fully AI-generated examples during training.

4.5 Ablation Studies

We examine two design choices of MixDetect: the composition of the training data and the separate prediction of edit occurrence and intensity.

Edit-Only vs. Full. Edit-Only is trained on human-written and AI-edited text, whereas Full additionally includes fully AI-generated text. Edit-Only performs better on fine-grained editing analysis,

Table 3: AI-text classification results on the EditLens test split and under domain and generator shifts. All binary and ternary results are reported with macro-F1.

	Binary Classification		Ternary Classification		
Method	Any AI	Full AI	EditLens	Enron	Unseen Llama
EditLens	0.921	0.943	0.881	0.674	0.860
EchoPrompt	0.722	0.768	0.607	0.660	0.432
Fast-DetectGPT	0.557	0.595	0.414	0.447	0.592
Binoculars	0.566	0.595	0.418	0.468	0.576
LastDE++	0.550	0.540	0.381	0.292	0.382
MIXDETECT (Edit-Only)	**0.929±0.002**	0.906±0.012	0.848±0.011	0.730	0.828
MIXDETECT (Full)	0.928±0.005	**0.980±0.004**	**0.916±0.005**	**0.865**	**0.893**

Table 4: Ablation of the prediction formulation. Word AUROC evaluates word-level edit localization, text-level ρ measures the Spearman correlation with the reference overall AI editing magnitude, and Ternary reports macro-F1 for human-written, AI-edited, and AI-generated classification.

Setting	Prediction	Word AUROC	Text-level ρ	Ternary	Separate Q, M
Edit-Only	Direct-s	**0.945±0.002**	**0.877±0.005**	–	No
	q only	0.885±0.005	0.822±0.009	**0.850±0.013**	No
	μ only	–	0.409±0.077	0.411±0.011	No
	$q\mu$	0.885±0.005	0.861±0.011	0.848±0.011	Yes
Full	Direct-s	**0.945±0.000**	**0.880±0.002**	–	No
	q only	0.885±0.004	0.813±0.004	**0.922±0.003**	No
	μ only	–	0.527±0.006	0.652±0.012	No
	$q\mu$	0.885±0.004	0.852±0.009	0.916±0.005	Yes

with higher correlations for editing intensity on substituted words (0.384 vs. 0.348) and requested editing degree on APT-Eval (0.578 vs. 0.551), as well as better localization of human-edited regions on LAMP (AUROC 0.769 vs. 0.646). In contrast, Full performs better when fully AI-generated text must also be distinguished, reaching a ternary classification macro-F1 of 0.916 compared with 0.848 for Edit-Only and showing stronger OOD classification performance. Overall, Edit-Only is better suited to fine-grained analysis of partial AI editing, while Full provides better coverage when fully AI-generated text is also considered.

Separate prediction of occurrence and intensity. We compare $\hat{s}_i = \hat{q}_i\hat{\mu}_i$ with variants that directly predict a single word-level score, edit occurrence only, or editing intensity only. Table 4 shows that Direct-s achieves the best scalar prediction performance, with a word-level AUROC of 0.945 and a text-level correlation of 0.877 under Edit-Only, but it cannot distinguish where editing occurs from how substantial the edit is. The q-only model provides edit localization but no intensity estimate, while the μ-only model performs substantially worse in estimating the overall AI editing magnitude because it does not model edit occurrence. Predicting q and μ separately therefore provides both editing scope and editing intensity while retaining strong word- and text-level performance.

5 CONCLUSION

We introduced MIXDETECT, a word-level framework for localizing and quantifying AI editing. For each word, MIXDETECT predicts edit occurrence and conditional editing intensity, providing both localized AI-editing scores and, when needed, aggregated text-level predictions of editing scope, intensity, and overall AI editing magnitude. Source–edited pairs are used only to construct word-level supervision during training, while inference requires only a single input text. Experiments show that the model accurately localizes AI-edited words, reflects differences in editing intensity, and produces meaningful changes in scope and intensity across editing degrees, operations, and editing directions; its aggregated predictions also remain effective for standard AI-text classification under domain and generator shifts. Overall, MIXDETECT provides a finer-grained view of AI-edited text by showing not only how much AI editing is present, but also where it occurs and how substantial the edits are.

ETHICS STATEMENT

This work studies the detection and characterization of AI-edited text. Our experiments use existing research datasets and publicly available language models, and do not involve the collection of new personal data or human-subject experiments. The proposed method estimates evidence of AI editing from textual patterns and does not establish the authorship or provenance of a text with certainty. Its predictions may also vary across domains, writing styles, languages, and generation or editing models not represented in the evaluation data. Therefore, MIXDETECT should not be used as the sole basis for high-stakes decisions about authorship, academic misconduct, or other individual-level judgments. We view fine-grained editing estimates as supporting evidence for analysis rather than definitive attribution.

REPRODUCIBILITY STATEMENT

The source-aligned supervision, word-level prediction heads, and training objective are specified in Sections 3.1–3.2, while inference and text-level aggregation are described in Section 3.3. Appendix A gives the complete alignment and label-construction procedure, including contextual word encoding and the monotonic dynamic program. Appendix C documents the model configuration, dataset roles, baselines, evaluation metrics, and classification threshold selection used in the experiments. All reported MIXDETECT results use fixed dataset splits and are averaged over the stated random seeds; validation-selected classification thresholds are frozen before test and out-of-distribution evaluation.

AI USE STATEMENT

In this work, we used generative AI tools to assist with method implementation, translation, dataset cleaning and reformatting, scientific figure creation, and editing the manuscript for readability. We did not use generative AI tools for other research tasks requiring disclosure under the ICLR 2027 policy, including developing research hypotheses or methodology, formulating theoretical or mathematical claims, or interpreting experimental results. All AI-assisted code, data processing, translations, figures, and manuscript edits were manually reviewed and verified against the intended methods, original data, and experimental results. All reported results were obtained from the verified implementation and processed data. We take full responsibility for the final content of this work.

REFERENCES

George Alexandru Adam, Alexander Cui, Edwin Thomas, Emily Napier, Nazar Shmatko, Jacob Schnell, Jacob Junqi Tian, Alekhya Dronavalli, Edward Tian, and Dongwon Lee. GPTZero: Robust detection of LLM-generated texts, 2026. URL https://arxiv.org/abs/2602.13042.

Ekaterina Artemova, Jason S Lucas, Saranya Venkatraman, Jooyoung Lee, Sergei Tilga, Adaku Uchendu, and Vladislav Mikhailov. Beemo: Benchmark of expert-edited machine-generated outputs. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 6992–7018, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.naacl-long.357. URL https://aclanthology.org/2025.naacl-long.357/.

Guangsheng Bao, Yanbin Zhao, Zhiyang Teng, Linyi Yang, and Yue Zhang. Fast-DetectGPT: Efficient zero-shot detection of machine-generated text via conditional probability curvature. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/6b8c6f846c3575e1d1ad496abea28826-Abstract-Conference.html.

Hongrui Bao, Yubing Ren, Yanan Cao, Jinhan You, Fang Fang, and Shi Wang. Once a response, always a response: Detecting LLM-generated text via latent prompt restoration, 2026a. URL https://arxiv.org/abs/2608.05741.

Hongrui Bao, Hangyu Rong, Zhuoshang Wang, Yubing Ren, and Yanan Cao. EVIL-Detect for NLPCC 2026 shared task 6: LLM-generated text detection, 2026b. URL https://arxiv.org/abs/2608.10698.

Tuhin Chakrabarty, Philippe Laban, and Chien-Sheng Wu. Can AI writing be salvaged? mitigating idiosyncrasies and improving human-AI alignment in the writing process through edits. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, pp. 1210:1–1210:33, New York, NY, USA, 2025. Association for Computing Machinery. doi: 10.1145/3706598.3713559. URL https://doi.org/10.1145/3706598.3713559.

Aaron Chatterji, Thomas Cunningham, David J. Deming, Zoe Hitzig, Christopher Ong, Carl Yan Shan, and Kevin Wadman. How people use ChatGPT. Working Paper 34255, National Bureau of Economic Research, September 2025. URL https://www.nber.org/papers/w34255.

Debby R. E. Cotton, Peter A. Cotton, and J. Reuben Shipway. Chatting and cheating: Ensuring academic integrity in the era of ChatGPT. *Innovations in Education and Teaching International*, 61(2):228–239, 2024. doi: 10.1080/14703297.2023.2190148. URL https://doi.org/10.1080/14703297.2023.2190148.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander M. Rush. GLTR: Statistical detection and visualization of generated text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pp. 111–116, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-3019. URL https://aclanthology.org/P19-3019/.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. doi: 10.48550/arXiv.2407.21783. URL https://arxiv.org/abs/2407.21783.

Abhimanyu Hans, Avi Schwarzschild, Valeriia Cherepanova, Hamid Kazemi, Aniruddha Saha, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Spotting LLMs with binoculars: Zero-shot detection of machine-generated text. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 17519–17537. PMLR, 2024. URL https://proceedings.mlr.press/v235/hans24a.html.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. DeBERTaV3: Improving DeBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing. In *International Conference on Learning Representations*, 2023. URL https://proceedings.iclr.cc/paper_files/paper/2023/hash/1b10b3e9c5f7ae964e7ffbcbf4a09f2f-Abstract-Conference.html.

Omid Kashefi, Tazin Afrin, Meghan Dale, Christopher Olshefski, Amanda Godley, Diane Litman, and Rebecca Hwa. ArgRewrite V.2: An annotated argumentative revisions corpus. *Language Resources and Evaluation*, 56(3):881–915, 2022. doi: 10.1007/s10579-021-09567-z. URL https://doi.org/10.1007/s10579-021-09567-z.

Bryan Klimt and Yiming Yang. The Enron corpus: A new dataset for email classification research. In *Machine Learning: ECML 2004*, volume 3201 of *Lecture Notes in Computer Science*, pp. 217–226. Springer, 2004. doi: 10.1007/978-3-540-30115-8_22. URL https://doi.org/10.1007/978-3-540-30115-8_22.

Eric Lei, Hsiang Hsu, and Chun-Fu Chen. PaLD: Detection of text partially written by large language models. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/d799b1d6a5e43546e67e7afdeffc067d-Abstract-Conference.html.

Yang Li, Qiang Sheng, Zhengjia Wang, Yehan Yang, Danding Wang, and Juan Cao. Beyond the final actor: Modeling the dual roles of creator and editor for fine-grained LLM-generated text detection. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5188–5203, San Diego, California, United States, July 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.235. URL https://aclanthology.org/2026.acl-long.235/.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019. doi: 10.48550/arXiv.1907.11692. URL https://arxiv.org/abs/1907.11692.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D. Manning, and Chelsea Finn. DetectGPT: Zero-shot machine-generated text detection using probability curvature. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 24950–24962. PMLR, 2023. URL https://proceedings.mlr.press/v202/mitchell23a.html.

Shakked Noy and Whitney Zhang. Experimental evidence on the productivity effects of generative artificial intelligence. *Science*, 381(6654):187–192, 2023. doi: 10.1126/science.adh2586. URL https://doi.org/10.1126/science.adh2586.

OpenAI. GPT-4o system card, August 2024. URL https://openai.com/index/gpt-4o-system-card/. Published August 8, 2024.

Shoumik Saha and Soheil Feizi. Almost AI, almost human: The challenge of detecting AI-polished writing. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 25414–25431, Vienna, Austria, July 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-acl.1303. URL https://aclanthology.org/2025.findings-acl.1303/.

Michele Salvagno, Fabio Silvio Taccone, and Alberto Giovanni Gerli. Can artificial intelligence help for scientific writing? *Critical Care*, 27(1):75, 2023. doi: 10.1186/s13054-023-04380-2. URL https://doi.org/10.1186/s13054-023-04380-2.

Katherine Thai, Bradley Emi, Elyas Masrour, and Mohit Iyyer. EditLens: Quantifying the extent of AI editing in text. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/c7d936e696f57468b6b80a7b1463178c-Abstract-Conference.html.

Pengyu Wang, Linyang Li, Ke Ren, Botian Jiang, Dong Zhang, and Xipeng Qiu. SeqXGPT: Sentence-level AI-generated text detection. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 1144–1156, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.73. URL https://aclanthology.org/2023.emnlp-main.73/.

Yihuai Xu, Yongwei Wang, Yifei Bi, Huangsen Cao, Zhouhan Lin, Yu Zhao, and Fei Wu. Training-free LLM-generated text detection by mining token probability sequences. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/2fcd27d2806a6d4f616cb4e6084d74f3-Abstract-Conference.html.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. doi: 10.48550/arXiv.2412.15115. URL https://arxiv.org/abs/2412.15115.

Qihui Zhang, Chujie Gao, Dongping Chen, Yue Huang, Yixin Huang, Zhenyang Sun, Shilin Zhang, Weiye Li, Zhengyan Fu, Yao Wan, and Lichao Sun. LLM-as-a-coauthor: Can mixed human-written and machine-generated text be detected? In *Findings of the Association for Computational Linguistics: NAACL 2024*, pp. 409–436, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-naacl.29. URL https://aclanthology.org/2024.findings-naacl.29/.

A SOURCE–EDITED ALIGNMENT DETAILS

This section provides the implementation details of the source–edited alignment used to construct the word-level supervision in Section 3.1. The procedure first identifies exact lexical anchors and then applies monotonic semantic matching within the unmatched intervals between anchors.

A.1 WORD PREPROCESSING AND CONTEXTUAL WORD REPRESENTATIONS

We first segment the source and edited texts into words while retaining their character spans and punctuation information. For lexical matching, each non-punctuation word is normalized by lowercasing and removing boundary punctuation:

$$\nu(w) = \mathrm{StripBoundaryPunct}\big(\mathrm{Lowercase}(w)\big). \tag{10}$$

To compute semantic similarity, we obtain contextual word representations for the source and edited texts independently using the same contextual encoder. Because a word may be split into multiple subword tokens, its contextual word representation is obtained by mean-pooling the hidden states of all subword tokens corresponding to that word:

$$\widetilde{\mathbf{e}}_i = \frac{1}{|\mathcal{T}(i)|} \sum_{k \in \mathcal{T}(i)} \mathbf{z}_k. \tag{11}$$

Here, $\mathcal{T}(i)$ denotes the set of subword tokens corresponding to word i, and $\mathbf{z}_k$ is the hidden state of subword token k.

For texts longer than the encoder context window, we encode the text using overlapping windows. If a word appears in multiple windows, its representations are averaged before computing the final normalized word representation:

$$\mathbf{e}_i = \frac{\frac{1}{|\mathcal{W}(i)|} \sum_{r \in \mathcal{W}(i)} \widetilde{\mathbf{e}}_i^{(r)}}{\left\| \frac{1}{|\mathcal{W}(i)|} \sum_{r \in \mathcal{W}(i)} \widetilde{\mathbf{e}}_i^{(r)} \right\|_2}. \tag{12}$$

Here, $\mathcal{W}(i)$ contains the windows covering word i. The resulting L_2-normalized word representations are used to compute the cosine similarities in Equation 3.

A.2 EXACT LEXICAL ANCHORS

We identify contiguous source–edited spans whose normalized word sequences match exactly. These matched spans are used as lexical anchors and are assigned KEEP. The anchors partition the remaining source and edited words into independent unmatched intervals. Semantic alignment is then performed separately within each interval rather than globally over the complete text.

A.3 MONOTONIC SEMANTIC ALIGNMENT

Consider an unmatched source interval $U = (u_1, \ldots, u_r)$ and edited interval $V = (w_1, \ldots, w_c)$. For each pair (u_i, w_j), we compute the compatibility

$$A_{ij} = \mathrm{clip}\left(\cos(\mathbf{e}_i^x, \mathbf{e}_j^y), 0, 1\right). \tag{13}$$

We then find the maximum-weight one-to-one matching subject to the monotonicity constraint that matched word order is preserved.

Let $D_{i,j}$ denote the maximum cumulative similarity obtainable from the prefixes $(u_1, \ldots, u_i)$ and $(w_1, \ldots, w_j)$. The dynamic program is

$$D_{i,j} = \max\left\{D_{i-1,j}, D_{i,j-1}, D_{i-1,j-1} + A_{ij}\right\}, \tag{14}$$

with $D_{0,j} = D_{i,0} = 0$. The first two transitions leave a source or edited word unmatched, respectively, while the third aligns u_i with w_j. The complete dynamic-programming procedure is summarized in Algorithm 1.

Matched non-identical pairs are treated as SUBSTITUTE, unmatched edited words as INSERT, and unmatched source words as deletions. The latter have no position in the edited text and therefore do not receive word-level supervision.

Algorithm 1 Monotonic Semantic Alignment

Require: Source interval $U = (u_1, \ldots, u_r)$; edited interval $V = (w_1, \ldots, w_c)$; compatibility matrix A
Ensure: Monotonic one-to-one matching $\mathcal{M}$
Initialize $D_{0,j} = D_{i,0} = 0$
for $i = 1, \ldots, r$ **do**
  for $j = 1, \ldots, c$ **do**
    $D_{i,j} \leftarrow \max\{D_{i-1,j}, D_{i,j-1}, D_{i-1,j-1} + A_{ij}\}$
    Store the corresponding backpointer
  end for
end for
Backtrack from (r, c) to recover matched word pairs
Break score ties by preferring more matched pairs, followed by a fixed deterministic operation priority
return $\mathcal{M}$

Algorithm 2 Source-Aligned Word Supervision

Require: Source text $x = (u_1, \ldots, u_m)$; edited text $y = (w_1, \ldots, w_n)$
Ensure: Word-level targets $\{q_i, \mu_i, s_i\}_{i=1}^{n}$
Segment x and y into words and identify punctuation
Normalize non-punctuation words for lexical matching
Encode x and y to obtain contextual word representations
Find contiguous exact lexical matches and use them as anchor blocks
for each edited word w_i in an exact anchor **do**
  Assign $w_i \leftarrow$ KEEP, $q_i \leftarrow 0$, $\mu_i \leftarrow \perp$, $s_i \leftarrow 0$
end for
Partition the unmatched source and edited words into anchor-delimited intervals
for each unmatched source–edited interval (U, V) **do**
  Compute pairwise contextual similarities c_{ji}
  $\mathcal{M} \leftarrow$ MONOTONICALIGN$(U, V, \{c_{ji}\})$
  for each matched pair $(u_j, w_i) \in \mathcal{M}$ **do**
    Assign $w_i \leftarrow$ SUBSTITUTE, $q_i \leftarrow 1$, $\mu_i \leftarrow 1 - c_{ji}$, $s_i \leftarrow 1 - c_{ji}$
  end for
  for each unmatched edited word $w_i \in V$ **do**
    Assign $w_i \leftarrow$ INSERT, $q_i \leftarrow 1$, $\mu_i \leftarrow 1$, $s_i \leftarrow 1$
  end for
end for
Exclude punctuation positions from supervision
Record unmatched source words as deletions without word-level labels
return $\{q_i, \mu_i, s_i\}_{i=1}^{n}$

A.4 Complete Label-Construction Procedure

Algorithm 2 summarizes the complete supervision construction used in the main text.

B Qualitative Example

Figure 6 shows a same-source example from the EditLens test set, illustrating the word-level AI-editing scores predicted by MIXDETECT for human-written, AI-edited, and fully AI-generated text.

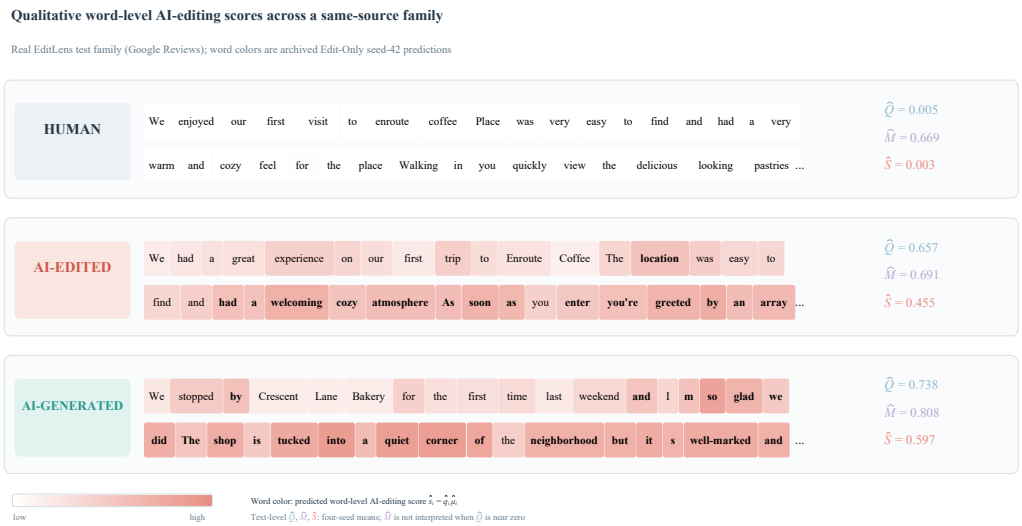

Figure 6: Qualitative word-level example from the EditLens test set. Word colors indicate the predicted word-level AI-editing score $\hat{s}_i = \hat{q}_i\hat{\mu}_i$, with text-level $\hat{Q}$, $\hat{M}$, and $\hat{S}$ shown on the right.

C Experimental Details

C.1 Implementation Details

We implement MixDetect using DeBERTa-v3-base (He et al., 2023) as the shared contextual encoder. When a word is split into multiple subword tokens, we mean-pool their final-layer hidden states to obtain its word representation $\mathbf{h}_i$. The occurrence and intensity predictors are implemented as two independent linear layers followed by sigmoid activations. Edit-Only and Full use the same model architecture and optimization settings and differ only in their training data.

We train the model with AdamW (Loshchilov & Hutter, 2019) for three epochs using a learning rate of 2×10^{-5} and a batch size of 8. The maximum input length is 512 tokens. Texts exceeding this length are processed with overlapping windows using a stride of 64. When a word appears in multiple windows, its predictions are averaged before computing the word-level outputs and text-level predictions.

All main MixDetect results are reported as the mean and standard deviation over four random seeds (42–45). The Direct-s ablation uses the same encoder, data, optimization settings, and training schedule, replacing the two prediction heads with a single sigmoid head trained directly on s_i; its reported results use seeds 42–44.

C.2 Datasets and Evaluation Settings

EditLens. The primary training, validation, and in-distribution test data come from the EditLens dataset (Thai et al., 2026), which contains human-written source texts, AI-edited variants, and fully AI-generated variants. Edit-Only is trained on the human-written and AI-edited examples, while Full additionally includes the fully AI-generated examples described in Section 3.2. The official EditLens training split is used for model fitting, its validation split is used only for threshold selection in text-level classification, and its test split is used for the main localization, text-level evaluation, and classification results. The `test_enron` and `test_llama` splits released with EditLens are used for domain- and generator-shift evaluation, respectively.

APT-Eval. APT-Eval (Saha & Feizi, 2025) provides human-written texts polished by multiple LLMs under requested editing degrees and requested editing proportions. We evaluate the frozen EditLens-trained MixDetect checkpoints directly on these examples without further tuning. For degree-based evaluation, Spearman correlation is computed between the ordered requested degree

and each text-level score. For percentage-based evaluation, the requested editing proportion is treated as an ordered variable and the corresponding score trajectories are reported.

Grammarly. The Grammarly evaluation follows the editing-operation setting released with EditLens (Thai et al., 2026). It contains nine independent Grammarly editing operations applied to the same human sources; the operations are evaluated independently rather than as sequential multi-pass edits. For each source–edited pair, we compute the paired changes in Q, M, and S and compare S with the realized word-level edit distance.

BEEMO. BEEMO (Artemova et al., 2025) contains AI responses followed by both LLM-based and expert-human edits. For AI-to-AI evaluation, each base response is paired with edits produced by Llama-3.1-70B-Instruct (Grattafiori et al., 2024) or GPT-4o (OpenAI, 2024) under the three BEEMO editing prompts P1–P3. For AI-to-human evaluation, the same base responses are paired with the corresponding expert-human edits. All reported changes are paired differences relative to the underlying AI base response.

LAMP. LAMP (Chakrabarty et al., 2025) contains AI-generated passages edited by professional writers and provides human-annotated edit regions. We compare text-level score changes before and after professional editing. For regional evaluation, annotated edit spans are mapped to model words and the score reduction inside those spans is compared with the reduction in background words from the same texts.

ArgRewrite. ArgRewrite (Kashefi et al., 2022) provides human Draft1–Draft2 editing pairs. We use these pairs as a human-to-human editing control and compare them with source-matched AI-edited alternatives in the evaluation construction. The reported 86 source families are those for which the original, human-edited, and source-matched AI-edited versions are all available.

Enron and unseen Llama. The Enron evaluation uses the email-domain split derived from the Enron corpus (Klimt & Yang, 2004) and released as `test_enron` with EditLens. The unseen-generator evaluation uses the EditLens `test_llama` split, whose generated texts come from a Llama model family (Grattafiori et al., 2024) not observed during training. No additional tuning is performed on either split.

C.3 Baselines

We compare against the official EditLens (Thai et al., 2026) RoBERTa-large (Liu et al., 2019) model and four representative AI-generated-text detectors: EchoPrompt (Bao et al., 2026a), Fast-DetectGPT (Bao et al., 2024), Binoculars (Hans et al., 2024), and LastDE++ (Xu et al., 2025). The zero-shot detectors use Qwen2.5-3B Base and Instruct models (Yang et al., 2024) as the proxy models where their scoring rules require base or instruction-tuned language models. Each baseline retains its own tokenizer and scoring rule, and score directions are fixed before test evaluation. We do not convert text-level baseline scores into artificial word-level scores for the localization experiments.

EchoPrompt requires the reconstructed prompt and response to fit within the model context window. When the prompt would be truncated, we exclude that example because the resulting score no longer corresponds to the complete prompt–response construction. Under this criterion, EchoPrompt covers 53.3% of the EditLens test set, 92.2% of Enron, and 56.5% of unseen Llama. Its reported results are computed over these valid subsets.

C.4 Evaluation Metrics

Word-level AUROC and AUPRC pool all valid non-punctuation words. $\mathrm{AP}_{\mathrm{text}}$ computes average precision independently within each text and then averages across texts, preventing longer texts from dominating the metric. For text-level evaluation, ρ_Q and ρ_M are partial Spearman correlations with the alignment-derived Q and M, respectively, while controlling for the other quantity. ρ_S is the Spearman correlation between the predicted and reference overall AI editing magnitude.

For paired editing-direction experiments, we report the mean paired score change and the fraction of pairs moving in the expected direction. APT-Eval additionally reports Spearman correlation with ordered requested editing degree or proportion. For LAMP regional evaluation, word-level score reduction is used to distinguish human-annotated edited regions from background words, reported with AUROC (and, where applicable, AUPRC).

C.5 Classification Protocol

We evaluate two binary tasks and one ternary task using the text-level scores. The *Any AI* task distinguishes human-written text from the union of AI-edited and fully AI-generated text. The *Full AI* task distinguishes fully AI-generated text from the union of human-written and AI-edited text. For each detector and each binary task, a decision threshold is selected on the EditLens validation split to maximize macro-F1.

For ternary classification, two ordered thresholds partition the scalar score into human-written, AI-edited, and fully AI-generated regions. The threshold pair is selected on the EditLens validation split to maximize ternary macro-F1. All selected thresholds are then frozen and applied unchanged to the EditLens test split, Enron, and unseen Llama evaluations.