COM S 5790 Final Project: Evaluating Note-Taking Quality Using Rule-Based, BERT, and LLM Models

Benjamin Jia Zhiang Kam **Mason Inman** **Jesus Soto Gonzalez**

Department of Computer Science
Iowa State University

bkam@iastate.edu, mjinman@iastate.edu, jhsoto@iastate.edu

Abstract

We frame IdeaUnit coverage in student notes as binary classification and compare a lexical rule-based baseline, a BERT sequence-pair classifier with a novel temporal alignment that extracts IdeaUnit-specific evidence to reduce truncation bias, and a prompted LLaMA 3 model. The rule-based method performs best overall (F1 0.75), BERT attains the highest recall with strong ranking quality (F1 0.69), and the LLM lags despite one-shot prompting (F1 0.54), highlighting that the task rewards lexical overlap while still benefiting from semantic modeling and task structure.

1 Introduction

Note-taking is a central learning strategy because it helps students identify, organize, and retain the main ideas presented in a lecture. However, students vary widely in what they capture in their notes, making it difficult for instructors to assess what information is being understood or overlooked. An automatic method to detect whether student-written notes contain key ideas from a lecture could support learning research and provide useful feedback to educators.

The data set we study contains, for each lecture topic, a set of instructor-defined IdeaUnits and a set of student notes divided into segments. The task is to determine whether a specific IdeaUnit appears in a corresponding segment of a student's notes. This setting is challenging because each topic includes only one labeled example, which creates a highly limited and noisy supervision signal.

Our work evaluates how different NLP approaches behave under this low-resource condition. The goal is to understand what types of modeling strategies are most effective in identifying whether a student has captured a key idea. In the following sections, we formally define the task and then describe three modeling approaches that reflect different assumptions about how idea coverage can be detected.

2 Task Definition

Given a lecture topic t, we are provided with a set of instructor-defined `IdeaUnits` $U_t = \{u_1, \ldots, u_m\}$ and a set of student note segments $S_t = \{s_1, \ldots, s_n\}$. Each training instance consists of a pair (u, s) with $u \in U_t$ and $s \in S_t$, along with a binary label $y \in \{0, 1\}$ indicating whether s expresses the idea described by u.

Our goal is to learn a function

$$f_\theta(u, s) \to y,$$

where θ denotes model parameters or decision rules. For probabilistic models, we estimate

$$p_\theta(y = 1 \mid u, s)$$

and predict

$$\hat{y} = \arg\max_{y \in \{0,1\}} p_\theta(y \mid u, s).$$

In rule-based approaches, a similarity score between the texts is computed as

$$\text{Score}(u, s),$$

and the prediction is given by

$$f(u, s) = \begin{cases} 1 & \text{if Score}(u, s) \geq \tau, \\ 0 & \text{otherwise.} \end{cases}$$

Because our learning setup uses only one labeled example per topic, the supervision signal is extremely limited, and the model must generalize from sparse and noisy student-written text.

3 Methodology

We frame the task as a binary decision function

$$f_\theta(u, s) \to y,$$

where u is an instructor-defined IdeaUnit, s is a student note segment, and $y \in \{0, 1\}$ indicates whether the idea is present. We compare three modeling approaches that embody different assumptions about how idea coverage can be detected in short, noisy student notes. The rule-based model relies on lexical similarity, the BERT classifier learns a parametric mapping from a single labeled example per topic, and the LLM model uses instruction-following with zero-shot and one-shot prompting. The subsections below summarize the design principles of each approach.

3.1 Rule-Based Model Methodology

We implement a relatively simple rule-based approach using **Fuzzy Ratio**. It is a string similarity metric derived from `Levenshtein Distance (LD)`. For this approach, the model utilizes the provided reference answer as the standard for scoring.

The dataset that the project works with is lexically short, consisting of few words or even single phrases. As a result of fewer tokens, each token carries high semantic weight, making character-level similarity a good perspective for correctness. This model is well-suited for short and 'more objective than subjective' datasets.

Normalized similarity Our model is based on the idea of LD, which measures the minimum number of single-character edits required to transform a string s_1 into another s_2. These edits consist of insertion, deletion and substitution.

Although LD operates at the character level, our model's token-sorted fuzzy ratio first tokenizes and alphabetically sorts words before computing character-level edit distance. This preprocessing step reduces sensitivity to word order while still penalizing missing, extra, or misspelled tokens. As a result, the method is well-suited for short-answer grading tasks where lexical overlap is more important than syntactic structure. We still measure character edits but only after neutralizing the effect of word order.

To ensure that the distance is comparable across strings of different lengths, we normalize LD into a similarity score:

Let a response string be R and the reference answer C. Tokenize each string into word sets and sort alphabetically:

$$\text{tokens}(R) = [r_1, \ldots, r_m] \xrightarrow{\text{sort}} R_s = r_{(1)}, \ldots, r_{(m)}$$

$$\text{tokens}(C) = [c_1, \ldots, c_n] \xrightarrow{\text{sort}} C_s = c_{(1)}, \ldots, c_{(n)}$$

$$\text{FuzzyRatio}(R, C) = 1 - \frac{LD(R_s, C_s)}{\max(|R_s|, |C_s|)}$$

where:

- $|R_s|$ and $|C_s|$ are the number of characters in the sorted strings,
- $LD(R_s, C_s)$ is the minimum number of single-character insertions, deletions, and substitutions required to transform R_s into C_s,
- The resulting $\text{FuzzyRatio}(R, C) \in [0, 1]$.

The denominator ensures that fuzzy ratio is within $[0, 1]$.

Classification Based on the fuzzy ratio score, each question is labeled according to empirically chosen thresholds. Thresholds can be selected using validation data to achieve individual goals for the model, to maximize F1-score and balance precision and recall.

Let τ_1 and τ_2 be empirically chosen similarity thresholds, where $0 \leq \tau_2 < \tau_1 \leq 1$. $C =$ Correct, $PC =$ Partially Correct, and $I =$ Incorrect. Given a response R, reference answer C, and the computed token-sorted fuzzy ratio $\text{FuzzyRatio}(R, C)$, we classify responses as follows:

$$\text{f}(R, C) = \begin{cases} 1, & \text{if } \text{f}(R, C) \geq \tau_1 \text{(C)} \\ 0, & \text{if } \tau_2 \leq \text{f}(R, C) < \tau_1 \text{(PC)} \\ -1, & \text{if } \text{f}(R, C) < \tau_2 \quad \text{(I)} \end{cases}$$

Here, τ_1 and τ_2 can be tuned using validation data to maximize desired metrics such as precision, recall, and F1-score.

Assumptions With this model, we are required to make several assumptions. Firstly, due to the nature of the project, we expect the responses to be short, such that the tokens are more greatly weighed allowing this model to work well. Secondly, we expect correct responses not to stray too far from the reference answer as the model is based on lexical overlap, not semantic similarity. Lastly, we assume that reference answers are available, allowing the direct comparison method by the model to work.

Limitations Due to a sole reliance on lexical overlap, the model misses out on the benefits that comes from semantic inference. The model is strict and rigid, where the correct idea must be expressed using lexically similar wording. As the model is built to perform well on short-stringed datasets, it cannot scale to longer stringed datasets. Additionally, it is restricted as it cannot be replicated on datsets that do not contain a reference answer as a scoring standard, making reference-free grading impossible.

3.2 BERT-Based Classifier with Temporal Alignment Methodology

We fine-tune `bert-base-uncased` (?) as a binary sequence-pair classifier. For each example, we encode an instructor IdeaUnit u and a note span x as `[CLS]` u `[SEP]` x `[SEP]` and use the final `[CLS]` representation with a linear head to estimate $p_\theta(y = 1 \mid u, x)$.

A standard BERT pair model must truncate long notes to `max_length`, which disproportionately removes evidence from later parts of a lecture. This creates a systematic failure mode: IdeaUnits discussed near the end of a lecture may be unobservable to the classifier if only the first `max_length` tokens of the notes are used. Our temporal alignment mechanism addresses this by selecting an IdeaUnit-specific span from the full note stream before truncation.

Note stream representation. For each student and topic, we form a single note stream by concatenating all available note segments:

$$N = s_1 \| s_2 \| s_3 \| s_4.$$

We tokenize N into L subword tokens $(w_0, \ldots, w_{L-1})$ (without special tokens), treating token position as a proxy for time within the lecture.

Temporal token alignment. Within a topic t, let $U_t = (u_0, \ldots, u_{K-1})$ be the ordered list of distinct IdeaUnits, where $K = |U_t|$. We approximate this lecture order by the first appearance of each IdeaUnit in the dataset for that topic. For an IdeaUnit u_j, we assign a normalized lecture interval

$$I_j = \left[\frac{j}{K}, \frac{j+1}{K}\right),$$

and expand it by a margin m to obtain

$$I'_j = \left[\max\left(0, \frac{j}{K} - m\right),\ \min\left(1, \frac{j+1}{K} + m\right)\right].$$

We map I'_j to token indices in the note stream:

$$a = \lfloor \alpha L \rfloor, \quad b = \lfloor \beta L \rfloor,$$

where $[\alpha, \beta] = I'_j$, and extract the aligned evidence span

$$x = \mathrm{decode}(w_a, \ldots, w_{b-1}).$$

Finally, we cap the span length so that the combined sequence `[CLS]` u `[SEP]` x `[SEP]` fits within `max_length`. Unlike naive truncation, this procedure ensures that IdeaUnits assigned to late lecture intervals are paired with note evidence from late portions of the notes, rather than being systematically dropped.

If an IdeaUnit is missing from the ordering map, we fall back to a coarse alignment based on the segment index (1–4), centering the interval at $(s - 0.5)/4$.

Training protocol and thresholding. By default, we perform a random 80/20 split of `train.csv` into train/validation; optionally, we evaluate a one-shot-per-topic regime by selecting exactly one labeled example per (*Experiment*, *Topic*) group. After fine-tuning, we tune a decision threshold τ on the validation set to maximize F1 and predict

$$\hat{y}_{\text{bert}} = \begin{cases} 1 & \text{if } p_\theta(y = 1 \mid u, x) \geq \tau, \\ 0 & \text{otherwise.} \end{cases}$$

3.3 LLM-Based Model Methodology

We treat idea coverage as a prompt-based binary classification task using the instruction-tuned LLaMA 3 8B model (?). For each IdeaUnit u and note segment s, the model receives a natural language prompt describing the task and must decide whether the idea is expressed.

Next-token classification. Instead of generating free-form text, we compare the model's next-token logits for the verbalizers `" YES"` and `" NO"` (**?**). The predicted label is

$$\hat{y} = \arg \max_{y \in \{\texttt{"YES"}, \texttt{"NO"}\}} \log p_\theta(y \mid P(u, s)),$$

which avoids output-parsing errors and yields a deterministic classifier without fine-tuning.

Prompt design. Student notes contain misspellings, abbreviations, and partial phrases. The zero-shot prompt therefore instructs the model to accept paraphrases and clear abbreviations, but to reject vague keyword matches or incomplete hints. This is intended to push the model toward semantic rather than lexical matching.

One-shot prompting. In the one-shot setting, the prompt additionally includes a single labeled example from the same lecture topic. To select this example, we compute a simple lexical-overlap score between each training pair (u, s) (**?**) and choose the positive example with the highest alignment. This provides the model with a clean demonstration of what counts as full idea coverage in that topic.

Assumptions. This approach is based on three main assumptions. First, an instruction-tuned LLM can evaluate semantics and match key lecture ideas and noisy student notes. Second, the model's next-token probabilities can be used to estimate $p(y = 1 \mid u, s)$. Third, a single carefully chosen example in the prompt can calibrate the model in a low-resource, per-topic setting.

Overall, the LLM method relies entirely on prompting and requires no gradient-based training, offering a flexible contrast to the rule-based and BERT-based approaches.

4 Experiments

4.1 Dataset

The dataset contains instructor-defined IdeaUnits and student note segments across multiple lecture topics. For each topic, instructors provide a set of short conceptual statements (*IdeaUnits*), and each student contributes up to four segmented notes. A labeled training instance pairs one IdeaUnit with the corresponding student note segment for that topic, and the label indicates whether the idea is expressed in the note.

The `Notes.csv` file contains 1,643 student–topic entries, each with up to four note segments. The training split provides 255 labeled IdeaUnit–segment pairs (124 positive and 131 negative), with at most one labeled example per topic in our one-shot setting. The test split contains 9,945 evaluation pairs, which remain unlabeled during training. The data is moderately imbalanced, and the student-written notes vary widely in length and quality, making the classification task challenging.

4.2 Evaluation Metrics

We evaluate all models using standard metrics for binary classification: accuracy, precision, recall, and F1-score. Accuracy measures overall correctness, while precision and recall capture performance on the positive class, which is important given the moderate class imbalance in the dataset. The F1-score summarizes precision and recall into a single harmonic mean.

Formally, let TP, FP, and FN denote true positives, false positives, and false negatives. Then

$$\text{Precision} = \frac{TP}{TP + FP}, \qquad \text{Recall} = \frac{TP}{TP + FN},$$

$$\text{F1} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}.$$

4.3 Experimental Setup

Rule-Based Model Experiments To determine the optimal thresholds, we ran a series of iterations over

BERT-Based Classifier Experiments We fine-tune `bert-base-uncased` as a sequence-pair classifier on temporally aligned note chunks (alignment margin $m = 0.3$). By default we use a seeded random 80/20 train/validation split of `train.csv` (batch size 8, max length 256, learning rate 2×10^{-5}, 4 epochs). We tune a fixed probability threshold τ on the validation split to maximize F1 and then apply the trained model and τ to generate predictions for `test.csv` pairs constructed using `Notes.csv`. We report accuracy, precision, recall, F1, ROC-AUC, and PR-AUC. ROC-AUC and PR-AUC not reported for rule and LLM based models, used for analysis of BERT model performance.

LLM Setup. The LLM experiments use LLaMA 3 8B Instruct in a prompt-based classification setting without fine-tuning. Each example is converted into either a zero-shot or one-shot prompt. The prediction is obtained from the model's next-token logits for the verbalizers "YES" and "NO". For one-shot prompting, the in-context example for each topic is selected from the training set by maximizing a lexical-overlap score between the IdeaUnit and its corresponding note segment. All prompts are truncated to a maximum length of 1024 tokens.

5 Results and Discussion

5.1 Main Results

Model	Accuracy	Precision	Recall	F1
Rule-Based	**0.81**	**0.93**	0.60	**0.75**
BERT-Based	0.72	0.72	**0.66**	0.69
LLM-Based	0.65	0.51	0.58	0.54

Table 1: Quantitative results of Rule, BERT, and LLM based models.

5.2 Analysis of Results

The rule-based model attains very high precision (0.93), indicating that many positive cases can be detected via strong lexical overlap between IdeaUnits and student notes; however, recall is lower (0.60) because paraphrases and fragmented mentions are missed. In contrast, BERT increases recall (0.66) with comparable precision (0.72), consistent with improved semantic matching. Our temporal alignment further supports this by extracting an IdeaUnit-specific evidence span from the full note stream, reducing the truncation bias that would otherwise drop late-lecture content under a fixed 256-token limit. The LLM underperforms (precision 0.51, F1 0.54) because predictions depend on prompt interpretation, which is brittle for short, noisy note segments; adding a topic-specific example (one-shot) improves calibration relative to zero-shot but remains weaker than supervised adaptation. Although the prompts explicitly account for misspellings, paraphrases, and incomplete phrasing, the short and noisy note segments still limit how much reliable semantic signal the model can extract.

5.3 Comparative Discussion

Overall, the results suggest this dataset strongly rewards surface-form similarity, explaining why a lexical rule-based baseline can outperform learned models on F1. By learning more complex features in the embedded space, we see an overall drop in performance at correctly classifiying IdeaUnits. At the same time, the gap in recall highlights that semantic generalization matters for a subset of examples, where BERT is more effective. The LLM's weaker performance indicates that prompt-only reasoning is not robust enough for short, incomplete notes in this low-resource setting, even when augmented with a single in-topic demonstration.

The LLM underperforms because it cannot rely on strong lexical cues like the rule-based model and does not benefit from supervised adaptation like BERT. Its predictions depend entirely on prompt interpretation, which remains difficult for brief or incomplete notes despite the use of noise-aware prompting. One-shot prompting improves calibration by providing an in-topic example, but this is not sufficient to overcome the structural limitations of the data.

6 Conclusion

We studied automatic detection of IdeaUnit coverage in student notes under a low-resource setting with short, noisy text. A lexical rule-based baseline achieved the strongest overall performance, indicating that this dataset often rewards surface-form overlap between instructor phrasing and student notes. Our BERT sequence-pair classifier with temporal token alignment improved semantic coverage detection and achieved the highest recall. The prompted LLM approach was the weakest overall: one-shot prompting improved calibration relative to zero-shot, but prompt-only inference remained brittle on brief, incomplete note segments. Overall, the results suggest that effective note-quality assessment benefits from combining dataset-matched lexical cues with semantic generalization, and that incorporating task structure (lecture order) can mitigate context-length limitations.

7 Limitations

First, the labeled data are small and noisy, and our validation split is limited, which can introduce variance in both training outcomes and threshold selection. Second, our temporal alignment assumes that note-taking follows lecture order; when students reorder ideas or omit transitions, the aligned span may miss relevant evidence, or "lecture recaps" at the end. Third, approximating IdeaUnit order from dataset appearance may not perfectly match true lecture order and can affect alignment quality. Fourth, all methods treat the task as independent binary decisions per (IdeaUnit, segment) pair and do not model dependencies across IdeaUnits or segments (e.g., repeated mentions or cross-segment aggregation). Finally, the LLM results are sensitive to prompt design and example selection strategy, and stronger performance may require more extensive prompt tuning or lightweight adaptation beyond the scope of this project.

Appendix

A LLM Results

Setting	Split	Accuracy	Precision	Recall	F1
Zero-shot	Train	0.51	0.50	0.93	0.65
One-shot	Train	0.56	0.56	0.48	0.51
Zero-shot	Test	0.40	0.37	0.96	0.54
One-shot	Test	0.65	0.51	0.58	0.54

Table 2: LLM-based classification results for zero-shot and one-shot prompting on the train and test splits.

L. Yujian and L. Bo, "A Normalized Levenshtein Distance Metric," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 6, pp. 1091-1095, June 2007, doi: 10.1109/TPAMI.2007.1078.

References