Diagnosing Fine-Grained Inconsistency Classification in Financial Disclosure Text

Anonymous ACL submission

Abstract

Financial disclosures contain numerical claims, temporal statements, entity references, policy commitments, and risk descriptions that may conflict in qualitatively different ways. Detecting that a conflict exists is only the first step: practical review workflows also need to identify what kind of inconsistency is present, since numerical, temporal, referential, factual, and normative conflicts require different evidence and downstream checks. We study this diagnostic task as fine-grained inconsistency classification. Using SBID-FD, a synthetic benchmark of financial disclosure passages with annotated inconsistency types and paired conflicting evidence spans, we compare frozen embedding classifiers, fine-tuned encoders, evidence-augmented classifiers, prompted large language models, and LoRA-adapted generative models under a shared 11-class evaluation setting. The results show that supervised task adaptation is a stronger performance lever than model scale among the systems tested: a fine-tuned 300M encoder reaches 61.9% accuracy, competitive with a LoRA-adapted Qwen3.5-9B model (61.5%) and GPT-5.4 (61.3%). Conflict localization provides an additional but only partially realized benefit: a gold-span oracle improves the fine-tuned encoder to 65.3%, while predicted spans recover only part of this gain. Per-class and confusion analyses further show that some categories, such as Referential, are primarily localization-limited, whereas others, such as Factual and Logical, remain difficult even with gold evidence. These findings suggest that compact task-adapted classifiers can provide an efficient diagnostic layer for financial inconsistency review, but that progress requires both stronger span extraction and better reasoning over closely related inconsistency types.

1 Introduction

Financial disclosures combine quantitative statements, temporal claims, entity references, policy commitments, legal qualifications, risk descriptions, and forward-looking projections. These statements are often produced and revised across different teams, data sources, and reporting cycles, creating opportunities for conflicts that differ substantially in form and in the reasoning required to diagnose them. A numerical inconsistency may require checking arithmetic or tabular aggregation; a temporal inconsistency may require comparing reporting periods or event sequences; a referential inconsistency may require tracking entities across clauses; and a normative inconsistency may require determining whether stated obligations or policies are mutually compatible. Inconsistency in this setting is not a single homogeneous phenomenon: different types depend on different evidence, reasoning patterns, and downstream remediation steps.

Financial text presents domain-specific challenges that generic models struggle to handle. Its lexical and semantic behavior diverges from general usage: general-purpose sentiment dictionaries can misclassify common financial terms in 10-K filings (Loughran and McDonald, 2011), and domain-specific pretrained models have been proposed for financial text mining (Liu et al., 2020). Financial question-answering benchmarks further show that understanding financial reports often requires reasoning over heterogeneous textual and tabular evidence, including numerical operations and multi-step derivations (Chen et al., 2021; Zhu et al., 2021). These studies motivate financial disclosures as a domain in which generic language understanding is insufficient, but they target sentiment analysis, representation learning, or question answering rather than fine-grained inconsistency typing.

Prior work has studied inconsistency and contradiction primarily through natural language inference (NLI), fact verification, and evidence-grounded prediction. Large-scale NLI benchmarks such as SNLI, MultiNLI, and ANLI frame con-

tradiction as a relation between a premise and a hypothesis, assigning one of entailment, contradiction, or neutral (Bowman et al., 2015; Williams et al., 2018; Nie et al., 2020). Fact-verification benchmarks such as FEVER instead ask whether a claim is supported, refuted, or unverifiable with respect to evidence (Thorne et al., 2018), while rationale-based evaluation work such as ERASER studies whether model predictions can be grounded in supporting text spans (DeYoung et al., 2020). More recent resources extend inference beyond sentence pairs: ContractNLI applies document-level inference and evidence identification to contracts (Koreeda and Manning, 2021), and ContraDoc studies self-contradictions within long documents (Li et al., 2024). These lines of work establish contradiction detection, claim verification, and evidence grounding as central problems, but they do not directly address the complementary question studied here: once a passage is known to contain an inconsistency, can a model identify which specific *type* it contains?

We refer to this task as *fine-grained inconsistency classification*. It is useful in two ways. Diagnostically, it surfaces which inconsistency types current models handle reliably and which remain difficult. Operationally, it can serve as a routing step in document-auditing workflows, where numerical, temporal, referential, normative, and other conflicts may require different downstream checks. As a controlled testbed we use SBID-FD (Anonymous, 2026), a synthetic benchmark of financial disclosure passages with one injected inconsistency each.

This paper makes four contributions.

1. **Systematic model comparison.** We systematically compare frozen embedding classifiers, fine-tuned encoders, evidence-augmented classifiers, prompted LLMs, and LoRA-adapted generative models under a shared 11-class evaluation setting.
2. **Effect of task-specific adaptation.** We show that task-specific adaptation of a small open encoder is a stronger performance lever than parameter count among the tested systems, matching much larger prompted and LoRA-adapted generative models.
3. **Gold-span oracle and localization analysis.** We introduce a gold-span oracle and control analyses to isolate the value of conflict localization from span-extraction quality, showing that correct conflicting spans help substantially but predicted spans recover only part of the oracle gain.
4. **Per-class diagnostic analysis.** We provide per-class and confusion diagnostics that separate localization-limited categories from knowledge- or reasoning-limited ones, offering concrete directions for improving inconsistency classifiers.

2 Task and Data

Task definition. Given an inconsistent input text T_j, the task is to predict an inconsistency label $\hat{c}_j$ from 11 predefined categories. The prediction is correct if $\hat{c}_j = c_j$, the ground-truth label.

Data. We use the SBID-FD benchmark (Anonymous, 2026), a synthetic dataset of financial disclosure passages in which each instance contains exactly one injected inconsistency. The benchmark covers 11 inconsistency types (Table 1), ranging from Numerical (arithmetic conflicts) to Referential (inconsistent entity naming) and Normative & Policy (conflicting compliance rules). Each instance includes the inconsistent passage, the inconsistency label, a reference explanation, and annotated evidence spans identifying the two conflicting claims. The full construction methodology, taxonomy rationale, and validation procedure are described in the companion paper (Anonymous, 2026); briefly, each instance begins from a financial-disclosure-style base passage, one localized inconsistency is injected according to the taxonomy label, and the passage is annotated with the label, an explanation, and two conflicting evidence spans. The present paper uses a fixed snapshot of 5,940 instances and makes no claim to the benchmark as a contribution.

We adopt a stratified 70–30 train-test split, yielding 4,158 training and 1,782 test examples.[1] The label distribution is moderately imbalanced, ranging from 109 instances (Referential) to 241 (Logical) in the test set; full per-class counts are given in Appendix Table 6. The inconsistent passages are short excerpts rather than full documents (median length about 87 words; the pre-injection passages, i.e. the original text before the inconsistency was introduced, have a median of about 78 words), and

[1] We release the exact train-test split (instance identifiers) used in this paper to support reproducibility; a link to the full benchmark will be added in the camera-ready version. The split is anonymized for review.

each instance is annotated with two conflicting evidence spans. To confirm that inconsistencies are introduced as localized edits rather than wholesale rewrites, we measure surface-form overlap between each original passage and its inconsistent variant: the median character-level edit (Levenshtein) distance is 72, and the longest common contiguous substring covers about 73% of the shorter passage at the median. The conflict is therefore a small, localizable part of an otherwise unchanged disclosure, which is what makes localization a meaningful lever to study. A worked example and full length and surface-overlap statistics are given in Appendix A.

Table 1: The 11 inconsistency types in SBID-FD, organized by theme.

Theme	Types
Structural & Quantitative	Numerical, Unit & Measurement, Temporal
Truth & Reasoning	Factual, Logical, Theoretical
Semantic & Referential	Referential, Terminological, Specificity & Scope
Procedural & Normative	Pragmatic, Normative & Policy

3 Methods

We compare two families of approaches. The first is traditional embedding-based classification: we embed the passage (and, optionally, predicted conflicting spans) and predict the inconsistency type with a small multilayer perceptron (MLP) head (Figure 1); this family covers the passage classifiers and the evidence-augmented classifiers below. The second is generative: we prompt instruction-tuned LLMs to emit the label directly, and also fine-tune two such models with LoRA. All approaches use the same task, split, and label set.

3.1 Passage Classifiers

We consider both frozen and fine-tuned passage encoders, evaluated under the same downstream classifier to keep the comparison fair.

Frozen setting. We embed the inconsistent passage with a frozen sentence encoder and train a two-layer MLP on top (hidden sizes [512, 256], dropout 0.2, lr $= 10^{-3}$). We compare five backbones: MiniLM-L6-v2 (Wang et al., 2020; Reimers and Gurevych, 2019), Qwen3-Embedding 0.6B and 8B (Zhang et al., 2025), EmbeddingGemma-300M (Vera et al., 2025), and DeBERTa-v3-base (He et al., 2023) (mean-pooled).

Fine-tuned setting. Among the frozen backbones, EmbeddingGemma-300M achieves the highest accuracy (27.1%), making it the natural candidate for full fine-tuning. For EmbeddingGemma-300M, fine-tuning proceeds in two steps. First, we fine-tune the full encoder end-to-end for 11-way classification using a linear head, adapting the encoder weights to the task. Second, we discard the linear head, freeze the adapted encoder, precompute passage embeddings, and train a new two-layer MLP classifier on those embeddings, using the identical MLP architecture as the frozen baseline. The reported fine-tuned numbers come from this second step, so the only difference from the frozen baseline is the quality of the underlying embeddings.

To test whether the frozen result is merely an artifact of a weak frozen backbone, we also fine-tune DeBERTa-v3-base as a passage classifier following the same two-step protocol, so the same backbone appears in both frozen and fine-tuned form. For each encoder we report both conditions: (i) the linear classifier from end-to-end fine-tuning, and (ii) the MLP classifier after freezing the fine-tuned encoder. Hyperparameters are in Appendix D.

3.2 Evidence-Augmented Classifiers

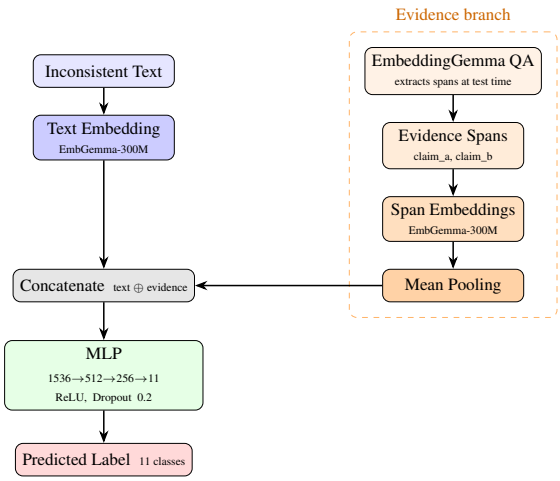

Figure 1: Evidence-augmented classifier. An EmbeddingGemma QA model predicts the two conflicting spans, which are pooled and concatenated with the passage embedding before the MLP head. The passage-only classifier omits the evidence branch; the gold-span oracle replaces the predicted spans with gold spans at test time.

Each SBID-FD instance annotates two evidence spans, `claim_a` (an anchoring claim) and `claim_b` (the conflicting claim), marking the pair of claims in tension. We hypothesize that localizing these spans provides stronger signal for the inconsistency

type than the passage as a whole. Because gold spans are unavailable at test time, we predict them with an extractive question-answering (QA) model and pass the predicted spans to the classifier (Figure 1).

Span prediction. We frame localization as extractive QA over the passage. The model is run twice per passage, once with a fixed role question for claim A and once for claim B, and decodes one contiguous span per role from the start and end token logits (the highest start+end logit sum over the top-25 start and end candidates per window, with a maximum span length of 80 tokens). Roles are assigned by which question is posed; the model does not predict the role itself. Each training row yields up to two QA examples, one per gold span. We compare several encoder backbones (Appendix B); our strongest extractor attaches an extractive-QA head to the EmbeddingGemma-300M backbone (Vera et al., 2025), which we deploy. EmbeddingGemma is released as an embedding model that outputs a single pooled representation rather than token-level start/end logits; we therefore discard the release pooling head and fine-tune the underlying transformer with a standard span head under the same QA protocol as the other backbones. This Gemma extractor attains the best span quality (Token-F1 0.679 vs 0.611 for DeBERTa-v3-base), and we use it for span prediction; the full backbone comparison, decoding details, and span-quality results are in Appendix B, and the analysis behind deploying the EmbeddingGemma extractor is in Appendix C.

Feature construction. Let $e(\cdot)$ denote the L2-normalized EmbeddingGemma-300M encoder. Given spans s_a, s_b (predicted at test time, gold during training), the classifier input is

$$\mathbf{f} = \left[e(T)\,;\, \frac{1}{2}\left(e(s_a) + e(s_b)\right) \right] \in \mathbb{R}^{1536},$$

the concatenation of the passage embedding with the mean-pooled span embeddings, fed to the same [512, 256] MLP head. We evaluate this branch in three encoder regimes: the frozen encoder; the fine-tuned EmbeddingGemma encoder used as a frozen feature extractor after passage fine-tuning (our deployed setting); and a fully end-to-end variant in which the encoder receives the passage and both spans and is trained with gradients flowing through it.

Matched training and the gold-span oracle. For every evidence-augmented model, the classifier is trained with *gold* spans on the training split; the conditions differ only in what is supplied at *test* time. The deployed pipeline uses predicted spans at test; the gold-span oracle uses gold spans at test. Because the encoder, head, and training data are identical across these conditions, any difference is attributable to the test-time span source, which lets us separate the value of localization from the quality of span prediction.

3.3 Generative Models

We prompt three instruction-tuned LLM families to output one of the 11 labels given the passage, label definitions, and one illustrative example per label (deterministic decoding: temperature $= 0$ for GPT-5.4, `do_sample=False` for local models; `max_new_tokens` set per model family: 8,192 for Qwen3.5, 96 for Gemma-4, 32 for GPT-5.4; full prompt in Appendix J): Qwen3.5 (4B, 9B, 27B, 35B-A3B) (Qwen Team, 2026), Gemma-4 (E4B-it, 26B-A4B-it, 31B-it) (Google DeepMind, 2026), and GPT-5.4 (OpenAI, 2026). All models output a single label string in practice; the budget differences do not affect results. We additionally fine-tune Qwen3.5-9B with QLoRA (quantized Low-Rank Adaptation; Hu et al., 2022; Dettmers et al., 2023) using supervised fine-tuning: each example pairs an instruction-style prompt (the excerpt and the label list) with the gold inconsistency-type string as the completion, optimizing the standard causal language-modeling objective over the prompt-label sequence (supervised fine-tuning, SFT); full hyperparameters are in Appendix F.

Metrics and protocol. We report Accuracy, Macro-F1, and Weighted-F1 on the fixed test set (1,782 examples). For the fine-tuned encoder models we report mean and standard deviation over three seeds (re-running encoder fine-tuning and classifier training per seed); the data split is held fixed. The LoRA model is run over three seeds. Frozen-encoder and prompted results are single-seed. We report multi-seed variance for the conditions where training variance most affects our headline comparisons; frozen-encoder MLP variance is small given fixed embeddings, and prompted results are deterministic under greedy decoding. Per-class F1 scores for all methods are reported in Appendices G, H, and I.

4 Results

4.1 Main Comparison

Table 2 reports the main comparison; random chance on this 11-class task is 9.1%.

Frozen embeddings are weak; fine-tuning the encoder is the dominant lever. A frozen EmbeddingGemma encoder with an MLP head reaches only 27.1% accuracy, and scaling the frozen backbone within the Qwen3-Embedding family helps little (0.6B: 20.4%, 8B: 23.3%); frozen DeBERTa-v3-base is similarly weak (23.4%). Fine-tuning the same 300M EmbeddingGemma encoder end-to-end raises accuracy to $61.9 \pm 0.6\%$ (Macro-F1 61.4 ± 0.5), a gain of more than 34 absolute points over the frozen version. The same effect holds on a different backbone: fine-tuning DeBERTa-v3-base and applying the same MLP classifier lifts it from 23.4% frozen to 49.6%, a +26.2-point jump. The finding holds across backbones: both encoders are weak when frozen and both improve sharply once fine-tuned, confirming that weak frozen performance reflects unadapted representations rather than an inherent difficulty of the task. Task adaptation, not embedding scale, is what closes most of the gap.

A small fine-tuned encoder is competitive with much larger models. The fine-tuned 300M encoder ($61.9 \pm 0.6\%$) matches or slightly exceeds Qwen3.5-9B + LoRA ($61.5 \pm 0.2\%$) and GPT-5.4 (61.3%), despite being more than an order of magnitude smaller than the LoRA model and requiring no proprietary API. Scaling the LoRA model to 27B barely moves the result (62.9%, single seed), despite being roughly ninety times larger than the encoder. All prompted models remain well below the fine-tuned encoder, even at 27B scale.

Conflict localization pushes further past the generative baselines. Adding predicted evidence spans to the fine-tuned encoder improves accuracy beyond the passage-only result. With the EmbeddingGemma QA extractor, the fine-tuned evidence-augmented model reaches 63.3% (Macro-F1 62.7%), clearly surpassing LoRA 9B (61.5%) and GPT-5.4 (61.3%), and competitive with LoRA 27B (62.9%, single seed). Training the encoder jointly with spans from the start (end-to-end) yields 63.5% with predicted spans and 67.4% with gold spans, the highest gold-oracle result across all conditions. These results show that conflict localiza-

Table 2: Performance on the 11-class inconsistency classification task (1,782 test examples). EmbGemma = EmbeddingGemma-300M; MiniLM = all-MiniLM-L6-v2; QwenEmb = Qwen3-Embedding; pred. = predicted evidence spans. (linear) and (+ MLP) follow the two-step protocol in §3. EmbGemma fine-tuned: mean ± std over three seeds; LoRA: mean ± std over three seeds; all other rows single-seed. Prompted models use label definitions and illustrative examples. †Single-seed (Gemma extractor, end-to-end rows, 27B LoRA).

Method	Acc.	Mac. F1	Wei. F1
Frozen Passage Encoder + MLP			
MiniLM	0.189	0.157	0.173
QwenEmb-0.6B	0.204	0.163	0.187
QwenEmb-8B	0.233	0.202	0.227
DeBERTa-v3-base	0.234	0.151	0.178
EmbGemma	0.271	0.220	0.248
Fine-Tuned Passage Encoder			
DeBERTa-v3-base (linear)	0.468	0.363	0.406
DeBERTa-v3-base (+ MLP)	0.496	0.434	0.471
EmbGemma (linear)	0.611	0.600	0.607
(± std, 3 seeds: 1, 3, 42)	0.010	0.013	0.013
EmbGemma (+ MLP)	0.619	0.614	0.620
(± std, 3 seeds: 1, 3, 42)	0.006	0.005	0.007
Evidence-Augmented (EmbGemma + spans)			
frozen, pred. (DeBERTa)	0.510	0.497	0.511
frozen, pred. (Gemma)	0.532	0.517	0.531
frozen, gold *(oracle)*	0.609	0.602	0.611
fine-tuned, pred. (DeBERTa)	0.625	0.621	0.625
fine-tuned, pred. (Gemma)†	0.633	0.627	0.634
fine-tuned, gold *(oracle)*	0.653	0.651	0.653
end-to-end, pred. (DeBERTa)†	0.623	0.611	0.620
end-to-end, pred. (Gemma)†	0.635	0.623	0.631
end-to-end, gold *(oracle)*†	0.674	0.665	0.670
Prompted Generative			
Qwen3.5-4B	0.274	0.244	0.256
Qwen3.5-9B	0.330	0.275	0.289
Qwen3.5-35B-A3B	0.389	0.324	0.349
Gemma-4-31B-it	0.404	0.366	0.385
Gemma-4-E4B-it	0.409	0.397	0.414
Gemma-4-26B-A4B-it	0.439	0.411	0.434
Qwen3.5-27B	0.487	0.472	0.482
GPT-5.4	0.613	0.589	0.600
Fine-Tuned Generative			
Qwen3.5-9B + LoRA SFT	0.615	0.596	0.604
(± std, 3 seeds)	0.002	0.002	0.003
Qwen3.5-27B + LoRA SFT†	0.629	0.610	0.622

tion is not merely a diagnostic tool: it provides a practical performance gain on top of task adaptation, with the evidence-augmented encoder giving the best non-oracle result among the tested configurations.

4.2 The Gold-Span Oracle and Predicted Spans

To study conflict localization independently of representation, we hold the encoder, head, and training data fixed and vary only the test-time span source (Table 2). On the fine-tuned encoder, the gold-span oracle reaches 65.3% accuracy (Macro-F1 65.1%), a gain of $+3.4$ points over the passage-only baseline (61.9%). The EmbeddingGemma QA extractor (Token-F1 0.679; Table 9) reaches 63.3% with predicted spans, recovering about a third of that gold gain. Localization yields a real and reproducible benefit, but current span prediction realizes only a fraction of the available gain. The barrier is span-prediction quality: better spans help, but a large gap to the gold oracle persists.

Per-class oracle decomposition. Table 3 decomposes performance by category under the frozen pipeline, separating types that are hard to *represent* from those hard to *localize*. Referential is the clearest localization-limited case: text-only F1 is near zero (0.016) but reaches 0.540 with gold spans. Factual and Logical remain low even with gold spans (0.454 and 0.493), pointing to knowledge or reasoning deficits that localization alone cannot resolve. Most categories benefit primarily from better span prediction; Factual and Logical require different remedies.

The oracle gain requires the right span, not any span. A potential concern is that the oracle gain is an artifact of the synthetic construction: if the classifier benefits from being handed *any* span rather than specifically the *correct* one, the gain would not reflect genuine localization. We test this directly by replacing the gold test spans with controls of the same form (Table 4). Random same-length spans drawn from the passage collapse accuracy to 0.515, and TF-IDF distractor spans to 0.502, both far below the gold oracle's 0.653 and near the passage-only level. The oracle gain is thus a genuine localization effect: it depends on supplying the correct conflicting span, not on the presence of any span.

Table 3: Per-class F1 under the fixed *frozen* EmbeddingGemma + MLP pipeline, isolating representation from span prediction. Text = no evidence; Pred. = predicted spans (DeBERTa-v3-base); Gold = gold-span oracle. Δ = Gold $-$ Pred. Rows ordered by Δ.

Category	**Text**	**Pred.**	**Gold**	Δ
Unit & Measurement	0.142	0.418	0.603	0.185
Referential	0.016	0.360	0.540	0.180
Temporal	0.207	0.472	0.611	0.139
Factual	0.106	0.339	0.454	0.115
Logical	0.244	0.378	0.493	0.115
Numerical	0.263	0.624	0.715	0.091
Specificity & Scope	0.038	0.515	0.601	0.086
Pragmatic	0.082	0.459	0.540	0.081
Theoretical	0.545	0.736	0.793	0.057
Terminological	0.261	0.486	0.539	0.053
Normative & Policy	0.521	0.683	0.735	0.052
Accuracy	0.271	0.510	0.609	
Macro F1	0.220	0.497	0.602	

Downstream accuracy tracks span quality. If span-prediction quality is the barrier, downstream accuracy should degrade as span quality falls. We confirm this with a controlled mixing sweep (Appendix E): for each test instance, each gold span is replaced by the model-predicted span independently with probability $p \in \{0, 0.2, 0.4, 0.6, 0.8, 1.0\}$, interpolating between full gold ($p = 0$, Token-F1 1.0, Macro-F1 0.645) and full EmbeddingGemma QA-predicted ($p = 1.0$, Token-F1 0.678, Macro-F1 0.627). Span Token-F1 falls steadily across the sweep; downstream Macro-F1 tracks the overall trend, though it is not strictly monotone at intermediate points. Improving the extractor directly improves classification accuracy.

The gap persists even with end-to-end span-aware training. One might expect that training the encoder end-to-end with spans, rather than reusing a passage-tuned encoder, would close the predicted-to-gold gap. Joint training helps only modestly: it reaches 0.674 accuracy with gold spans and 0.635 with predicted spans (single seed, Gemma extractor), versus the fine-tuned passage-only baseline of 61.9%: predicted spans add a small gain over passage-only, but still trail gold by roughly 4 points. The qualitative finding is unchanged across encoder regimes: gold spans help substantially, predicted spans help only partially (negligible with the DeBERTa extractor, a small gain with the EmbeddingGemma extractor), and

a sizable predicted-to-gold gap remains. Across these regimes the binding constraint is consistently span-prediction quality: with the classifier, representation, and training scheme held fixed, only the test-time span source moves the predicted-to-gold gap.

A note on the frozen pipeline: the apparent gain from predicted spans in the frozen setting (27.1% to 51.0%) largely reflects the weakness of the frozen baseline rather than a localization effect. The frozen and fine-tuned gold oracles (60.9% and 65.3%) define the ceiling achievable under perfect localization for each encoder regime.

Table 4: Oracle-artifact control (fine-tuned encoder, seed 42): gold test spans replaced by same-length random spans or high-TF-IDF distractor spans, training unchanged. Only the correct span recovers the oracle gain.

Test evidence	Acc.	Mac. F1
Gold spans *(oracle)*	0.653	0.645
Random same-length spans	0.515	0.502
TF-IDF distractor spans	0.502	0.481

4.3 Span-Prediction Quality by Role

The predicted-to-gold gap persists even with the strongest extractor. The EmbeddingGemma QA model attains overall Token-F1 = 0.679, but with a clear asymmetry: `claim_b` (the conflicting span) at 0.755 versus `claim_a` (the anchoring span) at 0.604 (Appendix Table 10). The conflicting claim is easier to localize than the claim it contradicts. This asymmetry, and the still-imperfect absolute Token-F1, explain why even the strongest extractor recovers only part of the oracle's gain, and where a better extractor would help further. The predicted-to-gold gap is not an artifact of how we combine spans with the passage: it is stable (around 3 points) across mean-pooling, two-slot concatenation, and max-pooling, and a single-sequence cross-encoder improves the gold oracle (to 0.671) without improving predicted spans (Appendix E).

4.4 Confusion Analysis

Per-class F1 identifies which categories are difficult but not how they fail. We inspect the row-normalized confusion matrix of the fine-tuned evidence-augmented pipeline (Table 5). Errors are structured rather than uniform. Two patterns stand out. First, Logical acts as a broad attractor, absorbing errors from Pragmatic (20.1% of its rows), Specificity & Scope (14.1%), Referential (15.6%), Theoretical (11.6%), and Normative & Policy (7.9%). Second, Factual and Temporal are mutually confused: Factual is most often mistaken for Temporal (20.4%) and Temporal for Factual (13.8%), suggesting that some factual conflicts surface through reporting-period or event-sequence cues. Referential is unstable, splitting its errors between Logical (15.6%) and Factual (14.7%). These patterns extend the oracle analysis: residual errors reflect both local overlap between neighboring labels and a broader collapse of difficult cases into reasoning-oriented categories, consistent with Factual and Logical being among the hardest classes even under the oracle.

Table 5: Largest off-diagonal confusions of the fine-tuned evidence-augmented pipeline, by row-normalized percentage of the gold class.

Gold	Predicted	Count	Row %
Factual	Temporal	29	20.4
Pragmatic	Logical	29	20.1
Referential	Logical	17	15.6
Specificity & Scope	Logical	21	14.1
Referential	Factual	16	14.7
Temporal	Factual	23	13.8
Pragmatic	Temporal	19	13.2
Theoretical	Logical	21	11.6
Logical	Normative & Policy	19	7.9
Normative & Policy	Logical	18	7.9

5 Conclusion

This paper studied fine-grained inconsistency classification in financial disclosures: the task of identifying which type of inconsistency is present once a passage is known to contain a conflict. Using SBID-FD, we compared frozen embedding classifiers, fine-tuned encoders, evidence-augmented classifiers, prompted LLMs, and LoRA-adapted generative models under a shared 11-class evaluation setting. The results show that task adaptation is more important than model scale for this setting. A fine-tuned 300M encoder matches or exceeds much larger prompted and LoRA-adapted generative models, suggesting that compact domain-adapted classifiers can be a strong and practical alternative to large-scale generative systems for inconsistency typing.

The experiments also show that conflict localization is useful but not yet fully exploited. Gold evidence spans consistently improve classification

performance, and control experiments confirm that the gain comes from the correct conflicting spans rather than from adding arbitrary text spans. However, predicted spans recover only part of the oracle improvement, indicating that span extraction is a major bottleneck. Per-class and confusion analyses further show that some categories, such as Referential, are strongly localization-limited, while others, such as Factual and Logical, remain difficult even with better evidence. These findings suggest that future progress will require both stronger evidence localization and more robust reasoning over closely related inconsistency types. Future work should also extend this evaluation from short synthetic passages to longer, document-scale financial disclosures, improve span prediction for both anchoring and conflicting claims, and study whether fine-grained inconsistency classifiers can support human reviewers in realistic disclosure review settings.

Reproducibility. We will release all code for the experiments in this paper, including the classifier and extractor training scripts, the evidence-augmented and oracle pipelines, and the evaluation and analysis scripts, together with the exact train-test split used here, upon publication. All hyperparameters, data splits, and seeds are specified in the appendices.

Limitations

The benchmark is synthetic, so our results measure performance on constructed inconsistencies rather than naturally occurring filing errors, and the instances are short passages rather than full documents, leaving cross-section inconsistencies uncaptured. The synthetic construction reflects a real data-access constraint: financial inconsistencies in internal drafts and revision histories are confidential and rarely released at scale. Validation on naturally occurring inconsistencies, ideally in a purpose-built long-document resource with annotated conflicting-span pairs and type labels, is the most important open direction.

The gold-span oracle is an upper bound not achievable at deployment, and the localization benefit it demonstrates may differ for naturally occurring inconsistencies where the conflicting span is not a localized synthetic edit. Because the split is stratified at the instance level, it does not fully rule out shared generation patterns across train and test; future work should evaluate on grouped or template-disjoint splits.

Our variance estimates are partial. The data split is held fixed across seeds, and the predicted spans come from a single QA run, so split and extractor variance are not captured. The end-to-end span-aware results and the oracle-artifact control are single-seed and should be treated as indicative. One encoder fine-tuning run was excluded after failing to converge, which we note as a reproducibility caveat: end-to-end fine-tuning of this encoder is occasionally unstable.

Societal impact. A reliable fine-grained inconsistency classifier could assist auditors and analysts in triaging financial disclosures, directing numerical, temporal, or normative conflicts to the appropriate downstream check. The same capability carries risk if over-trusted: a model that misses or misroutes a real inconsistency in a high-stakes filing could create false confidence, and our results show that even the best configurations are well short of perfect (and that two categories remain hard even with gold evidence). We therefore frame this work as a diagnostic and triage aid rather than an automated verifier, and caution against deploying such a classifier as a sole check on consequential disclosures. In realistic use, predictions should be presented alongside the supporting spans so that human reviewers can audit rather than blindly accept the model output.

Acknowledgments

The authors used AI writing assistants (Claude and ChatGPT) for coding support and paper editing. All experimental results, analyses, and scientific claims are the authors' own.

References

Anonymous. 2026. SBID-FD: A synthetic benchmark for inconsistency detection in financial disclosure text. Under review.

bitsandbytes Foundation. 2026. bitsandbytes. Official GitHub repository.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Kenneth Huang, Bryan R.

Routledge, and William Yang Wang. 2021. FinQA: A dataset of numerical reasoning over financial data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3697–3711. Association for Computational Linguistics.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. QLoRA: Efficient finetuning of quantized LLMs. In *Advances in Neural Information Processing Systems*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. 2020. ERASER: A benchmark to evaluate rationalized NLP models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4443–4458, Online. Association for Computational Linguistics.

Google DeepMind. 2026. Gemma 4: Byte for byte, the most capable open models. Official release page.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2023. DeBERTaV3: Improving DeBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing. In *International Conference on Learning Representations*.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Yuta Koreeda and Christopher Manning. 2021. ContractNLI: A dataset for document-level natural language inference for contracts. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1907–1919, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jierui Li, Vipul Raheja, and Dhruv Kumar. 2024. ContraDoc: Understanding self-contradictions in documents with large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 6509–6523, Mexico City, Mexico. Association for Computational Linguistics.

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

Zhuang Liu, Degen Huang, Kaiyu Huang, Zhuang Li, and Jun Zhao. 2020. FinBERT: A pre-trained financial language representation model for financial text mining. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pages 4513–4519. International Joint Conferences on Artificial Intelligence Organization.

Tim Loughran and Bill McDonald. 2011. When is a liability not a liability? textual analysis, dictionaries, and 10-Ks. *The Journal of Finance*, 66(1):35–65.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901, Online. Association for Computational Linguistics.

OpenAI. 2026. Introducing GPT-5.4. Official release page.

Qwen Team. 2026. Qwen3.5: Towards native multimodal agents. Official release page.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Henrique Schechter Vera, Sahil Dua, Biao Zhang, Daniel Salz, Ryan Mullins, and 1 others. 2025. EmbeddingGemma: Powerful and lightweight text representations. *arXiv preprint arXiv:2509.20354*.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *arXiv preprint arXiv:2002.10957*.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. 2025. Qwen3 embedding: Advancing text

embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*.

Fengbin Zhu, Wenqiang Lei, Youcheng Huang, Chao Wang, Shuo Zhang, Jiancheng Lv, Fuli Feng, and Tat-Seng Chua. 2021. TAT-QA: A question answering benchmark on a hybrid of tabular and textual content in finance. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 3277–3287. Association for Computational Linguistics.

A Dataset Examples

To make the task concrete, we show one SBID-FD instance, illustrating the structure our localization analysis relies on: two conflicting clauses placed inside a longer passage, each marked by a gold span, with the passage typed into one of 11 inconsistency categories. Table 6 gives the per-class counts for both splits.

Table 6: Per-class instance counts in the train and test splits (stratified 70–30).

Inconsistency type	**Train**	**Test**
Logical	564	241
Normative & Policy	531	228
Theoretical	423	181
Numerical	419	180
Temporal	390	167
Specificity & Scope	347	149
Pragmatic	337	144
Factual	330	142
Terminological	289	124
Unit & Measurement	273	117
Referential	255	109
Total	4,158	1,782

SBID-FD (11-way typing with paired spans). *Label:* Logical inconsistency. *Passage:* "Projected earnings per share of $2.40 assumes stable diesel fuel pricing across our regional distribution network. Freight rate increases exceeding 8% year-over-year would compress operating margins below our conservative targets. Operating margins are structurally insulated from freight cost volatility. Fleet utilization rates are forecast to reach 85% by Q4, contingent on hiring 500 new drivers. . . " *Gold spans:* CLAIM_A = "Freight rate increases exceeding 8% year-over-year would compress operating margins below our conservative targets"; CLAIM_B = "Operating margins are structurally insulated from freight cost volatility". *Why inconsistent:* the passage first states that higher freight rates would compress margins, then asserts margins are insulated from freight volatility; the two claims cannot both hold. The task is to predict the type (here Logical) given the passage; the evidence-augmented pipeline additionally supplies the two spans.

Length statistics. Table 7 reports length statistics for the base passages and their inconsistent variants over the 5,940 instances. Token counts use the Qwen3.5 tokenizer. The train and test splits are nearly identical on these statistics (the split is stratified), so we report the pooled figures.

Table 7: Length statistics for base passages and inconsistent variants over the 5,940 instances. Token counts use the Qwen3.5 tokenizer. Variants are slightly longer, reflecting the information added to introduce a conflict.

Metric	Mean	S.D.	Min	Max
Base passages				
Characters	574.9	125.4	73	1,099
Tokens (Qwen3.5)	100.9	23.3	12	200
Sentences	5.2	1.5	1	12
Inconsistent variants				
Characters	625.8	142.2	97	1,176
Tokens (Qwen3.5)	110.5	26.5	19	236
Sentences	5.6	1.7	1	14

Surface-form overlap. To quantify how localized the perturbations are, we measure character-level edit (Levenshtein) distance and longest common contiguous substring (LCS) between each base passage and its inconsistent variant (Table 8). The median edit distance is 72 characters and the median LCS covers about 73% of the shorter passage, confirming that inconsistencies are introduced through localized edits or insertions rather than wholesale rewrites.

B Evidence-Span QA: Implementation Details

Task Formulation

We model evidence-span prediction as an extractive QA problem over each sample's inconsistent passage. Each instance contains a gold evidence list with items of the form `{"quote": "...", "role": "claim_a"|"claim_b"}`, converted into up to two QA training examples:

Table 8: Surface-form overlap between base passages and their inconsistent variants over the 5,940 instances. Levenshtein distance is character-level; LCS is the longest common contiguous substring. Normalized LCS divides the LCS length by the shorter (min) or longer (max) passage length.

Metric	Mean	Median	P5	P95
Levenshtein distance	81.8	72	9	198
LCS length (chars)	420.1	404	213	682
LCS / min length	0.742	0.734	–	1.000
LCS / max length	0.661	0.655	–	–

- *Question:* "Extract the evidence span for claim A (verbatim substring)." *Answer:* gold quote with `role="claim_a"`

- *Question:* "Extract the evidence span for claim B (verbatim substring)." *Answer:* gold quote with `role="claim_b"`

Gold quotes are aligned to the context via exact substring match, then whitespace-normalized matching. Examples that cannot be aligned are skipped. The training split expands from 4,158 rows to 8,076 QA examples; the test split expands from 1,782 rows to 3,481 examples, with 34 rows excluded due to unalignable gold quotes.

Model and Training

We fine-tune five encoder backbones with an extractive QA head (`max_length=384`, `doc_stride=128`): `distilbert-base-uncased` (Sanh et al., 2019), `roberta-base` (Liu et al., 2019), `bert-base-uncased` (Devlin et al., 2019), `deberta-v3-base` (He et al., 2023), and the EmbeddingGemma-300M backbone. For the last, EmbeddingGemma is released as an embedding model with a mean-pooling head; we discard that head and attach a token-level start/end span head to the underlying `Gemma3TextModel` transformer, training it with the identical QA protocol. Each backbone is trained on the training split with a 10% passage-level validation slice held out for checkpoint selection; the test split is not used during training. We deploy the EmbeddingGemma QA extractor, which gives the best span quality (Table 9); training was stable in fp32 with no divergence.

Inference and Decoding

At test time, the model is run twice per instance, once per role. Candidate spans are scored by:

$$\text{score}(s, e) = \text{start_logit}[s] + \text{end_logit}[e] \quad (1)$$

searching over the top-25 start and end positions across all overflow windows, keeping the highest-scoring span of at most 80 tokens.

Table 9: QA backbone sweep results (seed 42; 10% train validation, test held out until inference). The EmbeddingGemma QA extractor is deployed.

Backbone	EM	Token-F1
`EmbeddingGemma-QA`	**0.3710**	**0.6790**
`deberta-v3-base`	0.3030	0.6110
`distilbert-base-unc.`	0.2784	0.5740
`bert-base-uncased`	0.2611	0.5464
`roberta-base`	0.1864	0.4529

Table 10: Evidence-span QA results by role for the deployed extractor (EmbeddingGemma QA, seed 42; trained with 10% train-only validation, test held out until inference).

Role	EM	Token-F1	n
`claim_a`	0.344	0.6040	1,748
`claim_b`	0.399	0.7550	1,733
Overall	0.3710	0.6790	3,481

C EmbeddingGemma as a Span Extractor

Our deployed extractor attaches a QA head to the EmbeddingGemma backbone. We summarize the experiments behind this choice.

Span quality. Under the identical QA protocol (seed 42, 10% passage validation, `max_length` 384, `doc_stride` 128), the EmbeddingGemma QA extractor attains overall Token-F1 0.679 (EM 0.371), versus 0.611 (EM 0.303) for DeBERTa-v3-base (Table 9). Training was stable in fp32 with no divergence. The released EmbeddingGemma checkpoint is an embedding model with a pooling head; we discard that head and train the underlying transformer with a token-level span head, confirming that the backbone supports extractive span prediction and, here, does so better than a standard QA encoder.

Downstream effect. Swapping only the test-time predicted spans in the deployed pipeline (fine-tuned EmbeddingGemma classifier, pooled MLP, seed 42) changes accuracy from 0.621 (DeBERTa spans) to 0.633 (Gemma spans) and Macro-F1 from 0.621 to 0.627, against a gold-oracle ceiling of 0.653 / 0.645. Higher span quality transfers to downstream accuracy, though a gap to the oracle remains.

A unified single model hurts classification. We also tried a single Gemma backbone trained multi-task (a span head and an 11-way classification head jointly), using its own predicted spans for classification. Its span quality stays high (Token-F1 0.675), but classification drops to 0.604 / 0.586, below the two-model stack (0.633 / 0.627 with separate extractor and classifier). We therefore keep span extraction and classification as separate models rather than unifying them.

D Classifier Training Details

All MLP heads use hidden layers [512, 256], dropout 0.2, lr $= 10^{-3}$, weight decay $= 10^{-4}$, batch size 256, 25 epochs. Frozen-encoder features are precomputed. The fine-tuned EmbeddingGemma encoder is trained end-to-end on passage-only 11-way classification (5 epochs, lr $= 2 \times 10^{-5}$, batch size 8); for the evidence-augmented and oracle conditions, this fine-tuned encoder is then used as a frozen feature extractor for both passage and span embeddings, and only the MLP head is trained. This is not full end-to-end span-aware fine-tuning; the encoder is adapted on passages and reused as an embedder. Fine-tuned DeBERTa-v3-base passage classification uses 5 epochs, lr $= 6 \times 10^{-6}$, warmup 0.1. The end-to-end span-aware variant trains the EmbeddingGemma encoder jointly with the MLP head over the passage-plus-spans input (5 epochs, lr $= 2 \times 10^{-5}$, seed 42).

Compute. Experiments were run on a node with four NVIDIA RTX 6000 Ada Generation GPUs, though most use only a single GPU; multiple GPUs were used only for the largest model (the 27B LoRA run). Approximate wall-clock training times (recovered from run logs) illustrate the cost profile. Frozen-encoder runs are dominated by a single embedding pass and take seconds to a few minutes. Fine-tuning the EmbeddingGemma passage classifier takes about four minutes per seed on a single GPU; the evidence-augmented head trains in well under a minute on top of cached features, and the end-to-end span-aware variant takes about twenty minutes. The deployed EmbeddingGemma QA extractor trains in about two and a half minutes. By contrast, the LoRA generative baselines are far more expensive: Qwen3.5-9B takes roughly fifty minutes per seed and Qwen3.5-27B about two and a half hours, despite reaching accuracy statistically indistinguishable from the four-minute encoder fine-tune. Prompted inference over the test set ranges from a few minutes for the smaller models to about an hour for the largest. These figures are approximate and recovered post hoc; a few runs were not logged with reliable timing.

E Feature-Fusion Ablation and Span-Quality Sweep

We ablate how the passage and span embeddings are combined, on the frozen EmbeddingGemma pipeline (seed 42), reporting both the predicted-span condition and the gold oracle (Table 11). Mean-pooling, two-slot concatenation, and max-pooling give similar results, with a stable predicted-to-gold gap of roughly 3 points. A single-sequence cross-encoder improves the gold oracle to 0.671 but does not improve predicted spans (0.607), consistent with the finding that the binding constraint is span-quality rather than fusion: better fusion of *correct* spans helps, but no fusion rescues noisy predicted spans.

Table 11: Feature-fusion ablation (frozen fine-tuned EmbeddingGemma, seed 42). Accuracy with predicted spans and gold spans (oracle).

Fusion	Pred.	Gold
Mean-pool	0.621	0.653
Two-slot concat	0.619	0.654
Max-pool	0.619	0.652
Cross-encoder	0.607	0.671

Span-quality mixing sweep. To verify that downstream accuracy tracks span quality, we vary the fraction of gold test spans via per-role Bernoulli substitution with probability p (probability of replacing a gold span with the DeBERTa-predicted span). Table 12 reports the results on the fine-tuned EmbeddingGemma pipeline. Span Token-F1 falls steadily as p increases; downstream Macro-F1 tracks the endpoints and the overall trend, though it is not strictly monotone at intermediate values.

Table 12: Span-quality mixing sweep on the fine-tuned EmbeddingGemma pipeline (seed 42). p = probability of substituting the EmbeddingGemma QA-predicted span for the gold span per role. $p = 0$: full gold spans; $p = 1$: full predicted spans. Downstream Macro-F1 at $p = 1$ matches the single-seed result in Table 2 (Macro-F1 0.627).

p	Avg. Token-F1	Macro-F1
0.0	1.000	0.645
0.2	0.936	0.644
0.4	0.872	0.635
0.6	0.807	0.637
0.8	0.744	0.634
1.0	0.678	0.627

F LoRA Fine-Tuning Details

We fine-tune `Qwen/Qwen3.5-9B` under 4-bit NF4 (Normal Float 4) quantization (`bitsandbytes`, bitsandbytes Foundation, 2026) with double quantization and `bf16` compute dtype. LoRA adapters are applied to attention projections (`q_proj`, `k_proj`, `v_proj`, `o_proj`), with rank $r = 16$, scaling $\alpha = 32$, and dropout 0.05. Training uses 1 epoch, `paged_adamw_8bit`, learning rate 2×10^{-4} with warmup ratio 0.03, effective batch size 16, and maximum sequence length 1,024 tokens. At inference, we use deterministic decoding with `max_new_tokens=32` and parse predictions via exact-line matching with a substring fallback. The `Qwen/Qwen3.5-27B` LoRA model uses the identical configuration (same rank, schedule, prompt, and decoding), changing only the base model, so the 9B-vs-27B comparison reflects scale alone; it is run for a single seed (42).

G Per-Class F1 Scores: Frozen-Encoder Methods

Table 13 reports per-class F1 for the frozen text-only backbones and for the frozen EmbeddingGemma pipeline with predicted and gold spans, the configuration used for the localization decomposition in Table 3.

H Per-Class F1 Scores: Fine-Tuned Encoder

Table 14 reports per-class F1 for the fine-tuned EmbeddingGemma encoder (seed 42) under passage-only, both predicted-span extractors (DeBERTa and the deployed Gemma), and the gold oracle. Per class, the Gemma extractor improves on DeBERTa for most categories (notably Referential 0.442 to 0.478 and Factual 0.404 to 0.431), yet every category still trails its gold-oracle value. This is the per-class form of the paper's central finding: better spans help, but a gap to perfect localization persists across the board.

I Per-Class F1 Scores: All Generative Models

Table 15 reports per-class F1 scores for all generative configurations, ordered by overall accuracy. Prompted Qwen3.5-9B remains weak and unevenly distributed across classes, with its lowest F1 on *Theoretical* (0.085) and over-prediction of *Logical* (recall 0.67 against precision 0.23), indicating a label bias without task-specific adaptation. GPT-5.4 performs strongly on structured categories such as *Numerical* (F1 = 0.796) and *Normative & Policy* (F1 = 0.745) but struggles on *Factual* (F1 = 0.258) and *Pragmatic* (F1 = 0.305). The LoRA-fine-tuned Qwen3.5-9B is the most balanced, with no category below F1 = 0.287 and strong results on *Theoretical* (0.774) and *Unit & Measurement* (0.756).

J Classification Prompt

Classification Prompt

```
You are a careful annotator.
Task: classify the given excerpt into exactly one inconsistency type.

Valid labels (choose exactly one), with a definition and one example each:
Factual, Logical, Normative & Policy, Numerical, Pragmatic, Referential, Specificity & Scope, Temporal, Terminological, Theoretical, Unit & Measurement.

Return JSON with exactly this schema:
{"label": "<one of the valid labels verbatim>"}

Excerpt:
{inconsistent_text}
```

Table 13: Per-class F1 scores for frozen-encoder MLP classifiers (1,782 test examples, seed 42). EmbGemma = EmbeddingGemma-300M; MiniLM = all-MiniLM-L6-v2; QwenEmb = Qwen3-Embedding; pred. = predicted evidence spans (DeBERTa-v3-base); gold = gold-span oracle.

Category	**n**	**MiniLM**	**QwenEmb-0.6B**	**QwenEmb-8B**	**DeBERTa**	**EmbGemma**	**EmbGemma (pred.)**	**EmbGemma (gold)**
Factual	142	0.095	0.084	0.133	0.124	0.106	0.339	0.454
Logical	241	0.131	0.180	0.184	0.244	0.244	0.378	0.493
Normative & Policy	228	0.335	0.428	0.507	0.306	0.521	0.683	0.735
Numerical	180	0.151	0.239	0.224	0.473	0.263	0.624	0.715
Pragmatic	144	0.127	0.120	0.194	0.000	0.082	0.459	0.540
Referential	109	0.031	0.047	0.043	0.000	0.016	0.360	0.540
Specificity & Scope	149	0.082	0.022	0.071	0.000	0.038	0.515	0.601
Temporal	167	0.178	0.109	0.226	0.241	0.207	0.472	0.611
Terminological	124	0.135	0.110	0.094	0.000	0.261	0.486	0.539
Theoretical	181	0.391	0.396	0.467	0.235	0.545	0.736	0.793
Unit & Measurement	117	0.069	0.062	0.078	0.033	0.142	0.418	0.603
Accuracy		0.189	0.204	0.233	0.234	0.271	0.510	0.609
Macro F1		0.157	0.163	0.202	0.151	0.220	0.497	0.602
Weighted F1		0.173	0.187	0.227	0.178	0.248	0.511	0.611

Table 14: Per-class F1 for the fine-tuned EmbeddingGemma encoder (seed 42), used as a frozen feature extractor with the [512,256] MLP head. Passage = no evidence; DeBERTa / Gemma = predicted spans from each extractor; Gold = gold-span oracle. The Gemma extractor is deployed.

Category	**Passage**	**DeBERTa**	**Gemma**	**Gold**
Factual	0.403	0.404	0.431	0.449
Logical	0.460	0.473	0.489	0.527
Normative & Policy	0.788	0.792	0.802	0.804
Numerical	0.731	0.760	0.751	0.767
Pragmatic	0.551	0.534	0.555	0.563
Referential	0.434	0.442	0.478	0.500
Specificity & Scope	0.609	0.625	0.641	0.672
Temporal	0.541	0.559	0.571	0.607
Terminological	0.648	0.657	0.654	0.652
Theoretical	0.781	0.773	0.786	0.798
Unit & Measurement	0.700	0.730	0.737	0.760
Accuracy	0.612	0.625	0.633	0.653
Macro F1	0.604	0.621	0.627	0.651

Table 15: Per-class F1 scores for generative models (1,782 test examples). All prompted models use label definitions and illustrative examples. Models ordered by overall accuracy. LoRA = Qwen3.5-9B fine-tuned on training split. All columns are single-seed (seed 42) for per-class detail; the LoRA aggregate here (seed 42) is the high end of the three-seed range whose mean (0.615 acc, 0.596 macro) is reported in Table 2.

Category	**n**	**Qwen3.5-4B**	**Qwen3.5-9B**	**Qwen3.5-35B-A3B**	**Gemma-4-31B**	**Gemma-4-E4B**	**Gemma-4-26B**	**Qwen3.5-27B**	**GPT-5.4**	**Qwen3.5-9B + LoRA**
Factual	142	0.000	0.138	0.207	0.153	0.152	0.165	0.171	0.258	0.287
Logical	241	0.222	0.343	0.342	0.346	0.414	0.384	0.447	0.566	0.501
Normative & Policy	228	0.547	0.537	0.651	0.622	0.655	0.621	0.687	0.745	0.734
Numerical	180	0.369	0.269	0.580	0.683	0.552	0.683	0.629	0.796	0.727
Pragmatic	144	0.098	0.123	0.221	0.133	0.106	0.093	0.321	0.305	0.590
Referential	109	0.227	0.229	0.372	0.412	0.380	0.331	0.478	0.556	0.448
Specificity & Scope	149	0.283	0.157	0.244	0.274	0.536	0.449	0.502	0.703	0.578
Temporal	167	0.249	0.470	0.531	0.529	0.523	0.539	0.584	0.661	0.627
Terminological	124	0.447	0.410	0.158	0.400	0.386	0.398	0.519	0.607	0.638
Theoretical	181	0.064	0.085	0.162	0.290	0.276	0.593	0.339	0.594	0.774
Unit & Measurement	117	0.180	0.267	0.097	0.185	0.390	0.265	0.514	0.694	0.756
Accuracy		0.274	0.330	0.389	0.404	0.409	0.439	0.487	0.613	0.622
Macro F1		0.244	0.275	0.324	0.366	0.397	0.411	0.472	0.589	0.605
Weighted F1		0.256	0.289	0.349	0.385	0.414	0.434	0.482	0.600	0.613