

Just Pass Twice: Efficient Token Classification with LLMs for Zero-Shot NER

Ahmed Ewais* Ahmed Hashish* Amr Ali

WitnessAI

{ahmed, ahmed.hashish, amr}@witness.ai

Abstract

Large language models encode extensive world knowledge valuable for zero-shot named entity recognition. However, their causal attention mechanism, where tokens attend only to preceding context, prevents effective token classification when disambiguation requires future context. Existing approaches use LLMs generatively, prompting them to list entities or produce structured outputs, but suffer from slow autoregressive decoding, hallucinated entities, and formatting errors.

We propose **Just Pass Twice (JPT)**, a simple yet effective method that enables causal LLMs to perform discriminative token classification with full bidirectional context. Our key insight is that concatenating the input to itself lets each token in the second pass attend to the complete sentence, requiring no architectural modifications. We combine these representations with definition-guided entity embeddings for flexible zero-shot generalization. Our approach achieves state-of-the-art results on zero-shot NER benchmarks, surpassing the previous best method by **+7.9 F1** on average across CrossNER and MIT benchmarks, being over 20× faster than comparable generative methods.

1 Introduction

Named Entity Recognition (NER) is a foundational natural language processing task, underpinning numerous downstream applications such as information extraction, privacy-preserving text processing, knowledge graph construction, entity linking, question answering, and document understanding pipelines (Keraghel et al., 2024). Early approaches framed NER as a token-level sequence labeling problem, most commonly using the BIO tagging scheme (Chiu and Nichols, 2016; Akbik et al., 2018; Qin et al., 2019; Devlin et al., 2019), where

*Equal contribution

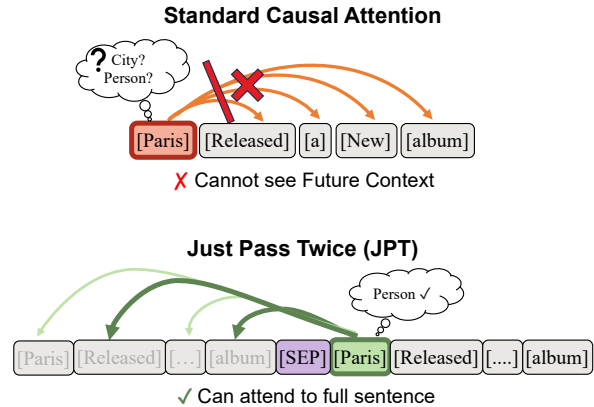


Figure 1: **Just Pass Twice (JPT) enables bidirectional token classification in causal LLMs.** (Top) Standard causal masking restricts the target token “Paris” from attending to future context like “album” (red barrier), leading to ambiguity. (Bottom) JPT duplicates the input to itself. In the second pass, the target token (green box) attends backwards to the *entire* original sequence (green arrows), resolving the entity type without architectural modifications.

every token is assigned a label indicating whether it begins, is inside, or is outside an entity mention.

The dominant approach to NER has been discriminative token classification using bidirectional encoders such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and DeBERTa (He et al., 2021). This paradigm underlies a wide range of NER systems, including BioNER based on BERT (Cocchieri et al., 2025), RoBERTa-based approaches such as NuNER (Bogdanov et al., 2024), and DeBERTa-based systems such as GLiNER (Zaratiana et al., 2024). These models naturally support token-level labeling: their bidirectional attention allows each token to attend to both preceding and subsequent context, enabling effective disambiguation. However, encoder-based methods face inherent limitations: they are typically small (<1B parameters), have limited context windows, and encode less world knowledge than larger mod-

els, which can limit generalization to novel entity types and domains.

Large language models (LLMs) offer a compelling alternative. With billions of parameters and training on vast corpora, LLMs encode rich world knowledge and demonstrate remarkable reasoning capabilities across diverse NLP tasks. These properties make them seemingly ideal for zero-shot NER.

However, existing approaches to LLM-based NER diverge from the sequence labeling paradigm, instead employing generative formulations. Recent work has fine-tuned open-source LLMs on diverse NER datasets to enhance domain adaptability: InstructUIE (Wang et al., 2023) trains on a wide range of information extraction datasets using instruction tuning; UniversalNER (Zhou et al., 2024) distills from ChatGPT and queries entity types one at a time for improved recall; and GoLLIE (Sainz et al., 2024) uses code-style annotation guidelines to improve zero-shot generalization. Empirical evaluations show that even “vanilla” prompting of large models like ChatGPT yields suboptimal results compared to smaller supervised baselines (Wei et al., 2024; Li et al., 2023).

Despite these advances, all generative NER approaches suffer from fundamental drawbacks: (1) **Speed**: autoregressive token generation is inherently slow compared to single-pass classification; (2) **Cost**: output tokens cost 3–4× more than input tokens in commercial APIs because decoding is sequential and memory-bound, requiring a forward pass per generated token; (3) **Hallucinations and Format Errors**: Generative models are prone to hallucinating entities not present in the input and can produce outputs that fail to parse into valid structured data, requiring error handling or re-prompting (Li et al., 2023; Wang et al., 2025b).

A natural question arises: why not apply the successful token classification paradigm from encoders directly to LLMs? The fundamental obstacle is architectural. Modern LLMs employ causal (unidirectional) attention, where each token can only attend to itself and preceding tokens. While essential for efficient autoregressive generation, this creates a critical limitation for token classification: when classifying a token, the model lacks access to subsequent context that may be essential for disambiguation.

Consider the sentence “*Paris released a new album*” (Figure 1). To correctly classify *Paris* as a person rather than a location, a model must see the

full sentence; the noun phrase “*a new album*” reveals this refers to the musician, not the city. Under causal attention, when the model processes *Paris*, it has not yet observed *released*, *new*, or *album*. The causal mask prevents the classifier from accessing this disambiguating future context, which is precisely why standard token classification fails with decoder-only LLMs.

We propose **Just Pass Twice (JPT)**, a method that bridges this gap through two key innovations.

First, we enable bidirectional token classification in causal LLMs without architectural changes. Our insight is simple: concatenating the input sequence to itself allows each token in the second pass to attend to the complete sentence from the first pass. We extract representations only from this second occurrence, where each token has effective bidirectional context, and use these for token classification. Despite doubling the input length, the operation occurs entirely in the highly parallel *prefill* phase. Unlike autoregressive decoding, which is sequential and memory-bound, this makes JPT dramatically faster than generative alternatives.

Second, we introduce definition-guided entity typing for flexible zero-shot generalization. Rather than encoding entity types by name alone (e.g., “PERSON”), we encode rich natural language definitions that precisely specify what each type encompasses. We inject definitions through two complementary channels: (1) as embeddings that the classifier matches against token representations, and (2) directly in the LLM’s input prompt, where they guide token encoding via attention. Inspired by recent work showing that definitions outperform simple label names for zero-shot NER (Cocchieri et al., 2025), this approach decouples the model from any fixed label vocabulary while providing fine-grained control: users can specify boundary cases directly in natural language, such as whether PRICE captures only explicit amounts (“\$50”) or also qualitative descriptors (“budget-friendly”). Since definitions remain fixed at inference time, entity embeddings are computed offline and cached, adding no runtime overhead.

We implement JPT by adding lightweight LoRA adapters (Hu et al., 2021), projection layers, and a bilinear classifier to frozen Qwen3 backbones (4B and 8B parameters), trained on an in-house Wikipedia-derived NER dataset with no overlap with evaluation benchmarks. On zero-shot evaluation across CrossNER and MIT benchmarks, JPT surpasses state-of-the-art methods by **+7.9 F1** on

average, while being over 20× faster than comparable generative methods. We further demonstrate JPT’s strong generalization on an extended benchmark of 20 diverse NER datasets spanning biomedical, social media, and multilingual domains.

Our main contributions are:

- A simple, effective method for enabling bidirectional context in causal LLMs through input duplication, requiring no architectural modifications to the base LLM architecture and leveraging efficient parallel prefill computation.
- Definition-guided entity typing that enables flexible zero-shot generalization through natural language type specifications, offering fine-grained control over what constitutes each entity type.
- State-of-the-art results on CrossNER and MIT benchmarks (+7.9 F1 over the previous best), with consistent improvements across 19 of 20 extended benchmarks and over 20× speedup versus generative methods.

Code and pretrained model weights will be released upon acceptance.

2 Related Work

The landscape of zero-shot Named Entity Recognition is characterized by a fundamental tension: efficient discriminative models with limited capacity and reasoning capabilities versus powerful generative LLMs with high latency and reliability issues.

2.1 Generative NER with LLMs

The dominant paradigm for LLM-based NER formulates the task as sequence generation. **UniversalNER** (Zhou et al., 2024) demonstrated that distilling ChatGPT into smaller generative models (e.g., LLaMA-7B) via targeted instruction tuning, querying one entity type at a time, can achieve impressive open-vocabulary performance. **InstructionUIE** (Wang et al., 2023) established a unified framework showing that multi-task instruction tuning captures inter-task dependencies. To handle complex schema definitions, **GoLLIE** (Sainz et al., 2024) fine-tunes models to follow annotation guidelines formatted as code, improving zero-shot generalization to unseen schemas. Building on these instruction-tuning paradigms, **GNER** (Ding et al., 2024) identifies that previous methods are overly

“entity-centric”; they propose incorporating negative instances (non-entities) into training to explicitly refine boundary detection and context awareness. **SaM** (Ding et al., 2025) dynamically selects and merges domain-specific LoRA adapters at inference time, achieving strong zero-shot performance but requiring a library of pre-trained expert weights.

However, generative approaches face inherent limitations. **GPT-NER** (Wang et al., 2025b) identified the “hallucination issue,” where LLMs overconfidently generate entities not present in the input, necessitating secondary verification steps. More fundamentally, autoregressive decoding introduces latency that scales with output length rather than input length.

Some recent approaches seek to improve accuracy by prompting LLMs for explicit explanations or justifications. For example, **PromptNER** (Ashok and Lipton, 2023) asks models to produce explanations supporting entity compatibility. While such methods can bolster interpretability and performance, they further increase generation costs and latency by requiring models to output rationales in addition to entity predictions. JPT sidesteps this trade-off entirely: we leverage the underlying reasoning capacity of LLM backbones but bypass autoregressive generation through discriminative projection.

2.2 Discriminative Encoder-Based Approaches

Parallel to generative methods, discriminative architectures treat NER as a semantic matching problem. **GLiNER** (Zaratiana et al., 2024) uses a bidirectional transformer (DeBERTa) to encode concatenated entity type prompts and text, enabling zero-shot detection via span-type matching in a shared latent space. **OpenBioNER** (Cocchieri et al., 2025) extends this paradigm using a cross-encoder architecture tailored to the biomedical domain, demonstrating that encoding entity *definitions*, rather than simple label names, significantly boosts zero-shot performance on rare concepts. **NuNER** (Bogdanov et al., 2024) pushes the encoder paradigm further by employing a bi-encoder architecture (separating text and concept encoding) pretrained on massive synthetic datasets annotated by LLMs.

While these encoder-based methods are efficient, they are inherently limited by the capacity of their backbone models (typically BERT/DeBERTa at <1B parameters), which may encode less

world knowledge than larger decoder models. JPT bridges this gap: we adopt the definition-augmented matching strategy of OpenBioNER but apply it to much larger decoder-only LLMs, while also injecting definitions directly into the prompt for dual-channel guidance.

Alternative Approaches to Bidirectional Context. Several techniques exist for obtaining bidirectional context in language models. Prefix-LM attention (Raffel et al., 2020; Tay et al., 2023) allows bidirectional attention over a designated prefix, but requires this pattern during pretraining. Fill-in-the-middle training (Bavarian et al., 2022) enables conditioning on future context, but only for specially-trained models. Simply removing the causal mask at inference fails because attention patterns become out-of-distribution. In contrast, JPT’s input duplication works with any off-the-shelf causal LLM: the model processes a standard causal sequence, but by repeating the input, tokens in the second pass attend to the complete sentence, requiring no architectural changes and no special pretraining.

2.3 Positioning Our Approach

JPT occupies a unique position in this landscape. Unlike generative models, we operate as a discriminative token classifier, eliminating autoregressive latency and generation-related hallucinations. Unlike encoder-only models, we leverage the massive parameter space and world knowledge of 7B+ decoder LLMs. And unlike multi-stage pipelines, we require only a single forward pass. Our approach demonstrates that the causal attention constraint can be overcome through a simple input transformation rather than architectural modifications or complex inference procedures.

3 Method

3.1 Overview

Figure 2 illustrates the JPT architecture. Given an input text and a set of entity types with definitions, JPT performs the following steps: (1) entity type definitions are encoded using a text embedding model (this can be done offline and cached), (2) the text is duplicated and processed by a causal LLM to obtain bidirectional token representations, (3) both representations are projected to a shared space R^{d_p} , and (4) a bilinear classifier computes matching scores between tokens and entity types.

3.2 Bidirectional Context via Input Duplication

The core insight of JPT is that causal attention’s unidirectional constraint can be overcome through input duplication. In a causal LLM, each token x_i can only attend to preceding tokens x_1, \dots, x_{i-1} . This prevents effective token classification, as disambiguating context often appears *after* the token of interest.

Given an input sequence $\mathbf{x} = (x_1, x_2, \dots, x_n)$, we construct the duplicated input:

$$\mathbf{x}' = (\underbrace{x_1, \dots, x_n}_{\text{first pass}}, [\text{SEP}], \underbrace{x_1, \dots, x_n}_{\text{second pass}}) \quad (1)$$

When the LLM processes \mathbf{x}' , each token x_i in the second pass can attend to: (1) all tokens from the first pass x_1, \dots, x_n , providing complete “future” context, and (2) preceding tokens in the second pass x_1, \dots, x_{i-1} , providing standard “past” context. The combination provides effective bidirectional attention without modifying the causal attention mechanism.

Attention Coverage. Crucially, for a token at position k in the second pass (position $n + 1 + k$ in \mathbf{x}'), causal attention permits attending to all positions $j \leq n + 1 + k$, which includes all n tokens of the first pass. This means *every* token in the second pass has access to the complete input sequence, exactly the bidirectional context required for accurate token classification. We visualize these attention patterns in Section 5.

Token Representation Extraction. We extract hidden states only from the second occurrence of tokens, as these contain the bidirectional context. Let $\mathbf{h}_i \in R^{d_{\text{llm}}}$ denote the final-layer hidden state of token x_i from the second pass. We apply a learned projection:

$$\mathbf{t}_i = \text{MLP}_{\text{token}}(\mathbf{h}_i) \in R^{d_p} \quad (2)$$

where the MLP consists of linear layers with LayerNorm and GELU activation. This projects from the LLM’s hidden dimension ($d_{\text{llm}} = 2560$ for Qwen3-4B, 4096 for Qwen3-8B) to the shared representation space ($d_p = 256$).

3.3 Definition-Guided Entity Typing

A key component enabling zero-shot generalization is our use of natural language definitions to represent entity types. Rather than encoding entity types by their names alone (e.g., “PERSON”),

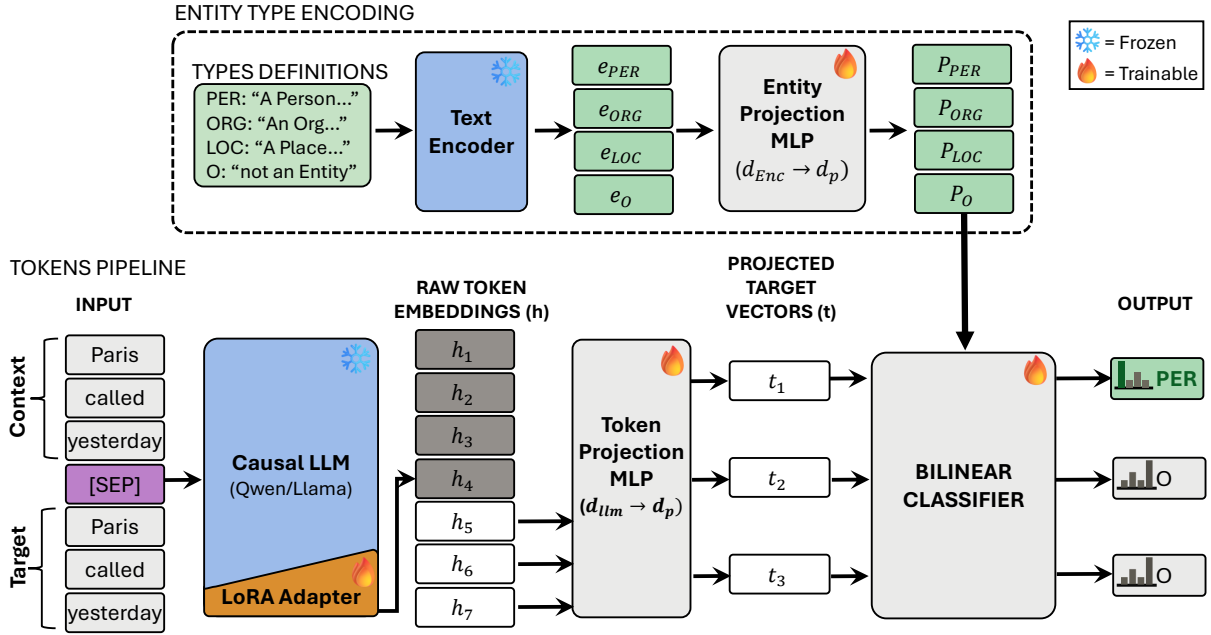


Figure 2: **Architecture of the proposed model.** (Top) Entity definitions are encoded via a text encoder (dimension d_{enc}) and projected into the shared space R^{d_p} using the Entity Projection MLP, yielding entity embeddings $\mathbf{p}_{per}, \mathbf{p}_{loc}, \dots$ (Bottom) The Causal LLM processes the duplicated input sequence. Hidden states from the second pass (h_5, \dots, h_7) are projected to token embeddings $\mathbf{t}_1, \dots, \mathbf{t}_n \in R^{d_p}$ and scored against entity embeddings via a bilinear classifier.

which relies on surface-level semantics, we encode rich definitions that specify exactly what should be tagged.

This approach enables flexible zero-shot transfer: new entity types can be specified at inference time without retraining, as the model learns to match token representations to definition semantics rather than memorizing fixed label vocabularies. Beyond generalization, definition-guided typing provides fine-grained control, allowing users to specify boundary cases directly in natural language. This makes JPT a controllable information extractor rather than a fixed NER model. We validate these properties in Section 5.1.

For each entity type j , we create a definition text def_j and encode it using a pre-trained text embedding model:

$$\mathbf{p}_j = \text{MLP}_{\text{entity}}(\text{enc}_j) \in R^{d_p} \quad (3)$$

where $\text{enc}_j = \text{Embed}(def_j) \in R^{d_{enc}}$ is the raw embedding from the text encoder (e.g., $d_{enc} = 4096$ for Qwen3-Embedding).

Definition Format. Definitions can range from simple (“A human individual’s name”) to detailed specifications that include boundary cases and disambiguation rules:

LOCATION: “Any word indicating WHERE: explicit place names (Boston, downtown), relative indicators (nearby, around), directional words (east, south side).”

This flexibility allows users to precisely control what gets tagged. For instance, they can specify whether “nearby” should be tagged as a LOCATION indicator or ignored as a common adjective.

Dual-Channel Definition Injection. In addition to encoding definitions as entity embeddings \mathbf{p}_j , we include the list of entity types and their definitions in the LLM’s input prompt (see Appendix C). This provides explicit semantic guidance during token encoding: the LLM’s attention mechanism can attend to definitions while processing each token, while the embedding space enables the classifier to match tokens to entity semantics. Our ablation study (Section 5.1) shows that both channels provide complementary benefits.

Embedding Caching. Entity type definitions remain constant at inference time, so their embeddings can be precomputed and cached, incurring no additional latency during prediction. Any text embedding model can be used, e.g., OpenAI’s text-embedding-3-small (1536-dim) or Qwen3-Embedding (4096-dim).

3.4 Classification

Given projected token representations $\mathbf{t}_i \in R^{d_p}$ and entity embeddings $\mathbf{P} = [\mathbf{p}_1, \dots, \mathbf{p}_N] \in R^{N \times d_p}$, we compute classification scores using a bilinear interaction:

$$s_{ij} = \mathbf{t}_i^\top \mathbf{W} \mathbf{p}_j + b_j \quad (4)$$

where $\mathbf{W} \in R^{d_p \times d_p}$ is a learned weight matrix and b_j is a learned type-specific bias.

We include an explicit “O” (outside/non-entity) class with a fixed embedding \mathbf{p}_O derived from the definition “A token that is not part of any named entity,” making this an $(N + 1)$ -way classification. The final prediction is:

$$\hat{y}_i = \arg \max_{j \in \{O, 1, \dots, N\}} s_{ij} \quad (5)$$

At inference, consecutive entity tokens are merged into spans, with span boundaries triggered by type changes.

3.5 Training

Parameter Efficiency. We freeze the LLM backbone and train only: (1) LoRA adapters (Hu et al., 2021) on the attention projections, (2) token and entity projection MLPs, and (3) the bilinear classifier. This yields strong performance while updating under 2% of backbone parameters (0.95% for JPT-4B, 1.71% for JPT-8B). Full details in Appendix A.

Loss Function. We use weighted classification losses (cross entropy and focal loss) to handle class imbalance, downweighting the “O” class relative to entity classes. Loss is computed only on tokens from the second pass. Full details on the loss functions are provided in Appendix A.

Training Data. We train on an in-house NER dataset derived from Wikipedia with automatic annotation, containing diverse entity types. Importantly, this dataset has *no overlap* with any evaluation benchmark, ensuring our evaluation is truly zero-shot with respect to test domains and datasets.

Full training data statistics and examples are provided in Appendix B.

4 Experiments

4.1 Experimental Setup

We evaluate on two established zero-shot NER benchmarks:

- **CrossNER** (Liu et al., 2020): Five specialized domains (AI, Literature, Music, Politics, Science) with 9–17 domain-specific entity types per domain.
- **MIT Movie/Restaurant** (Liu et al., 2013): Slot-filling NER for conversational queries; the Restaurant dataset contains 8 entity types and the Movie dataset contains 12.

We compare against state-of-the-art methods from both paradigms:

- *Generative*: UniNER-7B (Zhou et al., 2024), GoLLIE (Sainz et al., 2024), InstructUIE (Wang et al., 2023), GPT-NER (Wang et al., 2025b), SaM (Ding et al., 2025)
- *Discriminative*: GLiNER-L (Zaratiana et al., 2024)

Implementation Details. We use Qwen3-4B and Qwen3-8B as base LLMs. Entity embeddings use Qwen3-Embedding-8B. The shared projection dimension is $d_p = 256$. Training uses AdamW with learning rate 5×10^{-5} , effective batch size 8, and 5 epochs on a 4xH100 GPU machine. LoRA and projection configurations are detailed in Section 3.5.

4.2 Main Results

Table 1 presents our main results. **JPT-8B** achieves state-of-the-art performance, outperforming the previous best method (**SaM**) by +7.9 F1 overall and (**UniNER-7B**) by +12.3 F1. The largest gains appear on Music (+11.8 F1), Restaurant (+11.5 F1), and AI (+11.0 F1), domains with specialized terminology where LLM world knowledge proves particularly beneficial. Even **JPT-4B** surpasses all baselines (+4.7 F1 over SaM), with further gains from scaling to 8B, suggesting that larger LLM backbones continue to improve performance.

4.3 Extended Benchmark Results

Table 2 reports results on an extended suite of 20 datasets. **JPT-4B** outperforms both **GLiNER-L** and **UniNER-7B** on 19 out of 20 benchmarks.

4.4 Efficiency Analysis

We estimate cost using inference time \times \$5.07/hour for local models or API pricing for GPT-5, adopting the NER prompting template from Ye et al. (2023). All local models are benchmarked on the same A100 GPU with batch size 1 for fair comparison. Table 3 summarizes results.

Model	AI	Lit.	Music	Politics	Science	Movie	Rest.	Avg
<i>Generative Baselines</i>								
UniNER-7B	62.9	64.9	70.6	66.9	70.8	61.2	35.2	61.8
GoLLIE	59.1	62.7	67.8	57.2	55.5	63.0	43.4	58.4
InstructUIE	49.0	47.2	53.2	48.2	49.3	63.0	21.0	47.8
SaM (MoE)	60.9	66.9	73.5	74.4	62.6	72.1	52.9	66.2
<i>Discriminative Baselines</i>								
GLiNER-L	57.2	64.4	69.6	72.6	62.6	64.4	42.9	60.9
JPT-4B (Ours)	68.3	73.7	84.1	76.4	69.5	60.7	63.4	70.9
JPT-8B (Ours)	71.9	72.2	85.3	77.0	71.3	76.5	64.4	74.1

Table 1: Zero-shot F1 scores on CrossNER and MIT benchmarks. JPT-8B achieves the best overall average, improving by +7.9 F1 over the strongest baseline (SaM). Results for baselines are taken from prior work (Ding et al., 2025).

Dataset	UniNER-7B	GLiNER-L	JPT-4B
ACE05	36.9	27.3	44.6
AnatEM	25.1	33.3	37.2
bc2gm	46.2	47.9	54.8
bc4chemd	47.9	43.1	53.3
bc5cdr	68.0	66.4	70.4
Broad Twitter	67.9	61.2	71.2
CoNLL03	72.2	64.6	78.1
FabNER	24.8	23.6	27.8
FindVehicle	22.2	41.9	42.3
GENIA	54.1	55.5	50.8
HarveyNER	18.2	22.7	27.7
MIT Movie	42.4	57.2	73.4
MIT Restaurant	31.7	42.9	61.9
MultiNERD	59.3	59.7	65.7
NCBI	60.4	61.9	68.7
OntoNotes	27.8	32.2	43.1
PolyglotNER	41.8	42.9	47.4
TweetNER7	42.7	41.4	49.7
WikiANN	55.4	58.9	64.7
WikiNeural	69.2	71.8	77.3
Average	45.7	47.8	55.5

Table 2: Extended zero-shot F1 results across 20 NER benchmarks spanning biomedical, social media, and multilingual domains. Results for UniNER-7B and GLiNER-L are reported from (Zaratiana et al., 2024).

JPT processes all tokens in a single forward pass, while generative methods must decode token-by-token, with latency scaling with output length. **JPT-4B** is $\approx 22\times$ faster than **UniNER-7B**, and even **JPT-8B** (with a larger backbone and doubled input length) remains $\approx 13.5\times$ faster while achieving higher F1.

Why Doubling Input Length is Fast. While JPT doubles the input sequence length via concatenation (processing $2N$ tokens), this operation occurs entirely within the *prefill phase*, which exploits massive parallelism and is compute-bound, achieving high GPU utilization (Wang et al., 2025a). In contrast, generative approaches rely on the *decode phase*, which is inherently sequential and memory-

Method	Cost(\$) \downarrow	Time (sec) \downarrow	F1 \uparrow
<i>Generative Methods</i>			
UniNER-7B	2.77	1970.2	61.8
GNER	8.21	5831.5	75.8
GPT-5 \dagger	0.49	579.6	67.1
<i>Discriminative Methods</i>			
GLiNER-L	0.05	33.3	60.9
JPT-4B	0.13	89.7	76.4
JPT-8B	0.21	146.2	77.0

Table 3: Efficiency comparison on CrossNER-Politics. Cost: compute time \times \$5.07/hour (local) or API pricing (GPT). \dagger GPT results vary with prompting strategy.

bound; each generated token requires reloading model weights from memory for a single step of computation, leaving compute units underutilized (Patel et al., 2024). Consequently, a single forward pass over a duplicated input is orders of magnitude faster than autoregressively generating entity lists, effectively trading cheap “parallel” input tokens for expensive “serial” output tokens. This efficiency gain is reflected in commercial API pricing, where providers typically charge $3\text{--}5\times$ more for output tokens.

5 Ablation Studies and Analysis

5.1 Ablation Studies

We validate our two core design choices, input duplication and definition-guided typing, through ablation experiments. Table 4 summarizes results on CrossNER and MIT benchmarks.

Input Duplication. Removing input duplication (single pass) degrades performance by -15.2 F1, confirming that bidirectional context is essential for effective token classification with causal LLMs.

Entity Definitions. Using definitions in only one channel (prompt or embedding) provides limited

Configuration	Avg Micro F1
<i>Input Duplication</i>	
Single Pass	55.7
<i>Entity Definitions</i>	
No Definitions	58.3
Prompt-only definitions	63.3
Embedding-only definitions	65.2
Dual-channel definitions + Double Pass (JPT)	70.9

Table 4: Ablation studies on JPT-4B. Avg. micro-F1 across CrossNER and MIT benchmarks. Input duplication provides +15.2 F1; dual-channel definitions provide +12.6 F1 over no definitions.

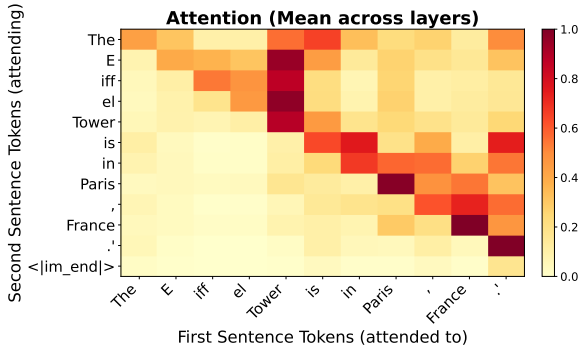


Figure 3: Attention weights averaged across all transformer layers from second-pass tokens (rows) to first-pass tokens (columns). The model attends to corresponding positions (diagonal) and semantically relevant context, suggesting effective bidirectional information flow.

gains. Jointly injecting definitions into both the prompt and embeddings yields the highest performance (+12.6 F1 over no definitions), suggesting that the two mechanisms provide complementary signals. The ability to precisely define entity boundaries proved especially valuable. For example, on MIT-Restaurant, specifying that LOCATION includes relative indicators like “nearby” improved type-specific F1 by +32.6 points (see Table 8 in Appendix D).

5.2 Understanding the Attention Patterns

The attention patterns reveal how JPT leverages bidirectional context for disambiguation. In Figure 3, subword tokens “The,” “E,” “iff,” and “el” in the second pass attend most strongly to “Tower” and to the remaining subwords of their entity from the first pass. This context appears *after* these tokens in the original sequence but becomes accessible through input duplication. This demonstrates JPT’s core mechanism: incomplete tokens use the first pass to “look ahead,” attending to complete words and surrounding context that would other-

wise be masked. This lookahead enables accurate entity boundary detection, correct type classification, and disambiguation of ambiguous mentions, allowing the model to form coherent entity representations despite the underlying causal constraint.

5.3 Error Analysis

A comprehensive error analysis including boundary detection errors, type confusion cases, and contrastive examples where SOTA baselines fail is provided in Appendix G.

6 Conclusion

We presented Just Pass Twice (JPT), a method enabling causal LLMs to perform discriminative token classification with bidirectional context via input duplication. Combined with definition-guided entity typing, JPT achieves state-of-the-art zero-shot NER results while being over 20× faster than generative alternatives. Our work demonstrates that causal attention constraints need not limit LLMs to generative approaches. The simplicity of our method suggests applicability to other token-level tasks beyond NER.

Limitations

- **Sequence length:** Input duplication doubles the effective sequence length, increasing attention complexity from $O(N^2)$ to $O((2N)^2)$, which may pose memory constraints for long contexts. In practice, this is mitigated by chunking long documents into shorter segments that can be batched together, a standard practice that JPT supports efficiently since it operates entirely in the parallel prefill phase.
- **Flat NER only:** JPT currently assigns one label per token, so nested entities are not supported and span boundaries depend on post-hoc merging of consecutive predictions. However, the architecture could be extended to support nested entities by using the sigmoid head’s independent per-type probabilities (predicting multiple types with probability > 0.5 for overlapping spans).
- **Training data:** Unlike prior zero-shot NER methods that train on existing benchmark training splits, we use an entirely separate Wikipedia-derived dataset with no overlap with any evaluation benchmark. While common entity types (e.g., PERSON, LOCATION) appear in our training data with different definitions, we validated

broad generalization across 20 diverse benchmarks spanning biomedical, social media, and multilingual domains.

References

- Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. [Contextual string embeddings for sequence labeling](#). In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA. Association for Computational Linguistics.
- Dhananjay Ashok and Zachary C. Lipton. 2023. [Prompter: Prompting for named entity recognition](#). *Preprint*, arXiv:2305.15444.
- Mohammad Bavarian, Heewoo Jun, Nikolas Tezak, John Schulman, Christine McLeavey, Jerry Tworek, and Mark Chen. 2022. [Efficient training of language models to fill in the middle](#). *Preprint*, arXiv:2207.14255.
- Sergei Bogdanov, Alexandre Constantin, Timothée Bernard, Benoit Crabbé, and Etienne P Bernard. 2024. [NuNER: Entity recognition encoder pre-training via LLM-annotated data](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 11829–11841, Miami, Florida, USA. Association for Computational Linguistics.
- Jason P.C. Chiu and Eric Nichols. 2016. [Named entity recognition with bidirectional LSTM-CNNs](#). *Transactions of the Association for Computational Linguistics*, 4:357–370.
- Alessio Cocchieri, Giacomo Frisoni, Marcos Martínez Galindo, Gianluca Moro, Giuseppe Tagliavini, and Francesco Candoli. 2025. [OpenBioNER: Lightweight open-domain biomedical named entity recognition through entity type description](#). In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 818–837, Albuquerque, New Mexico. Association for Computational Linguistics.
- 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 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Juyang Ding, Juntao Li, Pinzheng Wang, Zecheng Tang, Yan Bowen, and Min Zhang. 2024. [Rethinking negative instances for generative named entity recognition](#). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 3461–3475, Bangkok, Thailand. Association for Computational Linguistics.
- Zhuojun Ding, Wei Wei, and Chenghao Fan. 2025. [Selecting and merging: Towards adaptable and scalable named entity recognition with large language models](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9869–9886, Vienna, Austria. Association for Computational Linguistics.
- Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. [DeBERTa: Decoding-enhanced bert with disentangled attention](#). *Preprint*, arXiv:2006.03654.
- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. [Lora: Low-rank adaptation of large language models](#). *Preprint*, arXiv:2106.09685.
- Imed Keraghel, Stanislas Morbieu, and Mohamed Nadif. 2024. [Recent advances in named entity recognition: A comprehensive survey and comparative study](#). *Preprint*, arXiv:2401.10825.
- Bo Li, Gexiang Fang, Yang Yang, Quansen Wang, Wei Ye, Wen Zhao, and Shikun Zhang. 2023. [Evaluating chatgpt’s information extraction capabilities: An assessment of performance, explainability, calibration, and faithfulness](#). *Preprint*, arXiv:2304.11633.
- Jingjing Liu, Panupong Pasupat, Scott Cyphers, and Jim Glass. 2013. [Asgard: A portable architecture for multilingual dialogue systems](#). In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8386–8390.
- 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](#). *Preprint*, arXiv:1907.11692.
- Zihan Liu, Yan Xu, Tiezheng Yu, Wenliang Dai, Ziwei Ji, Samuel Cahyawijaya, Andrea Madotto, and Pascale Fung. 2020. [Crossner: Evaluating cross-domain named entity recognition](#).
- Pratyush Patel, Esha Choukse, Chaojie Zhang, Aashaka Shah, Íñigo Goiri, Saeed Maleki, and Ricardo Bianchini. 2024. [Splitwise: Efficient generative llm inference using phase splitting](#). *Preprint*, arXiv:2311.18677.
- Libo Qin, Wanxiang Che, Yangming Li, Haoyang Wen, and Ting Liu. 2019. [A stack-propagation framework with token-level intent detection for spoken language understanding](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2078–2087, Hong Kong, China. Association for Computational Linguistics.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. [Exploring the limits of transfer learning with a unified text-to-text](#)

- transformer. *Journal of Machine Learning Research*, 21(140):1–67.
- Oscar Sainz, Iker García-Ferrero, Rodrigo Agerri, Oier Lopez de Lacalle, German Rigau, and Eneko Agirre. 2024. **GoLLIE: Annotation guidelines improve zero-shot information-extraction**. In *The Twelfth International Conference on Learning Representations*.
- Yi Tay, Mostafa Dehghani, Vinh Q. Tran, Xavier Garcia, Jason Wei, Xuezhi Wang, Hyung Won Chung, Siamak Shakeri, Dara Bahri, Tal Schuster, Huaixiu Steven Zheng, Denny Zhou, Neil Houlsby, and Donald Metzler. 2023. **UI2: Unifying language learning paradigms**. *Preprint*, arXiv:2205.05131.
- Haonan Wang, Xuxin Xiao, Mingyu Yan, Zhuoyuan Zhu, Dengke Han, Duo Wang, Wenming Li, Xiaochun Ye, Cunchen Hu, Hongyang Chen, and Guangyu Sun. 2025a. **A systematic characterization of llm inference on gpus**. *Preprint*, arXiv:2512.01644.
- Shuhe Wang, Xiaofei Sun, Xiaoya Li, Rongbin Ouyang, Fei Wu, Tianwei Zhang, Jiwei Li, Guoyin Wang, and Chen Guo. 2025b. **GPT-NER: Named entity recognition via large language models**. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 4257–4275, Albuquerque, New Mexico. Association for Computational Linguistics.
- Xiao Wang, Weikang Zhou, Can Zu, Han Xia, Tianze Chen, Yuansen Zhang, Rui Zheng, Junjie Ye, Qi Zhang, Tao Gui, Jihua Kang, Jingsheng Yang, Siyuan Li, and Chunsai Du. 2023. **Instructuaie: Multi-task instruction tuning for unified information extraction**. *Preprint*, arXiv:2304.08085.
- Xiang Wei, Xingyu Cui, Ning Cheng, Xiaobin Wang, Xin Zhang, Shen Huang, Pengjun Xie, Jinan Xu, Yufeng Chen, Meishan Zhang, Yong Jiang, and Wenjuan Han. 2024. **Chatie: Zero-shot information extraction via chatting with chatgpt**. *Preprint*, arXiv:2302.10205.
- Junjie Ye, Xuanting Chen, Nuo Xu, Can Zu, Zekai Shao, Shichun Liu, Yuhan Cui, Zeyang Zhou, Chao Gong, Yang Shen, Jie Zhou, Siming Chen, Tao Gui, Qi Zhang, and Xuanjing Huang. 2023. **A comprehensive capability analysis of gpt-3 and gpt-3.5 series models**. *Preprint*, arXiv:2303.10420.
- Urchade Zaratiana, Nadi Tomeh, Pierre Holat, and Thierry Charnois. 2024. **GLiNER: Generalist model for named entity recognition using bidirectional transformer**. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5364–5376, Mexico City, Mexico. Association for Computational Linguistics.
- Yunyi Zhang, Ruozhen Yang, Xueqiang Xu, Rui Li, Jinfeng Xiao, Jiaming Shen, and Jiawei Han. 2025. **TELEClass: Taxonomy enrichment and llm-enhanced hierarchical text classification with minimal supervision**. In *WWW*. GitHub repository.
- Wenxuan Zhou, Sheng Zhang, Yu Gu, Muhao Chen, and Hoifung Poon. 2024. **UniversalNER: Targeted distillation from large language models for open named entity recognition**. In *The Twelfth International Conference on Learning Representations*.

A Model Architecture and Training Details

This section provides complete details on the JPT architecture, training configuration, and hyperparameters.

A.1 Architecture Configuration

Table 5 summarizes the architecture for both model variants. The base LLMs are frozen, with only lightweight adapters and projection layers trained.

A.2 Classifier and Loss Details

While Section 3 describes classification using a single bilinear scorer for simplicity, our implementation uses an ensemble of two classifiers for improved calibration:

- **Softmax head:** Models mutually exclusive token labels (including an explicit O-class). Trained with cross-entropy loss, where the O-class weight is reduced to $w_O = 0.25$ to handle class imbalance.
- **Sigmoid head:** Treats each entity type as an independent binary decision. Trained with focal loss ($\gamma = 2.5$, positive weight = 5.0) to focus on hard examples and upweight rare entities.

The two heads share the same projected token and entity embeddings. Final predictions are obtained by averaging their probability outputs. This ensemble provides complementary strengths: the softmax head yields decisive predictions while the sigmoid head better handles rare entities. The core method (input duplication + definition-guided typing) is independent of this design choice; the ensemble improves F1 by 1–2 points over a single softmax head.

A.3 Training Hyperparameters

Table 6 summarizes the training hyperparameters used for both **JPT-4B** and **JPT-8B**.

B Training Data

This section describes the training corpus used to train **JPT**, including dataset statistics, construction

Component	JPT-4B	JPT-8B
<i>Base LLM (Frozen)</i>		
Model	Qwen3-4B	Qwen3-8B
Hidden dim (d_{lm})	2560	4096
Layers / Attention heads	36 / 32	36 / 32
<i>LoRA Adapters (Trained)</i>		
Rank (r) / Alpha (α)	32 / 64	128 / 256
Target modules	q_proj, k_proj, v_proj, o_proj	
Parameters	\sim 23.6M	\sim 122.7M
<i>Projection MLPs (Trained)</i>		
Token MLP	$d_{lm} \rightarrow 1024 \rightarrow d_p$	$d_{lm} \rightarrow 1024 \rightarrow 512 \rightarrow d_p$
Entity MLP	$d_{enc} \rightarrow 1024 \rightarrow d_p$	$d_{enc} \rightarrow 1024 \rightarrow 512 \rightarrow d_p$
Shared dim (d_p)	256	256
Parameters	\sim 14.7M	\sim 19.4M
<i>Classifier (Trained)</i>		
Dual bilinear classifiers	\sim 131K	\sim 131K
Total trainable	\sim38.8M (0.95%)	\sim142.8M (1.71%)

Table 5: Detailed architecture configuration for JPT models. Only a small fraction of backbone parameters are trained via LoRA adapters and lightweight projection heads.

Hyperparameter	Value
Optimizer	AdamW
Learning rate	5×10^{-5}
LR scheduler	Cosine with warmup
Warmup ratio	10% of total steps
Effective batch size	8
Gradient accumulation	2
Training epochs	5
Max sequence length	4096
O-class weight (w_O)	0.25
Entity class weight	1.0
Hardware	$4 \times$ H100 GPU
Training time	\sim 1.5 hours

Table 6: Training hyperparameters used for both JPT-4B and JPT-8B.

procedure, entity-type distribution, and representative annotated examples.

B.1 Dataset Statistics

Table 7 reports the main training dataset statistics, including corpus size, token counts, and entity-type diversity.

B.2 Dataset Construction

Our training data consists of *natural text* from Wikipedia articles, sourced from the test partition of the DBpedia corpus in the TELEClass benchmark (Zhang et al., 2025). Each passage is associated with a three-level hierarchical topic taxonomy from the DBpedia ontology,

Statistic	Value
Total sentences	17,489
Total tokens	3,391,899
Total entity mentions	374,705
Unique entity types	5,009
Avg. sentence length	194.9 tokens
Entity/non-entity ratio	1:1.5

Table 7: Training dataset statistics.

progressing from broad categories (e.g., “agent,” “place”) through intermediate concepts (e.g., “athlete,” “natural_place”) to fine-grained types (e.g., “chess_player,” “mountain”)

We use Claude Sonnet 4.5 (with extended thinking) to automatically annotate these real passages in two stages:

- Type Generation:** Given a passage and its topic hierarchy, the model proposes domain-appropriate entity types with definitions (e.g., “Athlete,” “Team,” “Stadium” for sports articles).
- Entity Detection:** The model identifies entity spans in the text, followed by a gap-detection pass to catch missed mentions.

For quality validation, Claude Opus 4.5 assesses random samples across entity type appropriateness, definition actionability, and extraction accuracy. The resulting dataset comprises **17,489** training examples with **5,009** entity types and **2,500** test

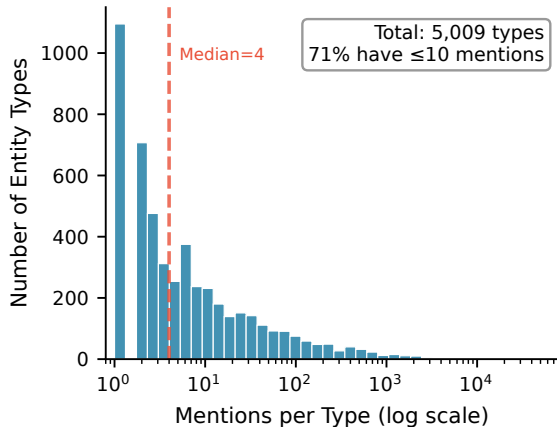


Figure 4: Distribution of entity type frequencies in training data. The long-tail distribution (71% of types have ≤ 10 mentions, median=4) encourages the model to leverage definition semantics rather than memorizing patterns.

examples with **1,947** entity types.

B.3 Entity Type Distribution

Figure 4 shows a long-tailed entity type distribution, with most types occurring infrequently, encouraging reliance on definition-based generalization rather than frequency-driven learning.

B.4 Training Examples

Figure 5 shows representative training examples with fine-grained entity annotations.

C Prompt Template

Figure 6 shows the prompt structure used during training and inference. Definitions are injected in the user turn, providing the dual-channel signal described in Section 3.3.

D Definition Engineering

This section provides guidance on crafting effective entity definitions and demonstrates their impact on recognition quality.

D.1 Impact of Definition Quality

Table 8 compares generic versus precise definitions. Precise definitions that specify boundary cases and provide examples yield substantial improvements.

D.2 Definition Writing Guidelines

Effective definitions should:

1. **Specify inclusions and exclusions:** What counts and what doesn't

2. **Provide concrete examples:** Representative instances of the type
3. **Address ambiguities:** Clarify edge cases (e.g., does “nearby” count as location?)

E Additional Ablations

This section presents additional ablation studies that analyze the effect of model scale and parameter-efficient adaptation on **JPT** performance. These experiments help characterize the trade-offs between accuracy, model capacity, and inference efficiency.

E.1 Impact of LLM Size

Table 9 examines the impact of the base LLM size on performance and inference latency. Increasing model size consistently improves token-level F1, but the gains diminish as scale grows. This highlights a practical trade-off between accuracy and efficiency, motivating the use of mid-sized backbones in resource-constrained settings.

E.2 Impact of LoRA Rank

Table 10 analyzes the effect of the LoRA rank on performance. Increasing the rank improves token-level F1, particularly when moving from frozen backbones to low-rank adaptation, but yields diminishing returns at higher ranks.

F Attention Visualization

Figure 7 provides additional attention heatmaps illustrating bidirectional information flow across different sentence structures.

G Error Analysis

We analyze common failure modes of **JPT** on the CrossNER and MIT benchmarks as well as custom examples.

G.1 Boundary Detection Errors

We observe that the most common failure mode of **JPT** involves boundary detection, where predicted spans partially overlap with the gold entities (Table 11). These errors typically manifest as over-extension or truncation of entity boundaries, often due to nearby descriptive modifiers or appositional phrases.

G.2 Entity Type Confusion

Figure 8 shows the entity-type confusion matrix, highlighting systematic confusions between semantically related categories.

Example 1: Sports Domain

“the men’s foil competition_[COMPETITION] in fencing_[SPORT] at the 2016 olympic games_[EVENT] in rio de janeiro_[CITY] was held on 7 august_[DATE] at the carioca arena 3_[VENUE]. the medals were presented by paul tergat_[PERSON], ioc_[ORG] member, kenya_[COUNTRY]”

Example 2: Natural Disaster Domain

“the 2014 mae lao earthquake_[EARTHQUAKE] occurred at 18:08:43_[TIME] indochina time_[TIMEZONE] on may 5_[DATE]. the epicenter was located 9 km_[DISTANCE] south of mae lao district_[DISTRICT], 27 km_[DISTANCE] southwest of chiang rai_[CITY], thailand_[COUNTRY]”

Example 3: Biographical Domain

“helga mühlberg-ulze_[ATHLETE] is an east german_[NATIONALITY] sprint canoeist_[SPORTDISC] who competed in the early to mid 1960s_[TIMEPERIOD]. she won gold_[MEDAL] (k-2 500m_[RACECAT]: 1966_[YEAR]) and two bronzes_[MEDAL]”

Figure 5: Training examples showing fine-grained entity types across diverse domains. Entity spans are underlined in blue with type labels in subscript. The dataset contains 5,009 unique entity types including domain-specific categories like EARTHQUAKE, TIMEZONE, and RACECATEGORY.

Type	Generic Definition	Precise Definition	$\Delta F1$
LOCATION	“A geographical place”	“Any word indicating WHERE: explicit places (NYC, downtown), relative indicators (nearby, around, close by), directional phrases (east, south side). Tag the location word itself.”	+32.6
PRICE	“A monetary value”	“Explicit monetary amounts (\$50, 100 dollars) AND qualitative price indicators (cheap, expensive, budget-friendly, overpriced, pricey).”	+34.9
AMENITY	“An available service”	“A feature, facility, or service offered by a restaurant. Includes: physical features (bar, parking), services (delivery, reservations, takeout), atmosphere descriptors (romantic, casual, family-friendly).”	+14.29

Table 8: Generic vs. precise entity definitions and their impact. F1 computed per-type on MIT Restaurant. Precise definitions with boundary cases and examples dramatically improve recognition.

Model	Params	Token F1
Qwen3-1.7B	1.7B	92.4
Qwen3-4B	4B	93.9
Qwen3-8B	8B	94.1
Qwen3-14B	14B	94.5

Table 9: Impact of base LLM size on token-level F1 (private evaluation set). Larger models yield consistent improvements, with diminishing returns beyond 8B parameters.

G.3 Failure Examples

Table 12 presents representative failure cases from **JPT-4B** across diverse benchmarks. These examples illustrate three systematic error patterns: type confusion between semantically related categories, missed entities in atypical contexts, and over-predicted entities where plausible mentions lack ground-truth annotations.

These failure modes suggest that errors primarily stem from surface-form ambiguity and fine-grained

LoRA Rank	Token F1
$r = 0$ (frozen)	82.0
$r = 16$	93.5
$r = 32$ (default)	93.9
$r = 64$	94.2
$r = 128$	94.5
$r = 256$	94.7

Table 10: Impact of LoRA rank on token-level F1 using JPT-4B (private evaluation set). Adaptation is essential ($r = 0$ drops 12 points), but returns diminish beyond $r = 32$.

semantic overlap rather than lack of contextual understanding. Type confusions often occur between closely related categories with overlapping definitions, missed entities frequently appear in descriptive or adjectival forms, and over-predictions arise when domain terms resemble entity mentions. Refining type definitions and adding targeted training examples for ambiguous and rare constructions

```

<|im_start|>system
You are an information-extraction assistant.
Task: Perform Named Entity Recognition (NER) on the user-supplied text.
The user will give you the supported entity types and their definitions.
You will read the types and definitions to understand what each entity type means.
The user will give you the text twice in the format "The first time: 'actual text' The second
time: 'actual text'".
Output Format: Output ONE annotated text with entities as <entity_text, ENTITY_TYPE>
Rules:
(1) Keep multi-word entities together;
(2) Only use provided types;
(3) Output once;
(4) No bare-noun labelling (e.g., don't label "museum" unless part of proper name);
(5) Output types exactly as listed;
(6) Only label if clearly matches definition.
<|im_end|>

<|im_start|>user
Supported entity types (3): ["PERSON", "ORGANIZATION", "LOCATION"]
Entity type definitions:
- "PERSON": "A named individual, including fictional characters"
- "ORGANIZATION": "A company, institution, or group with a formal name"
- "LOCATION": "A geographical place such as a city, country, or landmark"
<|im_end|>

<|im_start|>assistant
I have read the definitions. Please provide the text in the format 'The first time: <text> The
second time: <text>'
<|im_end|>

<|im_start|>user
The first time: '<Input_Sequence>' The second time: '<Input_Sequence>'
<|im_end|>

```

Figure 6: Complete prompt template for **JPT**. The system prompt specifies output format and labeling rules. Entity definitions are injected in the first user turn, and the input text is duplicated with explicit markers in the second user turn.

may reduce these errors.

G.4 SOTA Failures

Figure 9 provides contrastive disambiguation cases in which **JPT-4B** correctly predicts all entity spans and types, while **GLiNER** and **UniNER** exhibit errors. These examples highlight the benefit of definition-guided modeling for resolving polysemy in context.

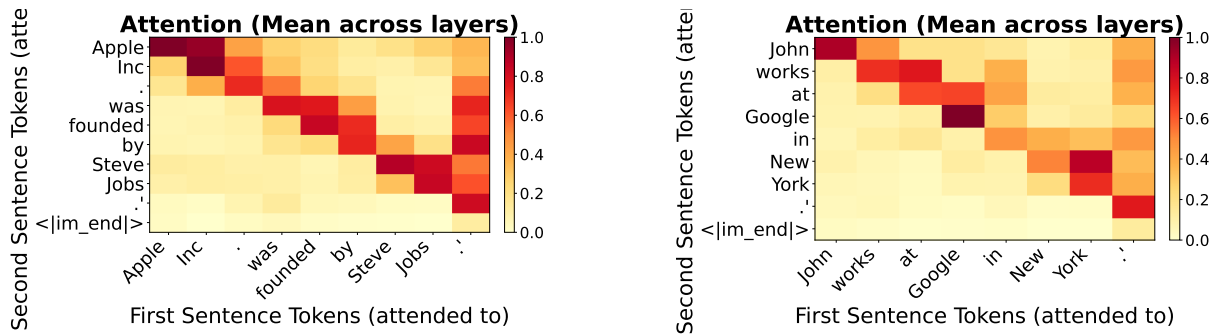


Figure 7: Additional attention visualizations. Rows: second-pass tokens. Columns: first-pass tokens. The diagonal pattern (self-attention across passes) combined with off-diagonal attention to context tokens demonstrates effective bidirectional information flow.

Dataset	Entity Type	Gold Span	Predicted Span
CrossNER-Science	ORGANIZATION	Virgo interferometer	Virgo interferometer collaboration
CrossNER-Science	ACADEMIC JOURNAL	the Springer Series Complexity	of the Springer Series Complexity
CrossNER-Politics	POLITICAL PARTY	Social Credit Party of Canada	national Social Credit Party of
CrossNER-Music	BAND	Collective Consciousness Society	The Collective Consciousness
CrossNER-Literature	LITERARY GENRE	Germany Romanticism	Romanticism era
MIT-Movie	TITLE	the reflecting skin	reflecting skin movie
MIT-Restaurant	LOCATION	within a mile	a mile of here

Table 11: Representative boundary detection errors across CrossNER and MIT. JPT often predicts the correct type and core mention but may over-extend or truncate spans when entities appear with modifiers, appositions, or colloquial phrasing.

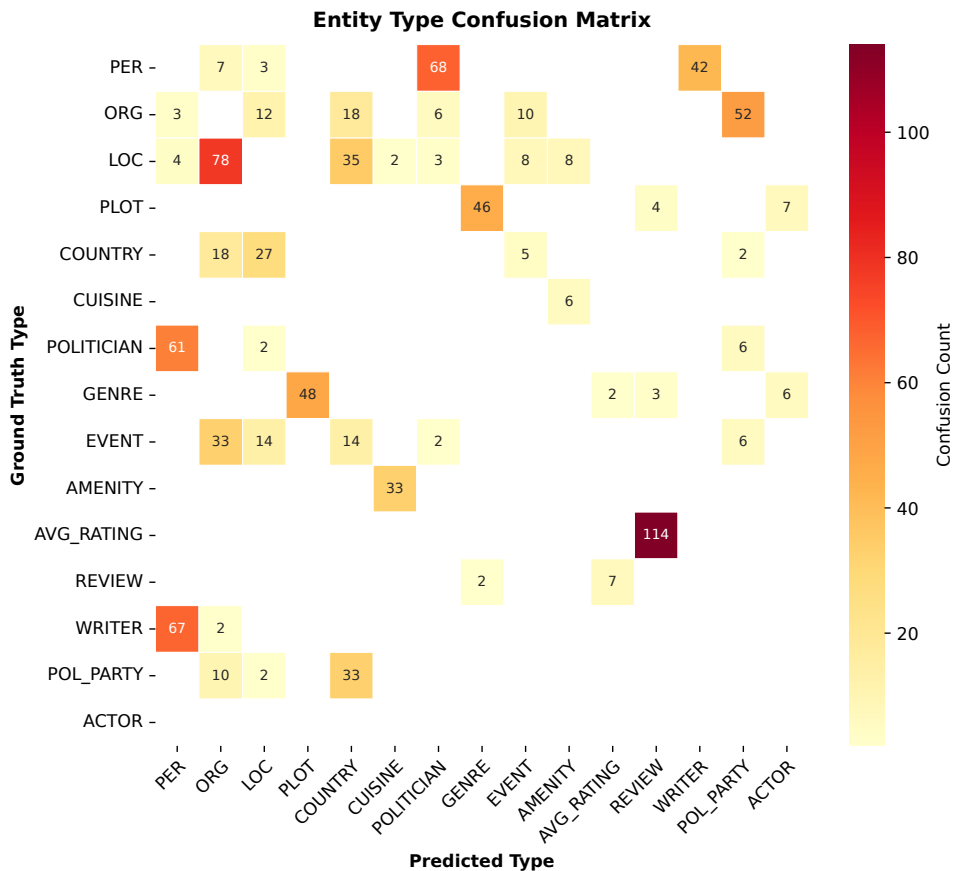


Figure 8: Entity type confusion matrix aggregated across CrossNER and MIT. Most confusions occur between semantically adjacent types (e.g., PER vs. ORG, LOC vs. COUNTRY, and POL_PARTY vs. POLITICIAN), suggesting that errors often stem from fine-grained boundary cases and overlapping semantic definitions rather than arbitrary label flips.

<p>Example 1: Dutch Universities (organization - completely missed)</p> <p><i>“This school initially consisted of nearly 200 faculty members and Ph.D. students from the <u>Vrije Universiteit</u>, <u>University of Amsterdam</u>, <u>Delft University of Technology</u>, and <u>Leiden University</u>.”</i></p> <p>Ground Truth: Vrije Universiteit, University of Amsterdam, Delft University of Technology, Leiden University → ORGANIZATION</p> <p>Model Errors: GLiNER: Missed all four universities entirely UniNER: Missed all four universities entirely JPT: Predicted all entities correctly</p>
<p>Example 2: Canadian Political Parties (political party vs organization)</p> <p><i>“In the 2006 Canadian federal election in Canada, the <u>Liberal Party of Canada</u> used attack ads against <u>Conservative Party of Canada</u> leader Stephen Harper.”</i></p> <p>Ground Truth: Liberal Party of Canada, Conservative Party of Canada → POLITICAL PARTY</p> <p>Model Errors: GLiNER: Missed both political parties entirely UniNER: Over-predicted types for both parties (includes ORGANIZATION in addition to POLITICAL PARTY) JPT: Predicted all entities correctly</p>
<p>Example 3: US Political Leaders (politician vs person)</p> <p><i>“<u>Lincoln</u> replaced <u>Buell</u> with <u>William Rosecrans</u>; and after the 1862 and 1863 United States House of Representatives elections he replaced <u>McClellan</u> with <u>Ambrose Burnside</u>.”</i></p> <p>Ground Truth: Lincoln, Buell, William Rosecrans, McClellan, Ambrose Burnside → POLITICIAN</p> <p>Model Errors: GLiNER: Labeled all five as PERSON instead of POLITICIAN UniNER: Mixed predictions per name due to multi-type outputs; includes PERSON in addition to POLITICIAN JPT: Predicted all entities correctly</p>
<p>Example 4: Historical Sovereign State (country vs organization)</p> <p><i>“The [attack] was part of the strategic bombing campaign waged by the United States of America against military and civilian targets and population centers of the <u>Empire of Japan</u> during the Japan home islands campaign in the closing stages of World War II.”</i></p> <p>Ground Truth: Empire of Japan → COUNTRY</p> <p>Model Errors: GLiNER: Labeled as ORGANIZATION instead of COUNTRY UniNER: Mixed predictions (includes ORGANIZATION in addition to COUNTRY) JPT: Predicted all entities correctly</p>

Figure 9: Examples from CrossNER Politics where our model correctly identifies fine-grained entity types. GLiNER confuses adjacent categories (POLITICAL PARTY/ORGANIZATION, POLITICIAN/PERSON, COUNTRY/ORGANIZATION). Because UniNER is inferred per entity type, it is more prone to over-prediction (multiple types per mention). Our model leverages type definitions to resolve these distinctions.

Error Type	Dataset	Text (excerpt)	Gold	Pred.	Analysis
Type Confusion	CrossNER-AI	“NIST also differs from Bilingual evaluation understudy in its calculation...”	METRICS	CONFERENCE	Acronym ambiguity
Type Confusion	CrossNER-Politics	“The provinces ceded to Augustus for that ten-year period...”	PERSON	POLITICIAN	Role vs. entity
Type Confusion	CrossNER-Music	“... on The Clash’s London Calling...”	ALBUM	LOCATION	Title–location ambiguity
Missed Entity	CrossNER-AI	“Rethink Robotics introduced Baxter as an industrial robot...”	PRODUCT	(none)	Descriptive form
Missed Entity	CrossNER-Politics	“growth in the Melanesian countries of Solomon Islands...”	LOCATION	(none)	Adjectival reference
Over-prediction	CrossNER-AI	“A frame language is a technology used for knowledge representation...”	(none)	PROG. LANG.	Over-generalization
Over-prediction	CrossNER-Politics	“reforms to the ALP under Gough Whitlam...”	(none)	POL. PARTY	Unannotated entity

Table 12: Representative JPT-4B error instances with context excerpts, gold labels, predictions, and diagnostics.