Learning Nested Named Entity Recognition from Flat Annotations

Igor Rozhkov
Lomonosov Moscow State University
Leninskie Gory, 1/4, Moscow, Russia
fulstocky@gmail.com

Natalia Loukachevitch
Lomonosov Moscow State University
Leninskie Gory, 1/4, Moscow, Russia
louk_nat@mail.ru

Abstract

Nested named entity recognition identifies entities contained within other entities, but requires expensive multi-level annotation. While flat NER corpora exist abundantly, nested resources remain scarce. We investigate whether models can learn nested structure from flat annotations alone, evaluating four approaches: string inclusions (substring matching), entity corruption (pseudo-nested data), flat neutralization (reducing false negative signal), and a hybrid fine-tuned + LLM pipeline. On NEREL, a Russian benchmark with 29 entity types where 21% of entities are nested, our best combined method achieves 26.37% inner F1, closing 40% of the gap to full nested supervision. Code is available at https://github.com/fulstock/Learning-from-Flat-Annotations.

1 Introduction

Named entity recognition (NER) is a fundamental task in information extraction (Tjong Kim Sang and De Meulder, 2003; Ratinov and Roth, 2009; Lample et al., 2016). While conventional NER systems assume non-overlapping entity spans, real-world text frequently contains nested entities where one mention is contained within another. For example, in "Ministry of Foreign Affairs of the United Kingdom," the entity "United Kingdom" (COUNTRY) is nested within the larger ORGANIZATION. Recognizing such nested structures can benefit downstream tasks (Finkel and Manning, 2009; Lu and Roth, 2015): relation extraction may leverage more complete entity information, entity linking—disambiguate entities at multiple nesting levels, and knowledge graph construction can capture more finer-grained entity relationships.

However, nested NER requires expensive annotation where annotators must identify entity mentions at every level of nesting. Early nested NER datasets include ACE 2004 (Doddington et al., 2004) and GENIA (Kim et al., 2003), with larger-scale resources appearing more recently (Ringland et al., 2019; Loukachevitch et al., 2021). The annotation cost creates a practical barrier: creating nested annotations requires much more effort than flat annotation. Furthermore, recent attempts to use large language models for automatic nested annotation have shown limited success, with LLMs struggling to maintain consistency across nesting levels (Kim et al., 2024).

In contrast, flat NER corpora—marking only non-overlapping entities—exist abundantly across languages and domains. Years of NER research have produced flat datasets for dozens of languages, various domains (news, biomedical, legal, social media), and different entity type systems. This abundance motivates our research questions:

- **RQ1:** Can we train models to recognize nested entities using only flat annotations?
- **RQ2:** Can large language models help bridge the gap between flat and nested supervision?
- **RQ3:** Can large language models compensate for missing nested annotations?

If successful, these approaches would enable leveraging existing flat NER resources for nested recognition without additional annotation effort.

We investigate these questions using NEREL (Loukachevitch et al., 2021), a Russian nested NER benchmark with 29 entity types—significantly more complex than typical nested NER datasets with 4–8 types. In NEREL, 21% of all entity mentions are nested within other entities. We systematically compare methods for recovering nested structure from flat annotations:

1. *Inclusions*: identifying nested entities through substring matching across the corpus.
2. *Entity corruption*: creating pseudo-nested training data by corrupting tokens within entities.

3. *Flat neutralization*: reducing false negative signal from unlabeled nested positions.
4. *Hybrid fine-tuned + LLM*: using fine-tuned models for outer entities and LLMs for nested detection.

Our experiments reveal several findings. String inclusions recover substantial nested structure, improving inner entity F1 from 3.84% to 21.36%. Among entity corruption strategies, end-position corruption consistently outperforms alternatives. Combining all three fine-tuning methods—inclusions, corruption, and neutralization—achieves 26.37% inner F1, closing 40% of the gap to full supervision. The hybrid fine-tuned + LLM approach achieves 70.16% overall F1 but underperforms fine-tuned methods on inner entities, indicating that current LLMs struggle with fine-grained nested structure across many entity types.

Our contributions are: (1) a systematic comparison of methods for learning nested NER from flat annotations (inclusions, corruption, neutralization) on a challenging 29-type Russian benchmark; (2) analysis of entity corruption strategies, finding that end-position corruption consistently outperforms alternatives; (3) a hybrid fine-tuned + LLM pipeline with evaluation of its limitations; (4) empirical evidence that simple methods close 40% of the gap between flat and full supervision.

2 Related work

Multiple approaches address nested NER: span-based biaffine parsing (Yu et al., 2020), layered models (Ju et al., 2018; Wang et al., 2020), set prediction (Tan et al., 2021), hypergraph methods (Yan et al., 2023), and bi-encoder approaches with contrastive learning (Zhang et al., 2023b) that represent entity types through natural language descriptions. All these approaches assume fully-annotated nested training data.

When annotations are incomplete, distant supervision (Lison et al., 2020; Meng et al., 2021) and data augmentation (Dai and Adel, 2020) provide alternatives. Our inclusions and corruption methods can be viewed as augmentation strategies specifically designed for generating pseudo-nested training signal.

As for learning nested entities from flat annotations, Zhu et al. (2022) propose excluding within-entity spans from negative sampling during training, achieving 54.8% nested F1 on ACE 2004. We build on this insight with a content-aware variant that selectively neutralizes spans matching known entity surface forms (Section 3.3.3). Rozhkov and Loukachevitch (2025) investigate this task on nested terms, demonstrating growing interest in learning nested structures from flat supervision.

LLM-based approaches (Wang et al., 2023) show prospect for NER but struggle with structured extraction tasks. Han et al. (2023) demonstrate that ChatGPT significantly underperforms fine-tuned models on information extraction, with performance degrading further for complex structures and large type inventories. These limitations motivate hybrid approaches combining fine-tuned models with LLM reasoning.

Several corpora support nested NER research. Early benchmarks include ACE 2004/2005 (Doddington et al., 2004) with 7 English entity types and GENIA (Kim et al., 2003) for biomedical English. NNE (Ringland et al., 2019) extends to 114 fine-grained types. Recent work has expanded nested NER to diverse languages and domains: DaN+ for Danish (Plank et al., 2020), Wojood for Arabic (Jarrar et al., 2022), historical documents (Tual et al., 2023), judicial Chinese (Zhang et al., 2023a), and BioNNE for Russian biomedical texts (Davydova et al., 2024). For Russian general-domain text, NEREL (Loukachevitch et al., 2021) provides 56K nested annotations across 29 types—the largest type inventory among major nested NER benchmarks—with 21% of entities nested. The RuNNE-2022 shared task (Artemova et al., 2022) established evaluation benchmarks on NEREL.

Building on these foundations, we systematically compare methods for learning nested NER from flat annotations on this challenging 29-type benchmark, including content-aware neutralization and hybrid fine-tuned+LLM pipelines.

3 Methodology

Figure 1 presents an overview of all approaches investigated in this work. We explore four complementary methods for recovering nested entities from flat annotations, organized into three categories: data augmentation techniques that create pseudo-nested training signal (Section 3.2), training modifications that adjust how the model learns from flat data (Section 3.3), and LLM-based approaches that leverage large language models (Section 3.4).

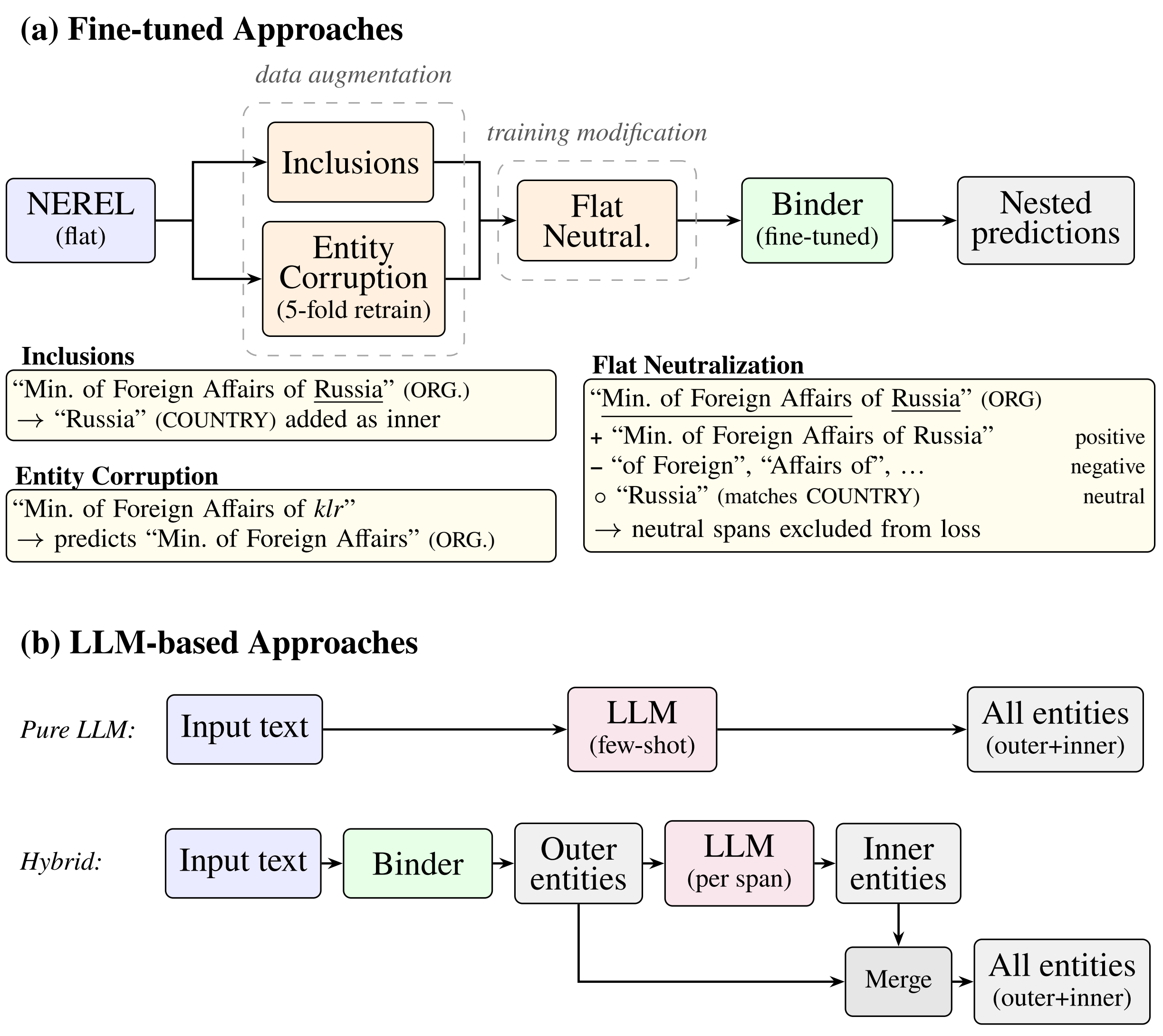
(a) Fine-tuned Approaches
data augmentation
training modification
NEREL (flat)
Inclusions
Entity Corruption (5-fold retrain)
Flat Neutral.
Binder (fine-tuned)
Nested predictions
Inclusions
"Min. of Foreign Affairs of Russia" (ORG.)
→ "Russia" (COUNTRY) added as inner
Entity Corruption
"Min. of Foreign Affairs of klr"
→ predicts "Min. of Foreign Affairs" (ORG.)
Flat Neutralization
"Min. of Foreign Affairs of Russia" (ORG)
+ "Min. of Foreign Affairs of Russia" positive
– "of Foreign", "Affairs of", … negative
○ "Russia" (matches COUNTRY) neutral
→ neutral spans excluded from loss
(b) LLM-based Approaches
Pure LLM:
Input text
LLM (few-shot)
All entities (outer+inner)
Hybrid:
Input text
Binder
Outer entities
LLM (per span)
Inner entities
Merge
All entities (outer+inner)

Figure 1: Overview of methods for learning nested NER from flat annotations. (a) Fine-tuned approaches augment flat data with pseudo-nested signal (inclusions, entity corruption) and modify training (flat neutralization) before training a Binder model. (b) LLM-based approaches use either a pure few-shot LLM or a hybrid pipeline where a fine-tuned Binder detects outer entities and an LLM identifies inner entities within each span.

3.1 Problem formulation

Given a text sequence $x = (x_1, x_2, \ldots, x_n)$ of n tokens, nested NER aims to identify all entity spans $e = (i, j, t)$ where i and j are the start and end positions, and t is the entity type. In nested NER, multiple entities can share the same tokens, allowing for overlapping and hierarchical structures.

In contrast, flat NER requires that no two mentions overlap. When converting nested annotations to flat annotations, we retain only the outermost mentions, removing all nested mentions within them. This creates a dataset where $\forall e_1, e_2$: either $e_1 \cap e_2 = \emptyset$ or one entity completely contains the other (but only the outer one is annotated).

3.2 Data augmentation

We first describe two methods that augment flat training data with pseudo-nested annotations.

3.2.1 Inclusions

We add inclusions to the training data—subsequences of flat mentions that match the surface form of other mentions in the dataset.

Formally, for each flat mention $e_i = (s_i, t_i)$ with text span s_i and type t_i, we extract all other flat mentions $\{e_j | j \neq i\}$ from the training set, find all strict substring matches (if s_j appears as a substring within s_i and $s_j \neq s_i$, create a new mention at that position with type t_j), and add these new mentions to the training data as positive examples.

For morphologically rich languages like Russian, the same entity may appear in different grammatical cases. We therefore also evaluate *lemmatized inclusions*: each mention is tokenized and each token is reduced to its dictionary form using pymorphy2 (Korobov, 2015), producing a canonical representation (sorted lemmatized tokens). Substring matching is then performed on these canonical forms rather than raw surface strings, recovering additional inclusions across inflectional variants. Lemmatization is applied only during training data preparation; at test time the model predicts spans directly from raw text.

3.2.2 Entity corruption

Another approach to create nested annotations is through "entity corruption". The idea is to train the model to recognize that when a word within a long entity is corrupted (replaced with a noise token), the remaining parts might still be valid entities.

We experiment with two corruption strategies, *early damage* and *late damage*.

In *early damage* strategy, we split the training data into 5 folds. For each fold, we train the model on 80% of the data where long flat mentions (length $>$ 2 words) have one word corrupted, predict on the remaining 20% (uncorrupted), combine predictions from all folds to create pseudo-nested annotations, and retrain on the combination of flat and pseudo-nested data.

In *late damage* strategy, similar to first, but we train the model on 80% of uncorrupted data, then predict on the remaining 20% with corrupted entities. The model learns to identify entities within corrupted contexts.

For illustration purposes, let us consider the flat entity "Ministry of Foreign Affairs of Russia" (ORGANIZATION). With end-position corruption using letters, the last word is replaced: "Ministry of Foreign Affairs of *klr*". In Strategy 2, a model trained on clean data predicts on this corrupted sentence and may recognize "Ministry of Foreign Affairs" as ORGANIZATION—a pseudo-nested annotation. Across the corpus, such predictions are collected and used as additional training signal.

Corruption symbols: We explore five types of replacement symbols: digits (e.g., "798")—random digit sequences unlikely to form meaningful words; letters (e.g., "klr")—random consonant sequences that violate phonotactic rules; diglets (e.g., "9z2")—mixed digits and letters for maximum "jumbledness"; semicolons (e.g., ";;;")—punctuation marks that signal phrase boundaries; and commas (e.g., ",,,"). The motivation for alphanumeric corruption is that these sequences are too "jumbled" to be interpreted as real words, acting as neutral placeholders. Punctuation-based corruption was hypothesized to provide stronger boundary signals but also affects subword tokenization more aggressively.

Corruption positions: We experiment with five positions within multi-word entities: start (corrupt the first word), end (corrupt the last word), middle (corrupt the middle word), random (corrupt a randomly selected word), and syntax (corrupt the syntactic root identified using dependency parsing). Different positions test different hypotheses: corrupting the end position preserves the entity's head word (often the first word in Russian noun phrases), while corrupting the start tests whether the model can recognize entities from their modifiers alone.

3.3 Fine-tuned model and training modifications

3.3.1 Binder model

Our baseline approach trains a nested NER model (Binder (Zhang et al., 2023b)) on flat annotations. Binder is a bi-encoder span-based model that achieves state-of-the-art results on several nested NER benchmarks including ACE 2004 and GENIA. It represents entity types through natural language prompts and predicts entities for all possible spans in a sentence using contrastive learning.

During training, Binder classifies each candidate span as either a positive entity match or a negative (non-entity) example. When training on flat data, the model learns to classify outer mention spans as positive and all other spans—including potential nested mentions—as negative.

The key challenge is that subsequences within flat mentions are treated as negative examples during training, even though they might be valid nested mentions in the true annotation. Our neutralization method (Section 3.3.3) addresses this by introducing a third category: neutral spans that are neither positive nor negative during training.

3.3.2 Binder prompts

The Binder model uses natural language *type descriptions* to represent entity types in its bi-encoder architecture (distinct from LLM prompting in Section 3.4). We evaluate six description strategies, following (Rozhkov and Loukachevitch, 2023): keyword (entity type name only), definition (natural language definition), most-frequent-class (most common entity example per type), context (sentence contexts), lexical-all-outer (full sentences with entity markers), and struct-nested (structural nesting information). This verifies that our weak supervision methods generalize across different type description configurations. Full details are provided in Appendix E.

3.3.3 Flat neutralization

Zhu et al. (2022) show that excluding within-entity spans from negative sampling improves nested entity recognition from flat supervision. Their approach uniformly ignores all spans geometrically contained within annotated entities. We propose a content-aware variant: rather than ignoring all within-entity spans, we selectively neutralize only those matching known entity surface forms via inclusion matching (Section 3.2.1), retaining negative signal for non-matching spans.

Specifically, we partition candidate spans into three sets: positive $\mathcal{P}$ (annotated mentions), negative $\mathcal{N}$ (spans not matching any known entity), and neutral $\mathcal{U}$ (within-entity spans matching known entities via inclusion). The training loss excludes neutral spans:

$$\mathcal{L}_{\text{neutral}}(s) = \begin{cases} \mathcal{L}(s) & \text{if } s \in \mathcal{P} \cup \mathcal{N} \\ 0 & \text{if } s \in \mathcal{U} \end{cases} \qquad (1)$$

This approach can be combined with inclusions: adding matched spans as positive examples while neutralizing remaining potential nested positions.

3.4 LLM-based approaches

3.4.1 Pure LLM

We investigate prompt-based approaches using DeepSeek-R1-32B (Guo et al., 2025) and RuAdapt-Qwen2.5-32B (Tikhomirov and Chernyshov, 2024; Tikhomirov and Chernyshev, 2023). We selected DeepSeek-R1-32B for its strong reasoning capabilities and RuAdapt-Qwen2.5-32B as a Russian-adapted model, both suitable for local deployment without API costs. Prompts explicitly define two nesting types (Kim et al., 2024): NDT (different type nesting) and NST (same type hierarchy), with definitions for all 29 entity classes (Appendix A). Output format is JSON: `{entity : type}`.

We compare zero-shot, one-shot, and five-shot configurations with three example selection strategies:

Random sampling: Examples selected uniformly at random from the training set.

Random: Examples selected uniformly at random from the training set.

MFE: Sentences containing the most frequently occurring entities, ensuring coverage of common types. Top entities: *США (USA)*, *России (Russia)*, *Москве (Moscow)*, *Владимир Путин (Vladimir Putin)*, ...

MFE-entwise: For each of the 29 entity types, the top examples by frequency are selected, ensuring balanced coverage. The per-type list appended to the prompt:

CITY:	*Москве, Нью-Йорке, Лондоне (Moscow, New York, London)*
COUNTRY:	*США, России, Россия (USA, of Russia, Russia)*
PERSON:	*Владимир Путин, Путин, Обама (Vladimir Putin, Putin, Obama)*

... [all 29 types]

MFE-entwise-sent: Same as MFE-entwise, but operates at sentence level with morphological normalization, grouping entity mentions by lemmatized forms: *России/Россия/Россию (of Russia / Russia / Russia$_{acc}$) $\rightarrow$ Россия (Russia).*

Figure 2: Example selection strategies for LLM prompts. MFE selects sentences by entity frequency; MFE-entwise adds a per-type entity list; MFE-entwise-sent additionally applies morphological normalization.

Most-frequent-entity (MFE): Sentences containing the most frequently occurring entities, ensuring coverage of common types.

Entity-wise selection (MFE-entwise): For each of the 29 entity types, we select top examples by frequency, ensuring balanced coverage. The variant MFE-entwise-sent operates at sentence level with morphological normalization, grouping entity mentions by lemmatized forms to handle Russian inflection.

Figure 2 illustrates the key differences between strategies.

To improve few-shot effectiveness, we documented characteristic nesting patterns for each entity class (Appendix B).

3.4.2 Hybrid fine-tuned + LLM

We propose a two-stage hybrid pipeline. First, a fine-tuned Binder model processes the input text to detect outer entities (82–83% F1 on this subtask). Second, for each detected outer entity span, we extract the text and query an LLM to identify nested entities within that span. The LLM receives the outer entity text, its predicted type, and a prompt describing common nesting patterns for that type (Appendix B). Predicted nested entities are merged with outer entities from stage one.

This decomposition leverages the strengths of both approaches: fine-tuned models excel at boundary detection and type classification, while LLMs can apply reasoning about entity composition. We test three variants: (a) *type-specific*: prompts tailored to each outer entity type's nesting patterns, (b) *full prompts*: comprehensive prompts covering all 29 types, and (c) *lemmatization matching*: morphological normalization to handle Russian inflection.

Inclusions and entity corruption were introduced in our prior work on nested term extraction (Rozhkov and Loukachevitch, 2025), which we adapt here to the more challenging setting of 29-type named entity recognition. Flat neutralization builds on Zhu et al. (2022)'s insight of excluding within-entity spans from negative sampling, with our contribution being content-aware selection via inclusion matching. The hybrid fine-tuned+LLM pipeline is novel to this work. Our primary contribution is the systematic comparison and combination of these methods on a challenging benchmark, along with new analysis of corruption position effects.

4 Experimental setup

4.1 Datasets

NEREL is a Russian nested NER dataset containing 29 entity types including PERSON, ORGANIZATION, LOCATION, DATE, etc. The dataset consists of news texts with rich nested entity annotations: train contains 44,680 entities (35,340 outer, 9,340 inner), dev contains 5,541 entities (4,587 outer, 954 inner), and test contains 5,790 entities (4,745 outer, 1,045 inner). Nesting depth reaches up to 6 levels, though the vast majority of nesting is shallow: across all splits, 83.4% of inner entities are at depth 2 (directly inside an outer entity), with only 3.5% of all entities at depth 3 or beyond. The RuNNE-2022 shared task (Artemova et al., 2022) evaluated nested NER on NEREL with a reduced training set; our full-supervision Binder baseline surpasses shared task results, so we use it as the upper bound. For flat training, we created the NEREL-outerflat dataset by removing all inner entities, retaining only the 35,340 outermost entities in the training set.

4.2 Model

We use Binder (described in Section 3.3.1) built on RuRoBERTa-large (Zmitrovich et al., 2024). Training uses the AdamW optimizer (Loshchilov and Hutter, 2019) with a learning rate of 1e-5, batch size of 8, and 64 epochs on a single NVIDIA RTX 4090 GPU. We run each experiment 5 times with different random seeds and report mean ± standard deviation. LLM experiments use DeepSeek-R1-32B and RuAdapt-Qwen2.5-32B (both 32B parameters), served locally on modern accelerators using the vLLM framework (v0.8.3). Each LLM experiment is run with 3 seeds.

For LLM experiments, we use temperature of 0 for near-deterministic outputs, repetition penalty of 1.05–1.1, top-p of 1.0, and maximum output length of 5,000 tokens. LLMs return a JSON dictionary `{entity: type}` enclosed in triple backticks. We extract text between backticks, strip formatting artifacts, and parse as JSON. Each predicted entity string is matched to the source text by exact string occurrence to determine character offsets; if a predicted string does not appear in the source text, it is discarded. Predicted types not in the 29-type inventory are retained but will not match any gold entity during evaluation. Failed JSON parses are treated as empty predictions (no entities for that input).

4.3 Evaluation metrics

Following standard practice in nested NER evaluation (Lu and Roth, 2015; Zhang et al., 2023b), we report micro F1 and macro F1 scores computed separately for three entity categories.

For each predicted and gold entity, we classify it as inner or outer based on containment relationships within the same set (predictions or gold, respectively): an entity e_i is inner if there exists another entity e_j in the same set such that e_j fully contains e_i (i.e., start$(e_j) \leq$ start(e_i) and end$(e_i) \leq$ end(e_j)), and outer otherwise. This means inner/outer classification is performed independently for gold and predicted entities. A predicted entity is compared against other predicted entities to determine if it is inner, and similarly for gold entities.

We compute overall F1 over all entities, inner F1 computed only on inner gold entities vs. inner predicted entities, and outer F1 computed only on outer gold entities vs. outer predicted entities. Micro F1 aggregates true positives, false positives, and false negatives across all documents before computing F1. Macro F1 computes F1 per document and averages.

5 Results

5.1 Inclusion statistics

For NEREL, we extracted 6,481 inclusions from the flat training data (18.34% of the 35,340 flat mentions). The distribution across entity types is highly skewed, with PERSON (2,851), PROFESSION (888), ORGANIZATION (668), and COUN-

Training	Prompt	Ovrl.	Inner	Outer
Flat	Keyword	76.59	3.84	83.28
Flat	Lex-all-out	76.72	3.11	83.40
Flat+Incl.	Keyword	76.70	21.36	83.05
Flat+Incl.	Lex-all-out	76.93	22.16	83.17
Full nested	Keyword	**85.81**	**65.48**	**83.95**
Full nested	Lex-all-out	85.55	65.33	83.59

Table 1: Baseline comparison (Micro F1, %) on NEREL test set. Flat and Flat+Inclusions represent weak supervision; Full nested is the upper bound with complete annotations. Full prompt comparison in Appendix E.

TRY (622) being the most frequent. Similar patterns appear in dev (739 inclusions, 16.71%) and test (814 inclusions, 17.68%) sets.

Named entities show high surface form reuse, with 18.34% of flat mentions containing potential nested mentions identified through substring matching. The distribution across entity types varies substantially: COUNTRY and ORGANIZATION inclusions have 82.55% and 59.82% precision respectively, while PERSON inclusions—the most numerous—have only 2.81% precision due to name components matching across unrelated entities (Appendix D).

Comparing the 6,458 exact inclusions against gold nested annotations, 1,488 match true inner entities (23.04% precision, 17.48% recall). When an inclusion span does match a gold inner entity, the type is correct 98.9% of the time (1,488 of 1,504 span matches), which confirms that substring matching reliably assigns the correct type. The remaining 76.7% of inclusions are spurious—their spans do not correspond to any gold inner entity. Lemmatized inclusions produce far more candidates (120,420) but with much lower precision (0.51%) and recall (7.22%). Despite this noise, training with inclusions substantially improves inner entity detection (3.84% $\rightarrow$ 21.36% F1), which suggests that even approximate nested signal provides some useful supervision.

5.2 Flat vs. inclusions vs. full training

Table 1 presents our main experimental results comparing flat training, training with inclusions, and full nested training on NEREL. We show results for the two best-performing prompt strategies (Keyword and Lex-all-outer); full comparison across all six prompt types is provided in Appendix E.

Flat training yields only 3–4% inner F1, demonstrating that models trained solely on outer entities cannot recognize nested structures. Adding inclusions increases inner F1 to 21–22%—a 5–7x improvement—with minimal degradation in outer entity performance. However, full nested training achieves 65% inner F1, indicating that inclusion-based methods, while effective, cannot replace full supervision. Different prompts yield similar results within each training regime (differences within 1–2% F1).

Symbol	Position	Overall	Inner	Outer
Digits	end	77.85	23.96	**82.61**
Letters	end	77.81	**25.92**	82.39
Diglets	start	**78.08**	22.43	82.27
Semicolon	start	74.59	19.16	77.13
Comma	start	74.23	19.46	76.38

Table 2: Best entity corruption results (Micro F1, %) for each symbol type using Strategy 1 (train on corrupted, predict on clean). Full results in Appendix F.

5.3 Entity corruption results

Table 2 summarizes the best corruption strategies for each symbol type. Full results across all positions and strategies are provided in Appendix F.

Training on corrupted data (Strategy 1) consistently outperforms corrupting at test time (Strategy 2). End-position corruption yields the highest inner F1 (25.92% with letters), likely because entity beginnings carry more type information. Alphanumeric corruption (digits, letters, diglets) maintains strong overall F1 (77–78%) while achieving 22–26% inner F1. Punctuation-based corruption (semicolons, commas) substantially degrades performance.

5.4 Flat neutralization results

Table 3 presents results for flat neutralization approaches, which mark potential nested mention positions as neutral during training rather than treating them as negative examples.

Flat neutralization alone improves inner F1 from 3.84% to 5.43%—a 41% relative improvement without explicit positive examples. Combining neutralization with inclusions (22.68%) modestly outperforms inclusions alone (21.36%). Lemmatization further improves matching by handling morphological variants (24.15%). Our best combined method (lemmatized inclusions + corruption + neutralization) achieves 26.37% inner F1 and 77.89% overall F1, closing 40% of the gap to full supervision while maintaining strong outer entity performance (82.54%).

Approach	Micro F1 (%)		
	Overall	Inner	Outer
Baseline			
Flat	76.59±0.23	3.84±0.77	**83.28**±0.23
Inclusions	76.70±0.17	21.36±1.53	83.05±0.26
Neutralization			
Flat+Neutral.	76.82±0.19	5.43±0.81	83.01±0.22
Incl.+Neutral.	77.01±0.15	22.68±1.27	82.93±0.18
Lem.Incl.+Neutral.	77.23±0.21	24.15±1.02	82.78±0.26
+Corruption			
Lem.Incl.+Corr.+Neu.	**77.89**±0.18	**26.37**±0.94	82.54±0.21

Table 3: Results for flat neutralization approaches on NEREL test set. Neutralization marks potential nested mentions as neutral (neither positive nor negative) during training. "Lem.Incl." refers to lemmatized inclusions.

Method	Overall	Inner	Outer
Traditional Approaches			
Flat	76.59±0.23	3.84±0.77	**83.28**±0.23
Flat+Inclusions	**76.70**±0.17	**21.36**±1.53	83.05±0.26
LLM Approaches (MFE)			
DeepSeek-R1 (0-shot)	38.35	2.69	38.86
DeepSeek-R1 (1-shot)	39.15	4.57	40.14
DeepSeek-R1 (5-shot)	**42.39**	**5.78**	**42.63**
RuAdapt (0-shot)	29.98	1.77	32.68
RuAdapt (1-shot)	29.23	2.00	31.82
RuAdapt (5-shot)	31.89	3.91	34.02
RuAdapt (MFE-entwise-sent)	45.91	2.42	46.51
Hybrid Approaches			
Hybrid (type-specific)	66.92	**20.24**	**82.73**
Hybrid (full prompts)	69.28	17.40	82.73
Hybrid (lem. matching)	**70.16**	18.84	75.83

Table 4: LLM and hybrid results on NEREL test set (Micro F1, %). Traditional approaches show mean±std across 3 runs; LLM approaches show mean across 3 seeds.

5.5 LLM results

Table 4 presents LLM and hybrid results. Pure LLM approaches substantially underperform fine-tuned methods: DeepSeek-R1 achieves only 42.39% overall F1 (5-shot) compared to 76.59% for flat training. The hybrid approach (70.16% overall F1) outperforms pure LLM methods but still underperforms fine-tuned approaches on inner entities (18.84% vs 21.36%). This suggests current LLMs struggle with fine-grained nested structure across 29 entity types.

5.6 Summary comparison

Table 5 presents a unified comparison of the best-performing variant from each method category.

The comparison reveals a clear hierarchy: pure LLM approaches substantially underperform fine-tuned methods, while hybrid approaches fall between pure LLM and fine-tuned weak supervision. Our best combined method achieves the highest inner F1 among weak supervision approaches, closing 40% of the gap to full nested supervision.

Method	Overall	Inner	Outer
Weak Supervision (Fine-tuned)			
Flat (baseline)	76.59	3.84	83.28
+ Inclusions	76.70	21.36	83.05
+ Corruption (letters, end)	77.81	25.92	82.39
+ Lem.Incl.+Corr.+Neutral.	**77.89**	**26.37**	**82.54**
LLM-based			
Pure LLM (DeepSeek 5-shot)	42.39	5.78	42.63
Hybrid (lem. matching)	70.16	18.84	75.83
Upper Bound			
Full nested supervision	<u>85.81</u>	<u>65.48</u>	<u>83.95</u>

Table 5: Summary comparison of best methods (Micro F1, %). Our best weak supervision method (Lem.Incl.+Corr.+Neutral.) closes 40% of the inner F1 gap between flat training and full supervision.

6 Conclusion

We investigated methods for learning nested named entity recognition from flat annotations, addressing the annotation cost that limits nested NER development. Using the Russian NEREL dataset with 29 entity types, we systematically compared string inclusions, entity corruption, flat neutralization, and hybrid fine-tuned with LLM approaches.

Our experiments demonstrate that nested structure can be recovered without gold nested annotations. String inclusions alone improve inner entity F1 from 3.84% to 21.36% by leveraging substring relationships between entities. Entity corruption creates pseudo-nested training examples, with our analysis revealing that end-position corruption consistently outperforms other positions—a finding not previously documented. Combining these methods with flat neutralization achieves 26.37% inner F1, closing 40% of the gap to full nested supervision.

However, a significant performance gap remains. Full nested supervision achieves 65.48% inner F1, indicating that our weak supervision methods, while effective, cannot fully substitute for nested annotations. The hybrid fine-tuned+LLM approach, despite leveraging large language model capabilities, underperforms fine-tuned methods on inner entity detection, suggesting that current LLMs struggle with fine-grained nested structure across many entity types.

Returning to our research questions:

RQ1: Yes, models can learn to recognize nested entities from flat annotations. String inclusions alone achieve 21.36% inner F1, and combining methods reaches 26.37%, closing 40% of the gap to full supervision.

RQ2: Partially. The hybrid fine-tuned + LLM pipeline achieves 70.16% overall F1, leveraging fine-tuned models for reliable outer entity detection (82–83% F1) while using LLMs for nested structure. However, this approach underperforms purely fine-tuned methods on inner entities (18.84% vs 26.37% inner F1), indicating that current LLMs provide limited benefit for fine-grained nested detection across many entity types.

RQ3: No, current LLMs cannot compensate for missing nested annotations. Pure LLM approaches achieve only 42.39% overall F1 compared to 76.59% for fine-tuned models, with inner entity detection at just 5.78% F1. LLMs struggle with the 29-type inventory, frequently hallucinating entity boundaries and types. While LLMs show some prospect for coarse-grained NER, they cannot substitute for proper supervision when fine-grained nested structure is required.

Future work should investigate more principled approaches to negative sampling in span-based models, cross-lingual transfer of weak supervision methods, and improved LLM prompting strategies for nested structure. The gap between weak and full supervision also motivates research into annotation-efficient approaches that selectively annotate nested structure where it provides the most benefit.

Limitations

Our experiments focus exclusively on Russian, using the NEREL dataset. While Russian's rich morphology provides a challenging test case, our findings may not generalize to other languages—especially those with very different structural properties. Morphological normalization is inherently language-specific, and its effectiveness in other languages remains to be verified. Additionally, because NEREL consists of news text, we cannot claim our results would hold in domains such as biomedical or social media data.

We evaluated on NEREL only, without cross-dataset validation. Although it is one of the largest nested NER resources available, testing on additional benchmarks would strengthen our conclusions. We leave experiments on other Russian nested NER datasets, such as NEREL-BIO, for future work.

All our comparisons use the Binder architecture; we did not test alternatives such as biaffine parsers or sequence-to-sequence models. Architecture comparisons are complicated by differences in preprocessing and evaluation, so we kept the focus narrow: our goal was to compare weak supervision strategies, not model designs. The strategies we studied (inclusions, corruption, neutralization) are architecture-agnostic and could be applied to other span-based NER models.

Class imbalance in NEREL is pronounced—PERSON and ORGANIZATION dominate. We did not adjust for this with re-sampling or loss weighting, so performance on rare types may be understated.

For our LLM experiments, we used DeepSeek-R1 and RuAdapt-Qwen2.5, chosen for strong Russian support and local deployability. Larger models such as GPT-4 may achieve better performance, but cost and reproducibility concerns ruled them out. Our LLM results should therefore be interpreted as demonstrating the approach rather than establishing upper bounds.

Results are averaged over three runs with standard deviations reported. We did not perform formal significance tests. With only a few seeds, smaller differences between methods should be treated cautiously, though the larger improvements (e.g., 3.84% to 21.36% inner F1) clearly exceed random variation.

Finally, we evaluated all experimental configurations on the test set rather than reserving it for final evaluation only. While we fixed hyperparameters upfront and our methods were linguistically motivated—not tuned on test metrics—this approach still carries some overfitting risk. A stricter design with dedicated development-phase tuning would have been stronger.

Acknowledgments

The study was supported by the Russian Science Foundation project 25-21-00206. The research was carried out using the MSU-270 supercomputer of Lomonosov Moscow State University.

References

Ekaterina Artemova, Natalia Loukachevitch, and Pavel Braslavski. 2022. RuNNE-2022 shared task: Recognizing nested named entities. In *Computational Linguistics and Intellectual Technologies: Papers from the Annual Conference "Dialogue"*.

Xiang Dai and Heike Adel. 2020. An analysis of simple data augmentation for named entity recognition. In *Proceedings of the 28th International Confer-*

ence on Computational Linguistics, pages 3861–3867, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Vera Davydova, Natalia Loukachevitch, and Elena Tutubalina. 2024. Overview of BioNNE task on biomedical nested named entity recognition at BioASQ 2024. In *CLEF Working Notes*.

George Doddington, Alexis Mitchell, Mark Przybocki, Lance Ramshaw, Stephanie Strassel, and Ralph Weischedel. 2004. The automatic content extraction (ACE) program – tasks, data, and evaluation. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal. European Language Resources Association (ELRA).

Jenny Rose Finkel and Christopher D. Manning. 2009. Nested named entity recognition. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 141–150, Singapore. Association for Computational Linguistics.

Daya Guo, Dejian Yang, He Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Ridong Han, Tao Peng, Chaohao Yang, Benyou Wang, Lu Liu, and Xiang Wan. 2023. Is information extraction solved by ChatGPT? an analysis of performance, evaluation criteria, robustness and errors. *arXiv preprint arXiv:2305.14450*.

Mustafa Jarrar, Mohammed Khalilia, and Sana Ghanem. 2022. Wojood: Nested Arabic named entity corpus and recognition using BERT. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 3626–3636, Marseille, France. European Language Resources Association.

Meizhi Ju, Makoto Miwa, and Sophia Ananiadou. 2018. A neural layered model for nested named entity recognition. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1446–1459, New Orleans, Louisiana. Association for Computational Linguistics.

Hongjin Kim, Jai-Eun Kim, and Harksoo Kim. 2024. Exploring nested named entity recognition with large language models: Methods, challenges, and insights. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8653–8670, Miami, Florida, USA. Association for Computational Linguistics.

Jin-Dong Kim, Tomoko Ohta, Yuka Tateisi, and Jun'ichi Tsujii. 2003. GENIA corpus—a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19(Supplement 1):i180–i182.

Mikhail Korobov. 2015. Morphological analyzer and generator for Russian and Ukrainian languages. In *Analysis of Images, Social Networks and Texts*, pages 320–332. Springer.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Pierre Lison, Jeremy Barnes, Aliaksandr Hubin, and Samia Touileb. 2020. Named entity recognition without labelled data: A weak supervision approach. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1518–1533, Online. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*.

Natalia Loukachevitch, Ekaterina Artemova, Tatiana Batura, Pavel Braslavski, Ilia Denisov, Vladimir Ivanov, Suresh Manandhar, Alexander Pugachev, and Elena Tutubalina. 2021. NEREL: A Russian dataset with nested named entities, relations and events. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 876–885, Held Online. INCOMA Ltd.

Wei Lu and Dan Roth. 2015. Joint mention extraction and classification with mention hypergraphs. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 857–867, Lisbon, Portugal. Association for Computational Linguistics.

Yu Meng, Yunyi Zhang, Jiaxin Huang, Xuan Wang, Yu Zhang, Heng Ji, and Jiawei Han. 2021. Distantly-supervised named entity recognition with noise-robust learning and language model augmented self-training. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10367–10378, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Barbara Plank, Kristian Nørgaard Jensen, and Rob van der Goot. 2020. DaN+: Danish nested named entities and lexical normalization. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6649–6662, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155, Boulder, Colorado. Association for Computational Linguistics.

Nicky Ringland, Xiang Dai, Ben Hachey, Sarvnaz Karimi, Cecile Paris, and James R. Curran. 2019. NNE: A dataset for nested named entity recognition in English newswire. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5176–5181, Florence, Italy. Association for Computational Linguistics.

Igor Rozhkov and Natalia Loukachevitch. 2023. Prompts in few-shot named entity recognition. *Pattern Recognition and Image Analysis*, 33(2):238–244.

Igor Rozhkov and Natalia Loukachevitch. 2025. Methods for recognizing nested terms. In *Proceedings of the 2025 Conference*.

Zeqi Tan, Yongliang Shen, Shuai Zhang, Weiming Lu, and Yueting Zhuang. 2021. A sequence-to-set network for nested named entity recognition. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI-21)*, pages 3936–3942.

Mikhail Tikhomirov and Daniil Chernyshev. 2023. Impact of tokenization on LLaMA Russian adaptation. In *2023 Ivannikov Ispras Open Conference (ISPRAS)*, pages 163–168. IEEE.

Mikhail Tikhomirov and Daniil Chernyshov. 2024. Facilitating large language model Russian adaptation with learned embedding propagation. *Journal of Language and Education*, 10(4):130–145.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Solenn Tual and 1 others. 2023. A benchmark of nested named entity recognition approaches in historical structured documents. In *International Conference on Document Analysis and Recognition*, pages 115–131.

Jue Wang, Lidan Shou, Ke Chen, and Gang Chen. 2020. Pyramid: A layered model for nested named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5918–5928, Online. Association for Computational Linguistics.

Shuhe Wang, Xiaofei Sun, Xiaoya Li, Rongbin Ouyang, Fei Wu, Tianwei Zhang, Jiwei Li, and Guoyin Wang. 2023. GPT-NER: Named entity recognition via large language models. *arXiv preprint arXiv:2304.10428*.

Yukun Yan, Tao Gui, Junqi Ye, and Qi Zhang. 2023. Nested named entity recognition as building local hypergraphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 13855–13863.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. Named entity recognition as dependency parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6470–6476, Online. Association for Computational Linguistics.

Hao Zhang and 1 others. 2023a. Judicial nested named entity recognition method with MRC framework. *International Journal of Cognitive Computing in Engineering*, 4:118–126.

Sheng Zhang, Hao Cheng, Jianfeng Gao, and Hoifung Poon. 2023b. Optimizing bi-encoder for named entity recognition via contrastive learning. In *The Eleventh International Conference on Learning Representations (ICLR)*.

Enwei Zhu, Yiyang Sheng, Yanping Chen, and Jinpeng Li. 2022. Recognizing nested entities from flat supervision: A new NER subtask, feasibility and challenges. *arXiv preprint arXiv:2211.11116*.

Dmitry Zmitrovich, Aleksandr Abramov, Andrey Kalmykov, Vitaly Kadulin, Maria Tikhonova, Ekaterina Taktasheva, Danil Astafurov, Mark Baushenko, Artem Snegirev, Tatiana Shavrina, Sergei S. Markov, Vladislav Mikhailov, and Alena Fenogenova. 2024. A family of pretrained transformer language models for Russian. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 507–524, Torino, Italia. ELRA and ICCL.

A Base LLM prompt template

The base prompt template used for all pure LLM experiments:

```
Given entity label set: ['AGE',
    ↪ 'AWARD', 'CITY', 'COUNTRY',
    ↪ 'CRIME', 'DATE', 'DISEASE',
    ↪ 'DISTRICT', 'EVENT', 'FACILITY',
    ↪ 'FAMILY', 'IDEOLOGY', 'LANGUAGE',
    ↪ 'LAW', 'LOCATION', 'MONEY',
    ↪ 'NATIONALITY', 'NUMBER',
    ↪ 'ORDINAL', 'ORGANIZATION',
    ↪ 'PENALTY', 'PERCENT', 'PERSON',
    ↪ 'PRODUCT', 'PROFESSION',
    ↪ 'RELIGION', 'STATE_OR_PROVINCE',
    ↪ 'TIME', 'WORK_OF_ART'].
You are an excellent linguist and
    ↪ annotator. Based on the given
    ↪ entity label set, please
    ↪ recognize the named entities in
    ↪ the given text. Consider there
    ↪ might be a nested case, where one
    ↪ entity contains another. There
    ↪ are two possible type of nested
    ↪ entities:
NDT: It consists of an entity
    ↪ containing a shorter entity
    ↪ tagged with a different type.
NST: This case usually occurs when
    ↪ entities are originally
    ↪ represented by a hierarchy.
```

```
Give me ONLY entities in format of a
    ↪ json dictionary with named
    ↪ entities as keys and their types
    ↪ as values like this: {entity :
    ↪ type}. Do not write any
    ↪ additional text. Enclose answer
    ↪ in ```.
```

For methods with entity definitions (e.g., type-specific hybrid), Russian definitions are appended after the base prompt.

B Entity-specific nesting patterns

For hybrid fine-tuned+LLM approaches, we augment the base prompt with entity-specific nesting patterns derived from the training data. Below are examples for selected entity types:

```
For class ORGANIZATION most common
 ↪  nested entity classes are:
 ↪  ORGANIZATION, COUNTRY, EVENT, CITY,
 ↪  PROFESSION

Here are some examples of the
 ↪  ORGANIZATION class as outermost
 ↪  entity and its nested entities:
Outermost entity: ```Федеральный штаб
 ↪  народного ополчения``` (Federal
 ↪  Headquarters of the People's
 ↪  Militia), nested are:
 ↪  ```[{"Федеральный штаб":
 ↪  "ORGANIZATION"}, (Federal
 ↪  Headquarters) {"штаб народного
 ↪  ополчения": "ORGANIZATION"},
 ↪  (headquarters of the people's
 ↪  militia) {"ополчения":
 ↪  "ORGANIZATION"}]``` (militia)

For class PERSON most common nested
 ↪  entity classes are: PERSON,
 ↪  PROFESSION, ORDINAL, CITY,
 ↪  ORGANIZATION

Here are some examples of the PERSON
 ↪  class as outermost entity and its
 ↪  nested entities:
Outermost entity: ```Эрнст Теодор Амадей
 ↪  Гофман``` (Ernst Theodor Amadeus
 ↪  Hoffmann), nested are: ```[{"Эрнст":
 ↪  "PERSON"}, {"Амадей": "PERSON"},
 ↪  {"Гофман": "PERSON"}]``` (Ernst,
 ↪  Amadeus, Hoffmann - name components)

For class DATE most common nested entity
 ↪  classes are: DATE, NUMBER, AGE,
 ↪  ORDINAL, PERSON

Here are some examples of the DATE class
 ↪  as outermost entity and its nested
 ↪  entities:
Outermost entity: ```21 октября 1952
 ↪  года``` (October 21, 1952), nested
 ↪  are: ```[{"1952": "DATE"}, {"21":
 ↪  "ORDINAL"}, {"года": "DATE"},
 ↪  {"октября": "DATE"}]``` (1952, 21,
 ↪  year, October)

... [patterns for all 29 entity classes]
```

C Russian entity definitions

For experiments requiring entity definitions, we provide Russian descriptions with English translations:

AGE Возраст: Это число, которое показывает, сколько лет кому-то или чему-то. — A number that shows how old someone or something is.

AWARD Награда: Это признание заслуг или достижений. — Recognition of merits or achievements.

CITY Город: Место, где живут люди, обычно больше, чем деревня. — A place where people live, usually larger than a village.

COUNTRY Страна: Большая территория с определенным населением и правительством. — A large territory with a defined population and government.

CRIME Преступление: Действия, запрещенные законом. — Actions prohibited by law.

DATE Дата: Указание времени, когда что-то произошло. — An indication of when something happened.

DISEASE Болезнь: Состояние, при котором организм не работает нормально. — A condition in which the body does not function normally.

DISTRICT Район: Часть страны или города, имеющая свои границы и управление. — Part of a country or city with its own boundaries and governance.

EVENT Событие: Что-то важное, что произошло. — Something important that happened.

FACILITY Объект инфраструктуры: Строение или место, используемое для определенной цели. — A building or place used for a specific purpose.

FAMILY Семья: Группа людей, связанных кровными узами. — A group of people related by blood ties.

IDEOLOGY Идеология: Набор идей и убеждений, определяющих поведение и политику. — A set of ideas and beliefs that determine behavior and policy.

LANGUAGE Язык: Система общения, состоящая из слов и правил. — A communication system consisting of words and rules.

LAW Закон: Правила, установленные государством. — Rules established by the state.

LOCATION Место: Где что-то находится. — Where something is located.

MONEY Деньги: Средства обмена, используемые для покупки товаров и услуг. — A medium of exchange used to purchase goods and services.

NATIONALITY Национальность: Принадлежность к определенной стране или народу. — Belonging to a particular country or nation.

NUMBER Число: Цифра или количество чего-либо. — A digit or quantity of something.

ORDINAL Порядковый номер: Указывает на позицию в ряду. — Indicates a position in a sequence.

ORGANIZATION Организация: Группировка людей с общей целью. — A group of people with a common purpose.

PENALTY Наказание: Последствия за нарушение закона. — Consequences for violating the law.

PERCENT Процент: Доля от целого, выраженная в сотых долях. — A fraction of the whole expressed in hundredths.

PERSON Человек: Индивидуальное лицо. — An individual.

PRODUCT Продукт: То, что создано или произведено для продажи или использования. — Something created or produced for sale or use.

PROFESSION Профессия: Вид деятельности, которым человек зарабатывает на жизнь. — A type of activity by which a person earns a living.

RELIGION Религия: Вера и система верований. — Faith and a system of beliefs.

STATE_OR_PROVINCE Штат или провинция: Административная единица внутри страны. — An administrative unit within a country.

TIME Время: Конкретный момент или период. — A specific moment or period.

WORK_OF_ART Художественное произведение: Произведения искусства, созданные человеком. — Works of art created by humans.

D Inclusion statistics per entity type

Table 6 shows inclusion counts and precision for each of the 29 entity types. *Inclusions* are pseudo-nested entities identified by exact substring matching; *Lem. inclusions* additionally apply morphological normalization. *Precision* indicates the percentage of inclusions that correspond to true nested annotations.

E Full prompt comparison

Table 7 presents complete results across all six prompt strategies.

F Full entity corruption results

Table 8 presents complete results for all entity corruption strategies.

Type	True inner	Incl.	Prec. (%)	Lem. incl.	Prec. (%)
PERSON	433	2,845	2.81	41,347	0.07
PROFESSION	1,820	884	28.05	14,042	0.97
ORGANIZATION	2,001	662	59.82	13,456	1.43
COUNTRY	1,773	619	82.55	13,874	1.28
NUMBER	194	456	3.29	2,477	0.04
EVENT	424	300	25.00	14,059	0.09
CITY	634	159	37.11	4,186	0.74
DATE	325	111	9.91	3,813	0.18
STATE_OR_PROV.	213	73	72.60	807	1.49
ORDINAL	324	60	11.67	1,377	0.29
PRODUCT	30	53	20.75	1,203	0.42
AWARD	0	31	0.00	739	0.00
IDEOLOGY	0	31	0.00	1,030	0.00
FACILITY	53	23	56.52	1,453	0.21
LAW	0	21	0.00	763	0.00
NATIONALITY	52	21	9.52	760	0.00
AGE	17	18	0.00	867	0.00
LOCATION	127	17	29.41	469	0.64
DISEASE	0	16	0.00	260	0.00
CRIME	0	13	0.00	1,096	0.00
WORK_OF_ART	0	11	0.00	789	0.00
LANGUAGE	0	10	0.00	87	0.00
PENALTY	0	9	0.00	320	0.00
RELIGION	0	8	0.00	67	0.00
DISTRICT	59	5	40.00	74	1.35
MONEY	22	1	0.00	155	0.00
TIME	9	1	0.00	202	0.00
FAMILY	3	0	0.00	561	0.18
PERCENT	0	0	0.00	87	0.00
Total	**8,513**	**6,458**	**23.04**	**120,420**	**0.51**

Table 6: Inclusion statistics per entity type on training data, sorted by exact inclusion count. *True inner*: gold nested entities. *Incl.*: exact substring inclusions. *Lem. incl.*: lemmatized substring inclusions. Precision measures the fraction of inclusions matching gold inner entities. Entity types like COUNTRY and ORGANIZATION have high exact-match precision, while PERSON produces many inclusions with low precision due to name components matching across unrelated entities. Lemmatization vastly increases inclusion counts but reduces precision, as morphological normalization over-generalizes matching.

Training	**Prompt**	**Micro F1 (%)**			**Macro F1 (%)**		
		Overall	Inner	Outer	Overall	Inner	Outer
Flat Training (NEREL-outerflat)							
	Keyword	76.59±0.23	3.84±0.77	83.28±0.23	76.67±0.17	3.30±0.58	82.94±0.27
	Definition	76.43±0.17	4.42±0.52	82.96±0.31	76.57±0.17	3.76±0.79	82.66±0.32
	Most-freq	76.60±0.21	3.66±0.46	83.08±0.56	76.67±0.24	3.23±0.53	82.80±0.55
	Context	75.32±0.17	3.32±0.72	81.38±0.23	75.42±0.12	2.79±0.69	81.05±0.30
	Lex-all-out	76.72±0.17	3.11±0.61	83.40±0.26	76.69±0.18	2.76±0.26	83.08±0.20
	Struct-nest	76.57±0.15	4.04±1.15	82.88±0.46	76.63±0.15	3.39±1.04	82.52±0.39
Flat Training with Inclusions							
	Keyword	76.70±0.17	21.36±1.53	83.05±0.26	76.53±0.24	18.17±1.14	82.65±0.29
	Definition	76.79±0.29	21.97±0.67	83.01±0.26	76.66±0.33	18.47±0.60	82.69±0.38
	Most-freq	76.89±0.10	21.30±0.85	82.88±0.90	76.74±0.09	17.95±0.59	82.65±0.91
	Context	75.39±0.39	18.64±1.77	81.04±0.31	75.32±0.38	15.76±1.34	80.74±0.39
	Lex-all-out	76.93±0.19	22.16±1.93	83.17±0.27	76.82±0.15	18.60±1.45	82.90±0.29
	Struct-nest	76.72±0.09	20.50±0.77	83.15±0.25	76.61±0.08	17.56±0.73	82.87±0.24
Full Nested Training							
	Keyword	85.81±0.19	65.48±0.94	83.95±0.28	85.60±0.21	54.53±0.70	83.56±0.31
	Definition	85.82±0.14	65.90±0.79	83.89±0.23	85.58±0.19	54.93±0.44	83.52±0.26
	Most-freq	85.72±0.21	65.68±1.49	83.88±0.22	85.45±0.25	54.87±1.83	83.41±0.34
	Context	83.91±0.07	54.62±0.95	81.48±0.09	83.69±0.05	45.94±0.86	80.99±0.15
	Lex-all-out	85.55±0.10	65.33±0.62	83.59±0.14	85.24±0.16	54.26±0.59	83.17±0.22
	Struct-nest	85.68±0.08	65.93±0.39	83.85±0.11	85.42±0.16	54.92±0.29	83.44±0.13

Table 7: Full prompt comparison on NEREL test set. Six prompt strategies tested: Keyword (entity type names), Definition (natural language definitions), Most-freq (most frequent entity example per type), Context (sentence contexts), Lex-all-out (full sentences with entity markers), Struct-nest (structural nesting information).

Corruption	Position	Micro F1 (%)			Macro F1 (%)		
		Overall	Inner	Outer	Overall	Inner	Outer
Digits Corruption							
Early	random	77.98±0.04	21.23±2.16	82.74±0.01	78.15±0.01	17.41±1.86	82.53±0.05
	start	77.74±0.10	23.32±0.45	82.68±0.25	77.76±0.11	19.38±0.67	82.33±0.29
	end	77.85±0.21	23.96±0.73	82.61±0.30	77.95±0.24	20.35±0.82	82.33±0.22
	middle	78.12±0.16	22.07±1.77	82.45±0.12	78.29±0.12	18.45±1.23	82.24±0.07
	syntax	77.92±0.05	22.63±0.84	82.76±0.04	78.01±0.06	18.95±0.42	82.52±0.10
Late	random	76.72	17.43	81.60	77.04	14.75	81.50
	start	75.69	18.69	81.85	75.80	15.12	81.52
	end	75.51	13.28	82.31	75.65	11.59	81.98
	middle	77.30	17.87	82.73	77.41	15.64	82.62
	syntax	76.31	23.14	82.64	76.35	18.85	82.37
Letters Corruption							
Early	random	78.08±0.09	23.80±2.34	82.10±0.15	78.19±0.06	19.77±2.03	81.83±0.12
	start	78.11±0.12	22.51±1.44	82.15±0.25	78.24±0.16	19.03±1.13	81.88±0.27
	end	77.81±0.15	25.92±0.24	82.39±0.21	77.92±0.11	21.54±0.43	82.19±0.06
	middle	77.46±0.23	22.82±0.83	82.42±0.24	77.57±0.27	19.57±0.87	82.33±0.20
	syntax	77.57±0.15	23.19±1.20	82.55±0.28	77.67±0.20	19.70±0.85	82.28±0.27
Late	random	77.39	17.82	82.86	77.68	14.82	82.85
	start	77.78	23.30	83.01	77.89	19.65	82.95
	end	76.29	14.61	83.09	76.61	12.33	82.92
	middle	77.30	16.93	82.53	77.36	13.28	82.29
	syntax	77.29	23.84	82.40	77.31	20.79	82.11
Diglets Corruption (mixed digits+letters)							
Early	start	78.08	22.43	82.27	78.16	18.86	82.03
	end	77.55	22.37	82.83	77.60	18.50	82.62
Late	start	77.04	20.20	81.74	77.23	16.34	81.44
	end	75.64	13.85	82.13	76.10	11.48	82.05
Semicolon Corruption							
Early	start	74.59	19.16	77.13	75.11	16.36	77.45
	end	74.37	18.34	74.65	74.90	16.66	75.05
Late	start	76.24	3.44	83.67	76.21	2.77	83.10
	end	76.58	6.08	83.43	76.62	5.66	83.00
Comma Corruption							
Early	start	74.23	19.46	76.38	74.84	17.88	76.91
	end	73.93	18.86	73.00	74.38	17.02	73.53
Late	start	76.79	5.05	83.33	76.73	3.72	82.88
	end	76.30	5.22	83.56	76.43	4.57	83.30

Table 8: Full entity corruption results on NEREL test set. "Early" (early damage) refers to training on corrupted data and predicting on clean data; "Late" (late damage) refers to training on clean data and predicting on corrupted data. Positions: random, start, end, middle, syntax (syntactic root).