ReProCon: Scalable and Resource-Efficient Few-Shot Biomedical Named Entity Recognition

Jeongkyun Yoo
Ain Hospital
Incheon, South Korea
luxiante@gmail.com

Nela Riddle
Indiana University
Indianapolis, USA
nmriddle@iu.edu

Andrew Hoblitzell
Purdue University
Indianapolis, USA
ahoblitz@purdue.edu

Abstract

Named Entity Recognition (NER) in biomedical domains faces challenges due to data scarcity and imbalanced label distributions, especially with fine-grained entity types. We propose ReProCon, a novel few-shot NER framework that combines multi-prototype modeling, cosine-contrastive learning, and Reptile meta-learning to tackle these issues. By representing each category with multiple prototypes, ReProCon captures semantic variability, such as synonyms and contextual differences, while a cosine-contrastive objective ensures strong interclass separation. Reptile meta-updates enable quick adaptation with little data. Using a lightweight fastText + BiLSTM encoder, with much lower memory use, ReProCon achieves a macro-F_1 score close to BERT-based baselines (~99% of BERT performance). The model remains stable with a label budget of 30% and only drops 7.8% in F_1 when expanding from 19 to 50 categories, outperforming baselines such as SpanProto and CONTaiNER, which see 10–32% degradation in Few-NERD. Ablation studies highlight the importance of multi-prototype modeling and contrastive learning in managing class imbalance. Despite difficulties with label ambiguity, ReProCon demonstrates state-of-the-art performance in resource-limited settings, making it suitable for biomedical applications.

Disclaimer: This work is licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0).
Correspondence: Nela Riddle (nmriddle@iu.edu), Andrew Hoblitzell (ahoblitz@purdue.edu)

ACM Reference Format:
Jeongkyun Yoo, Nela Riddle, and Andrew Hoblitzell. 2025. ReProCon: Scalable and Resource-Efficient Few-Shot Biomedical Named Entity Recognition. In . ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Named Entity Recognition (NER) identifies and categorizes entity mentions in unstructured text, serving as a foundation for downstream biomedical NLP tasks, including knowledge graph construction, literature curation, and clinical decision support. In biomedical domains, NER is particularly challenging due to the high cost of annotation and the long-tailed label distributions in public corpora, such as MedMentions [1], BC2GM [2], and NCBI-Disease [3], which often contain contextually variable entity categories.

Few-shot learning mitigates the data bottleneck by training on only a handful of labeled examples. Although recent work shows promise on general domain benchmarks (e.g., Few-NERD [4]), NER has achieved limited success in the biomedical domain with a large number of entity types. In such cases, existing approaches often suffer from poor performance due to the overwhelming diversity of categories and a lack of ontological awareness.

To address these challenges, we propose **ReProCon**, a prototype-based cosine-contrastive NER framework that addresses these gaps. The key components are the following:

(1) **Multi-prototype modeling**: Each category has M distinct prototypes that capture various contexts of biomedical entities, designed to be angularly separated for better differentiation.
(2) **Cosine supervised-contrastive objective**: The optimization aims for the entity spans to align closely with their corresponding category prototypes while being angularly distanced from others.
(3) **First-order meta-updates (Reptile)**: Episode-based updates enable rapid parameter adaptation without the computational burden of second-order gradients [5].

Empirical evaluation in MedMentions demonstrates that ReProCon effectively tackles imbalanced samples and hierarchical complexity of the dataset, such as concept structures based on the Unified Medical Language System (UMLS) [6], while achieving performance comparable to or exceeding state-of-the-art methods, such as BioBERT [7] and SpanProto [8], with improved computational efficiency.

2 Related Work

2.1 Domain-specific Language Models and Biomedical Resources

Biomedical PLMs. Large-scale pre-training on domain-specific corpora significantly enhances biomedical NLP performance. Models like BioBERT, PubMedBERT [9], and SciBERT [10] utilize PMC articles or PubMed abstracts instead of generic BERT training data, achieving substantial improvements in biomedical NER and question answering. These results underscore the data mismatch, differences in vocabulary, and context between the general and biomedical domains as the main bottleneck for domain transfer.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
Conference'17, Washington, DC, USA
© 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

UMLS.. The Unified Medical Language System (UMLS) integrates more than 100 biomedical vocabularies into a unified ontology. Despite its extensive coverage, UMLS's heterogeneous sources, biannual updates, and complex licensing pose challenges for few-shot evaluations, particularly in data preparation and model adaptation. To address this, we derive a dynamic semantic type hierarchy from MedMentions annotations [1], enabling ReProCon to capture the latest ontological structures while avoiding licensing constraints.

2.2 Prototypical Network

Several studies have implemented the prototypical network for the NER few-shot learning task. Classic prototypical networks assume one centroid per category [11]. ReProCon extends hyperspherical prototype networks by incorporating multiple prototypes per category, enabling a finer-grained representation of biomedical entities.

Mettes et al. applied hyperspherical prototype networks to NER tasks, including those within and between domains. They adapted classic prototypical networks by training prototypes as part of the end-to-end process, using a loss term designed to maximize the separation between prototypes [12]. In this study, we adopted the concept of calculating the repulsion loss between prototypes.

2.3 Meta-Learning

Model-Agnostic Meta-Learning learns a single parameter initialization that can fine-tune new tasks with only a few gradient updates [13]. It employs an inner loop for task-specific fine-tuning within an outer loop for generalized parameter optimization that computes second-order gradients. This approach aims to produce model parameters that are adaptable to new tasks with minimal data. However, backpropagating through gradient steps introduces significant computational and memory overhead, often causing instability in deep or noisy tasks.

Reptile is a first-order meta-learning algorithm that learns a task-agnostic parameter initialization by repeatedly fine-tuning the sampled tasks and adjusting the shared weights to the adapted ones. Unlike MAML, Reptile eliminates second-order derivatives, so its meta-update is both simpler and substantially faster, while retaining strong few-shot performance. This outer loop translation of inner loop progress approximates higher-order information.

2.4 Supervised Contrastive Learning

Supervised contrastive learning improves representation robustness in a few-shot setting and has also been applied in recent NER tasks to improve representation robustness under limited supervision [14]. For example, Das et al. implemented contrastive learning for the NER task by calculating losses over entity spans [15]. However, existing formulations assume one-to-one prototype alignment, which struggles with fine-grained biomedical categories. We extended this approach by introducing a multi-prototype contrastive loss that uses category-wise minimum pooling to align projected spans with the closest prototype in each category. This loss function enables both semantic flexibility and interclass separation, which is crucial for fine-grained classification of biomedical entities.

3 Data Processing

3.1 Sentence and Entity Preprocessing

We first normalize whitespace and remove text enclosed in parentheses, brackets, braces, or angle brackets, as they often duplicate named entities, introducing redundant contextual variability. We applied the same preprocessing to both the sentences and their entity strings.

3.2 Semantic Type Normalization and Hierarchical Disambiguation

The MedMentions corpus annotates entities with semantic type IDs of the Unified Medical Language System (UMLS), many of which are fine-grained (e.g., `Organization` , `Substance`), exhibiting highly imbalanced sample distributions. Moreover, a single span can carry multiple types of labels, introducing ambiguity. To enhance learning efficiency by focusing on well-represented categories, we design a two-stage procedure through sparsity reduction and PageRank-based disambiguation to select a single type when multiple labels are present.

3.2.1 Category Integration and Pruning. We map every semantic type ID to a human-readable name using a lookup table provided from the Medmentions dataset. Let Y denote the original set of types and $\text{freq}(y)$ the frequency of type $y \in Y$. Then, to reduce sparsity and avoid underrepresented categories, two rules are applied:

(1) **Depth constraint** If a semantic type ID corresponds to a node deeper than a specific level in the UMLS hierarchy, we promote it to its closest ancestor. We tested two constraint levels, levels 3 and 4, to compare the model's performance according to the number of categories.

(2) **Low-frequency merging** If a type has fewer than a specific threshold number of samples, it is recursively promoted to the closest ancestor until the sample number becomes larger than the threshold, regardless of its depth. We set thresholds of 100 for level 3 constraints and 50 for level 4 constraints.

After applying the above rules, we discarded any top-level category with fewer than the threshold number of instances. The resulting taxonomies yield 19- and 50-way classification settings; the list of category statistics is in Table 6 and Table 5 (Appendix).

3.2.2 Resolution of Multi-Label Entities. As described in Algorithm 1, we constructed an undirected co-occurrence graph where nodes represent semantic types, with edges reflecting the frequency of co-occurrence within entity annotations. We then used PageRank centrality to rank the types, assuming that the most central types are broader or more representative categories [16]. For entities with multiple semantic types, we retained only the one with the highest PageRank score, reducing the labeling noise.

3.3 Span Generation and Label Assignment

We used SpaCy [17] to process tokenization before generating text spans. Inspired by EPNet [18], we generated text spans from sentence token sequences $S = [t_1, t_2, ..., t_n]$ by extracting consecutive tokens of up to 8 tokens, since 99.95% of the entity spans contain fewer than nine tokens. To preserve contextual order, we create samples by marking the generated token spans with a special marker

`[MARK_POSITION]` within the sentences to disambiguate multiple instances of the same word inside a sentence. We organized these samples by semantic type category.

Algorithm 1 PageRank-based Construction of a Semantic-Type Hierarchy

Require: Annotated corpus $D = \{d_1, \ldots, d_{|D|}\}$ with entity spans
Ensure: Ordered list H of semantic types ranked by PageRank

```
C ← [ ]                                   ▷ list of multi type combinations
V ← ∅                                     ▷ set of all unique semantic type IDs
for all document d ∈ D do
    for all entity span e ∈ d do
        Y ← {y ∈ e | freq(y) ≥ threshold}   ▷ Filter types below frequency threshold
        if |Y| > 1 and Y ∉ C then
            append Y to C
            V ← V ∪ Y
        end if
    end for
end for
A ← 0_{|V|×|V|}                           ▷ undirected co-occurrence matrix
for all Y ∈ C do
    for all (y_i, y_j) ∈ Combinations(Y, 2) do   ▷ pairwise type combinations
        A[y_i, y_j] ← A[y_i, y_j] + 1
        A[y_j, y_i] ← A[y_j, y_i] + 1     ▷ symmetry
    end for
end for
Build graph G = (V, E) with edge weights A[y_i, y_j]
PR ← PageRank(G)                          ▷ damping factor 0.85
H ← types in V sorted by PR[y] in descending order
return H
```

3.4 Task Sampling Strategy for Reptile Meta-Learning

Let K denote the number of support shots and define the split ratio of category-wise training as $r_{\text{train}} \in \{0.3, 0.4, \ldots, 0.8\}$. [1] For each episode, we:

(1) sample N (either 19 or 50 in this study) categories uniformly,

(2) draw K support examples per category from the training pool,

(3) draw validation and query examples from their respective pools.

To control memory usage, we limit pool sizes to 30,000 (support), 500 (validation), and 400 (query) instances per category and pregenerate 200 episodic task sets. The learning process performs inner loop updates followed by a meta-update with step size α.

4 Model Training

Notation. Let N be the number of semantic categories, M the number of prototypes per category, and $D = 50$ the dimensionality of each prototype. The shared span-representation space has dimensionality $d_{\text{representation}} = 512$. For the fastText branch, we use static embeddings of size $d_{\text{embed}} = 300$ and sinusoidal positional encodings of size $d_{\text{position}} = 200$. We initialized all weights with Xavier Uniform [20] under a fixed seed 42 for reproducibility. We ran all experiments on a single NVIDIA L4 GPU (22.5GB VRAM) within Google Colab's high-RAM environment. [2]

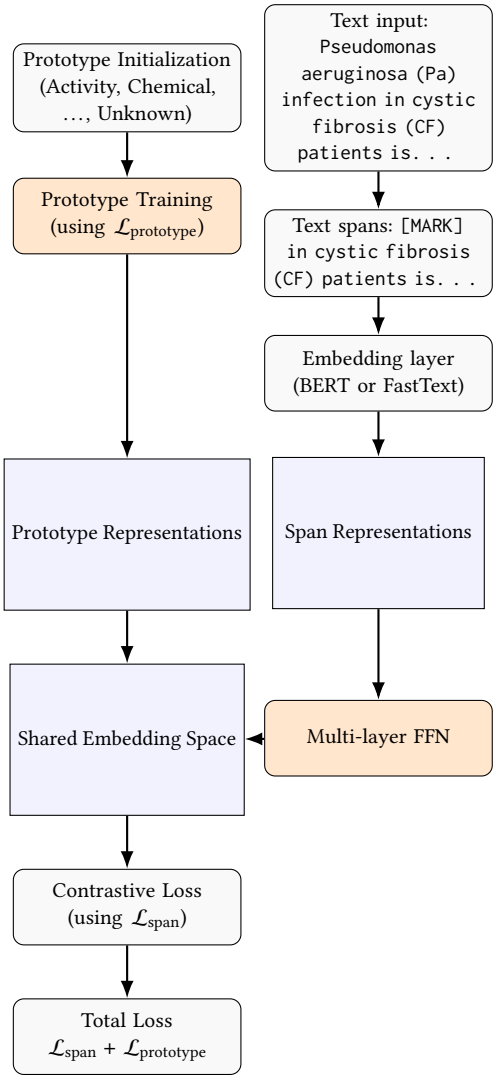

Figure 1: A high-level overview of the model pipeline for a single task, where orange boxes indicate learnable parameters. The system begins with the initialization and training of the prototype. Text inputs are tokenized into spans, embedded (e.g., via BERT [19]), and transformed into span representations.

4.1 Prototype Matrix

Each category maintains $M = 10$ learnable prototypes to capture semantic diversity. The resulting tensor is

$$\mathbf{P} \in \mathbb{R}^{(N \times M) \times D}.$$

[1] Validation and query splits are fixed at 10% and $(1 - r_{\text{train}}) - 0.10$, respectively.

[2] All codes and experiment results are in the following GitHub repository: https://github.com/FewshotMedicalNer/fewshot_medical_ner

Every prototype vector is ℓ_2 normalized after each meta-update to calculate cosine similarity.

4.2 Context Encoder (fastText Setting)

Given a token sequence of length $T = 300$, we concatenate static fastText embeddings[21] with positional encodings, described as

$$\mathbf{X} \in \mathbb{R}^{T \times (d_{\text{embed}} + d_{\text{position}})}.$$

A single-layer bidirectional LSTM[22] with hidden size $h = 1024$ per direction produces contextual states:

$$\mathbf{H} = \text{BiLSTM}(\mathbf{X}) \in \mathbb{R}^{T \times 2h}.$$

A linear projection $\mathbf{W}_p \in \mathbb{R}^{2h \times d_{\text{representation}}}$ projects token representations in the $d_{\text{representation}}$ space, to align with the output of the BERT encoder in our alternative configuration.

4.3 Sinusoidal Positional Encoding

We adopt the sinusoidal encoding from [23]. For positions $p \in [0, T-1]$ and dimensions $i \in \left[0, \frac{d_{\text{pos}}}{2} - 1\right]$

$$P_{p,\,2i} = \sin\left(p/10000^{2i/d_{\text{position}}}\right), \tag{1}$$

$$P_{p,\,2i+1} = \cos\left(p/10000^{2i/d_{\text{position}}}\right), \tag{2}$$

The matrix $\mathbf{P} \in \mathbb{R}^{T \times d_{\text{position}}}$ is concatenated with $\mathbf{X}$ along the feature dimension.

4.4 Span Projection Network

The span embedding of each candidate $\mathbf{s} \in \mathbb{R}^{d_{\text{representation}}}$ is processed by a feed-forward network of L layers:

$$\mathbf{v}^{(\ell+1)} = \text{Dropout}\Big(\text{GELU}\Big(\text{BatchNorm}\big(\mathbf{W}^{(\ell)}\mathbf{v}^{(\ell)} + \mathbf{b}^{(\ell)}\big)\Big)\Big), \ \mathbf{v}^{(0)} = \mathbf{s}.$$

A final affine transformation followed by Layer Normalization [24] projects into the prototype space:

$$\mathbf{z} = \text{LayerNorm}\big(\mathbf{W}^{(\text{out})}\mathbf{v}^{(L)} + \mathbf{b}^{(\text{out})}\big) \in \mathbb{R}^{D}.$$

We computed cosine similarity between the $\mathbf{z}$ and prototype vectors during training and inference. In this study, we set $L = 1$ to avoid overfitting.

4.5 Span Representation Embedding

We evaluate two span-encoding strategies:

(1) **fastText + BiLSTM branch** For each SpaCy token, we obtain a fastText vector $\mathbf{e} \in \mathbb{R}^{d_{\text{embed}}}$ and apply the weighted mean for multi-word tokens:

$$\mathbf{e} = \sum_{i=0}^{C-1} w_i\,\hat{\mathbf{e}}_i, \quad w_i = 1 + 0.1\left(\frac{C-i}{2}\right)^2, \quad \hat{\mathbf{e}}_i = \frac{\mathbf{e}_i}{\|\mathbf{e}_i\|_2}.$$

The token sequence is right-padded to length T with zeros, and concatenated with positional encodings:

$$\mathbf{X} = \left[\mathbf{E} \parallel \mathbf{P}\right] \in \mathbb{R}^{T \times (d_{\text{embed}} + d_{\text{position}})}.$$

A single layer BiLSTM (h=1024 per direction) produces contextual states $\mathbf{H} \in \mathbb{R}^{T \times 2h}$. The hidden state at the marker position becomes the span representation $\mathbf{s} \in \mathbb{R}^{d_{\text{representation}}}$.

(2) **BERT branch** Let $(s_{\text{start}}, s_{\text{end}})$ denote the range of subtoken indexes of the marked span (computed by aligning SpaCy tokens with the BERT wordpiece sequence and skipping all [CLS]/[SEP] markers). We feed the entire sequence into the vanilla BERT-base-cased model and obtain hidden states $\mathbf{H} \in \mathbb{R}^{T \times d_{\text{bert}}}$. The span representation $\mathbf{s} \in \mathbb{R}^{d_{\text{representation}}}$ is a pooled tensor, using the mean-max scheme. As BERT inherently captures bidirectional context, this branch does not need an extra BiLSTM layer.

4.6 Loss Functions

The total objective is the sum of a *prototype repulsion loss* term and a *span alignment loss* term:

$$\mathcal{L} = \mathcal{L}_{\text{proto}} + \mathcal{L}_{\text{span}}.$$

Prototype repulsion loss. Inspired by a methodology suggested in EPNet, we maximize the angular separation between $N \times M$ prototypes $\{\mathbf{p}_0, \ldots, \mathbf{p}_{N \times M - 1}\}$:

$$\mathcal{L}_{\text{proto}} = \frac{1}{NM} \sum_{i=0}^{NM-1} \max_{j \neq i}\big(\cos(\mathbf{p}_i, \mathbf{p}_j) + 1\big).$$

The offset +1 ensures a positive quantity and penalizes highly aligned pairs.

Span alignment loss. Let $\{\mathbf{q}_0, \ldots, \mathbf{q}_{NK-1}\}$ be the projected span vectors of a training episode (K spans per category). We first compute the squared angular distance $H_{ij} = (1 - \cos(\mathbf{p}_i, \mathbf{q}_j))^2$. For each category c, we take the minimum distance between its M prototypes:

$$\widetilde{H}_{c,j} = \min_{cM \le k < (c+1)M} H_{kj}.$$

The span alignment loss encourages each query span to align closely with the prototype of its category while remaining distant from the others, enabling a robust classification of biomedical entities:

$$\mathcal{L}_{\text{span}} = \sum_{c=0}^{N-1} \frac{\frac{1}{K}\sum_{j=cK}^{(c+1)K-1} \widetilde{H}_{c,j}}{\sum_{j=0}^{NK-1} \widetilde{H}_{c,j}}.$$

4.7 Meta-Learning with Reptile

Each meta-episode involves (1) sampling a few-shot task from the pool, (2) fine-tuning a task model in its support set under the contrastive objective of Section 4.6, and (3) performing a first-order meta-learning update to the meta model. The algorithm 2 describes the procedure.

We set inner epochs to $E = 5$ (fastText) and $E = 3$ (BERT), with learning rates $\gamma = 5 \times 10^{-4}$ and $\gamma = 10^{-3}$, respectively, optimized by cross-validation. The meta-step sizes are $\alpha = 0.4$ and $\alpha = 0.5$. Cosine decay scheduling [25] and gradient clipping [26] (max norm 1.0) stabilize the training.

4.8 Hard-Negative Training

After the initial meta-training phase, we identify hard-negative spans that the model assigns high scores to despite belonging to other categories. We conducted a second meta-training round using the resampled support sets under the same hyperparameters. We compared the performance of the model against a model trained without hard-negative mining to evaluate changes in classification accuracy.

Algorithm 2 Reptile Meta-Training Loop

Require: Initial meta-parameters θ_0, pool of episodic tasks $\mathcal{T}$, validation set $\mathcal{V}$, meta step size α, task learning rate γ, inner epochs $\mathcal{E}$, outer epochs M, patience $\mathcal{P}$
Ensure: Best meta-parameters θ^*

```
θ ← θ_0
for outer epoch m = 1 to M do
    Sample few-shot task S ~ T
    clone meta-model: θ'_0 ← θ
    Initialize task optimizer (learning rate = γ) and cosine scheduler
    for inner epoch ⌉ = 1 to E do
        Generate span projections
        Compute L_proto, L_span
        L ← L_proto + L_span
        Perform a gradient descent step on θ'
    end for
    θ'_1 ← θ'                                   ▷ fine-tuned parameters
    Reptile update:
        θ ← θ + α(θ'_1 − θ'_0)                  (Reptile)
    ℓ_2-normalize all prototype vectors in θ
    Evaluate F_1 on V; keep best weights θ*
    if no F_1 improvement for P outer epochs then
        break                                   ▷ early stopping
    end if
end for
return θ*
```

5 Discussion

5.1 Prototype Representation and Optimization

We adopt a multi-prototype strategy, allocating M distinct prototypical vectors per category to capture the variability within the same category. We chose cosine similarity over Euclidean distance for two reasons:

(1) In high-dimensional spaces, Euclidean distances are less effective metrics [27], whereas the cosine distance remains discriminative under ℓ_2 normalization [28].
(2) Our projection vectors are already normalized, so angular objectives converge more smoothly and yield more precise decision boundaries.

Figure 2 visualizes the vectors projected from the query set trained with a 0.3 training split ratio of the NER model based on fastText of 19 ways via the UMAP dimensionality reduction [29] condition. The plot indicates that some categories have converged into discrete clusters; however, the number of clusters is less than the number of prototypes assigned per category that we set.

5.2 Performance Comparison

We benchmarked two embedding backbones: a lightweight fastText + BiLSTM encoder and a contextualized BERT model. As reported in Table 1, fastText embedding achieved a comparable macro-F_1 performance compared to BERT embedding. Considering its minimal memory footprint, faster inference, and straightforward deployment, the fastText pipeline may be a cost-effective option when working with constrained data and computing resources.

Table 1: Macro-F_1 (%) of fastText vs. BERT under various training-query split ratios for 19-way and 50-way classification.

19-way	0.8	0.7	0.6	0.5	0.4	0.3
fastText	43.28	46.39	49.45	47.58	50.18	50.80
BERT	46.37	45.97	48.26	47.68	48.09	51.03
Mean	fastText = 47.95 | BERT = 47.90					
50-way	0.8	0.7	0.6	0.5	0.4	0.3
fastText	46.83	50.50	49.89	51.89	50.71	46.85
BERT	53.79	54.60	53.78	52.03	51.11	46.73
Mean	fastText = 49.45 | BERT = 52.01					

Table 2: Scalability comparison. ΔF_1 is the absolute Macro-F_1 drop; "Rel. drop" is $\Delta F_1/F_1^{small} \times 100$ (%). Few-NERD results are from cited papers; MedMentions scores are our 19-way to 50-way experiment.

Method	**Smaller set**		**Larger set**		ΔF_1	Rel. drop (%)
	Ways	F_1	Ways	F_1		
NNShot [30]	5	35.74	10	27.67	8.07	22.6
StructShot [30]	5	38.83	10	26.39	12.44	32.0
CONTaiNER	5	53.70	10	43.87	9.83	18.3
Decomposed [31]	5	63.23	10	56.84	6.39	10.1
ESD [32]	5	52.14	10	42.15	9.99	19.2
SpanProto	5	65.89	10	59.37	6.52	9.9
Three-stage [33]	5	65.22	10	58.35	6.87	10.5
Ours	19	50.80	50	46.85	3.95	**7.8**

We further assessed robustness under varying data budgets, training in 30% -80% of available samples. In particular, F_1 remains stable or improves with fewer examples (Table 1), underscoring the data efficiency of our reptile-based few-shot paradigm.

The confusion matrix in Figure 3 reveals the performance of one of our models, which uses a token embedding based on fastText + BiLSTM, trained on 30% of the total samples. This visualization reveals patterns of misclassification that can inform future improvements. The model demonstrates reasonable accuracy, with strong performance in specific categories such as `Group` (352 correct predictions), `Injury or Poisoning` (346), and `Organization` (326), where the recall exceeds 80%. These categories appear to have more distinct features, possibly due to their semantic uniqueness in the biomedical domain.

Notable confusion patterns include misclassifications between `Anatomical Structure` and `Substance` (35 and 55 instances, respectively), likely due to shared lexical features in biological texts (e.g., tissue or compound). `Phenomenon or Process` and

Table 3: Ablation on 19-way task (Macro-F_1, %).

Configuration	Change	Macro-F_1
Complete model	—	**50.80**
Single prototype/category	$M = 1$	49.13
Hard-neg. Off (random)	disabling hard-neg. mining	56.66
CE loss	cross-entropy (no contrastive)	2.70

`Natural Phenomenon or Process` show reciprocal errors (37 each), indicating potential label ambiguity in the training data or insufficient examples to learn fine-grained distinctions. The `Activity` is confused with `Occupational Activity` (45) and `Phenomenon or Process` (54), underscoring the model's difficulty in distinguishing between action-oriented and procedural entities, a challenge exacerbated by limited shots in our few-shot setup.

5.3 Class Scalability Analysis

Caveat. Previous work on class scalability has mainly used *Few-NERD*, a general domain benchmark. Therefore, absolute scores are not directly comparable to our biomedical runs. Instead, we focus on the relative performance drop when the number of categories roughly doubles; this ratio is domain independent and serves as a practical proxy for scalability.

Our model loses only **7.8%** when the label space expands from 19 to 50 biomedical types, while representative baselines show 10 to 32% degradation when categories double in Few-NERD (Table 2). We could not train baseline models on MedMentions due to VRAM limitations in Google Colab.

We attribute this robustness to our supervised contrastive objective. By minimizing category-wise minimum distances between spans and multiple prototypes, gradient domination from primary categories is alleviated, yielding balanced decision boundaries even under extreme category imbalance.

5.4 Ablation Study

To isolate the contribution of each architectural decision, we ablate three key components while keeping all other hyperparameters fixed (fastText + BiLSTM backbone, 0.3 split). Table 3 investigates the contribution of each architectural or training ingredient to macro-F_1 in general.

Replacing the supervised-contrastive objective with vanilla cross entropy causes a collapse to near-random performance (2.7%). It highlights that the contrastive learning process makes a substantial contribution in a few-shot setting. This property is crucial when there is an extreme imbalance of samples in each category.

Collapsing each category with a single centroid reduces F_1 by approximately 1.7 percentage points. The result confirms our intuition that biomedical entity categories are semantically diverse (e.g., gene symbols vs. full protein names) and therefore benefit from distributed rather than monolithic representations.

Turning off hard-negative sampling and returning to uniformly selected negatives increased performance to 56.66%. We hypothesize that gradient instability causes this, which is exacerbated by hard-negative mining a handful of high-loss examples, producing steep, noisy gradients that counteract the regularizing effect of contrastive loss. We believe that when the training set is already small, this variance outweighs the potential benefit of focusing on hard-negative samples.

Table 4: Meta-model F_1 scores across 3 iterations (mean ± 95% CI). Full indicates that all categories were used in F_1 calculation, while Base includes only the labels used in the first phase of training.

Split	Phase	Full F_1 (%)	Base F_1 (%)
A	1	-	51.36 [47.74, 54.98]
	2	52.47 [49.25, 55.69]	56.21 [53.36, 59.06]
B	1	-	55.53 [51.55, 59.51]
	2	52.43 [52.27, 52.60]	54.30 [53.48, 55.13]
C	1	-	53.59 [50.87, 56.31]
	2	51.06 [48.10, 54.02]	51.70 [50.55, 52.86]

5.5 Tag Set Extension

In addition to validating the model on a fixed set of labels, we experimented with adding previously unseen labels during the training process to assess the model's ability to retain previously learned information while integrating new information.

From the 19-way label set, we randomly split the 18 labels except for `UnknownType` into six groups of three. For each split, the model was trained for 100 epochs on only two-thirds of the labels not in the split, alongside `UnknownType` in the initial phase (phase 1). We then trained the model for an additional 100 epochs on the complete set of labels (Phase 2). We repeated the process three times with different seeds for each iteration (42, 123, 999).

- **Split A**: Anatomical Structure, Idea or Concept, Injury or Poisoning, Intellectual Product, Occupation or Discipline, Organization
- **Split B**: Conceptual Entity, Manufactured Object, Natural Phenomenon or Process, Organism Attribute, Phenomenon or Process, Substance
- **Split C**: Activity, Behavior, Finding, Group, Occupational Activity, Organism

Table 4 describes the performance on the Base subset of labels, which remains similar or improved despite the addition of previously unseen labels. F_1 in Split A showed a statistically significant performance improvement ($p = 0.039$) in the categories of the Base subset. Split B and C showed a performance decrease, but the effects were minimal ($p = 0.554$ for split B, $p = 0.211$ for split C). These results demonstrate the robustness of our model to changes in the distribution of labels.

Comparison of the full F_1 and the base F_1 score for each split showed that new labels can be added late in the training process without a decrease in performance ($p = 0.653$ for split A, $p = 0.128$ for split B, $p = 0.218$ for split C).

6 Conclusion

We proposed ReProCon, a cosine-contrastive few-shot NER framework for biomedical text that integrates supervised contrastive

learning with Reptile meta-updates. Using a lightweight **fastText + BiLSTM** encoder, ReProCon achieves ~99% of a baseline Macro-F_1 based on BERT while requiring significantly less memory. Performance remains stable with a label budget of 30% and decreases by only 3.95 percentage points (7.8%) when expanding from 19 to 50 categories. These results demonstrate that state-of-the-art biomedical few-shot NER is achievable without large language models, making ReProCon well-suited for resource-constrained environments.

In future work, we plan to:

(1) Conduct cross-domain meta-training from PubMed abstracts to clinical notes and patent texts to evaluate domain-shift robustness.

(2) Comparison of performances between models from other studies, by training on biomedical datasets.

Acknowledgments

We acknowledge the Discord user `aneesh_qai`, who suggested valuable ideas, including Xavier uniform initialization for the prototype tensor, cosine decay scheduling, and mean-max pooling for span representation using BERT.

References

[1] Sunil Mohan and Donghui Li. Medmentions: A large biomedical corpus annotated with umls concepts, 2019.

[2] M Krallinger, A Morgan, L Smith, F Leitner, L Tanabe, J Wilbur, L Hirschman, and A Valencia. The biocreative ii-critical assessment for information extraction in biology challenge. *Genome Biology*, 9:S1, 2008. doi: https://doi.org/10.1186/gb-2008-9-s2-s8.

[3] Rezarta Islamaj Doğan, Robert Leaman, and Zhiyong Lu. Ncbi disease corpus: a resource for disease name recognition and concept normalization. *Journal of biomedical informatics*, 47:1–10, 2014. doi: https://doi.org/10.1016/j.jbi.2013.12.006.

[4] Ning Ding, Guangwei Xu, Yulin Chen, Xiaobin Wang, Xu Han, Pengjun Xie, Haitao Zheng, and Zhiyuan Liu. Few-nerd: A few-shot named entity recognition dataset, 2021.

[5] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms, 2018.

[6] Olivier Bodenreider. The unified medical language system (umls): integrating biomedical terminology. *Nucleic Acids Research*, 32:D267–D270, 2004. doi: https://doi.org/10.1093/nar/gkh061.

[7] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, 2020. doi: https://doi.org/10.1093/bioinformatics/btz682.

[8] Jianing Wang, Chengcheng Han, Chengyu Wang, Chuanqi Tan, Minghui Qiu, Songfang Huang, Jun Huang, and Ming Gao. Spanproto: A two-stage span-based prototypical network for few-shot named entity recognition, 2022.

[9] Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computing for Healthcare*, 3(1):1–23, 2021. doi: https://doi.org/10.1145/3458754.

[10] Iz Beltagy, Kyle Lo, and Arman Cohan. Scibert: A pretrained language model for scientific text, 2019.

[11] Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning, 2017. URL https://doi.org/10.48550/arXiv.1703.05175.

[12] Pascal Mettes, Elise van der Pol, and Cees G. M. Snoek. Hyperspherical prototype networks, 2019.

[13] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks, 2017.

[14] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning, 2021.

[15] Sarkar Snigdha Sarathi Das, Arzoo Katiyar, Rebecca Passonneau, and Rui Zhang. Container: Few-shot named entity recognition via contrastive learning, 2022.

[16] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1–7):107–117, 1998. doi: https://doi.org/10.1016/S0169-7552(98)00110-X.

[17] Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. spacy: Industrial-strength natural language processing in python, 2020.

[18] Bin Ji, Shasha Li, Shaoduo Gan, Jie Yu, Jun Ma, and Huijun Liu. Few-shot named entity recognition with entity-level prototypical network enhanced by dispersedly distributed prototypes, 2022.

[19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies*, volume 1, pages 4171–4186, Minneapolis, Minnesota, 2019. Association for Computational Linguistics. doi: https://doi.org/10.18653/v1/N19-1423.

[20] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, volume 9, pages 249–256, Chia Laguna Resort, Sardinia, Italy, 2010. PMLR. URL https://proceedings.mlr.press/v9/glorot10a.html.

[21] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017. doi: https://doi.org/10.1162/tacl_a_00051.

[22] Alex Graves, Santiago Fernández, and Jürgen Schmidhuber. Bidirectional lstm networks for improved phoneme classification and recognition. In *Artificial Neural Networks: Formal Models and Their Applications – ICANN 2005*, pages 799–804, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. doi: https://doi.org/10.1007/11550907_163.

[23] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[24] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016.

[25] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts, 2017.

[26] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, volume 28, pages 1310–1318, Atlanta, Georgia, USA, 2013. PMLR. URL https://proceedings.mlr.press/v28/pascanu13.html.

[27] Charu C. Aggarwal, Alexander Hinneburg, and Daniel A. Keim. On the surprising behavior of distance metrics in high dimensional space. In *Proceedings of the 8th International Conference on Database Theory*, page 420–434, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg. doi: https://doi.org/10.1007/3-540-44503-X_27.

[28] Tuomo Korenius, Jorma Laurikkala, and Martti Juhola. On principal component analysis, cosine and euclidean measures in information retrieval. *Information Sciences*, 177(22):4893–4905, 2007. doi: https://doi.org/10.1016/j.ins.2007.05.027.

[29] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction, 2020.

[30] Yi Yang and Arzoo Katiyar. Simple and effective few-shot named entity recognition with structured nearest neighbor learning, 2020.

[31] Tingting Ma, Huiqiang Jiang, Qianhui Wu, Tiejun Zhao, and Chin-Yew Lin. Decomposed meta-learning for few-shot named entity recognition. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1584–1596, Dublin, Ireland, 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-acl.124.

[32] Guanghai Wang, Yudong Liu, and James Hearne. Few-shot learning for sumerian named entity recognition. In *Proceedings of the Third Workshop on Deep Learning for Low-Resource Natural Language Processing*, pages 136–145, Hybrid, 2022. Association for Computational Linguistics. doi: https://doi.org/10.18653/v1/2022.deeplo-1.15.

[33] Shengjie Ji and Fang Kong. A novel three-stage framework for few-shot named entity recognition. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 1293–1305, Torino, Italy, 2024. ELRA and ICCL. URL https://aclanthology.org/2024.lrec-main.116/.

Appendix

Table 5: 50 Categories and Frequencies of Merged Categories.

Type	Frequency	Type	Frequency
UnknownType	6,947,708	Phenomenon or Process	2,239
Biologic Function	47,840	Injury or Poisoning	2,215
Chemical	39,839	Sign or Symptom	2,104
Qualitative Concept	33,479	Clinical Attribute	2,001
Health Care Activity	28,221	Professional or Occupational Group	1,984
Functional Concept	26,191	Substance	1,910
Fully Formed Anatomical Structure	23,592	Food	1,538
Quantitative Concept	18,252	Classification	1,449
Finding	16,521	Body Substance	1,403
Spatial Concept	14,354	Organism	1,234
Research Activity	10,644	Individual Behavior	1,127
Idea or Concept	10,296	Biomedical Occupation or Discipline	1,019
Intellectual Product	10,167	Social Behavior	1,015
Temporal Concept	10,159	Laboratory or Test Result	987
Group	10,066	Occupational Activity	969
Eukaryote	9,288	Daily or Recreational Activity	944
Activity	8,643	Anatomical Abnormality	868
Population Group	6,738	Family Group	863
Manufactured Object	4,082	Educational Activity	643
Organism Attribute	3,864	Occupation or Discipline	614
Conceptual Entity	3,413	Health Care Related Organization	554
Natural Phenomenon or Process	3,162	Human-caused Phenomenon or Process	509
Medical Device	2,638	Anatomical Structure	445
Bacterium	2,243	Organization	369
Virus	1,412	Governmental or Regulatory Activity	193

Table 6: 19 Categories and Frequencies of Merged Categories.

Type	Frequency	Type	Frequency
UnknownType	6,947,708	Organism	14,160
Idea or Concept	112,603	Intellectual Product	11,092
Natural Phenomenon or Process	51,014	Activity	9,044
Substance	44,679	Manufactured Object	6,955
Occupational Activity	40,945	Organism Attribute	5,865
Anatomical Structure	24,858	Conceptual Entity	3,413
Group	19,616	Phenomenon or Process	2,744
Finding	19,594	Behavior	2,670
Injury or Poisoning	2,215	Occupation or Discipline	1,633
Organization	923		

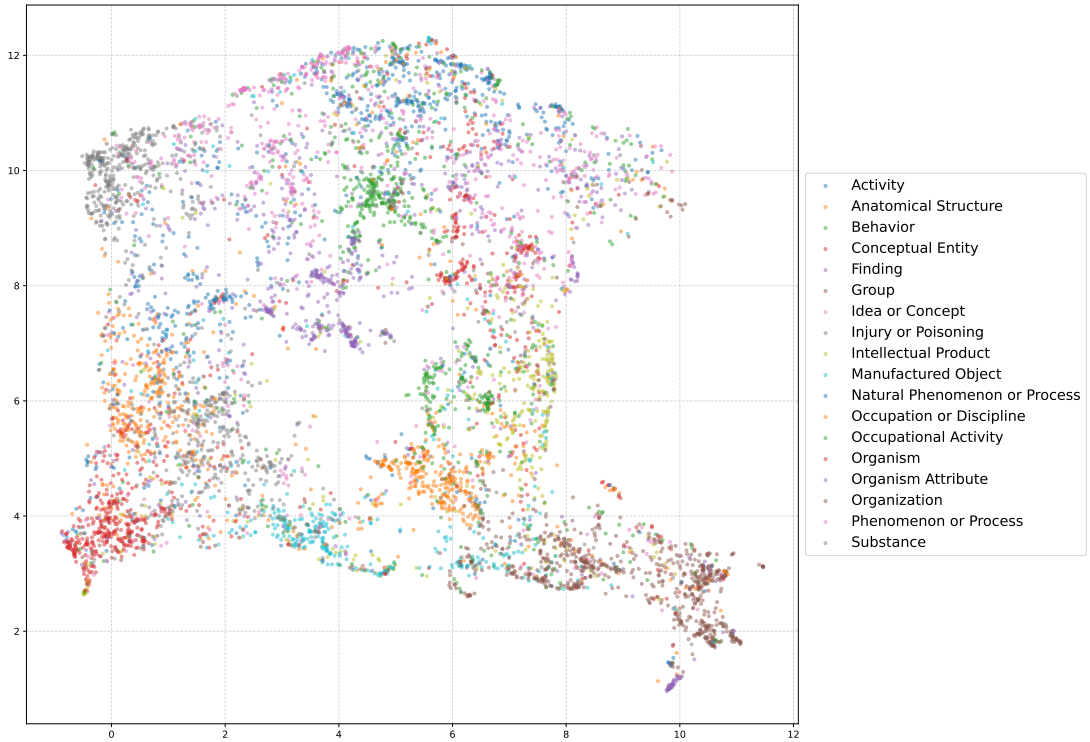

Figure 2: Two-dimensional UMAP projection of span representations.

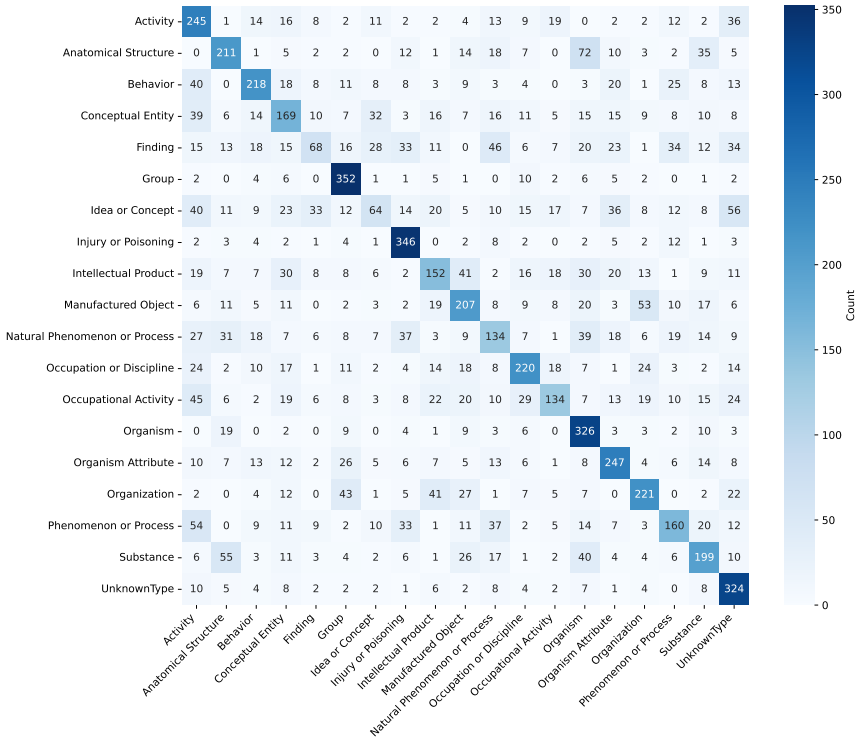

Figure 3: A confusion matrix of test samples.