Improving Few-Shot Cross-Domain Named Entity Recognition by Instruction Tuning a Word-Embedding based Retrieval Augmented Large Language Model

Subhadip Nandi
IIT Kanpur

Neeraj Agrawal
IISc Bangalore

Introduction

Few-Shot Cross-Domain NER is the process of leveraging knowledge from data-rich source domains to perform entity recognition on data-scarce target domains. However, most previous SOTA approaches suffer from one of the below problems:

- Require finetuning or architecture modifications for use in target domain
- Require making calls to proprietary LLMs like GPT4

In this work, we propose **IF-WRANER** (Instruction Finetuned Word-embedding based Retrieval Augmented large language model for Named Entity Recognition), a retrieval augmented LLM, finetuned for the NER task. By virtue of the regularization techniques used during LLM finetuning and the adoption of word-level embedding over sentence-level embedding during the retrieval of in-prompt examples, IF-WRANER is able to outperform previous SOTA Few-Shot Cross-Domain NER approaches.

The main contributions of our work are summarised as follows:

- Our approach finetunes an open-source LLM (Llama2 7B) once on source domain data, making it adaptable to new domains without further finetuning
- Our regularized finetuning method prevents model overfitting on source domain data.
- The use of word-level embedding similarity instead of the conventional sentence-level embedding similarity boosts entity recognition performance.
- We also introduce a smaller version of our model, obtained by finetuning Tinyllama, and this is particularly well suited for domains with low latency and high throughput requirements.

Methodology

Problem Definition

The NER task involves identifying sequences of words in a sentence as entities and categorizing them into predefined entity types. With Few-Shot Cross-Domain NER, we have limited labelled target domain examples but can leverage labelled data from a source domain for model building.

Model Input:
1. User Query -> "Chavez's party, The United Socialist Party of Venezuela drew 48% votes"
2. Entity Types -> politician, person, political party, event, election etc.

Model Output:
{"politician": ["Chavez"], "political party":["The United Socialist Party of Venezuela"]}

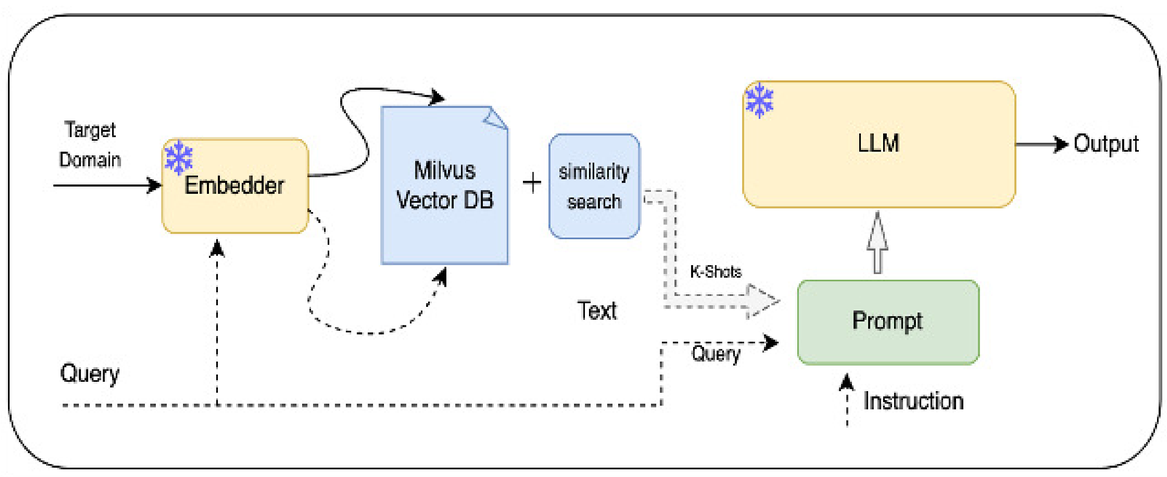
Prompting LLM for NER

We treat NER as a text generation task instead of sequence labelling. Our experimental prompts follow this format:

- Task Description (part of Instruction in figure)
- Entity Definitions (part of Instruction in figure)
- Input Output Examples (K-shots in figure)
- User Query (Query in figure)

Retrieval Augmented Generation (RAG)

For selecting relevant examples to include in LLM prompt we employ the RAG framework.



Retrieval Augmented Generation with LLM

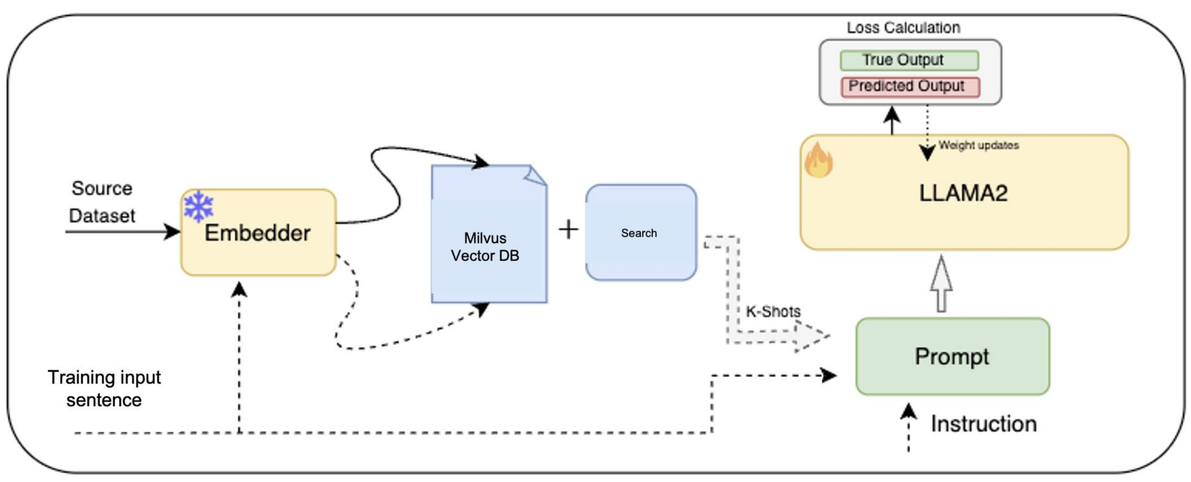
	Performance	Cost
RAG with GPT4	Adequate	High
RAG with Llama2	No structured output	Low

Finetuning open-source LLMs for Few-Shot Cross-Domain NER

Finetuning open-source models is necessary to ensure prompt adherence and reduce costs at scale. We finetune the Llama2-7B model on this source domain data.

Finetuning teaches the model to:

1. Perform NER
2. Adhere to output structure specified in prompt



Finetune Llama2-7B on source domain data

Training Regularization

Model overfitting. Not following instruction in prompt.

- Input:
  - Utterance: "President John F. Kennedy was shot and killed on November 22, 1963"
  - Entity-type list: ["politician", "person", "organization", "political party"]
- Prediction:
  - {"person": "John F. Kennedy"}

John F. Kennedy was tagged as a "person" instead of "politician" by the model despite the explicit instruction in Person definition that only non-politicians were to be tagged as Person.

To address this, we applied the following regularization techniques:

- Delete some of the entity types randomly from input and output of some training examples. This forces the model to learn to only output entity types defined in the input.
- We randomly shuffled the order of entity types in some training examples, to train the model to be agnostic to entity type ordering in prompt.

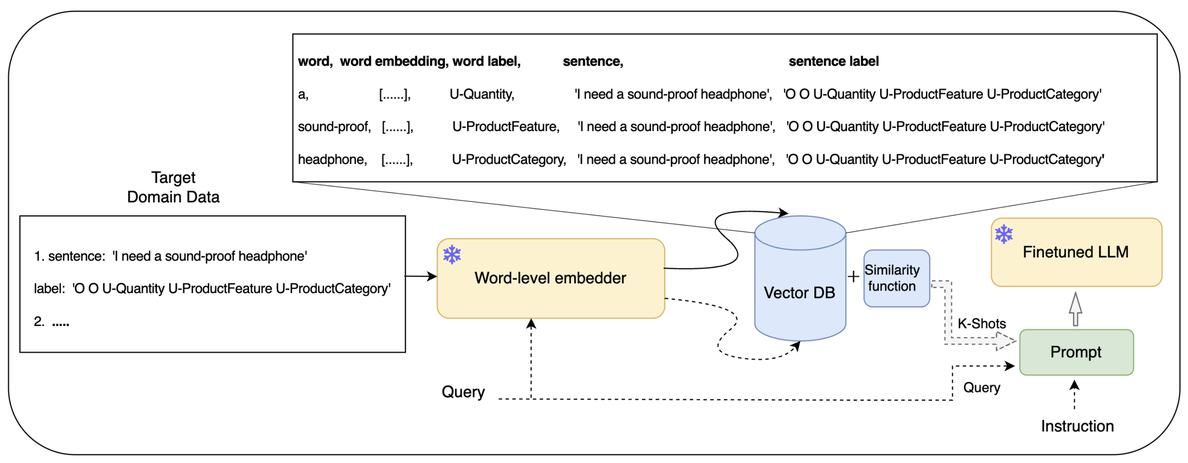
Using word-level embedding instead of sentence-level embedding

NER is a word-level task that focuses more on local evidence rather than a sentence-level task

Query sentence: I want to buy a *16-inch macbook* from store.

Relevant "in-prompt" example for LLM?
Option 1: I want to buy a table from store. (sentence-level embedding) v/s
Option 2: Show me a *16-inch macbook*. (word-level embedding)

Option 2 is much better for entity recognition!
Solution: Select top k "in-prompt" examples by comparing similarity scores for word-level embeddings instead of sentence-level embeddings.



Using word-level embedding instead of sentence-level embedding

Results

Model	Politics	Natural Science	Music	Literature	AI	Average
LST-NER (Zheng et al., 2022)	73.25	70.07	76.83	70.76	63.28	70.84
LANER (Hu et al., 2022)	74.06	71.83	78.78	71.11	65.79	72.31
CP-NER (Chen et al., 2023)	74.25	75.82	79.10	72.17	67.95	73.86
GPT-NER (Wang et al., 2023)	74.71	70.77	78.30	62.18	66.07	70.41
PromptNER (GPT3.5) (Ashok and Lipton, 2023)	71.74	64.83	77.78	64.15	59.35	67.57
PromptNER (GPT4) (Ashok and Lipton, 2023)	78.61	72.59	84.26	74.44	64.83	74.95
RAG + GPT4 using sentence embeddings	78.2	73.52	83.61	71.32	66.91	74.71
RAG + GPT4 using word embeddings	78.63	73.95	84.25	74.68	68.19	75.94
IF-WRANER (ours)	**79.8**	**75.31**	**85.43**	**75.52**	**68.81**	**76.97**

Comparison of previous Cross-Domain NER SOTA models with IF-WRANER in terms of F1 scores(%)

Model	Performance			Latency (s)	QPS	Cost/month ($)
	CrossNER	Domain A	Domain B			
IF-WRANER (ours)	76.97	83.72	79.95	2X	1X	1X
Tiny-IF-WRANER (ours)	73.62	79.64	76.46	1X	1X	1X
RAG with GPT 4 (our implementation)	74.71	80.95	78.04	4X	1X	120X
PromptNER with GPT4(Ashok and Lipton, 2023)	74.95	81.15	78.12	4.2X	1X	120X

Comparison of model performance, latency, throughput and cost for IF-WRANER, Tiny-IF-WRANER, RAG+GPT4 and PromptNER. F1 score(%) is used as model performance metric.

Model Deployment

We serve IF-WRANER on Triton Inference Server using TensorRT framework, with separate instances serving each domain based on traffic and latency needs. While IF-WRANER achieves good latency and throughput numbers on A100 GPUs, for domains with very low latency needs we developed **Tiny-IF-WRANER**, a smaller finetuned LLM with Tinyllama (1.1B parameter variant of Llama2) as its base model. Although Tiny-IF-WRANER has lower F1 scores, it meets strict latency requirements. Compared to GPT-4 models, IF-WRANER and Tiny-IF-WRANER provide similar throughput with much lower latency and cost. We have deployed both IF-WRANER and Tiny-IF-WRANER for different customer care domains of an e-commerce enterprise based on their latency and throughput requirements.

Conclusion

In this work, we have introduced IF-WRANER, a cost-effective, retrieval-augmented instruction following LLM that excels in. Few-Shot Cross-Domain NER without the need for finetuning or structural changes to adapt to new domains. It's user-friendly, requiring only entity type definitions and a few labelled examples from end-users. For low latency domains, we proposed Tiny-IF-WRANER, which uses Tinyllama as its base LLM.