Study of long-context LLM capabilities for Fine-grained Named Entity Recognition in Indian Languages

A B. Tech Project Report Submitted
in Partial Fulfillment of the Requirements
for the Degree of

Bachelor of Technology

by

Adittya Gupta
(210101007)

Gautam Sharma
(210101042)

under the guidance of

Prof. Ashish Anand

to the

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI
GUWAHATI - 781039, ASSAM

CERTIFICATE

*This is to certify that the work contained in this thesis entitled "**Study of long-context LLM capabilities for Fine-grained Named Entity Recognition in Indian Languages**" is a bonafide work of **Adittya Gupta** (**Roll No. 210101007**) and **Gautam Sharma** (**Roll No. 210101042**), carried out in the Department of Computer Science and Engineering, Indian Institute of Technology Guwahati under my supervision and that it has not been submitted elsewhere for a degree.*

Supervisor: **Prof. Ashish Anand**

Professor,

April, 2025

Department of Computer Science & Engineering,

Guwahati.

Indian Institute of Technology Guwahati, Assam.

Acknowledgements

We would take this opportunity to thank **Prof. Ashish Anand**, Department of Computer Science and Engineering, IIT Guwahati, **Mr. Prachuryya Kaushik**, PhD Candidate, Department of Computer Science and Engineering, IIT Guwahati, and **Ms. Ajanta Maurya**, PhD Candidate, Department of Computer Science and Engineering, IIT Guwahati, for their invaluable support during the Problem Statement Formulation, and helping in gaining extensive insight into the result of the experiments that were conducted during the making of this thesis project. I also thank **AMal Lab**, IIT Guwahati, for providing the necessary resources for conducting experiments.

Abstract

Vast training data has powered current advancements in machine and deep learning models. Recently, the demand for such datasets has increased in multiple ways to accommodate new and expanding domains. Previously, such datasets were made using excessive human intervention, such as crowdsourcing through a large population, hiring annotators, etc. Such methods are time-consuming, resource-consuming and have low scalability. This thesis project tries to automate the process of ***Fine-grained Named Entity Recognition for Indic Languages****, a task that is fundamental in creating large and accurate datasets, by studying long context large language models, making the process of creating large and labelled datasets fast and cheaper compared to older methods. Through this project, we propose a method to create Fine-grained Named Entity Recognition datasets for Indic languages and support the validity of our method with various experiments.*

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Recent breakthroughs in natural language processing (NLP) have been largely attributed to the availability of vast amounts of training data and the emergence of large-scale language models. Among these, Fine-grained Named Entity Recognition plays a crucial role in enhancing the quality and applicability of NLP tasks, especially for creating high-quality annotated datasets. Although much of the research has focused on high-resource languages like English, and tasks like Coarse-grained Named Entity Recognition, there remains a significant gap in the availability of such resources for Indian languages.

Indian languages, with their diverse scripts, grammar structures, and contextual nuances, pose unique challenges in NLP. Traditional methods of generating Fine-grained NER datasets, such as manual annotation and crowd-sourcing, are time-consuming, costly, and not scalable. This thesis explores the potential of long-context Large Language Models (LLMs) to automate and improve the identification of Fine-grained NER tags in Indic languages, thereby reducing reliance on manual annotation and expertise and increasing scalability and efficiency.

This project presents a systematic study of the capabilities of long-context LLMs to handle complex linguistic structures and maintain contextual coherence over longer text spans

in Indian languages. We propose a methodology to generate fine-grained NER datasets using LLMs and validate the approach with comprehensive experiments and evaluation metrics.

1.1 Organization of the Report

This chapter provides the necessary background and outlines the motivation for the work undertaken in this thesis. We began by discussing the challenges in Fine-grained NER for Indian languages and how long-context LLMs can potentially address them.

The rest of the report is organized as follows. In Chapter 2, we provide a comprehensive review of the existing literature on Fine-grained NER, LLM, and NLP for Indian languages. Chapter 3 describes the proposed methodology, including dataset preparation, model architecture, and prompt design. Chapter 4 presents the experimental setup, evaluation metrics, and results. Chapter 5 discusses the implications of the findings and limitations of our approach. Finally, Chapter 6 concludes the thesis and outlines possible directions for future research.

Chapter 2

Review of Prior Works

2.1 Popular datasets in the domain

We first looked at various datasets that are currently present in the domain in which we are working to get an understanding of the current SOTA datasets on which we have to improve. We studied two such datasets: **Few-NERD[3]** and **MultiCoNER-2[4]** in this context. The comparison between the fine-grained entity types is depicted in Fig. 2.1.

2.1.1 FewNERD [3]

- The paper introduces Few-NERD, a manual hand-labelled dataset designed for Fine-grained Named Entity recognition tasks.
- This is the largest manually hand-labeled fine-grain dataset currently available for English.
- This paper is important to get a basic understanding of how important good fine-labeled data is in terms of training the model for tasks and how costly it is to get such high-quality data. This work motivates the need for alternative, cheaper methods of fine-labeling datasets.

2.1.2 MultiCoNER-v2 [4]

- The paper introduces MultiCoNER-v2, a fine-grain labelled dataset designed for a fine-grained entity recognition task.
- This dataset consists of **12** languages, including multilingual subsets, with a total of **2.3M** instances.
- The authors used Wikidata by creating a gazetteer for the entities and then fine labelling using many of the Wikipedia sentences.

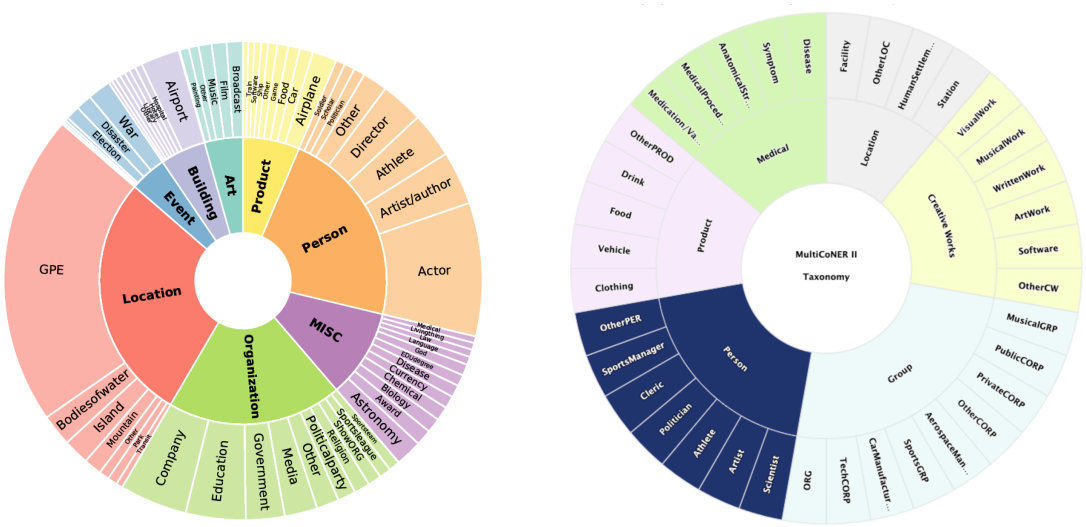


Figure 2.1: Comparison of the fine grained entities of *Few-NERD(left)* and *MultiCoNER-2(right)*. The outer circle represents the fine-grained entities, while the inner circle represents the coarse-grained ones. Few-NERD has in total **66** fine-grained entity types, while MCN-2 has only **33**.

2.2 State of the Art (SOTA) methods of dataset labelling for NER task

To our knowledge, no efforts have been made to create fine-grained multilingual named entity recognition datasets through programmatic labelling.

Dataset	Grain Type	Number of Types	Training Set Size
Naamapadam [5]	Coarse	3	400k (11 languages)
MultiCoNER-1 [6]	Coarse	6	1.8 Million (11 languages)
MultiCoNER-2	Fine	33	Hi: 25k, En: 12k
FINER [7]	Fine	94 (claimed 113)	2 Million (En)
OntoNotes [8]	Fine	88	251k (En)
HAnDS [9]	Fine	118	37 Million (En)
FewNERD [10]	Fine	66	180k (En)

Table 2.1: Comparison of NER Datasets (not exhaustive)

2.2.1 TaLLoR [11]

- This paper focuses to solve on the problem of coarse-grained entity labelling for raw datasets.
- The authors introduce a new approach **TaLLoR** (Tagging with Learnable Logical Rules), that automatically learns new logical rules from unlabeled data with the help of a small seed set of rules.
- The approach consists of iterative labelling of the dataset with the given set of rules followed by a Neural-net, which can learn new rules from this labelled data. Simply put the new rules into the set based on some threshold performance and repeat this step.
- The approach is indeed quite ingenious and yields impressive results. However, it is primarily effective for generating coarse labels due to the limited generalization capabilities of the generated logical rules. These rules tend to be less effective in generalizing for finer tags, which limits their applicability to our fine-grained labelling task.

2.2.2 Snorkel [12]

- This paper focuses on creating a development tool that can be used for getting a final label for each data point with many labelling functions at once, even with conflicting

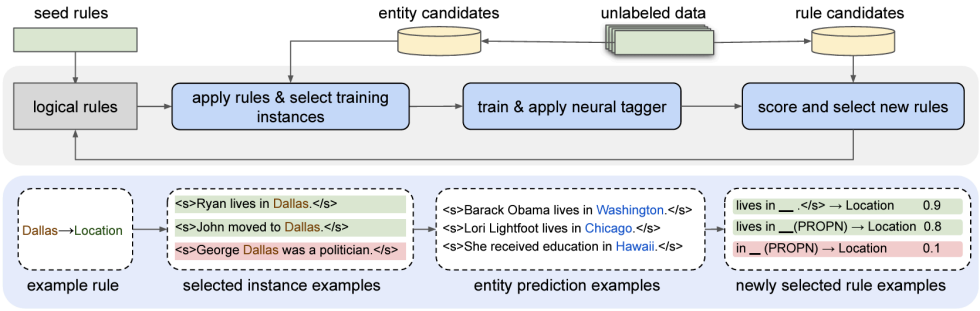


Figure 2.2: Overview of **TaLLoR** framework

labels.

- Snorkel's framework consists of three main stages: writing labelling functions, automatically modelling their accuracies and correlations, and using the probabilistic labels to train a discriminative model.

- A key feature of Snorkel is its ability to estimate the accuracies of these noisy labels and their correlations, facilitating the generation of probabilistic training labels.

- This is a very useful framework as it can be very useful in our case where we can have a large number of labelling functions that would be generated by LLM, and we need a consensus between them.

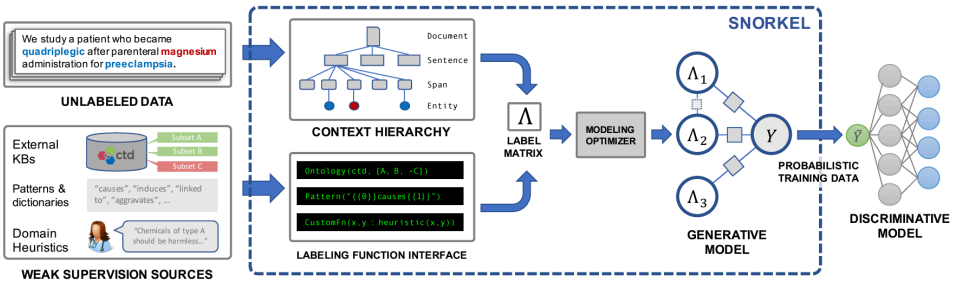


Figure 2.3: Overview of **Snorkel** framework

2.2.3 Coarse2Fine [13]

- This paper focuses on our problem of generating fine-grained labels for a dataset from its coarse-grained labels.
- Coarse (C2F) fine-tunes language models (like GPT-2) in a label-conditioned manner, where label surface names are added as prompts to the documents.
- The framework starts by generating pseudo-training data iteratively through the fine-tuned language model and then training a text classifier so as to refine its weak supervision.
- This framework only applies to dataset containing sentences with one entity. So basically, it cannot do context mapping to any other named entity, which is a major limitation of this method.

2.3 SOTA methods for labeling using LLM

We reviewed some works that use LLM for labelling datasets. This labelling is not for the NER task, but it was important to understand how good, in general, LLMs are in dataset labelling.

2.3.1 Can Large Language Models Design Accurate Label Functions? [14]

- This paper introduces a framework **Datasculpt** which basically can create labeling functions through LLM prompting
- Their focus is on the sentiment analysis rather than the NER task.
- They use different types of thresholds and heuristics to filter out the bad or noisy LFs in order to have confident predictions.

- They concluded that LLMs can perform well in designing keyword-based LFs for tasks requiring general knowledge. However, they are less effective in designing pattern-based LFs, and they are also less accurate in tasks requiring specific domain expertise.

2.3.2 Language Models in the Loop [15]

- This paper introduces the idea of prompt-labeled functions.
- , In other words,, it proposes an idea of using labelling functions that are not like normal Python functions, but are some kind of prompt questions about the text.
- The LLM will return some output to this prompt question.
- Later, a consensus is achieved among all the outputs of these prompt questions, and the final label is obtained.
- This paper solves the problem specifically for the spam detection task.

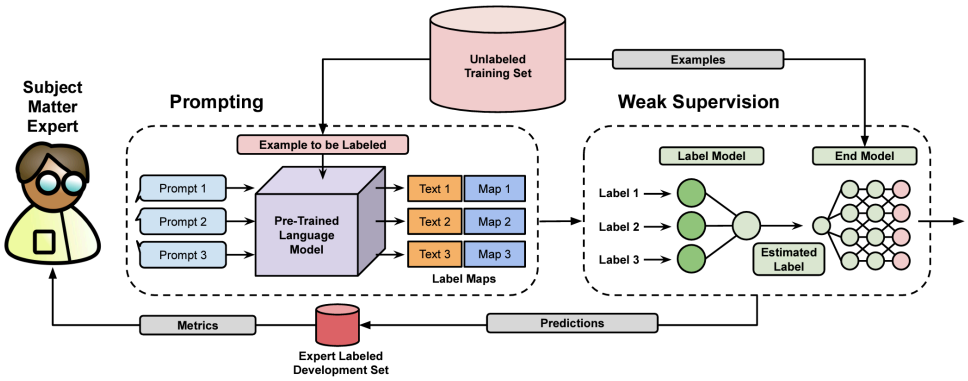


Figure 2.4: Language models in the loop

2.4 Few-shot learning models in NER

We also reviewed the current SOTA in the field of few-shot NER, specifically **CONTaiNER** [**16**]. Given its promising results on the Few-NERD dataset, it made sense to evaluate the quality of our dataset using this model.

2.4.1 CONTaiNER [16]

- This paper introduces a few-shot NER model via Contrastive Learning.
- Unlike traditional contrastive learners that optimize similarity objectives between point embeddings, CONTaiNER optimizes distributional divergence, effectively modelling Gaussian Embeddings.
- The model first trains on the source dataset by tuning its parameters for the divergence loss and then fine tunes itself for the given support set.
- It then uses nearest neighbor inference to find the best matching label for a token in query set

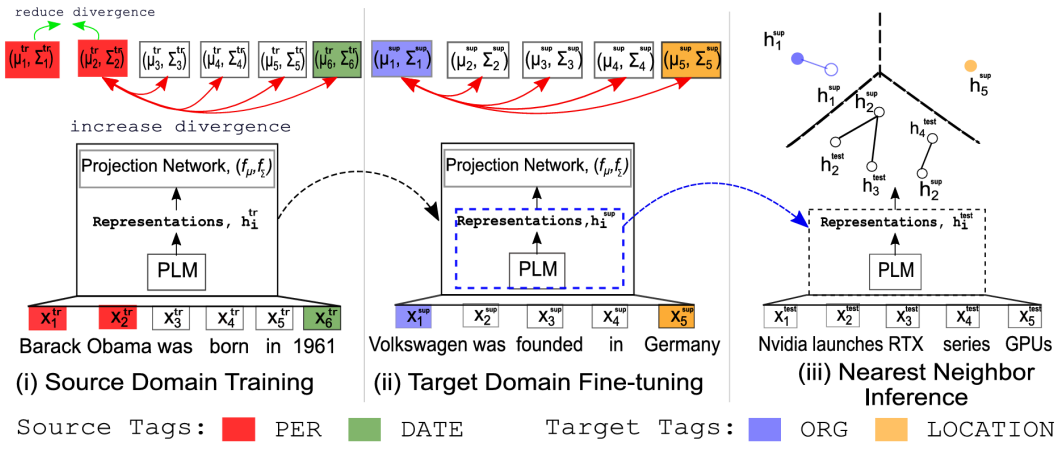


Figure 2.5: Overview of CONTaiNER framework. Here, PLM stands for pre-trained language model (BERT in this case).

2.5 Prompt Learning

Prompt learning is a technique used to guide large pre-trained language models (LLMs) to perform specific tasks by crafting natural language or structured prompts. Instead of fine-tuning the model's parameters, prompt learning exploits the model's existing knowledge

by framing the input in a way that elicits the desired output. Prompt learning can take various forms, including manual prompt engineering, automated prompt optimisation, and soft prompts (learnable vectors).

2.5.1 Prompt-Learning for Fine-Grained Entity Typing [1]

- The work presents a novel application of prompt-learning to the task of fine-grained entity typing. It demonstrates that prompt-based methods—especially those leveraging pretrained language models (PLMs)—can be adapted effectively to this task.

- The authors propose a **simple yet effective prompt-learning pipeline** that includes two key components: *entity-oriented templates* and *verbalizers*. Templates are carefully designed sentences that embed the entity mention within a cloze-style prompt (e.g., "`[Entity] is a type of [MASK]`"), enabling the PLM to predict the entity type using masked language modeling. Verbalizers then map type labels to words or phrases that align with the vocabulary and semantics of the PLM, facilitating accurate predictions.

- A key contribution of the paper is its extension to the **zero-shot setting**, where no labeled training data is available. To address this, the authors introduce a **self-supervised distributional optimization method** that clusters the contextual representations of entity mentions and aligns them with type representations derived from the label space. This strategy allows the PLM to capture the semantics of fine-grained types purely from the unlabeled input distribution, enabling meaningful predictions without supervision.

2.5.2 OpenPrompt: An Open-source Framework for Prompt-learning [2]

- The paper introduces **OpenPrompt**, a unified and extensible toolkit designed to standardise and facilitate the development of prompt-learning methods with pretrained

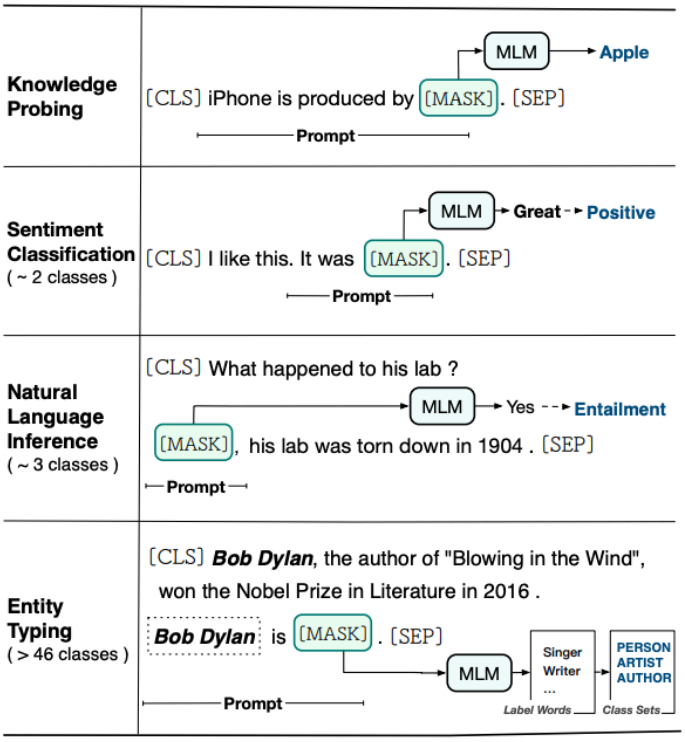


Figure 2.6: MLM based learning tasks in Prompt Learning

language models (PLMS).

- OpenPrompt provides a **research-friendly, modular, and extensible framework** that supports a broad spectrum of prompt-learning paradigms, including cloze-style prediction, autoregressive modeling, and sequence-to-sequence generation. The toolkit abstracts key components in prompt-learning—such as *template construction*, *verbalizer design*, and *PLM configuration*—allowing users to mix and match modules to suit different tasks and models.

- A contribution of OpenPrompt is its support for multiple **task formats and PLMs** in a unified interface. Users can rapidly deploy prompting frameworks with various combinations of language models (e.g., BERT, GPT, T5), task types (e.g., classification, generation, token-level prediction), and prompting strategies.

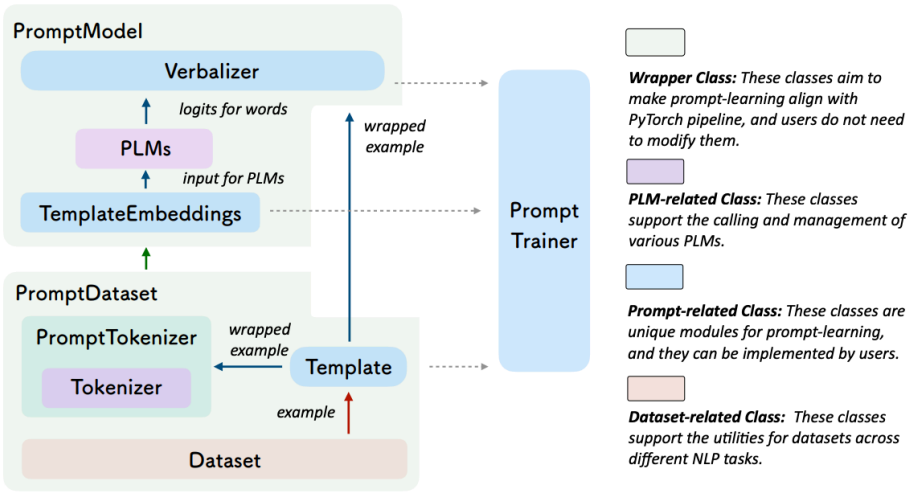


Figure 2.7: Architecture of OpenPrompt

2.6 Prompt Engineering Techniques

2.6.1 Automatic Prompt Engineering [17]

- This work introduces a technique known as Automatic Prompt Engineer (APE), which is aimed at generating and optimizing task instructions autonomously. The task of creating instructions is framed as a language generation challenge and is approached as a black-box optimization problem. Large language models (LLMs) are employed to explore and assess a space of potential instruction candidates.

- In the initial phase, a large language model is supplied with output examples and tasked with producing candidate instructions for a given task. These generated instructions serve as starting points for an optimization process. A separate model executes each instruction, and the most effective one is selected based on quantitative performance evaluations.

- This contribution has inspired further advancements in automated prompt generation strategies, such as **AutoPrompt** [18], **Prompt Tuning** [19], and **Prompt-OIRL** [20].

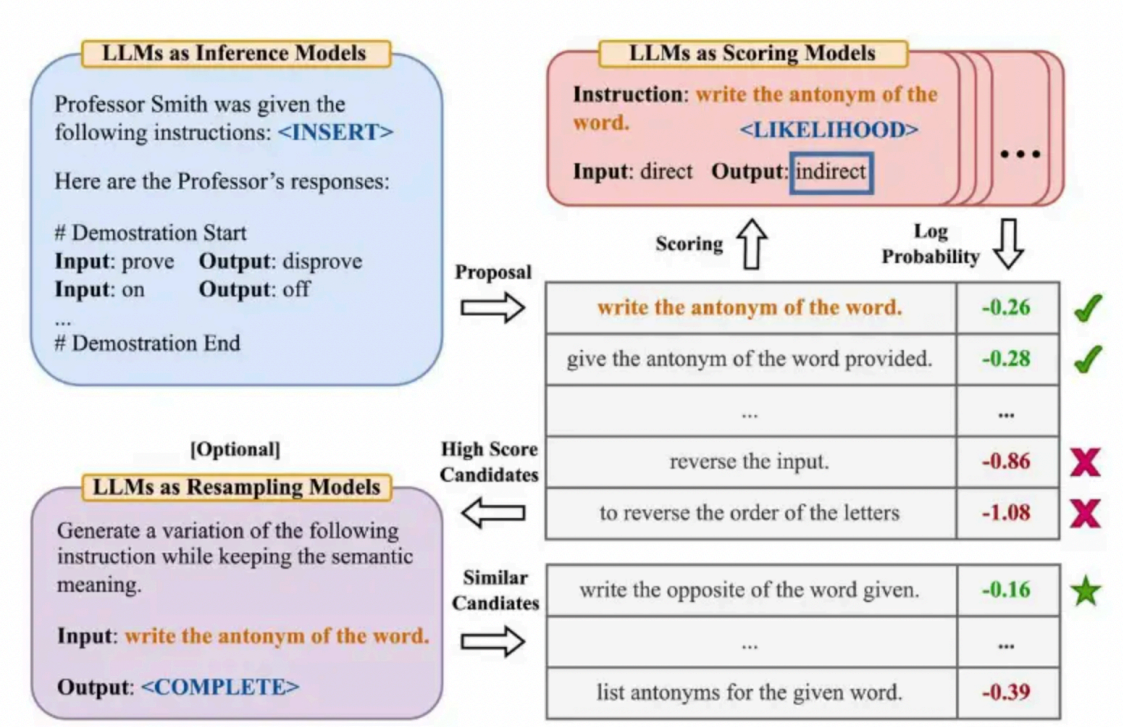


Figure 2.8: A depiction of how APE works. A LLM is prompted to generate a prompt based on sample inputs/outputs. A different LLM then scores the generated prompts on test input/outputs, followed by selecting high-scoring prompts. A third prompt can be optionally used to resample the generated prompts.

2.6.2 Chain of Thought Prompting [21]

- Chain-of-thought (CoT) prompting enhances a model's ability to handle tasks requiring multi-step reasoning by encouraging intermediate thought processes. When integrated with few-shot examples, it often leads to improved performance on tasks that demand logical progression before an answer is given.
- A more recent innovation in this area is known as **Zero-shot CoT** [22], which involves appending phrases like "Let's think step by step" to prompts in order to elicit structured reasoning from the model without requiring any examples.
- The authors report that, in numerous complex scenarios, Zero-shot CoT can outperform traditional few-shot prompting approaches.

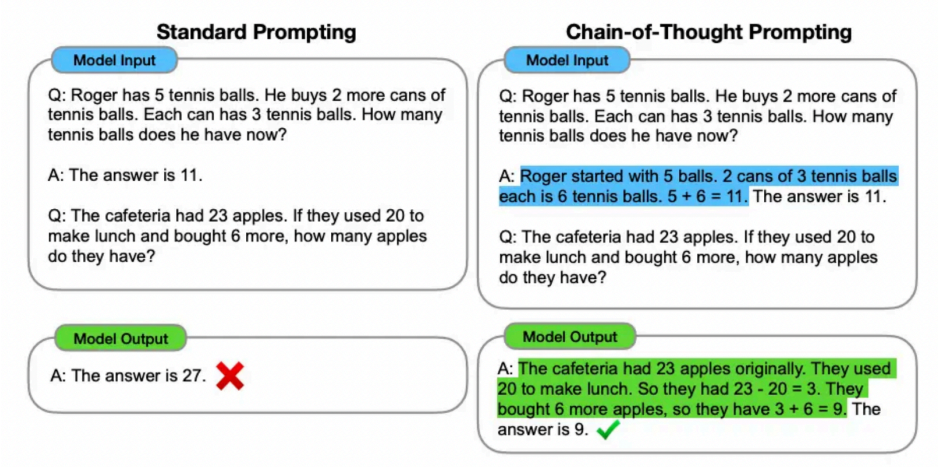


Figure 2.9: A contrastive difference on the working of standard prompting and CoT prompting. Note how in standard prompting, we directly get the incorrect answer, but in CoT, thinking through intermediate help the LLM to solve the task

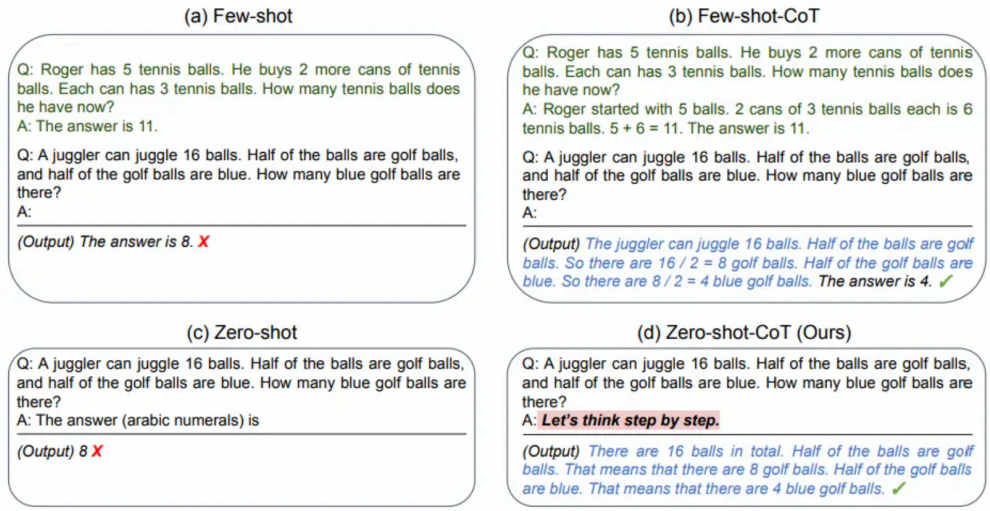


Figure 2.10: This figure shows the Zero-CoT technique where just by adding the "Let's think this step-by-step" clause, we get performance better than few-shot prompting

2.6.3 Tree of Thought prompting

We will discuss two main kinds of work that have been done in this domain. The first one uses some kind of algorithm to prune the nodes in the thought tree, whereas the second uses ingeniously designed prompts to mimic the ToT technique.

- The Tree of Thoughts (ToT) methodology organizes reasoning steps as a branching structure, where each node captures a logical segment contributing to the final solution. This setup allows the language model to assess its own progress incrementally and engage in structured, reflective problem-solving.
- As this area of research is still in its infancy, only two primary contributions have emerged. One was introduced by **Yao et al.** [23], and the other by **Long** [24].
- Both techniques enhance large language models' problem-solving abilities by enabling them to navigate through a reasoning tree using iterative exchanges. A key distinction lies in the search mechanism: *Yao's* method utilizes search strategies like depth-first, breadth-first, and beam search, whereas *Long* employs a learned "ToT Controller" using reinforcement learning to make decisions on traversal depth and when to revisit earlier reasoning branches.
- A separate approach by Hulbert simplifies the Tree of Thoughts concept by compressing the evaluation of reasoning steps into a single prompt, thus retaining the core idea while minimizing procedural complexity.

2.6.4 Graph of Thought Prompting [25]

- The **Graph of Thoughts (GoT)** framework introduces a non-linear reasoning paradigm where thoughts (intermediate reasoning steps) are structured as nodes in a graph. These thoughts are generated, evaluated, and composed asynchronously by LLMs, enabling richer and more flexible problem-solving pipelines.

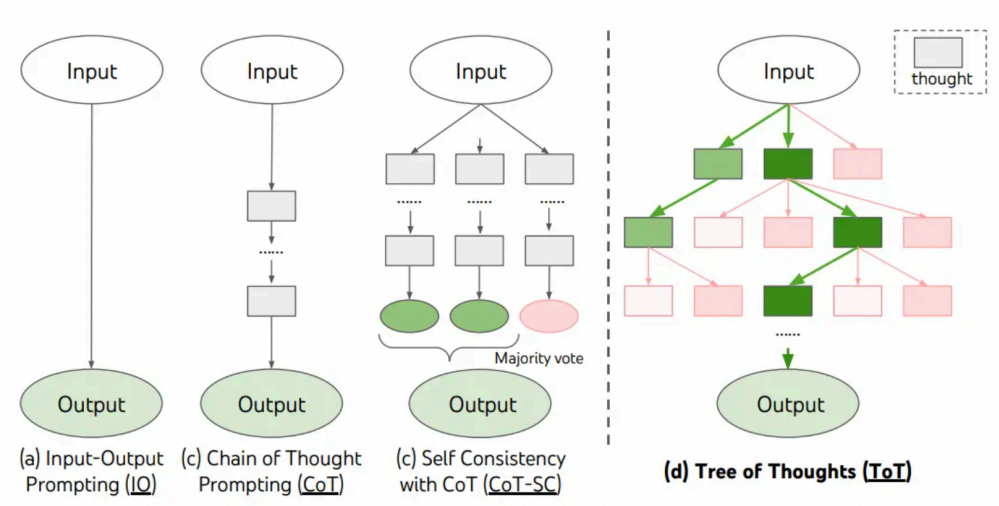


Figure 2.11: A visual depiction of the difference between standard prompting, CoT, ToT

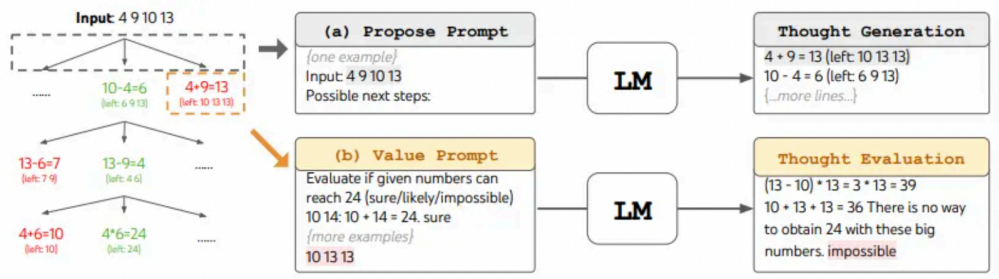


Figure 2.12: An example to show how ToT works. The example is the famous *Game of 24* where you are given 3 numbers, and you have to insert mathematical operations between these 3 to make the number 24. The upper path depicts the thought generation,n while the lower depicts thought evaluation.

```
Imagine three different experts are answering this question.
All experts will write down 1 step of their thinking,
then share it with the group.
Then all experts will go on to the next step, etc.
If any expert realises they're wrong at any point then they leave.
The question is...
```

Figure 2.13: An example prompt to depict the idea of Hulbert's Tree of Thought prompting

- The GoT prompting pipeline operates in three stages:

    1. **Thought Generation**: Multiple reasoning steps are generated in parallel as possible solution fragments or perspectives.
    2. **Thought Evaluation**: LLMs evaluate each generated thought using a quality criterion (e.g., factuality, coherence) to select the most promising ideas.
    3. **Thought Composition**: High-quality thoughts are merged or extended to form new nodes in the graph, progressing toward a final answer.

- Unlike chain-of-thought prompting, GoT allows parallel exploration and revisiting of earlier ideas, akin to how humans explore multiple approaches before converging on a solution. This makes GoT particularly suited for tasks involving synthesis, abstraction, and multi-step reasoning.

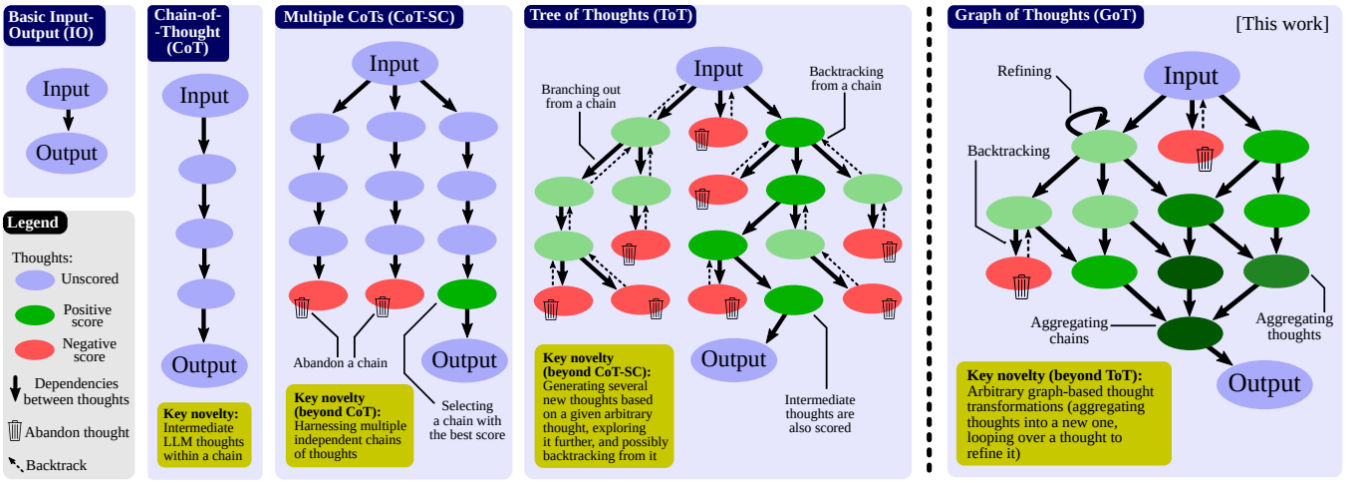


Figure 2.14: Comparing GoT along with other prompting techniques

Chapter 3

Problem Statement and Methodology

3.1 Problem

We will be exploring two kinds of problem in our thesis namely: *Translation with Annotation* **(TrA)** and *Raw to Fine* **(R2F)**. These will be explained subsequently.

3.1.1 Raw to Fine (R2F)

Input

Let $\mathcal{C}$ denote the set of coarse-grained NER labels. The input is any sentence given as tokens $x = [x_1, x_2, x_3, ..., x_t]$. Optionally, the sentence can be given a set of coarse labels $c_i \in \mathcal{C}$ in the form of a list as $c = [c_1, c_2, ..., c_t]$ where each c_i denotes the coarse-grained label of the input token, to aid the task.

['एक', 'टाइल', 'की', 'छत', 'के', 'साथ', 'सफेदी', 'वाली', 'ईंट', 'में', 'पनचक्की', '।']

Figure 3.1: Example Input of the R2F problem

Output

Let $\mathcal{F}$ denote the set of fine-grained NER labels. The output will be a list of fine-grained NER labels in $f = [f_1, f_2, f_3, ..., f_t]$. Each $f_i \in \mathcal{F}$ is the fine label assigned to the token x_i in the input, which matches the category of labels in which x_i falls given the list of fine labels. If x_i does not fall into any category, O is assigned to x_i.

['O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'Facility', 'O']

Figure 3.2: Example Output of the R2F problem

3.1.2 Translation with Annotation (TrA)

This problem deals with translating a sentence from a given source language (whose labeled data is abundantly available) to a target language (a low-resourced language) while maintaining a proper propagation of fine labels.

Input

Again let $\mathcal{F}$ denote the list of fine-grained NER labels and let $\mathcal{S}$ be the set of all possible tokens in the source language. Let the input sentence be given as tokens $x = [x_1, x_2, x_3, \ldots, x_n]$ in the source language i.e. $\forall\ 1 \leqslant i \leqslant n\ , x_i \in \mathcal{S}$. Also let $f = [f_1, f_2, f_3, \ldots, f_n]$ be the list of fine labels such that $\forall\ 1 \leqslant i \leqslant n\ , f_i \in \mathcal{F}$.

Output

Let $\mathcal{T}$ denote the set of all possible tokens in the target language. In output we expect another sentence in the target language i.e. we expect $x' = [x'_1, x'_2, x'_3, \ldots, x'_m]$ such that $\forall\ 1 \leqslant i \leqslant m\ , x'_i \in \mathcal{T}$ and with that we also want $f' = [f'_1, f'_2, f'_3, \ldots, f'_m]$ such that $\forall\ 1 \leqslant i \leqslant m\ , f_i \in \mathcal{F}$ which is the list of fine labels for the list of tokens that are outputted.

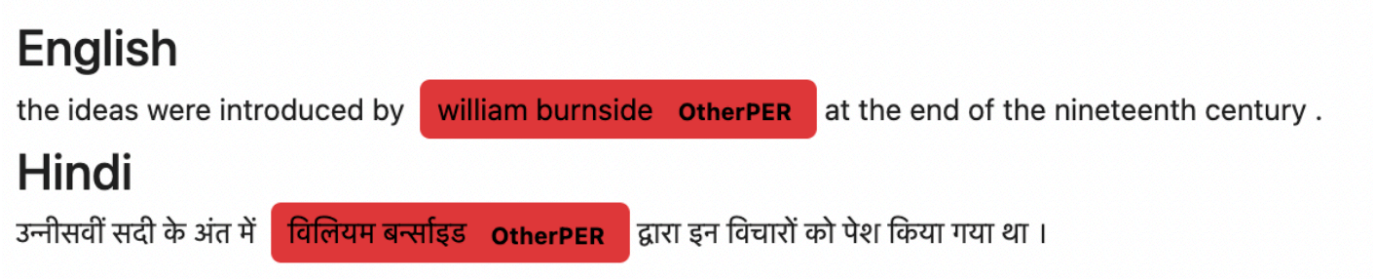


Figure 3.3: Example Input/Output of the TrA problem

3.2 Challenging nature of the Problems

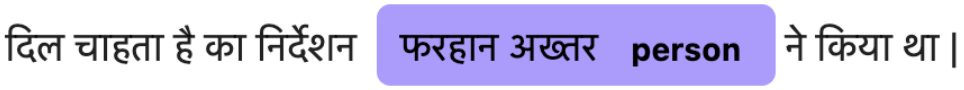
- **Importance of Context**: Fine-labelling of NER data can't be done without understanding the context of the sentence, which makes the problem of fine-labelling NER data more challenging in comparison to coarse-labelling NER data. To understand the importance of context, take the example of a person who is an Actor, Director, and Producer in cinema. While assigning a fine-grained label to the person, the context of the sentence in question has to be analysed to provide the label.
- **Low resources for Indic Languages**: Indic languages don't have a lot of training data to train a model accurately to cater to the needs of the language, so finding a coarse-labelled dataset for the above languages is a difficult task. Let alone a fine-labelled one.

3.3 Proposed Methodology

3.3.1 R2F: Raw to Fine

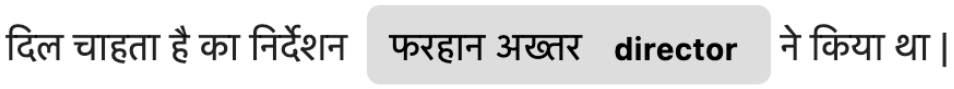
The methodology is composed of the below steps.

1. Prompt preparation
   (a) Example sentence selection



(a): Coarse-labelling of entity to *person*

(b): Fine-labelling of same entity to *actor*



(c): Fine-labelling of same entity to *director*

Figure 3.4: Comparing the complexity of coarse-labelling vs. fine-labelling

(b) Feeding the unlabelled sentence to the prompt

2. Feeding the unlabelled sentence through the prompt to the long context large language model for labelling.

3. Creating a training dataset from the labels generated by the long context large language model

4. Fine tuning a **Pre-trained language model** (XLM-RoBERTa [33] in this case) on the training dataset to make the final model.

To validate this model, we compare the F1-score of the final model obtained from above with the model trained on the original train MCN2 dataset, tested on the MCN2 test set as shown in here

Table 3.1: Comparison of Training Dataset Sizes for Indic Languages and English in Coarse and Fine-grained NER Tasks

Language	Coarse-grained Dataset	Tokens	Fine-grained Dataset	Tokens
English	CoNLL-2003 [26]	300K	OntoNotes 5.0 [27]	1.5M
			MultiCoNER v1 [28]	2.2M
			MultiCoNER v2 [29]	2.3M
Hindi	AI4Bharat NER [30]	350K	WikiNER [31]	43K
			MultiCoNER v1 [28]	200K
			MultiCoNER v2 [29]	210K
Tamil	AI4Bharat NER [30]	310K	WikiNER [31]	40K
Bengali	L3Cube-Bengali [32]	180K	WikiNER [31]	36K
Marathi	L3Cube-Marathi [32]	150K	Not Available	–
Malayalam	AI4Bharat NER [30]	200K	Not Available	–
Kannada	AI4Bharat NER [30]	220K	Not Available	–

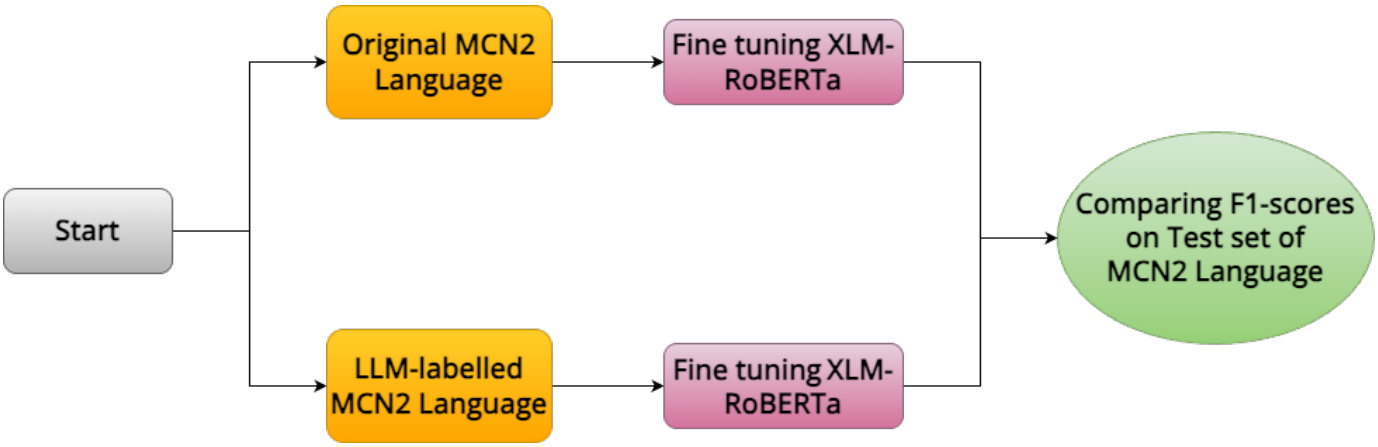


Figure 3.5: R2F: Comparing F1 methodology

Prompt Preparation

The prompt preparation consisted of the following parts:

- **Instruction about the task**
- **Giving examples for in-context learning**
- **Listing out the sentences to be labelled**

Each prompt follows this general rubric, and the prompts differ in the way the instruction is provided, and the number of sentences that are given to be labelled at a time.

```
I want to initialise this prompt for fine labelling of sentences task. The task would be as follows, you will be given a sentence as a list of tokens where each token is a word.

You just need to output the list of fine labels (size of this list would be same as that of token list) based on the data given. I am also going to explain the hierarchy among these coarse and fine labels two below for your help:

Location (LOC) : Facility, OtherLOC, HumanSettlement, Station
Creative Work (CW) : VisualWork, MusicalWork, WrittenWork, ArtWork, Software
Group (GRP) : MusicalGRP, PublicCORP, PrivateCORP, AerospaceManufacturerSportsGRP, CarManufacturer, ORG
Person (PER) : Scientist, Artist, Athlete, Politician, Cleric, SportsManagerOtherPER
Product (PROD) : Clothing, Vehicle, Food, Drink, OtherPROD
Medical (MED) : Medication/Vaccine, MedicalProcedure, AnatomicalStructureSymptom, Disease

The fine label can be one among the above or it will be 'O' if the token doesn't belong to any of the above categories.

Just return the labels for the tokens in the sentence. Label only the proper nouns.
DO NOT OUTPUT ANYTHING CONVERSATION OR CONFIRMATION, JUST OUTPUT THE LISTS.
DON'T EVEN OUTPUT ``` AT THE START AND END OF THE LIST.

Examples:
Sentence: ['जीभ', 'अधिकांश', 'कशेरुक', 'के', 'मुंह', 'के', 'फर्श', 'पर', 'एक', 'मांसपेशियों', 'के', 'हाइड्रोस्टेट']
Fine Labels: ['AnatomicalStructure', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']
```

(a): Instructing LLM about the task

```
Unlabelled Sentences:
Sentence: ['निराश', 'है', 'कि', 'उसका', 'बेटा', 'अपना', 'सारा', 'समय', 'नाइट', 'क्लब', 'में', 'बिताता', 'है', 'उसके', 'अमीर', 'पिता', 'एक', 'गिरोह', 'के', 'लिए', 'उसे', 'सबक', 'सिखाने', 'के', 'लिए', 'अपहरण', 'करने', 'की', 'व्यवस्था', 'करते', 'हैं।']
Sentence: ['उन्हें', 'कोलम्बस,', 'ओहायो', 'में', 'दफनाया', 'गया', 'था।']
Sentence: ['सेवानिवृत्ति', 'के', 'बाद', 'वह', 'अभ्यास', 'करने', 'के', 'लिए', 'प्रयागराज', 'पर', 'स्थानांतरित', 'हो', 'गया।']
Sentence: ['यह', 'बेलारूस', 'के', 'साथ', 'सीमा', 'के', 'पास', 'है।']
Sentence: ['कला', 'के', 'टिश', 'स्कूल', 'ने', 'एक', 'फिल्म', 'निर्माता', 'के', 'रूप', 'में', 'उनकी', 'रुचि', 'दिखाई', 'लेकिन', 'उन्होंने', 'इसके', 'बजाय', 'स्टैंड', 'अप', 'कॉमेडी', 'में', 'अपना', 'करियर', 'बनाने', 'का', 'फैसला', 'किया।']
Sentence: ['वहाँ', 'उसे', 'स्मारक', 'के', 'एक', 'टोकरे', 'के', 'साथ', 'प्रस्तुत', 'किया', 'गया', 'था', 'जिसमें', 'उसने', 'टिप्पणी', 'की', 'थी', 'कि', 'उसके', 'पति', 'राजकुमार', 'फिलिप', 'आनंद', 'लेंगे।']
Sentence: ['१८६३', 'से', '१८६५', 'तक', 'उन्होंने', 'मिस्र', 'उस्मानी', 'साम्राज्य', 'का', 'हिस्सा', ')', 'की', 'यात्रा', 'की।']
Sentence: ['इनका', 'राष्ट्रवादियों', 'ने', 'फ़्रांस', 'और', 'स्पेन', 'दोनों', 'से', 'स्वतंत्रता', 'के', 'बारे', 'में', 'अपनी', 'मांगों', 'को', 'आगे', 'बढ़ाने', 'के', 'लिए', 'एक', 'मंच', 'दिया।']
```

(b): Unlabelled sentences given to the LLM for labelling

Figure 3.6: An example prompt for R2F

Figure 3.7: R2F visualized

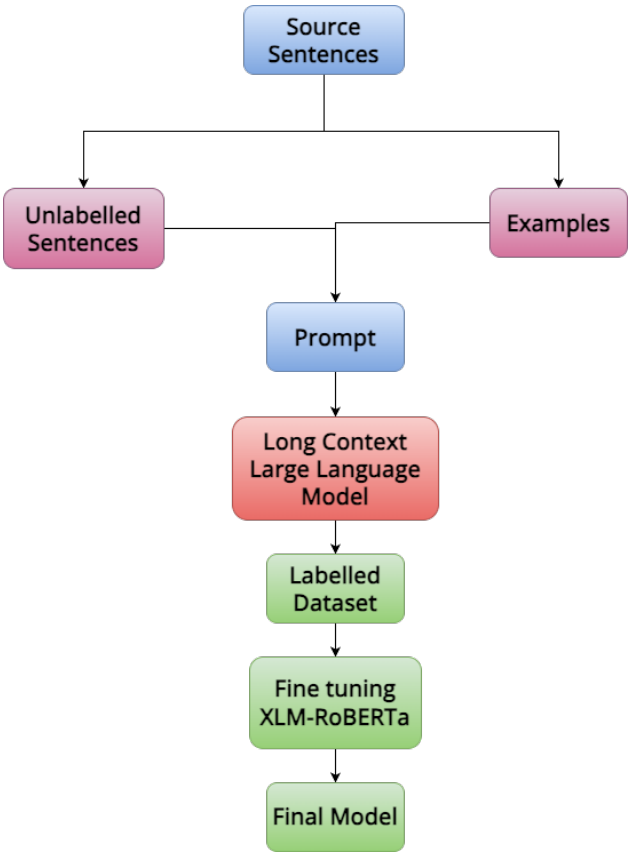


Figure 3.8: R2F Pipeline visualized

3.3.2 TrA: Translation with annotation

The methodology consists of the following steps in order:

1. **Prompt generation** (Refer to section 3.3.2).
2. Finalize the query using generated prompt and the query sentence.
3. Use the large context LLM to generate the output for the given query.
4. Cleaning of the generated output and selecting good sentences.

5. Fine tuning a **Pre-trained language model** (XLM-RoBERTa [33] in this case) on the training dataset to make the final model.

6. Use this model for testing on the original MultiCoNER-2 test data split and comparing the f1's

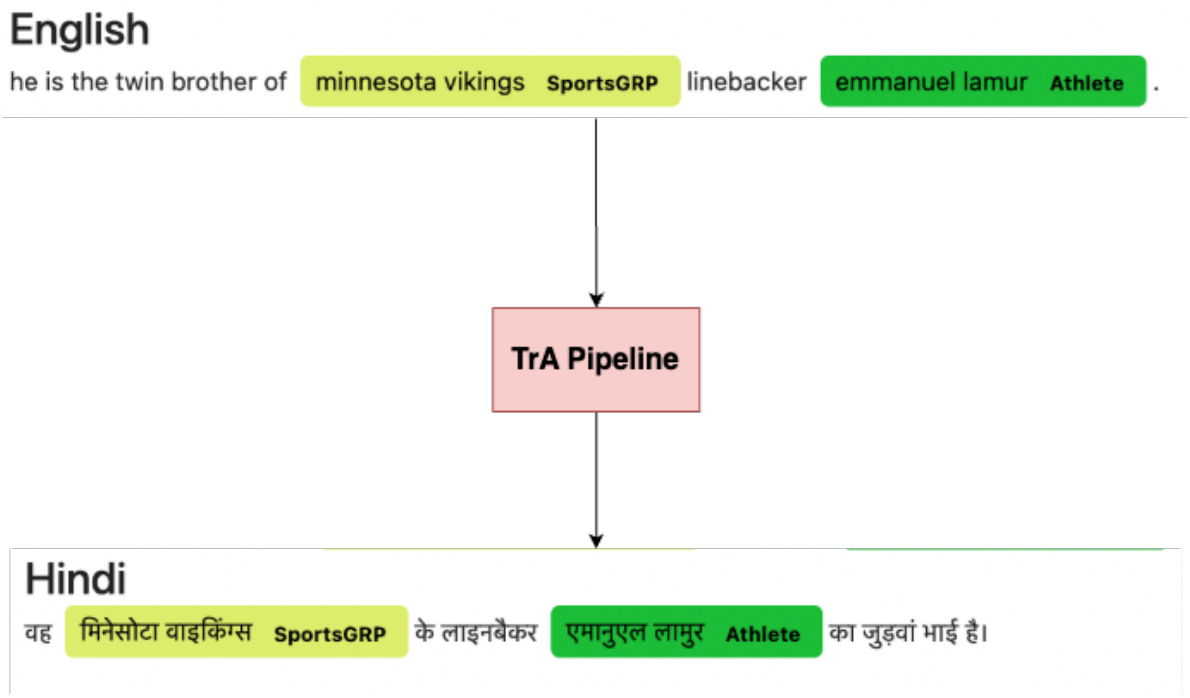


Figure 3.9: An example of what TrA is doing

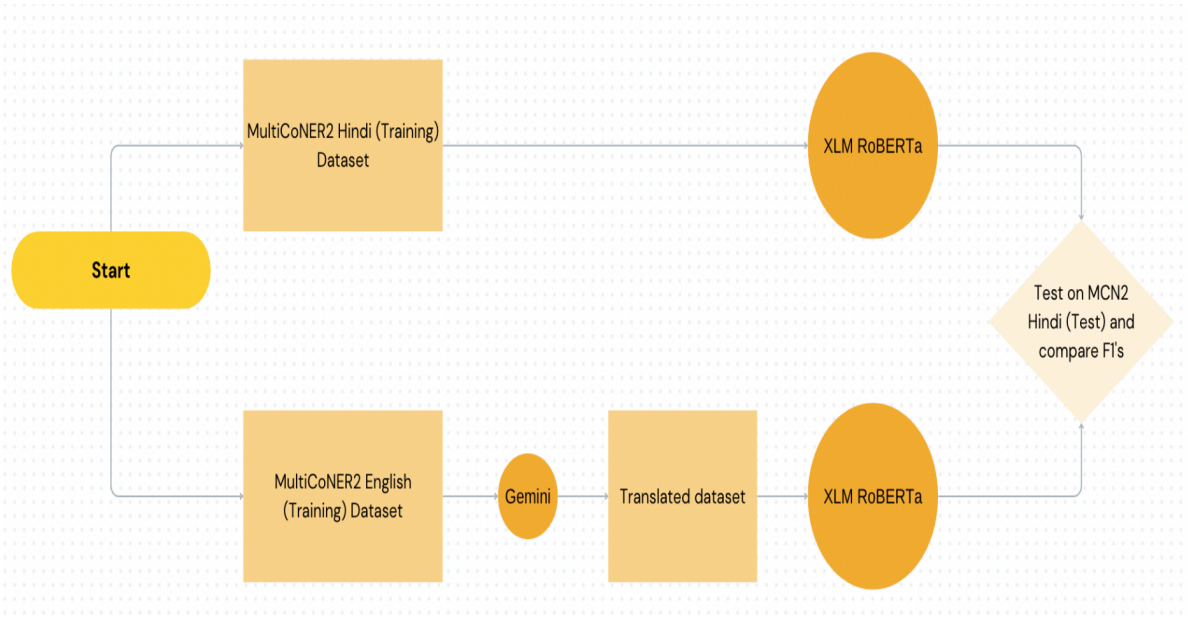


Figure 3.10: TrA pipeline visualised

Prompt generation

Prompt design have a significant impact on the level of generated output. This was readily seen in the results as well (Refer to section **??**). Here we will just be giving a template of a vanilla prompt which follows the prompt guidelines given here

A vanilla prompt template is shown at figure 3.11

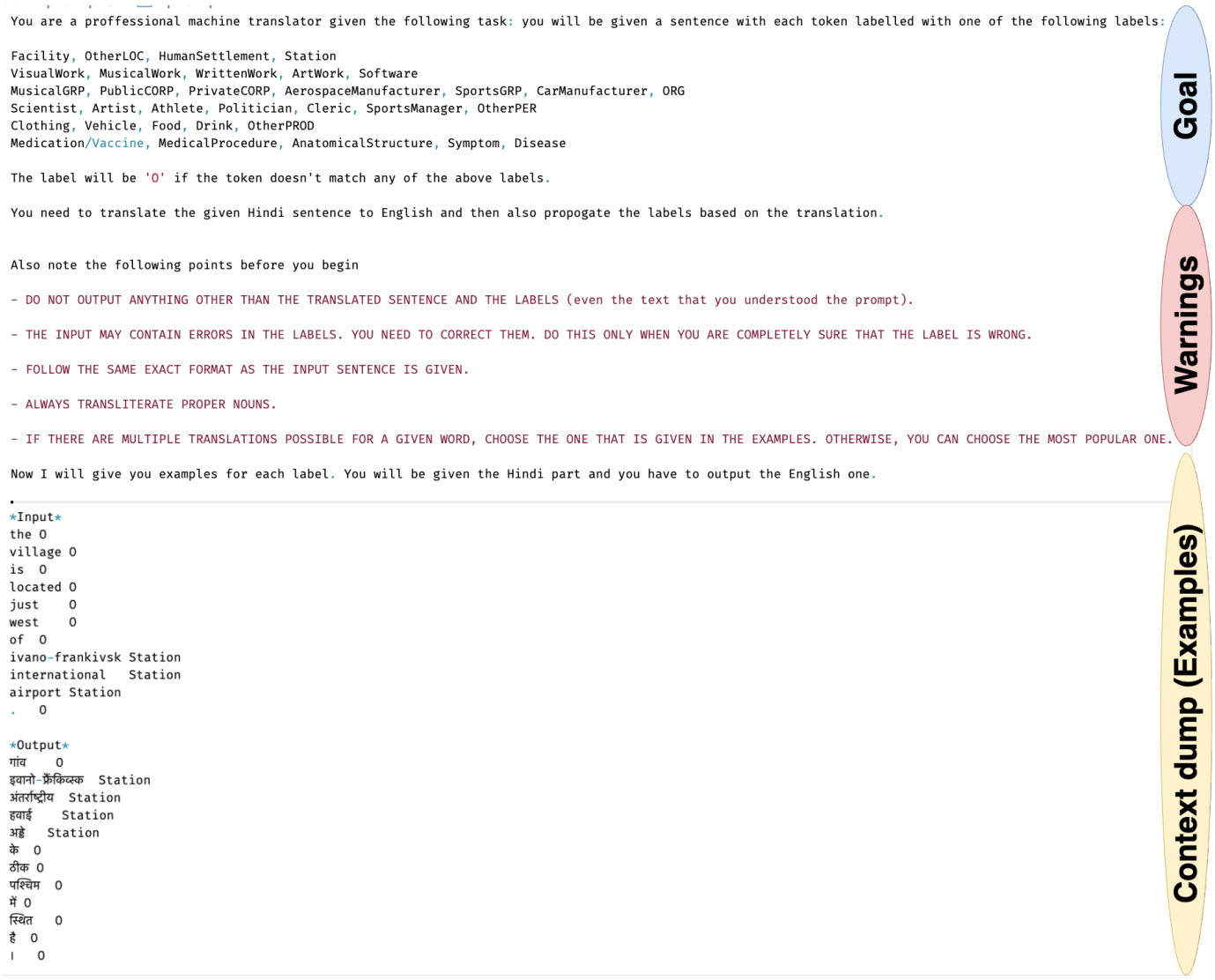


Figure 3.11: An example of prompt for TrA

3.3.3 Use of Gemini for Long-Context Modeling

To effectively model long-context dependencies in bilingual fine-grained Named Entity Recognition (NER), we utilized **Gemini**, a state-of-the-art long-context language model developed by Google DeepMind. Gemini was chosen due to its ability to handle extended sequences of text while preserving contextual coherence, which is critical for hierarchical and semantically nuanced labeling tasks. Its long-context capabilities arise from several key architectural features:

- **Segment-aware attention:** Gemini uses attention mechanisms that explicitly account for token positions across long documents by segmenting the input and conditioning on segment-level memory representations.
- **Memory-efficient attention variants:** Techniques such as sparse attention and local-global token mixing allow Gemini to scale to longer sequences without quadratic memory overhead.
- **Chunked context representation:** The model processes large inputs in context chunks and fuses their representations, enabling it to maintain semantic consistency over long ranges.
- **Retrieval-augmented generation (RAG):** Gemini can optionally retrieve relevant content from external memory or documents to support inference, especially useful when key information lies far apart in text.
- **Position interpolation and extended rotary embeddings:** These allow the model to generalize positional understanding beyond the original training context window.

Furthermore, Gemini offers robust multilingual understanding, which aligns well with the bilingual nature of our dataset comprising aligned English-Hindi sentences.

Chapter 4

Experiments & Results

Prior to detailing the experimental procedures, we first define the computational environment in which all experiments were conducted.

All experiments employ **Gemini**, a long-context large language model developed by Google DeepMind. This model was selected due to its favorable cost-performance ratio and the demonstrated efficacy reported in recent benchmarks. Experiments were executed via the publicly available (free-tier) API, which imposes rate limitations. Consequently, the time required for dataset annotation—impacted by these rate limits—is a non-negligible factor and is explicitly reported in the context of each experiment.

Secondly, all labeled and unlabeled datasets are derived from the **MultiCoNER 2** dataset collection (for details, see Section 2.1.2). The unlabeled datasets were created by removing entity labels systematically from corresponding labeled instances. In addition, to maintain the integrity of the evaluation, no test data was ever seen by the Gemini model during any phase of the process, thus ensuring that the test set remained strictly unseen during training and inference phases.

Also as for the Pre-Trained Model for testing the generated dataset we used **XLM-RoBERTa (Large)**[33] for our purposes.

For the Translation with Annotation (TrA) problem, the source language is English while

the target language can be *Hindi* or *Bangla*. Both of them are supported by Gemini and their datasets are available in MultiCoNER-2.

For the R2F Problem, the unlabelled sentences are either in *Hindi* or *Bangla*, and selected from the *train* set of the MultiCONER-II dataset.

4.1 Raw to Fine (R2F)

For the experiments, the example sentences that were provided to aid the LLM for in-context learning weren't changed throughout the experiments. These examples are listed below.

4.1.1 Experiment-0: Qualitative Testing of the LLM

LLM Used: Gemini 1.5 Flash

We started by qualitative analysis of the outputs by the LLM to see whether the LLM can be used to fine-label named entities in Indic languages.

In this experiment, there is one instruction prompt fig:r2f-exp2-instruction-prompt, coupled with examples for in-context learning of the LLM and **10** unlabelled sentences for labelling.

- **Sub experiment (A): Biased Prompting Setting**: In this sub-experiment, the examples and the unlabelled sentences were chosen so that they shared the same entity type in the training dataset of MCN2. Implies **33 sentences** of one entity type given as examples, and **10 sentences** given for labelling, which were labelled as the same entity type as the 33 example sentences in the training dataset.

- **Sub experiment (B): Zero-Shot Prompting Setting**: In this sub-experiment, the examples and the unlabelled sentences were chosen so that they did not share the entity type in the training dataset of MCN2. Implies **33 sentences** of **32** entity types given as examples, and **10 sentences** given for labelling, which were labelled as the entity type not in the 33 example sentences in the training dataset.

- **Sub experiment (C): Randomly generated Prompts Setting**: In this sub-experiment, the examples and the unlabelled sentences were chosen randomly from the training dataset of MCN2.

Results

The qualitative analysis yielded positive results, as many sentences were correctly labelled in the first and last sub-experiments. Regarding correctly assigning labels, sub-experiment A produced the best results, followed by sub-experiment C, and then sub-experiment B.

Sub-experiment B was particularly interesting, as the LLM had no examples to learn from in the prompt, and hence assigned many of the named entities the label 'O', instead of their true fine-label, **clearly showing the importance of examples given in the prompt**.

Since the initial results were promising, further experiments were done to study the labelling capabilities of the LLM.

Findings

- **One-off errors**: The LLM labels the surrounding corpus of the named entity many times, leading to one-off errors.
- **Not using the context**: The LLM many times, doesn't use the context of the sentence, but rather it's own pre-trained knowledge to fine-label an entity.

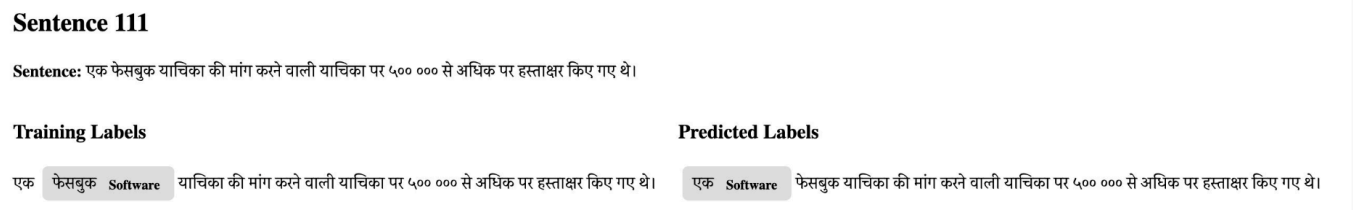


Figure 4.1: R2F: Experiment-0 One Off Errors: Instead of labelling फेसबुक , the LLM labels the token just before it as Software

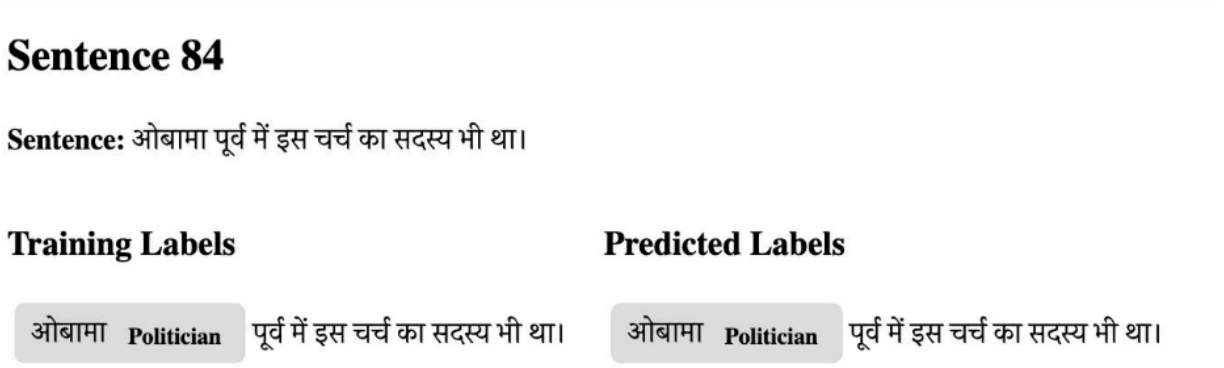


Figure 4.2: R2F: Experiment-0 Not using context: From the sentence it isn't clear that ओबामा is a Politician, but the LLM uses it's pre-trained knowledge to label that.

4.1.2 Experiment-1: Finding the acceptable number of sentences that can be labelled in one prompt

LLM Used: Gemini 1.5 Flash

In the second experiment, instead of a qualitative analysis of the LLM output, we tried to find out the practicality of the solution, and started with finding the number of sentences that can be labelled in one prompt without the LLM going into hallucination.

Findings

- **Hallucination**: The LLM doesn't always follow the prompt as asked to. For example in the prompt it is clearly mentioned not to output "', still the LLM outputs it.
- **Not using the context**: The LLM many times, doesn't use the context of the sentence, but rather it's own pre-trained knowledge to fine-label an entity. For example, the LLM labelled a coach as sportsperson, overlooking the context of the sentence.
- **Thirty unlabelled sentences**:
  - **Setting**: In this case, in the prompt **33** examples were given, each example covering one fine-grained label of the MultiCONER-II (MCN2) dataset, and along with these examples, **30** sentences were given to be labelled.

- **Result**: The LLM clearly hallucinated, giving **hardcoded labelling functions** for **every unlabelled sentence given**.

- **Fifteen unlabelled sentences**:
  - **Setting**: In this case, in the prompt **33** examples were given, each example covering one fine-grained label of the MultiCONER-II (MCN2) dataset, and along with these examples, **15** sentences were given to be labelled.
  - **Result**: The LLM clearly hallucinated, giving **hardcoded labelling functions** for **most of the unlabelled sentence given**, this is a slight reduction from the prompt, where **30** examples were given.

- **Ten unlabelled sentences**:
  - **Setting**: In this case, in the prompt **33** examples were given, each example covering one fine-grained label of the MultiCONER-II (MCN2) dataset, and along with these examples, **10** sentences were given to be labelled.
  - **Result**: The LLM **did not hallucinate for half of the sentences**, and gave the output in the desired form as required for analysis. So, around **4,600** sentences were labelled according to the specification provided in the prompt as compared to the **9,200** sentences in total. The resulting **F1-score** when trained and tested came out to be **38.54**, note that the baseline score of the MCN2 original train dataset is **68.05**.

```
I want to initialise this prompt for fine labelling of sentences task. The task would be as follows, you will be given a sentence as a list of tokens where each token is a word.

You just need to output the list of fine labels (size of this list would be same as that of token list) based on the data given. I am also going to explain the hierarchy among these coarse and fine labels two below for your help:

Location (LOC) : Facility, OtherLOC, HumanSettlement, Station
Creative Work (CW) : VisualWork, MusicalWork, WrittenWork, ArtWork, Software
Group (GRP) : MusicalGRP, PublicCORP, PrivateCORP, AerospaceManufacturerSportsGRP, CarManufacturer, ORG
Person (PER) : Scientist, Artist, Athlete, Politician, Cleric, SportsManagerOtherPER
Product (PROD) : Clothing, Vehicle, Food, Drink, OtherPROD
Medical (MED) : Medication/Vaccine, MedicalProcedure, AnatomicalStructureSymptom, Disease

The fine label can be one among the above or it will be 'O' if the token doesn't belong to any of the above categories.

Just return the labels for the tokens in the sentence. Label only the proper nouns.
DO NOT OUTPUT ANYTHING CONVERSATION OR CONFIRMATION, JUST OUTPUT THE LISTS.
DON'T EVEN OUTPUT ``` AT THE START AND END OF THE LIST.
```

Figure 4.3: R2F: Experiment-1 Instruction Prompt

```
Sentence: सेवानिवृत्ति के बाद वह अभ्यास करने के लिए प्रयागराज पर स्थानांतरित हो गया।
Ground truth labels: ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'HumanSettlement', 'O', 'O', 'O', 'O']
Labels:     labels = ['O'] * len(sentence)
```

Figure 4.4: R2F: LLMs returning Hardcoded Labelling Functions

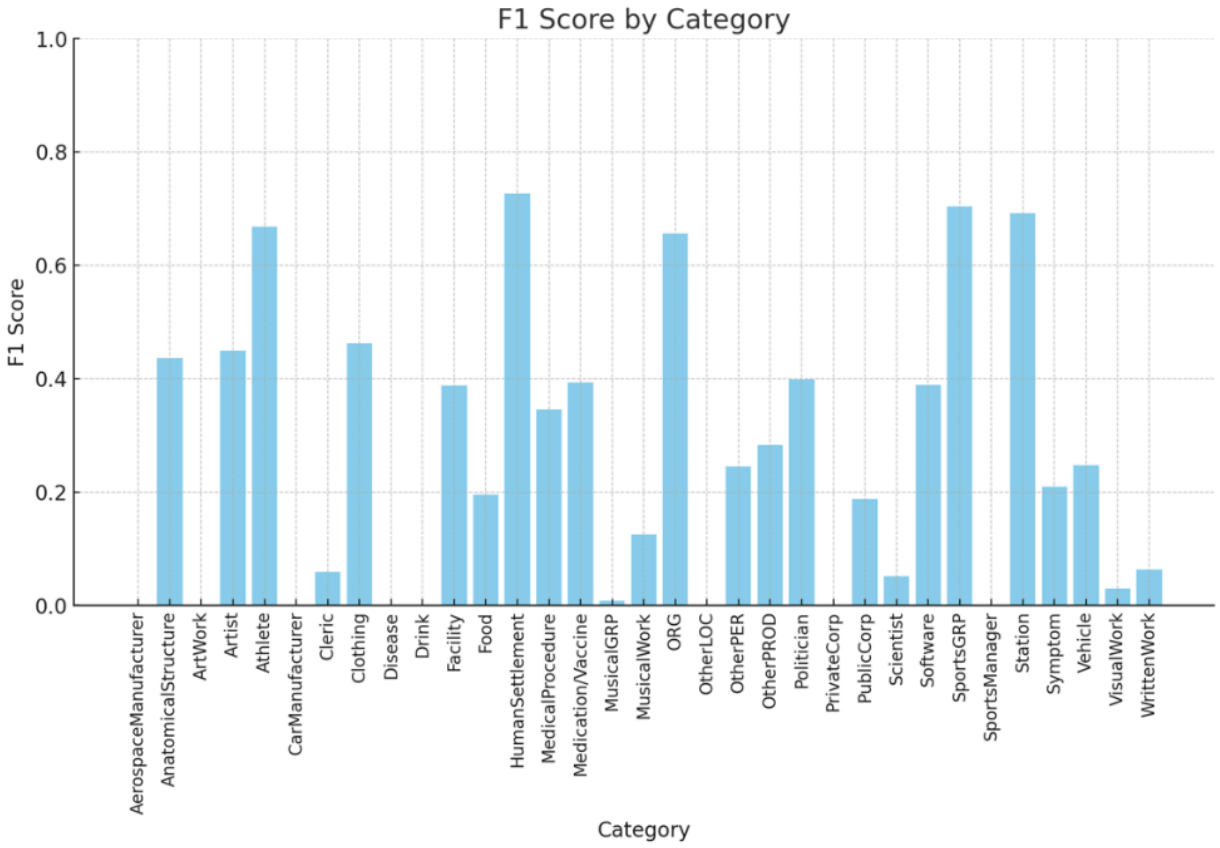


Figure 4.5: R2F: Experiment-1 F1 Score by category

4.1.3 Experiment-2: Fine-tuning LLM

LLM Used: Gemini 1.5 Flash

- **Setting**: We fine-tuned the above LLM to improve the results of experiment 1. **500 sentences** were randomly picked for fine-tuning the LLM. In this case, in the prompt **33** examples were given, each example covering one fine-grained label of the MultiCONER-II (MCN2) dataset, and along with these examples, **10** sentences were given to be labelled.

- **Results**: The LLM **did not hallucinate for over half of the sentences**, and gave the output in the desired form as required for analysis. So, around **5,000** sentences were labelled according to the specification provided in the prompt as compared to the

9,200 sentences in total. The resulting **F1-score** when trained and tested came out to be **43.62**, note that the baseline score of the MCN2 original train dataset is **68.05**.

Table 4.1: Machine Learning Training Parameters

Term	Definition	Value Used
Epochs	A full training pass over the entire training set such that each example has been processed once.	5
Batch size	The set of examples used in one training iteration. The batch size determines the number of examples in a batch.	16
Learning rate	A floating-point number that tells the algorithm how strongly to adjust the model parameters on each iteration.	0.0002
Learning rate multiplier	The rate multiplier modifies the model's original learning rate.	1.0

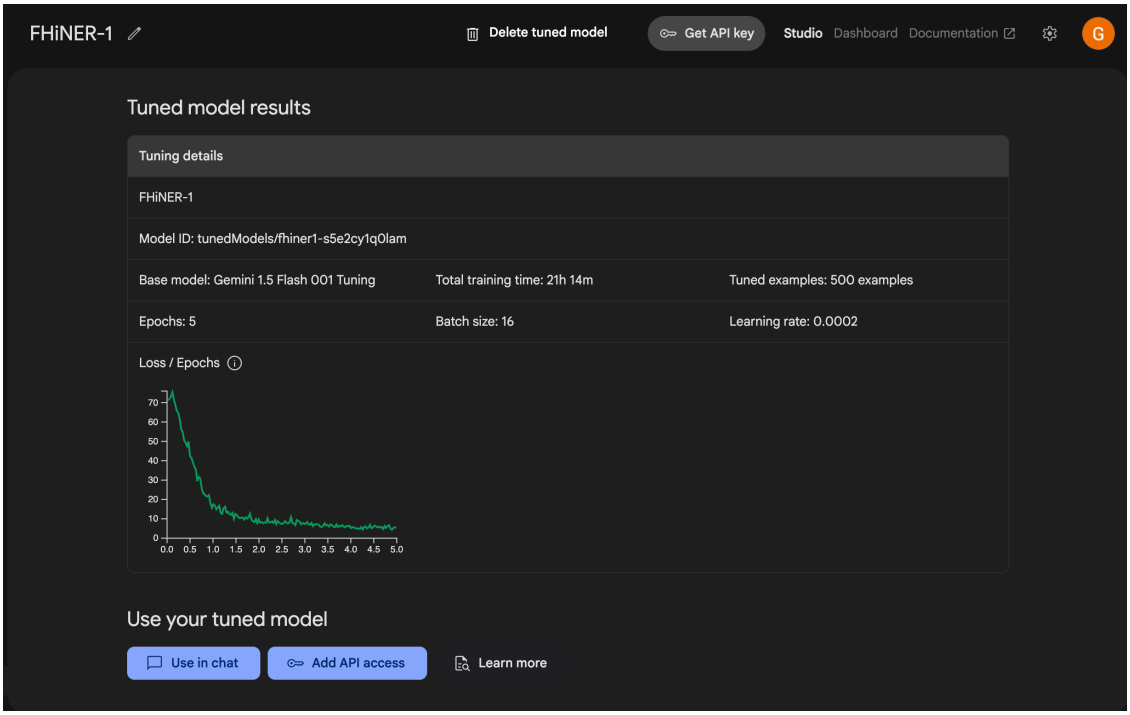


Figure 4.6: R2F: Experiment-2 Fine Tuning LLM

4.1.4 Experiment-3: Changing the Model

LLM Used: Gemini 2.0 Flash

- **Setting**: In this experiment, instead of Gemini-1.5 Flash, the **Gemini 2.0 Flash** model was used, with the prompt having **33** examples, each example covering one fine-grained label of the MultiCONER-II (MCN2) dataset, and along with these examples, **10** sentences were given to be labelled.
- **Result**: The LLM did not hallucinate for more half of the sentences, and gave the output in the desired form as required for analysis. The number of sentences at hand for training was more than that in experiment 2. The resulting F1-score when trained and tested came out to be **47.07**. Note that the baseline score of the MCN2 original train dataset is 68.05.

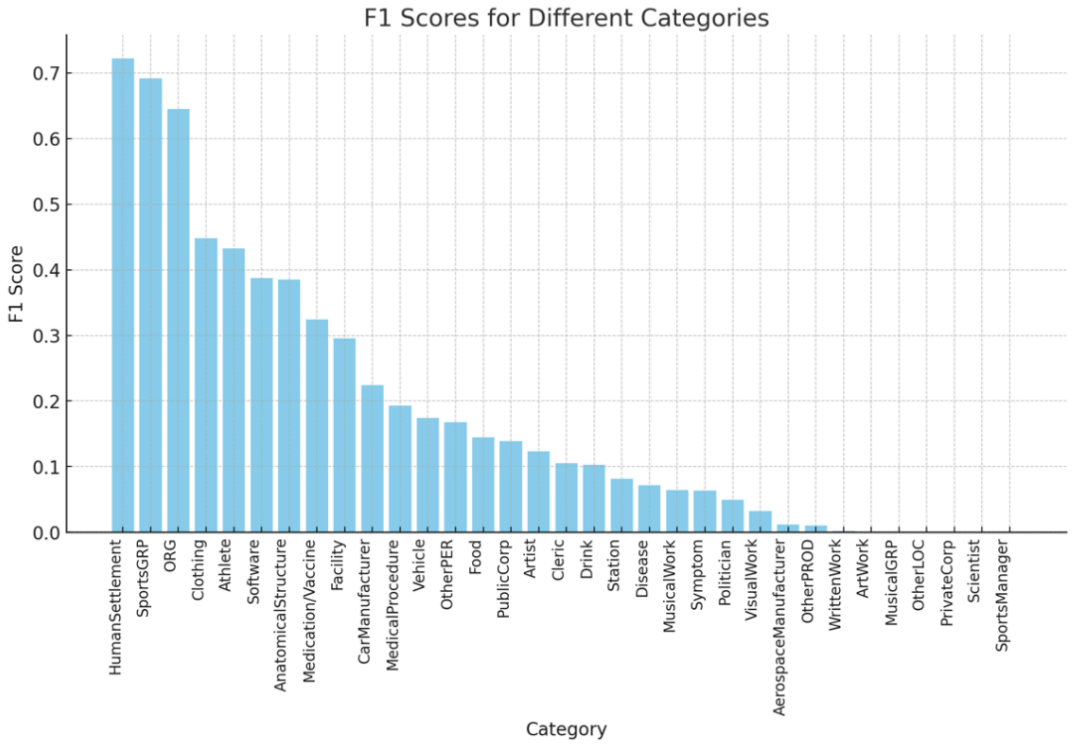


Figure 4.7: R2F: Experiment-3 F1 Score by category

4.1.5 Experiment-4: Better Instruction Prompts

LLM Used: Gemini 2.0 Flash

- **Setting**: In this experiment, the instruction prompt was changed to explain the LLM terms related to Natural Language Processing. It was also explained what each fine label meant. This experiment was done to find out the LLM's capability to learn from the prompt. Also the number of sentences to be labelled in one prompt was **reduced to one**.
- **Results**: Better prompt gave better results as the F1 score jumped to **63.71**, compared to the baseline of **68.05**.

```
### Fine-Grained Named Entity Labeling Prompt

You will be given a **sentence as a list of tokens**, where each token is a word. Your task is to return a list of **fine-grained en

The size of the output list must be **the same as the input token list**. If a token belongs to a specific entity category, assign i

### Definitions

#### Proper Noun
A **proper noun** is the specific name of a person, place, organization, brand, or unique entity. It is capitalized in languages tha

#### Common Noun
A **common noun** is a general term for people, places, objects, or concepts. It does not refer to a specific entity and is often us

---

### Label Hierarchy and Definitions

Each fine-grained label belongs to one of the following **coarse categories**:

#### Location (LOC)
- **Facility** → A physical structure built for a specific purpose, such as a building, airport, or stadium.
- **OtherLOC** → A geographical or physical feature that is not a human settlement, such as a river, mountain, or desert.
- **HumanSettlement** → A populated place where people live, such as a city, town, or village.
- **Station** → A transportation hub such as a bus station, train station, or metro stop.

#### Creative Work (CW)
- **VisualWork** → A creative work that is primarily visual in nature, such as a movie, TV show, or painting.
- **MusicalWork** → A musical composition, album, or song.
- **WrittenWork** → A book, article, poem, or other literary work.
- **ArtWork** → A piece of fine art, including sculptures and paintings.
- **Software** → A named software program, application, or operating system.

```

Figure 4.8: R2F: Experiment-4 Instruction Prompt

4.1.6 Experiment-5: LLM Voting Algorithm

The LLM Voting Algorithm leverages multiple prompts to the same LLM for the same sentence, collects the outputs, and uses majority voting to determine the final label for each

token. The key intuition is that consistent outputs across multiple generations are more likely to be reliable. The various steps in this method are described below.

1. **Prompting:** For a given input sentence, the LLM is queried three times using the same prompt template to obtain token-level labels.

2. **Label Collection:** Each of the three LLM responses is parsed to extract a sequence of token-label pairs.

3. **Voting Mechanism:** For each token:

    - The three predicted labels are collected.
    - A majority vote is taken. The label that appears most frequently among the three predictions is selected as the final label.
    - In case of a tie, i.e., three different labels, any one of the labels is selected at random.

4. **Output Generation:** The final labelled sequence, consisting of majority-voted labels, is produced and returned.

This method, which requires three times more API calls, and is slower compared to previous methods, has its own advantages outlined below.

- **Noise Reduction:** Smooths out random outputs and inconsistencies in LLM predictions.
- **Model-Agnostic:** Can be applied to any LLM without the need for fine-tuning.

Results: This method proved to be an improvement over the last experiment, and further pushed the F1-score to **65.19** compared to the baseline of **68.05**.

Token	LLM-1 Label	LLM-2 Label	LLM-3 Label	Final Label
स्टीवन	SportsManager	SportsManager	SportsPerson	SportsManager
मिनीस्कर्ट	Clothing	Clothing	Clothing	Clothing
ब्राउनिंग	Artist	Artist	Scientist	Artist

Table 4.2: Example of majority voting across LLM outputs

Table 4.3: R2F: Model Performance Comparison (F1 Scores)

Experiment	Description	F1 Score
Experiment-1	Ten unlabelled sentences in one prompt given to Gemini 1.5 Flash	38.54
Experiment-2	Fine-tuned Gemini 1.5 Flash	43.62
Experiment-3	Ten unlabelled sentences in one prompt given to Gemini 2.0 Flash	47.07
Experiment-4	Better Instruction Prompts	63.71
Experiment-5	LLM Voting	65.19
Baseline	Baseline given by MCN2 authors	68.05

4.2 Translation with Annotation (TrA)

Translation with annotation is a problem interesting in it's own right. Here we don't need just the translation of a given sentence but we also need the LLM to propagate the given labels correctly in the target language thus thereby testing both the translation as well as the reasoning capabilities of the Large Language model.

For the experiments, the example sentences that were provided to aid the LLM for in-context learning weren't changed throughout the experiments. These examples are listed in the appendix.

4.2.1 Experiment 0: Qualitative Analysis of TrA

This experiment was aimed as a Proof of Concept for the problem statement that we set up. The LLM was manually prompted and the results were qualitatively understood.

The prompt were of two kinds:

- Prompt 1: Manually labeled examples will have only one fine label per sentence.
- Prompt 2: Manually labeled examples can have multiple fine labels per sentence.

The prompt size was around 520+ Number of sentences to be labeled × Avg. token size per sentence. Gemini claims to have a context size of 1 million tokens and in the worst case we were still utilizing 0.01% of the maximum context length

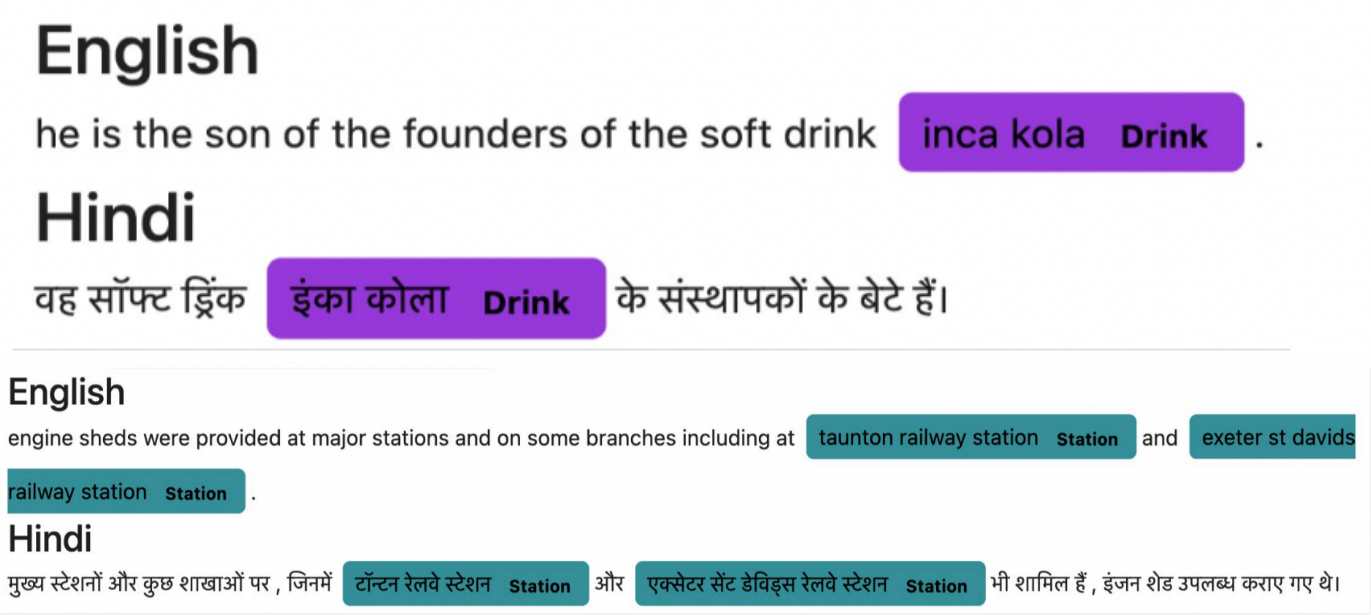


Figure 4.9: Gemini was able to translate as well as label single-label English sentences with great precision no matter how long or small these sentences were.

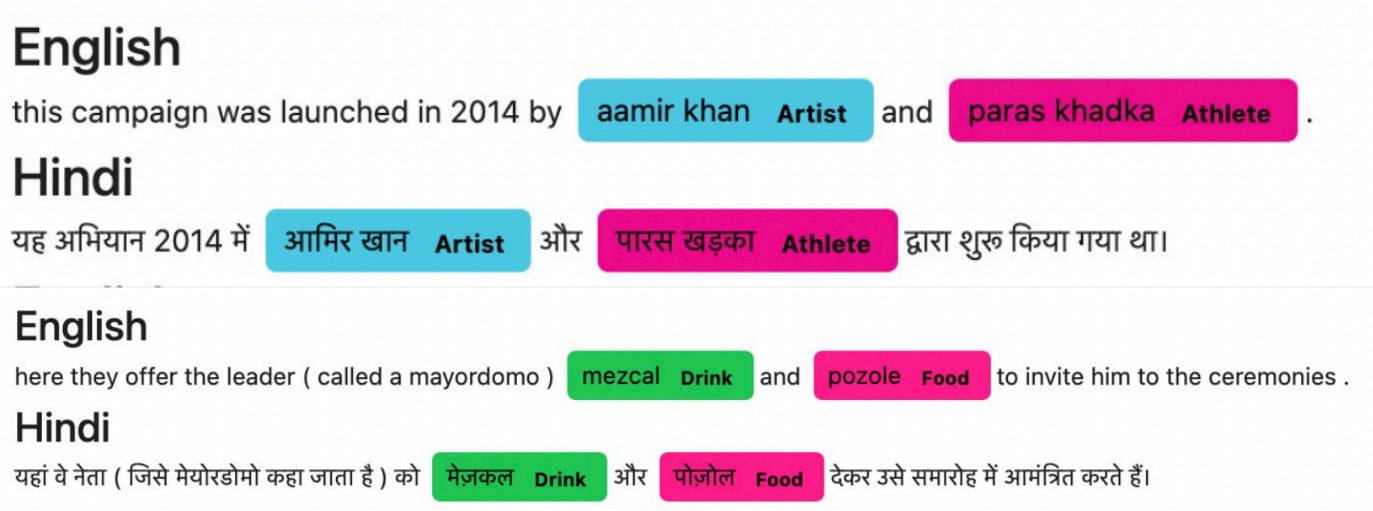


Figure 4.10: Gemini was able to translate multi fine label sentences as well without any difficulties.

The sentences as well as labels in the MultiCoNER-2 dataset were given in the form of lists. The first list consists of tokens and the second list (of the same size of the first one) contains fine labels for each of the token in the sentence list. So a natural way was to use in-context few shot prompt learning with input sentence and labels given in the form of lists. This bring us to our first experiment.

4.2.2 Experiment 1: Vanilla prompt

LLM Used: Gemini 1.5 flash

As explained above the first prompt consists of input sentences and labels in the form of lists and in output we expect the same list-type format i.e. two lists one for the sentence in the target language (Hindi for this experiment) and another the fine label list that match the size of the outputted sentence list.

The prompt also contains few shot examples for each of the fine label. We limited ourselves to 1 example per fine label. So in the prompt there were in total of 33 examples of manually labeled sentences. These sentences were selected on random basis from the English part of the MultiCoNER-2 dataset and then were manually translated and labeled.

The prompt is shown in figure 4.11

```
I want to initialise this prompt for fine labelling of sentences task with translation. The task would be as follows you will be given a sentence as a list of
tokens where each token is a word. You will also be given a list of fine labels (of same size as that of tokens list).
Here each fine label will be one among the following:

Facility, OtherLOC, HumanSettlement, Station
VisualWork, MusicalWork, WrittenWork, ArtWork, Software
MusicalGRP, PublicCORP, PrivateCORP, AerospaceManufacturer, SportsGRP, CarManufacturer, ORG
Scientist, Artist, Athlete, Politician, Cleric, SportsManager, OtherPER
Clothing, Vehicle, Food, Drink, OtherPROD
Medication/Vaccine, MedicalProcedure, AnatomicalStructure, Symptom, Disease


You need to translate the given English sentence to Hindi and then also propogate the fine labels based on the translation.

The fine label can be one among the above or it will be 'O' if the token doesn't match any of the above fine labels.

Also note the following points before you begin

- The output of the generated fine label list should be exactly same as to the generated token list

- Each token in the output list should correspond to a word or punctuation and should not be a group of words and/or punctuations

Now I will give you examples for each fine label. You will be given the English part and you have to output the Hindi one.

Fine Label: Station
*English*
Tokens: ['the', 'village', 'is', 'located', 'just', 'west', 'of', 'ivano-frankivsk', 'international', 'airport', '.']
FineLabels: ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'Station', 'Station', 'Station', 'O']

*Hindi*
Tokens: ['गांव', 'इवानो-फ्रैंकिव्स्क', 'अंतर्राष्ट्रीय', 'हवाई', 'अड्डे', 'के', 'ठीक', 'पश्चिम', 'में', 'स्थित', 'है', '।']
FineLabels: ['O', 'Station', 'Station', 'Station', 'Station', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

....
```

Figure 4.11: The vanilla prompt used in TrA

The model that we used for this experiment was *Gemini Flash 1.5*. It had a rate limit of 1500 requests per day. For conducting the experiments for **16,800** English sentences, it took around **19** hours after using multiple API keys.

The obtained translations appeared to be decent (see example in figure 4.12) which can be credited to the vast amount of training data of Google Translate that Gemini would be trained on. Gemini seem to maintain the overall context and grammar of the query sentence. But there was a problem in propagation of the labels because of a inherent problem in this vanilla prompt; This prompt relies on the assertion that the LLM would output token and fine label list of same length in the target language and this can be a problem. If the outputted token and label list is not of the same length, then there is no way to fix this alignment through some pragmatic way and we have to eventually ignore those sentences.

It turned out that out of the 16,800 sentences that we gave to Gemini, only **8700** had same length of the outputted token and label list. Even among these 8700, there is no

guarantee that there are no internal alignment issues i.e. label boundaries being shifted and are not properly aligned (see example in figure 4.13)

Results: The obtained *F1-score* from fine-tuning a PLM in the generated data set was **32.23** while the same was **68.05** for the original MultiCoNER-2 dataset. This is a huge difference which majorly stems from the problem of having most of the sentences removed out. This gives rise to the next experiment.

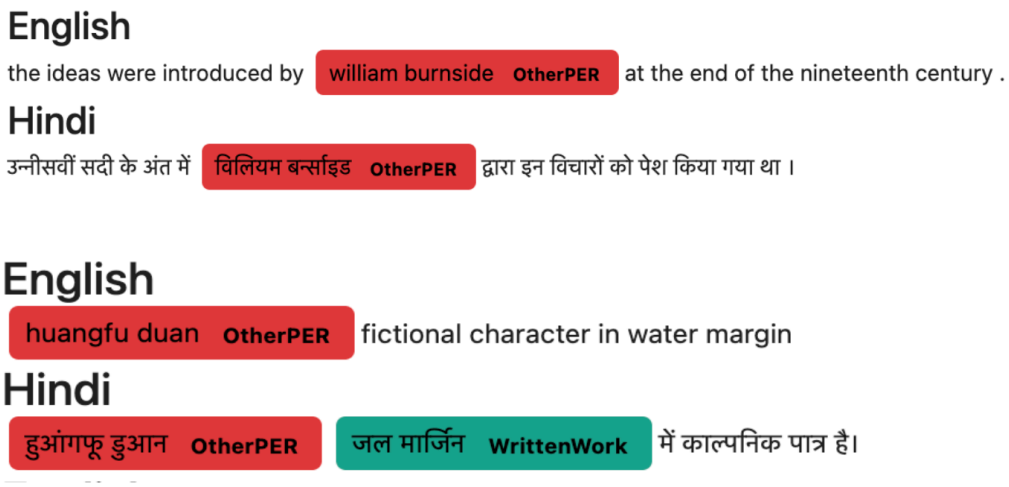


Figure 4.12: The first example shows that Gemini is not just translating the sentence word by word but also taking into context the whole sentence before translating. In the second example we can see that it is able to label tokens that aren't even labeled in the original sentence

4.2.3 Experiment 2: Chained output prompting

LLM Used: Gemini 2.0 flash

The shortcomings of last style of prompting was clear. The LLM was not able to remember how many label it needs to output and at what positions are the named entities. The problem clearly lies in dealing with list style of input/output operations. So we changed the prompt so that input sentence is given in multiple lines where each line would contain only two words the first one is the token and the second is it's fine label (see figure 4.15). The output is also expected in the same style.

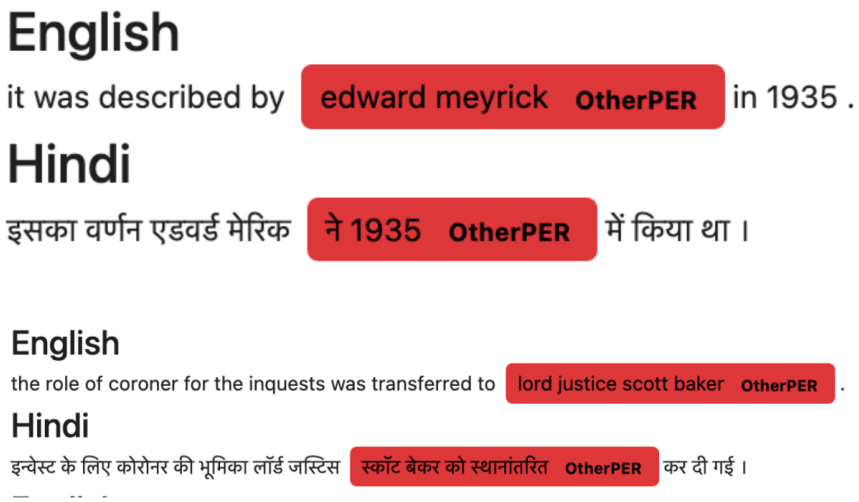


Figure 4.13: Example 1 depicts the internal alignment issues where the fine label tokens are shifted by 2 steps. Similar thing has happened in example 2 too.

This clearly removes any assertion to be maintained by the LLM because now whenever LLM will output a token it must also output its corresponding label. Although the translation ability will take a toll because of this but it's still manageable (see example in figure 4.14). Also for this experiment we used *Gemini Flash 2.0* which had a rate limit of **2000** requests per day and it nearly took **18** hours to perform the experiment.

Results: The F1-score jumped significantly from 32.23 in the last experiment to **56.15** in this experiment. The baseline is still **68.05**. Hence, we were successfully able to mitigate the shortcomings of the last experiment. Now we will try to improve on this.

4.2.4 Experiment 3: Self-Consistent Prompting

LLM Used: Gemini 2.0 flash

The major learning from the last experiment is that due to linear nature of LLMs(generating word by word), it becomes difficult for the LLM to output good translations when it also has to output labels in between them. To solve this we will be using a second LLM in series. The sole purpose of this LLM is to check on the generated output of the first LLM and output the final sentence. Also we made sure that this time instead of passing all the query

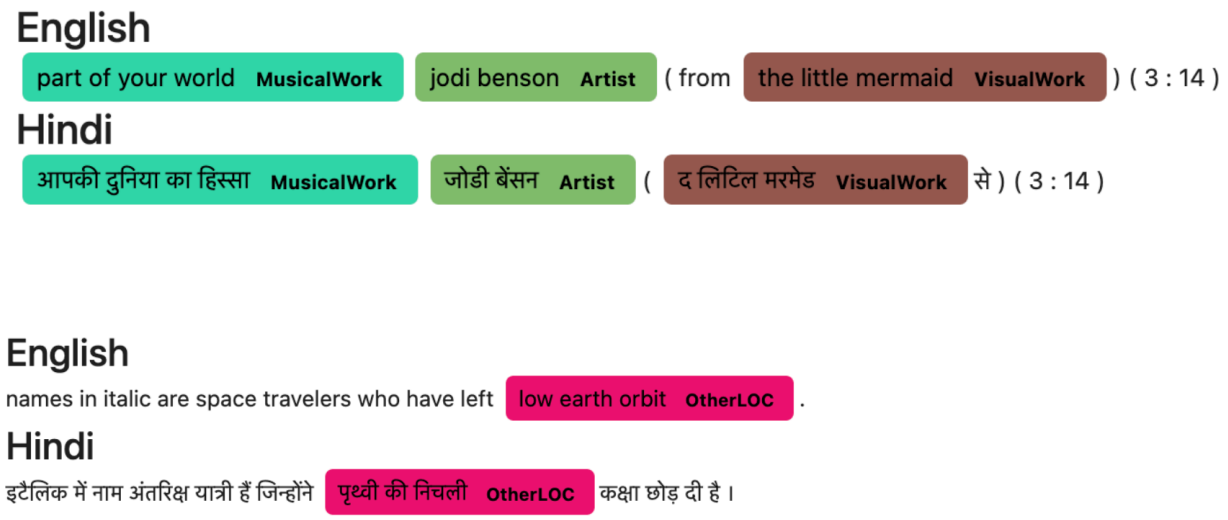


Figure 4.14: Translation challenges in COP: First example shows an instance where the LLM in translating the name of song "part of your world" whereas it should be transliterated as was done in case of "the little mermaid". Second example shows the LLM leaving some of the tokens unlabeled

sentences one after the another, we will refresh the context for each query so that the LLM doesn't take into its generation the earlier conversation.

Till this point we were certain that we have done the most that we could do using simple prompting, but still we were considerably behind the baseline dataset. So we moved on to compare the F1 scores for each of the labels (see figure 4.16) and found out that the major backliers were the labels 'Clothing', 'Drink' and 'OtherLOC'. After some inspection we found out that the major reason for this is the problem of synonyms. The problem is that whenever LLM tries to translate a given word it does so by selecting the most popular translation but this can cause problem. Let's consider an English word flower. Now there are multiple Hindi translations for this word like फूल and पुष्प . Now it may be possible that only the words labeled पुष्प are the ones labeled in the test corpus while the LLM generated a dataset where all those labels are given to फूल . So this can confuse the PLM in testing phase.

Results: Self Consistent prompting did give better results where the F1-score jumped to **57.51** from 56.15 of the last experiment. The base score is still **68.05**.

```
I want to initialise this prompt for fine labelling of sentences task with translation. The task would be as follows you will be given a sentence as a list
of tokens where each token is a word. You will also be given a list of labels (of same size as that of tokens list).
Here each label will be one among the following:

Facility, OtherLOC, HumanSettlement, Station
VisualWork, MusicalWork, WrittenWork, ArtWork, Software
MusicalGRP, PublicCORP, PrivateCORP, AerospaceManufacturer, SportsGRP, CarManufacturer, ORG
Scientist, Artist, Athlete, Politician, Cleric, SportsManager, OtherPER
Clothing, Vehicle, Food, Drink, OtherPROD
Medication/Vaccine, MedicalProcedure, AnatomicalStructure, Symptom, Disease


You need to translate the given English sentence to Hindi and then also propogate the labels based on the translation.

The label can be one among the above or it will be 'O' if the token doesn't match any of the above labels.

Also note the following points before you begin

- Each token in the output list should correspond to a word or punctuation and should not be a group of words and/or punctuations.

- There may be noise in the input data as well. You have to take that into account and generate noiseless output.

- You should not translate any non-'O' labeled tokens but transliterate them. For example, if the input is ["book", "the", "little", "mermaid"] having labels
["O", "WrittenWork", "WrittenWork", "WrittenWork"] then the output should be ["पुस्तक", "द", "लिटिल", "मर्मेड"] and not ["पुस्तक", "नन्ही", "जलपरी"].

Now I will give you examples for each label. You will be given the English part and you have to output the Hindi one.

Fine Label: Station
*Input*
the O
village O
is  O
located O
just    O
west    O
of  O
ivano-frankivsk Station
international   Station
airport Station
.   O

*Output*
गांव    O
इवानो-फ्रैंकिव्स्क  Station
अंतर्राष्ट्रीय  Station
हवाई    Station
अड्डे   Station
के  O
ठीक O
पश्चिम  O
में O
स्थित   O
है  O
।   O
```

Figure 4.15: The *Chained Output Prompting* prompt

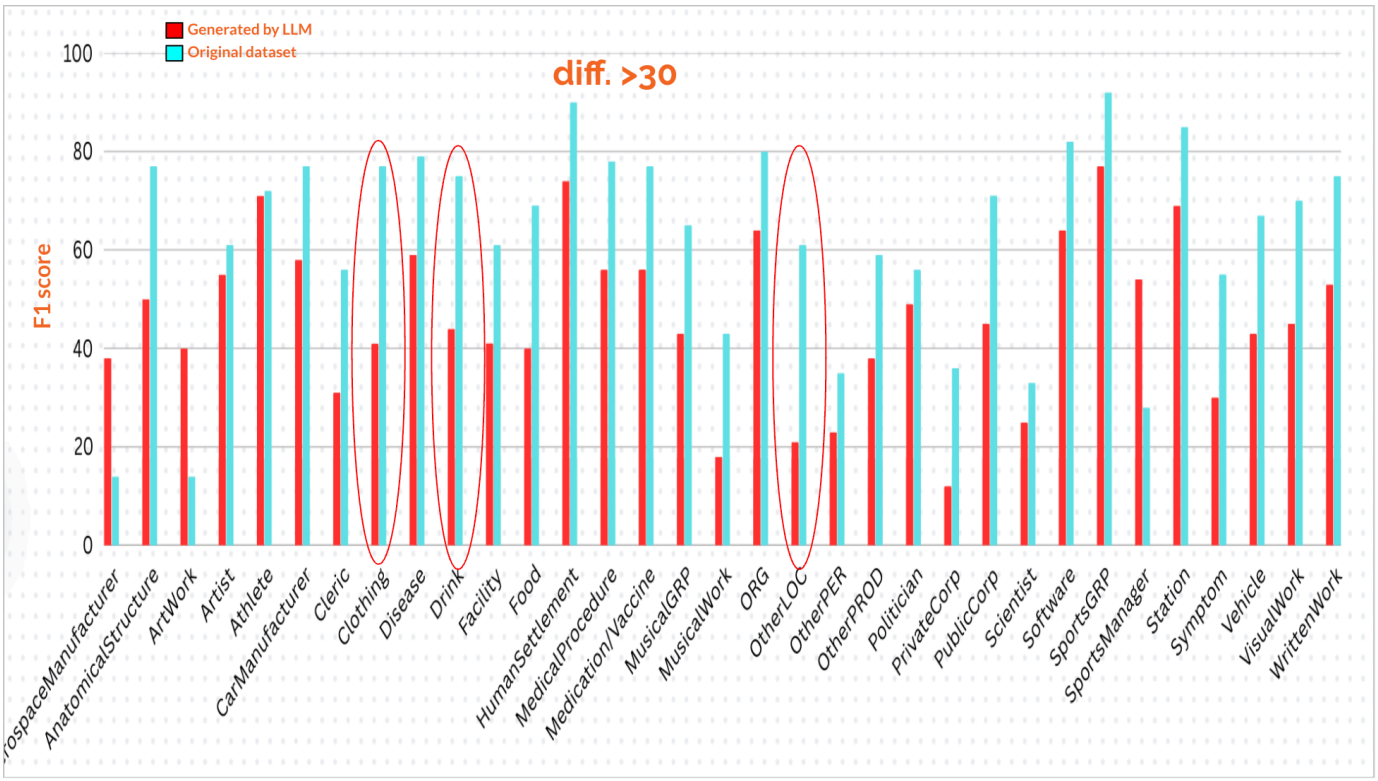


Figure 4.16: F1 score comparison for each label for the generated dataset as well as the original dataset

4.2.5 Experiment 4: Biased Few Shot Prompting

LLM Used: Gemini 2.0 flash

The main learning from last experiment is that we need a way to tell the Large Language Model that while translating a given word it should prefer a particular translation more than the others. This can be given in the form of few-shot examples that we were initially giving (33 examples per prompt).

The difference lies in the fact that this time instead of selecting random instances from the English part of MultiCoNER-2 dataset, we would be selecting random labeled instances from the Hindi part of MultiCoNER-2 and then we can sort of *reverse TrA* them to get there English counterparts and put it in the prompt. (See figure 4.17)

Also we will only give those examples that contains labels that belong to the label set of the query sentence. More formally let L_Q represent the set of labels of the query sentence and let L_S denote the set of labels of the example sentence. Then an example will become the part of prompt if and only if $L_S \subseteq L_Q$. Thus we will be having dynamic prompting instead of static prompting.

Moreover since we are only selecting important sentences this also means that we can increase the number of examples per label from 1 to 5. Hence we would be having around $5 \times 33 = 165$ examples in our prompt dataset. (See figure 4.17)

We also changed the rules part of the vanilla prompt to specify new type of rules that would be useful. (See figure 4.18)

We also modified the code to use parallel prompting and dynamic delays which reduced the prompting time significantly and to this end we were able to label all the sentences within **4 hours** ($\frac{1}{5}^{th}$ of the initial time)

Results: Biased few shot prompting didn't upset. It made the F1-score to inflate to **60.25** where the baseline was **68.05**. Till this point it was the best we could have done but still we still lagged behind the original F1-score. The problems seems not to be in the approach but in the MultiCoNER-2 dataset itself. This is discussed in section A

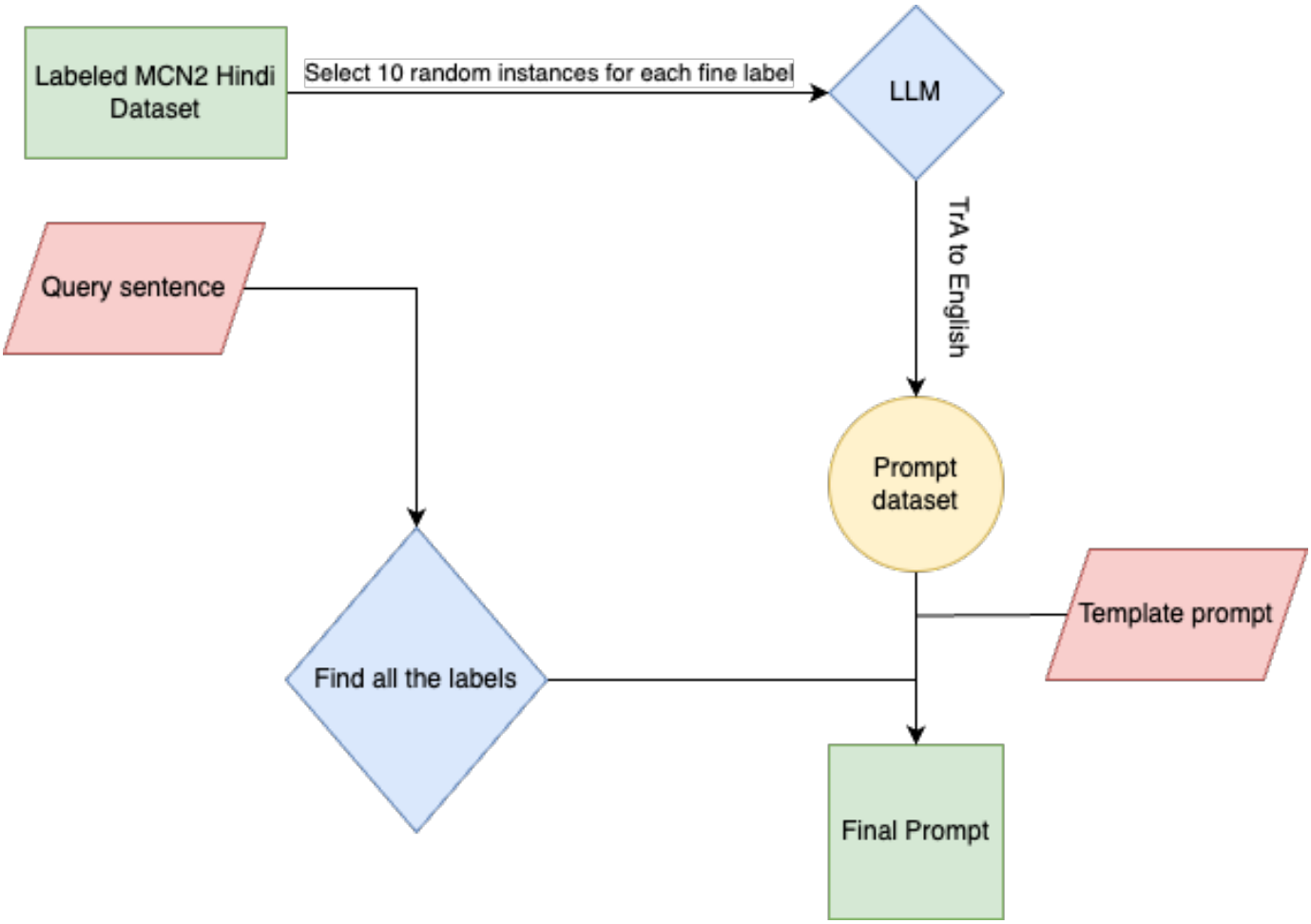


Figure 4.17: A depiction of pipeline for Biased Few Shot Prompting

```
Also note the following points before you begin


- THE INPUT MAY CONTAIN ERRORS IN THE LABELS. YOU NEED TO CORRECT THEM. DO THIS ONLY
WHEN YOU ARE COMPLETELY SURE THAT THE LABEL IS WRONG.

- FOLLOW THE SAME EXACT FORMAT AS THE INPUT SENTENCE IS GIVEN.

- ALWAYS TRANSLITERATE PROPER NOUNS.

- IF THERE ARE MULTIPLE TRANSLATIONS POSSIBLE FOR A GIVEN WORD, CHOOSE THE ONE THAT IS
GIVEN IN THE EXAMPLES. OTHERWISE, YOU CAN CHOOSE THE MOST POPULAR ONE.
```

Figure 4.18: New kind of prompting rules for Biased Few Shot Prompting

4.2.6 Experiment 5: Tree of Thoughts prompting

LLM Used: Gemini 2.0 flash

Based on the discussion in section 2.6.3 it is evident that Tree of Thoughts(ToT) prompting is in fact a very powerful technique to increase the reasoning capabilities of LLM. We can leverage the power of ToT in our statement.

The idea is to do *prompt chaining*; It refers to the practice of breaking down a complex task into sub-tasks and prompt the LLM for each of the task sequentially. In out case we can divide TrA into two tasks: translation & annotation. In translation part we will just be giving the unlabeled English sentence and expect an unlabeled Hindi sentence (pure translation). In the second task we will give the labeled English sentence along with the unlabeled Hindi sentence and expect the final labeled Hindi sentence (assisted annotation). The second part won't be any challenge so we would use simple prompting for that (see fig 4.19)

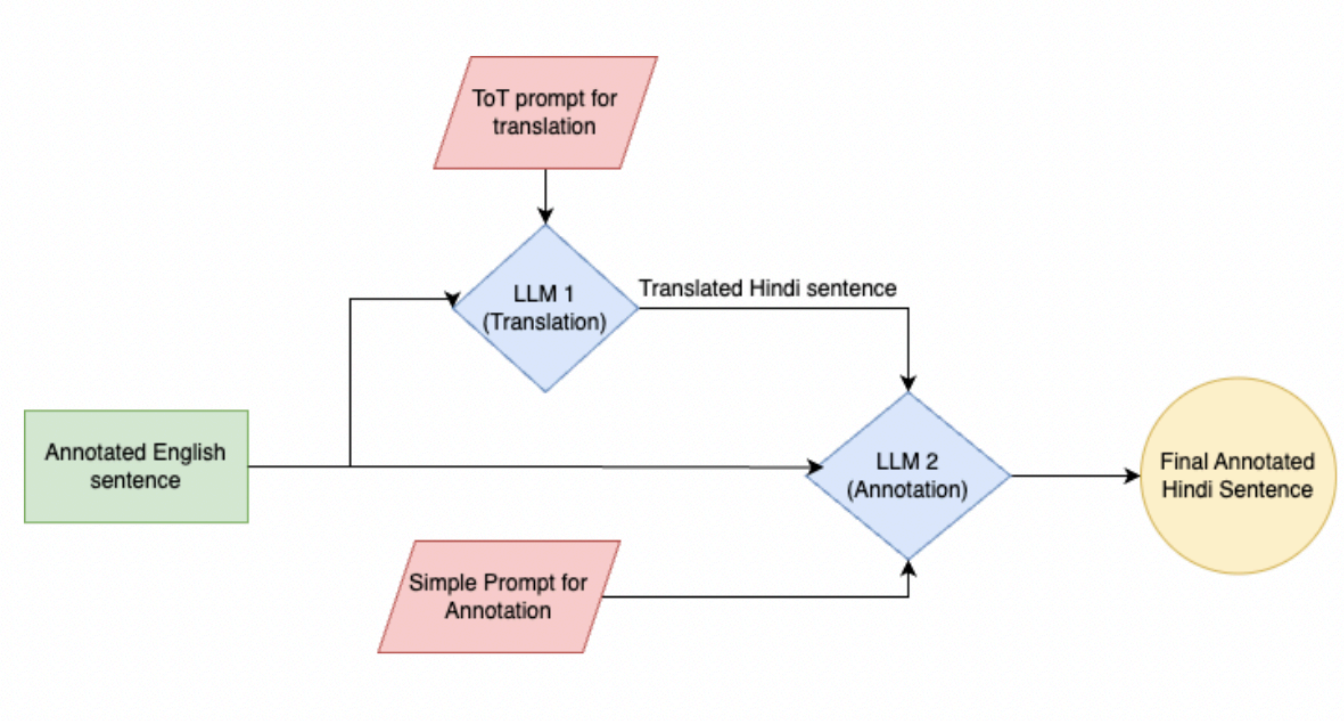


Figure 4.19: Magnified Tree of Thoughts pipleine for TrA

For the first part we can use Hulbert's ToT prompting (refer to section 2.6.3) in the hope that this will give us better translations. (see fig 4.20)

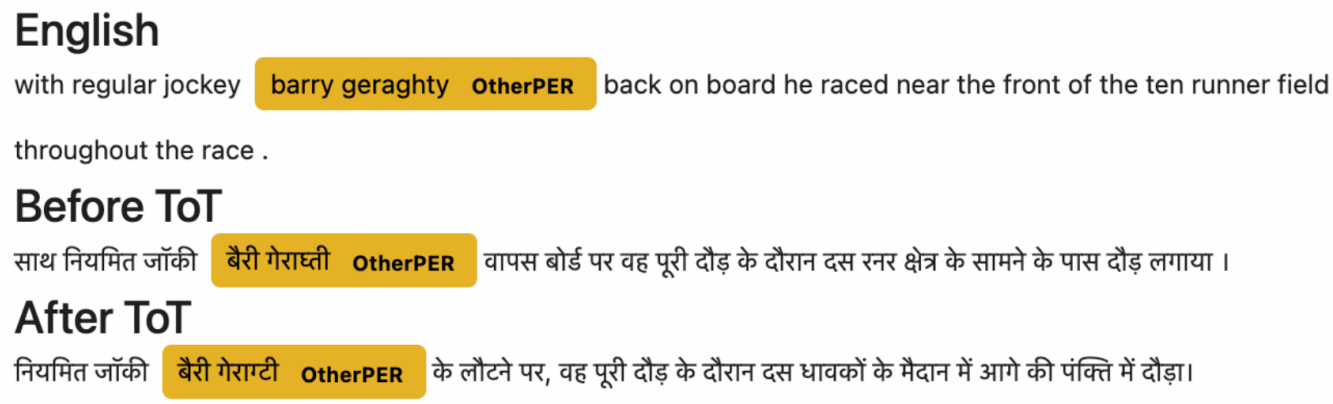


Figure 4.20: Example sentence that ToT improves translation. Notice how by simple prompting the translation was simple word to word but after ToT it became more natural

Results: Tree of thought prompting significantly increased the F1 score to **62.81** from 60.25. The base score was **68.05** (Again there is a limit by A)

4.2.7 Final results: Few shot testing, Bangla NER

With this we conclude all the experiments that we did in TrA. The best final F1-score that we reach was **62.81** (This is limited by A). All the results are summarized in table 4.4

We also checked the performance of the generated dataset in the few-shot learning paradigm. In other words instead of fine-tuning a PLM (supervised) learning, we partitioned the dataset $\mathcal{D}$ into two parts $\mathcal{S}$ and $\mathcal{Q}$. Let $L(\mathcal{D})$, $L(\mathcal{S})$ and $L(\mathcal{Q})$ denote the set of labels in $\mathcal{D}$, $\mathcal{S}$ and $\mathcal{Q}$ respectively. Then the partition requires $L(\mathcal{S}) \cap L(\mathcal{Q}) = \phi$ and $L(\mathcal{S}) \cup L(\mathcal{Q}) = L(\mathcal{D})$. The set $\mathcal{S}$ was used as the training set and the set $L(\mathcal{Q})$ was used to generate support set and query set. We modified **CONTAiNER** [16] by changing its base PLM to *Google's Muril large cased* [34] and used it for this purpose. The results for this experiment are given in table 4.5

Table 4.4: Model Performance Comparison (F1 Scores) for Hindi

Experiment	**Description**	**F1 Score**
Vanilla Prompting	Sentences and labels given in list format and expected in the same format	32.23
Chained Output Prompting	More natural syntax of giving input with a better model	56.15
Self consistent prompting	Another LLM in series to check on for mistakes, Refreshing context	57.51
Biased few shot prompting	Dynamic prompts instead of static one, examples taken from different corpus	60.25
Tree of thoughts prompting	Using Hulbert's tree of thoughts methods	62.81
Baseline	Original MCN-2 dataset (Hindi)	68.05

Dataset	**# of training instances**	**5-way**		**10-way**		**Avg.**
		1∼2 shot	**5∼10 shot**	**1∼2 shot**	**5∼10 shot**	
Generated dataset	$\sim 16,000$	32.23	38.81	**32.70**	31.45	33.80
Original Hindi MCN2	$\sim 9,000$	**34.00**	**39.90**	31.70	**33.98**	**34.89**

Table 4.5: F1 scores comparison on CONTAiNER(Few shot learning).

We also performed TrA experiment on *Bangla* language i.e. the source language remains as English but the target language is now Bangla. We were able to get similar results in this language too. The result is given in table 4.6

Table 4.6: Model Performance Comparison (F1 Scores) for Bangla language

Experiment	**F1 Score**
Generated Dataset	55.14
Original MCN2 dataset(Bangla)	63.55

Since this problem has translation as it's sub-part, there comes a natural question on how good translations given by Gemini are. So we compared various translation scores for both English to Hindi translation as well as English to Bangla translation. These results are given in table 4.7 and 4.8

Table 4.7: Translation scores comparison for English to Hindi translation by Gemini. It is desired to have bigger *BLEU, chrF, COMET* scores and smaller *TER* score

Experiment	**BLEU Score**	**TER score**	**chrF score**	**COMET score**
Google Translate	49.16	36.09	**71.09**	**0.87**
Bing Translate	42.03	55.08	57.11	0.71
Gemini	**50.30**	**35.90**	71.07	0.86

Table 4.8: Translation scores comparison for English to Bangla translation by Gemini. It is desired to have bigger *BLEU, chrF, COMET* scores and smaller *TER* score

Experiment	**BLEU Score**	**TER score**	**chrF score**	**COMET score**
Google Translate	37.22	47.23	68.37	**0.91**
Bing Translate	**39.10**	**44.73**	**69.63**	0.90
Gemini	28.45	55.68	62.60	0.90

Chapter 5

Conclusion and Future Work

5.1 Future Directions

This study unearthed many advantages and at the same time disadvantages of using long context LLM. Below we discuss some future directions which can be used to tackle the disadvantages posed by the current state of the art long context LLMs in downstream NLP tasks.

1. **Using a mixture of LLMs**: Different LLMs offer different advantages, abilities which can exploited for fine-grained labelling named entities, using reinforcement learning or voting algorithms.

2. **Building a RAG**: In many cases, since the long context LLM is using it's pre-trained knowledge to label entities without seeing the context, a retrievel-augmented generation model (RAG) can be used to minimize the effect of harnessing pre-trained knowledge. Up-to-date knowledge can be given in the knowledge source, and enhancements in the information retrieval mechanism can reduce the biasness introduced by the pre-trained knowledge of the long context LLM.

3. **Building an AI Agent**: Another direction could involve building a context-aware AI agent that can actively query and disambiguate information from large documents or

external databases when faced with uncertainty in entity labeling. Such an agent would leverage tool usage (e.g., search, lookup, summarization) and use LLMs as a decision-making module, combining them with memory and reasoning abilities to produce more accurate and consistent labels over long contexts.

4. **Incorporating Model Editing Techniques**: As long-context LLMs heavily rely on their pre-trained knowledge, they can often produce hallucinations or incorrect labels based on outdated or biased information. Model editing [35] offers a promising direction to address these issues without the need for full-scale retraining. By applying model editing techniques, specific factual updates or corrections relevant to named entities can be introduced directly into the model's parameters. This is especially useful in scenarios where new entities emerge or fine-grained distinctions evolve over time (e.g., geopolitical changes, new public figures, emerging terminology).

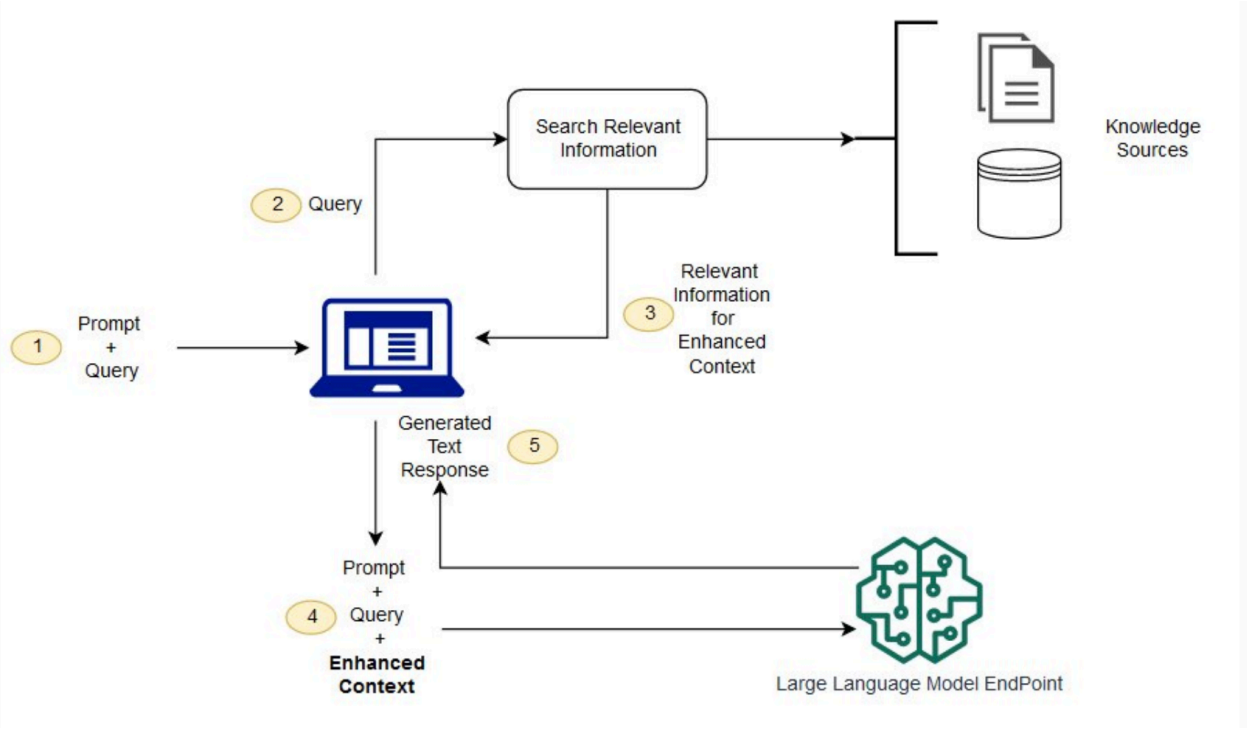


Figure 5.1: Working of RAG model

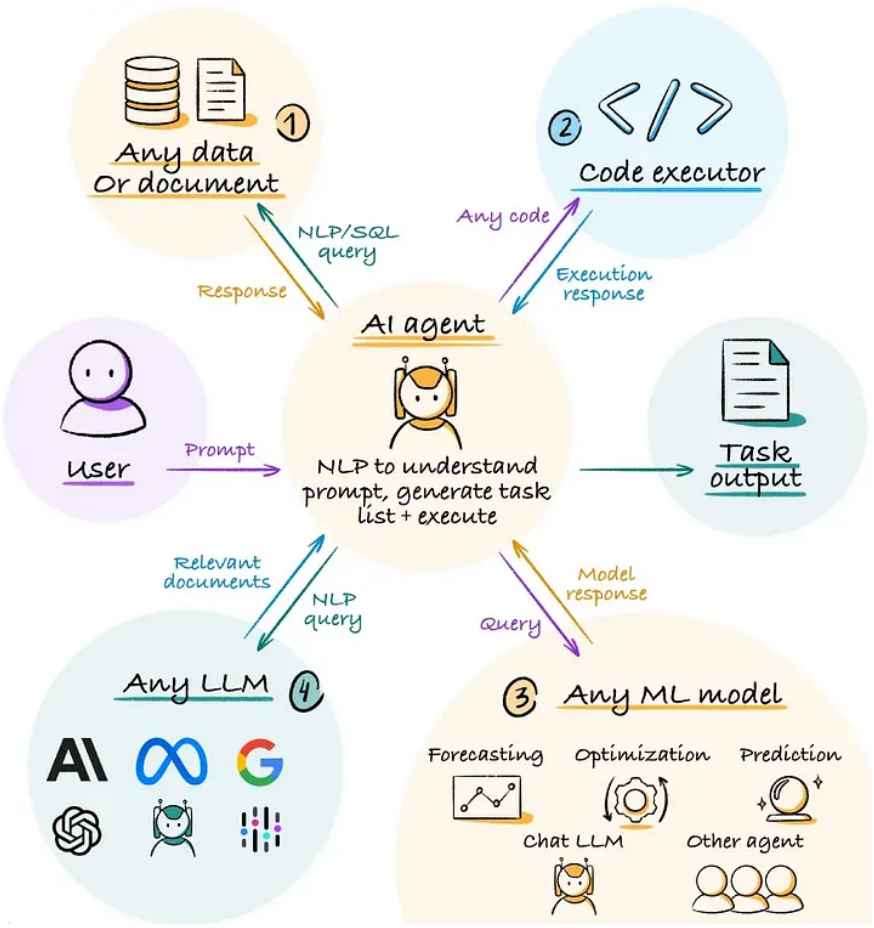


Figure 5.2: Working of an AI Agent

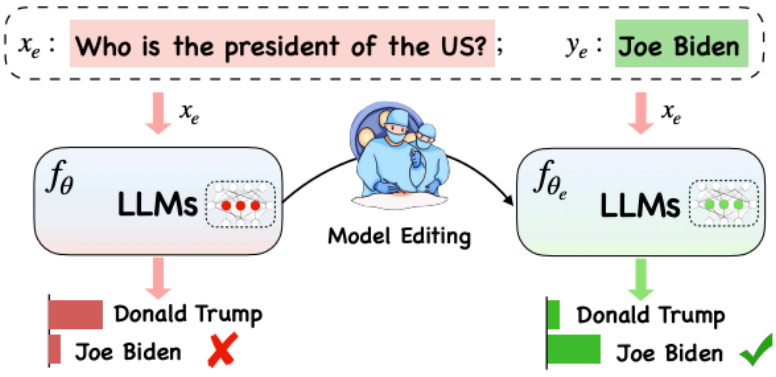


Figure 5.3: Model editing of LLMs

Bibliography

[1] N. Ding, Y. Chen, X. Han, G. Xu, X. Wang, P. Xie, H. Zheng, Z. Liu, J. Li, and H.-G. Kim, "Prompt-learning for fine-grained entity typing," in *Findings of the Association for Computational Linguistics: EMNLP 2022* (Y. Goldberg, Z. Kozareva, and Y. Zhang, eds.), (Abu Dhabi, United Arab Emirates), pp. 6888–6901, Association for Computational Linguistics, Dec. 2022.

[2] N. Ding, S. Hu, W. Zhao, Y. Chen, Z. Liu, H. Zheng, and M. Sun, "OpenPrompt: An open-source framework for prompt-learning," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations* (V. Basile, Z. Kozareva, and S. Stajner, eds.), (Dublin, Ireland), pp. 105–113, Association for Computational Linguistics, May 2022.

[3] N. Ding, G. Xu, Y. Chen, X. Wang, X. Han, P. Xie, H. Zheng, and Z. Liu, "Few-NERD: A few-shot named entity recognition dataset," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)* (C. Zong, F. Xia, W. Li, and R. Navigli, eds.), (Online), pp. 3198–3213, Association for Computational Linguistics, Aug. 2021.

[4] B. Fetahu, Z. Chen, S. Kar, O. Rokhlenko, and S. Malmasi, "Multiconer v2: a large multilingual dataset for fine-grained and noisy named entity recognition," 2023.

[5] A. Mhaske, H. Kedia, S. Doddapaneni, M. M. Khapra, P. Kumar, R. Murthy, and A. Kunchukuttan, "Naamapadam: A large-scale named entity annotated data for Indic languages," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (A. Rogers, J. Boyd-Graber, and N. Okazaki, eds.), (Toronto, Canada), pp. 10441–10456, Association for Computational Linguistics, July 2023.

[6] S. Malmasi, A. Fang, B. Fetahu, S. Kar, and O. Rokhlenko, "MultiCoNER: A large-scale multilingual dataset for complex named entity recognition," in *Proceedings of the 29th International Conference on Computational Linguistics* (N. Calzolari, C.-R. Huang, H. Kim, J. Pustejovsky, L. Wanner, K.-S. Choi, P.-M. Ryu, H.-H. Chen, L. Donatelli, H. Ji, S. Kurohashi, P. Paggio, N. Xue, S. Kim, Y. Hahm, Z. He, T. K. Lee, E. Santus, F. Bond, and S.-H. Na, eds.), (Gyeongju, Republic of Korea), pp. 3798–3809, International Committee on Computational Linguistics, Oct. 2022.

[7] T. Ruokolainen, P. Kauppinen, M. Silfverberg, and K. Lindén, "A finnish news corpus for named entity recognition," *Language Resources and Evaluation*, pp. 1–26, 2019.

[8] S. Pradhan, A. Moschitti, N. Xue, H. T. Ng, A. Björkelund, O. Uryupina, Y. Zhang, and Z. Zhong, "Towards robust linguistic analysis using OntoNotes," in *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, (Sofia, Bulgaria), pp. 143–152, Association for Computational Linguistics, Aug. 2013.

[9] A. Abhishek, S. B. Taneja, G. Malik, A. Anand, and A. Awekar, "Fine-grained entity recognition with reduced false negatives and large type coverage," in *Proceedings of the 1st Conference of the Automated Knowledge Base Construction*, (Amherst, USA), Automated Knowledge Base Construction, May 2019.

[10] N. Ding, G. Xu, Y. Chen, X. Wang, X. Han, P. Xie, H.-T. Zheng, and Z. Liu, "Few-nerd: A few-shot named entity recognition dataset," 2021.

[11] J. Li, H. Ding, J. Shang, J. McAuley, and Z. Feng, "Weakly supervised named entity tagging with learnable logical rules," 2021.

[12] A. Ratner, S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré, "Snorkel: rapid training data creation with weak supervision," *Proceedings of the VLDB Endowment*, vol. 11, p. 269–282, Nov. 2017.

[13] D. Mekala, V. Gangal, and J. Shang, "Coarse2fine: Fine-grained text classification on coarsely-grained annotated data," 2021.

[14] N. Guan, K. Chen, and N. Koudas, "Can large language models design accurate label functions?," 2023.

[15] R. Smith, J. A. Fries, B. Hancock, and S. H. Bach, "Language models in the loop: Incorporating prompting into weak supervision," 2022.

[16] S. S. S. Das, A. Katiyar, R. J. Passonneau, and R. Zhang, "Container: Few-shot named entity recognition via contrastive learning," 2022.

[17] Y. Zhou, A. I. Muresanu, Z. Han, K. Paster, S. Pitis, H. Chan, and J. Ba, "Large language models are human-level prompt engineers," 2023.

[18] T. Shin, Y. Razeghi, R. L. L. IV, E. Wallace, and S. Singh, "Autoprompt: Eliciting knowledge from language models with automatically generated prompts," 2020.

[19] B. Lester, R. Al-Rfou, and N. Constant, "The power of scale for parameter-efficient prompt tuning," 2021.

[20] H. Sun, A. Hüyük, and M. van der Schaar, "Query-dependent prompt evaluation and optimization with offline inverse rl," 2024.

[21] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou, "Chain-of-thought prompting elicits reasoning in large language models," 2023.

[22] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, "Large language models are zero-shot reasoners," 2023.

[23] S. Yao, D. Yu, J. Zhao, I. Shafran, T. L. Griffiths, Y. Cao, and K. Narasimhan, "Tree of thoughts: Deliberate problem solving with large language models," 2023.

[24] J. Long, "Large language model guided tree-of-thought," 2023.

[25] M. Besta, N. Blach, A. Kubicek, R. Gerstenberger, L. Gianinazzi, J. Gajda, T. Lehmann, M. Podstawski, H. Niewiadomski, P. Nyczyk, and T. Hoefler, "Graph of Thoughts: Solving Elaborate Problems with Large Language Models," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, pp. 17682–17690, Mar 2024.

[26] E. F. Tjong Kim Sang and F. De Meulder, "Introduction to the conll-2003 shared task: Language-independent named entity recognition," in *Proceedings of CoNLL-2003*, 2003.

[27] S. Pradhan and et al., "Towards robust linguistic analysis using ontonotes," *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, 2013.

[28] S. Malmasi and et al., "Findings of the multiconer shared task on multilingual complex named entity recognition," in *Proceedings of the 13th Workshop on Computational Approaches to Subjectivity, Sentiment, and Social Media Analysis (WASSA)*, 2022.

[29] N. Zhang and et al., "Findings of the multiconer ii shared task on multilingual complex named entity recognition and multilingual knowledge extraction," in *Proceedings of SemEval 2023*, 2023.

[30] D. Kakwani and et al., "Indicnlpsuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for indian languages," in *Findings of EMNLP 2020*, 2020.

[31] X. Pan and et al., "Cross-lingual name tagging and linking for 282 languages," in *Proceedings of ACL 2017*, 2017.

[32] H. Patil and et al., "L3cube-mahaner: A marathi named entity recognition dataset for deep learning," *arXiv preprint arXiv:2206.13113*, 2022.

[33] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov, "Unsupervised cross-lingual representation learning at scale," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, eds.), (Online), pp. 8440–8451, Association for Computational Linguistics, July 2020.

[34] S. Khanuja, D. Bansal, S. Mehtani, S. Khosla, A. Dey, B. Gopalan, D. K. Margam, P. Aggarwal, R. T. Nagipogu, S. Dave, S. Gupta, S. C. B. Gali, V. Subramanian, and P. Talukdar, "Muril: Multilingual representations for indian languages," 2021.

[35] K. Meng, D. Bau, A. Andonian, and Y. Belinkov, "Locating and editing factual associations in GPT," *Advances in Neural Information Processing Systems*, vol. 36, 2022. arXiv:2202.05262.

[36] H. R. Ehrenberg, J. Shin, A. J. Ratner, J. A. Fries, and C. Ré, "Data programming with ddlite: putting humans in a different part of the loop," in *Proceedings of the Workshop on Human-In-the-Loop Data Analytics*, HILDA '16, (New York, NY, USA), Association for Computing Machinery, 2016.

[37] S. Malmasi, A. Fang, B. Fetahu, S. Kar, and O. Rokhlenko, "Multiconer: A large-scale multilingual dataset for complex named entity recognition," 2022.

[38] E. F. Tjong Kim Sang and F. De Meulder, "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*, pp. 142–147, 2003.

[39] R. Ma, Z. Lin, X. Chen, X. Zhou, J. Wang, T. Gui, Q. Zhang, X. Gao, and Y. W. Chen, "Coarse-to-fine few-shot learning for named entity recognition," in *Findings of the*

Association for Computational Linguistics: ACL 2023 (A. Rogers, J. Boyd-Graber, and N. Okazaki, eds.), (Toronto, Canada), pp. 4115–4129, Association for Computational Linguistics, July 2023.

[40] G. Winata, S. Wu, M. Kulkarni, T. Solorio, and D. Preotiuc-Pietro, "Cross-lingual few-shot learning on unseen languages," in *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)* (Y. He, H. Ji, S. Li, Y. Liu, and C.-H. Chang, eds.), (Online only), pp. 777–791, Association for Computational Linguistics, Nov. 2022.

[41] A. Ratner, C. D. Sa, S. Wu, D. Selsam, and C. Ré, "Data programming: Creating large training sets, quickly," 2017.

Appendices

Appendix A

Text corpus limitation in MultiCoNER-2

The problem of TrA revolves around translating and propagating labels from a given dataset in source language to a dataset in target language. We then later fine tune a PLM using this dataset and test it on the test split of *Hindi MultiCoNER-2*.

But there is an unavoidable problem in this statement. Let's say the English MultiCoNER-2 dataset was generated using some English wikipedia corpus and then later divided into test and train split. Similar thing would have been done for Hindi as well. So both the train and test split of Hindi and English would have been taken from their respective corp-uses.

But what we were doing in TrA is taking the train split of English MCN-2 (say from some corpus A). Translating it into Hindi and then testing it against test split of Hindi MCN-2 (would be from some other corpus B). This would mean that there would be many type of entities that the PLM would have never seen in this generated dataset but it was thrown onto it while testing.

For example consider the fine label clothing. In figure A.1, we depict the entities (top 5 in count) that are labeled by clothing in both Hindi and English part of MultiCoNER-2. It is clearly evident from figure that there are entities like पगड़ी and तावीज़ don't exist as their

English translation in the English part of MultiCoNER-2. Hence these type of entities will never be encountered by the PLM in it's training phase.

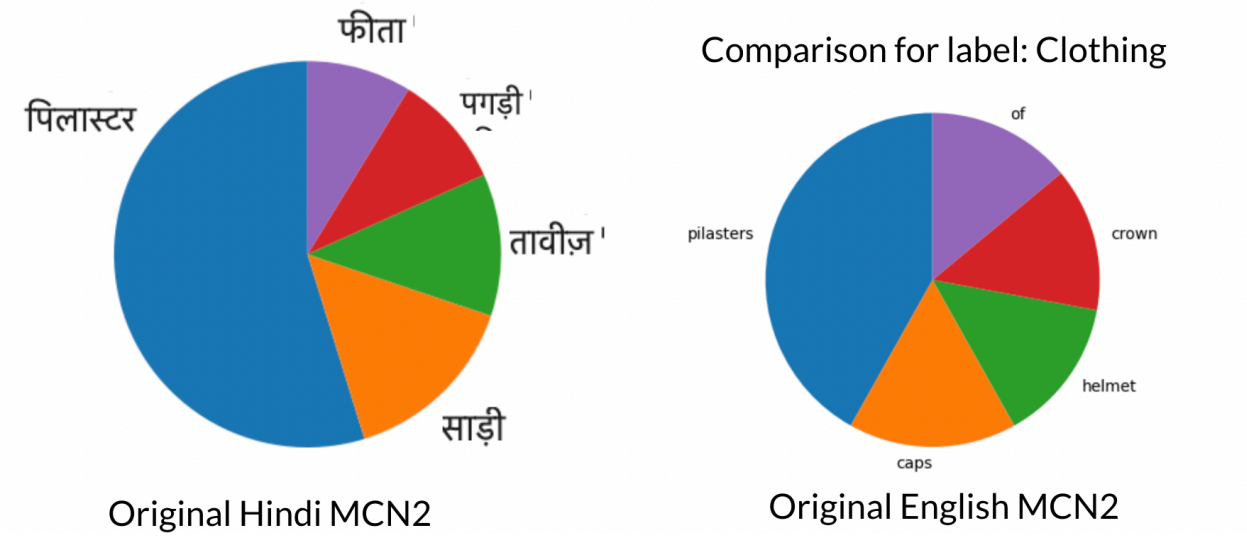


Figure A.1: A comparison of top-5 most labeled entites for both English and Hindi MCN-2

Now to prove this point we will set up an experiment. The experiment is as follows. First take a Hindi dataset and reverse TrA it to generate a noisy English dataset. Then again TrA it to get Hindi dataset back again. Now although this generated dataset would be very noisy but it would have same corpus that the original Hindi training split had (See figure A.2)

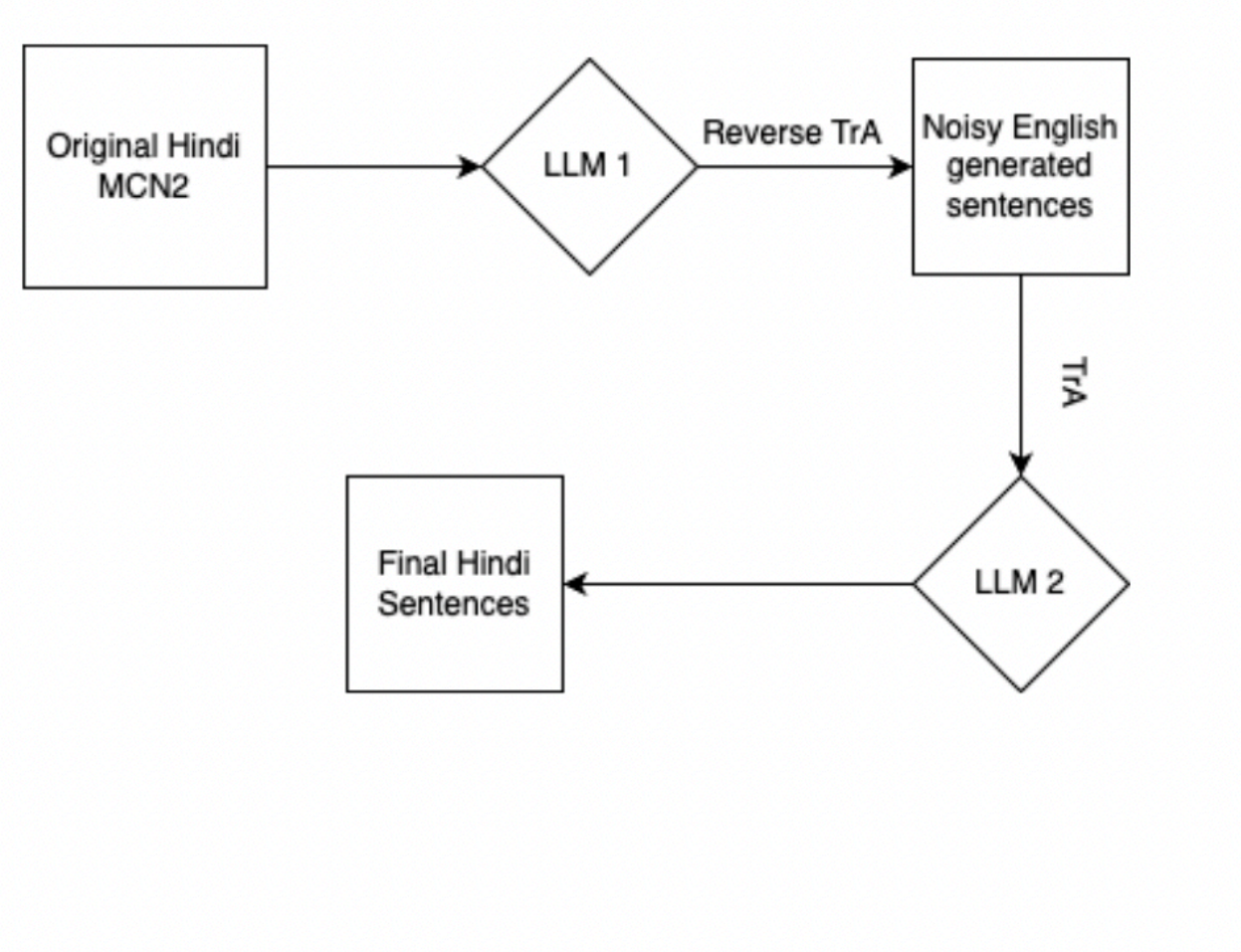


Figure A.2: Pipeline for text corpus comparison experiment

One would expect to get a F1-score lower than 60(which was the best that was obtained till now) if the corpus didn't matter. But we got an F1-score of **65.02** where the baseline was **68.05**. So it clearly depicts that there is a limit on F1-score that we can reach in TrA and no matter what we do after that we won't be able to increase it further.

Appendix B

Examples used for LLM in-context learning in R2F

For the experiments done for R2F, the example sentences that were provided to aid the LLM for in-context learning weren't changed throughout the experiments. These examples are listed below.

1. Sentence: **['जीभ', 'अधिकांश', 'कशेरुक', 'के', 'मुंह', 'के', 'फर्श', 'पर', 'एक', 'मांसपेशियों', 'के', 'हाइड्रोस्टेट']**

   Fine Labels: ['AnatomicalStructure', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

2. Sentence: **['यह', 'बताया', 'गया', 'कि', 'वे', 'अपनी', 'मौत', 'के', 'लिए', 'इन्तर्नास्योनाल', 'गाते', 'हुए', 'गए', 'थे।']**

   Fine Labels: ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'MusicalWork', 'O', 'O', 'O', 'O']

3. Sentence: **['एक', 'सामुदायिक', 'साइट', 'जहां', 'उपयोगकर्ता', 'सामग्री', 'बदलने', 'के', 'लिए', 'पंजीकरण', 'करते', 'हैं', 'लेकिन', 'इसे', 'देखने', 'के', 'लिए', 'नहीं', '(', 'उदाहरण', ':', 'विकिपीडिया', ')']**

   Fine Labels: ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'ORG', 'O']

4. Sentence: **['एक', 'टाइल', 'की', 'छत', 'के', 'साथ', 'सफेदी', 'वाली', 'ईंट', 'में', 'पनचक्की', '।']**

Fine Labels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’Facility’, ’O’]

5. Sentence: **['खुफिया', 'मंत्री', 'के', 'रूप', 'में', 'उनके', 'कार्यकाल', 'में', 'मेहदी', 'हाशमी', 'का', 'मामला', 'गिर', 'जाता', 'है।']**

   Fine Labels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’Cleric’, ’Cleric’, ’O’, ’O’, ’O’, ’O’, ’O’]

6. Sentence: **['क्लब', 'ने', 'वर्षों', 'की', 'अनिर्दिष्ट', 'संख्या', 'के', 'लिए', 'यांकी', 'स्टेडियम', 'पर', 'खेलने', 'की', 'योजना', 'बनाई', 'है।']**

   Fine Labels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’OtherLOC’, ’OtherLOC’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

7. Sentence: **['गीत', 'का', 'उपयोग', 'विज्ञापन', 'पेय', 'के', 'लिए', 'गिनीज़', 'के', 'लिए', 'किया', 'गया', 'था।']**

   Fine Labels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’Drink’, ’O’, ’O’, ’O’, ’O’, ’O’]

8. Sentence: **['यौगिक', 'फेनॉल', 'के', 'साथ', 'मौलिक', 'एल्यूमीनियम', 'की', 'प्रतिक्रिया', 'द्वारा', 'तैयार', 'किया', 'जा', 'सकता', 'है', ':']**

   Fine Labels: [’O’, ’Medication/Vaccine’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

9. Sentence: **['फास्ट', 'फूड', 'विज्ञापन', 'विज्ञापन',** ’fast’, ’food’, ’of’, ’usa’, ’hindi’, **'उत्पाद', 'और', 'जनता', 'तक', 'पहुंचने', 'के', 'लिए', 'कई', 'पहलुओं', 'का', 'उपयोग', 'करता', 'है।']**

   Fine Labels: [’O’, ’O’, ’O’, ’O’, ’Food’, ’Food’, ’Food’, ’Food’, ’Food’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

10. Sentence: **['हथियारों', 'को', 'एक', 'स्थानीय', 'कार', 'फर्म', 'पोर्शे', 'के', 'ट्रेडमार्क', 'में', 'भी', 'देखा', 'जा', 'सकता', 'है।']**

    Fine Labels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’CarManufacturer’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

11. Sentence: **['वह', 'कुछ', 'भी', 'नहीं', 'पहनती', 'है', 'लेकिन', 'एक', 'मिनीस्कर्ट', 'और', 'एक', 'गोरिल्ला', 'मास्क।']**

Fine Labels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’Clothing’, ’O’, ’O’, ’O’, ’O’]

12. Sentence: **['सिंथेटिक', 'वायरस', 'को', 'संभावित', 'जीन', 'चिकित्सा', 'टूल', 'के', 'रूप', 'में', 'भी', 'शोध', 'किया', 'गया', 'है।']**

    Fine Labels: [’O’, ’O’, ’O’, ’O’, ’MedicalProcedure’, ’MedicalProcedure’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

13. Sentence: **['मिडलसेक्स', 'काउंटी', 'क्रिकेट', 'क्लब', '१३', 'मैचों', 'और', '३', 'हार', 'के', 'साथ', '२६', 'मैचों', 'में', 'से', '८', '.', '१८२', 'के', 'साथ', 'रनर', 'अप', 'था।']**

    Fine Labels: [’SportsGRP’, ’SportsGRP’, ’SportsGRP’, ’SportsGRP’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

14. Sentence: **['यह', '१९३०', 'के', 'दशक', 'में', 'विद्युतमस्तिष्कलेखन', 'के', 'आविष्कार', 'के', 'साथ', 'शुरू', 'हुआ', 'जिसने', 'मस्तिष्क', 'की', 'इमेजिंग', 'को', 'सक्षम', 'किया', 'जैसा', 'कि', 'पहले', 'कभी', 'नहीं', 'देखा', 'गया', 'था।']**

    Fine Labels: [’O’, ’O’, ’O’, ’O’, ’O’, ’OtherPER’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

15. Sentence: **['स्वेतलाना', 'कुज़नेतसोवा', 'डिफेंडिंग', 'चैंपियन', 'था', 'लेकिन', 'तीसरे', 'दौर', 'में', 'मारिया', 'कि-रिलैंको', 'में', 'हार', 'गया।']**

    Fine Labels: [’Athlete’, ’Athlete’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’Athlete’, ’Athlete’, ’O’, ’O’, ’O’]

16. Sentence: **['यह', 'झियान', 'चीन', 'के', 'केंद्र', 'भाग', 'में', 'स्थित', 'है।']**

    Fine Labels: [’O’, ’HumanSettlement’, ’HumanSettlement’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

17. Sentence: **['बुदायूँ', 'रेलवे', 'स्टेशन', 'गांव', 'से', '७', 'किलोमीटर', 'की', 'दूरी', 'पर', 'स्थित', 'है।']**

    Fine Labels: [’Station’, ’Station’, ’Station’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

18. Sentence: **['जब', 'पहला', 'वाणिज्यिक', 'इलेक्ट्रॉनिक', 'कंप्यूटर', '१९५२', 'में', 'आइ.बी.एम', 'द्वारा', 'पेश', 'किया', 'गया', 'था', 'तो', 'मशीन', 'को', 'बनाए', 'रखना', 'और', 'महंगा', 'था।']**

Fine Labels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’OtherPROD’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

19. Sentence: **['द', 'क्रिएशन', 'ऑफ', 'आदम', '(', 'सी।', '१५११', ')', 'के', 'लेखक']**
Fine Labels: [’ArtWork’, ’ArtWork’, ’ArtWork’, ’ArtWork’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

20. Sentence: **['नाना', 'अकुफो-अडो', 'ने', 'भी', 'उनके', 'लिए', 'एक', 'श्रद्धांजलि', 'लिखी।']**
Fine Labels: [’Politician’, ’Politician’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

21. Sentence: **['रॉबर्ट', 'ब्राउनिंग', 'एक', 'हल्की', 'महिला', 'में', 'एक', 'व्यक्ति', 'के', 'रूप', 'में', 'बेसिलिस्क', 'को', 'शामिल', 'किया', 'गया।']**
Fine Labels: [’Artist’, ’Artist’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

22. Sentence: **['२०२०', 'का', 'पुरस्कार', 'बीटीएस', 'पर', 'चला', 'गया।']**
Fine Labels: [’O’, ’O’, ’O’, ’MusicalGRP’, ’O’, ’O’, ’O’]

23. Sentence: **['स्कूल', 'फेसबुक', 'पर', 'पाया', 'जा', 'सकता', 'है।']**
Fine Labels: [’O’, ’Software’, ’O’, ’O’, ’O’, ’O’, ’O’]

24. Sentence: **['प्रसवपूर्व', 'स्ट्रोक', 'के', 'कारण', 'उनके', 'पास', 'प्रमस्तिष्क', 'अंगघात', 'है।']**
Fine Labels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’Disease’, ’Disease’, ’O’]

25. Sentence: **['अस्थिभंग', 'जैसी', 'चोटों', 'के', 'कारण', 'कई', 'प्रदर्शनकारियों', 'को', 'अस्पताल', 'में', 'भर्ती', 'कराया', 'गया', 'था।']**
Fine Labels: [’Symptom’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

26. Sentence: **['व्यापारियों', 'के', 'अवहेलना', 'के', 'मूड', 'पर', 'टिप्पणी', 'करते', 'हुए', 'डॉन', '(अखबार)', 'में', 'एक', 'संपादकीय', 'ने', 'हर', 'किसी', 'से', 'मध्य', 'मैदान', 'खोजने', 'का', 'आग्रह', 'किया।']**
Fine Labels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’WrittenWork’, ’WrittenWork’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

27. Sentence: ['उन्हें', 'विनाशक', 'पोत', 'पर', 'सवार', 'कर', 'ड्यूटी', 'सौंपी', 'गई', 'थी।']
    Fine Labels: ['O', 'Vehicle', 'Vehicle', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

28. Sentence: ['पूर्ण', 'हस्तांतरण', 'से', 'पहले', 'सभी', 'उड़ानों', 'को', 'विशेष', 'रूप', 'से', 'टर्किश', 'एयरलाइंस', 'द्वारा', 'संचालित', 'किया', 'गया', 'था।']
    Fine Labels: ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'PublicCorp', 'PublicCorp', 'O', 'O', 'O', 'O', 'O']

29. Sentence: ['यह', 'संस्करण', 'स्पाइडर-मैन', 'सलाहकार', 'के', 'रूप', 'में', 'कार्य', 'करता', 'है।']
    Fine Labels: ['O', 'O', 'Scientist', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

30. Sentence: ['उनके', '४', '-', '४', '-', '२', 'के', 'गठन', 'को', 'स्टीवन', 'जेरार्ड', 'मिडफील्ड', 'के', 'बाईं', 'ओर', 'पोजिशनिंग', 'के', 'साथ', 'पुरानी', 'के', 'रूप', 'में', 'निकाला', 'गया', 'था।']
    Fine Labels: ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'SportsManager', 'SportsManager', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

31. Sentence: ['२०००', 'में', 'संपत्ति', 'के', 'प्रवेश', 'द्वार', 'के', 'पास', 'भूमि', 'का', 'एक', 'महत्वपूर्ण', 'खाली', 'भूखंड', 'मौजूद', 'था', 'जो', 'मूल', 'रूप', 'से', 'वर्साचे', 'फ्लैगशिप', 'स्टोर', 'के', 'लिए', 'था।']
    Fine Labels: ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'PrivateCorp', 'O', 'O', 'O', 'O', 'O']

32. Sentence: ['कई', 'विकल्पों', 'का', 'अध्ययन', 'बोइंग', 'निगम', 'द्वारा', 'किया', 'गया', 'था।']
    Fine Labels: ['O', 'O', 'O', 'O', 'AerospaceManufacturer', 'O', 'O', 'O', 'O', 'O']

33. Sentence: ['फिल्म', 'को', 'समीक्षकों', 'द्वारा', 'प्रशंसित', 'किया', 'गया', 'था', 'और', '८', '2000', 'अकादमी', 'पुरस्कार', 'सहित', 'कई', 'पुरस्कारों', 'के', 'लिए', 'नामांकित', 'किया', 'गया', 'था।']
    Fine Labels: ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'VisualWork', 'VisualWork', 'VisualWork', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

Appendix C

Examples used for LLM in-context learning in TrA

For the experiments, the example sentences that were provided to aid the LLM for in-context learning weren't changed throughout the experiments. These examples are listed below.

1. **Fine Label**: **Station**

   English

   Tokens: ['the', 'village', 'is', 'located', 'just', 'west', 'of', 'ivano-frankivsk', 'international', 'airport', '.']

   FineLabels: ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'Station', 'Station', 'Station', 'O']

   Hindi

   Tokens: **['गांव', 'इवानो-फ्रैंकिव्स्क', 'अंतर्राष्ट्रीय', 'हवाई', 'अड्डे', 'के', 'ठीक', 'पश्चिम', 'में', 'स्थित', 'है', '।']**

   FineLabels: ['O', 'Station', 'Station', 'Station', 'Station', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

2. **Fine Label**: **Artist**

   English

   Tokens: ['james', 'i', 'took', 'a', 'greater', 'interest', 'in', 'naval', 'power', '.']

   FineLabels: ['Artist', 'Artist', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

Hindi

Tokens: **['जेम्स', 'आई.', 'ने', 'नौसैनिक', 'शक्ति', 'में', 'अधिक', 'रुचि', 'ली', '।']**

FineLabels: [’Artist’, ’Artist’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

3. **Fine Label**: **HumanSettlement**

English

Tokens: [’the’, ’county’, ’has’, ’one’, ’city’, ’:’, ’tehran’, ’.’]

FineLabels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’HumanSettlement’, ’O’]

Hindi

Tokens: **['काउंटी', 'में', 'एक', 'शहर', 'है', ':', 'तेहरान', '।']**

FineLabels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’HumanSettlement’, ’O’]

4. **Fine Label**: **SportsGRP**

English

Tokens: [’he’, ’has’, ’also’, ’pitched’, ’in’, ’the’, ’minor’, ’leagues’, ’for’, ’the’, ’atlanta’, ’braves’, ’and’, ’boston’, ’red’, ’sox’, ’.’]

FineLabels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’SportsGRP’, ’SportsGRP’, ’O’, ’SportsGRP’, ’SportsGRP’, ’SportsGRP’, ’O’]

Hindi

Tokens: **['उन्होंने', 'माइनर', 'लीग्स', 'में', 'भी', 'अटलांटा', 'ब्रेव्स', 'और', 'बोस्टन', 'रेड', 'सॉक्स', 'के', 'लिए', 'पिचिंग', 'की', 'है', '।']**

FineLabels: [’O’, ’O’, ’O’, ’O’, ’O’, ’SportsGRP’, ’SportsGRP’, ’O’, ’SportsGRP’, ’SportsGRP’, ’SportsGRP’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

5. **Fine Label**: **Food**

English

Tokens: [’goulash’, ’soup’, ’;’, ’it’, ’is’, ’possible’, ’to’, ’cook’, ’gulyás’, ’like’, ’a’, ’stew’, ’as’, ’well’, ’(’, ’e.g.’, ’székelygulyás’, ’)’, ’.’]

FineLabels: [’Food’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’Food’, ’O’, ’O’]

Hindi

Tokens: **['गुलाश', 'सूप', ';', 'गुल्याश', 'को', 'एक', 'स्टू', 'की', 'तरह', 'भी', 'पकाना', 'संभव', 'है', '(', 'जैसे', 'स्ज़ेकेलीगुल्याश', ')', '।']**

FineLabels: [’Food’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’Food’, ’O’, ’O’]

6. **Fine Label**: **MusicalGRP**

English

Tokens: [’initially’, ’def’, ’leppard’, ’was’, ’scheduled’, ’to’, ’play’, ’at’, ’the’, ’festival’, ’.’]

FineLabels: [’O’, ’MusicalGRP’, ’MusicalGRP’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

Hindi

Tokens: **['शुरुआत', 'में', 'डेफ', 'लेपर्ड', 'त्योहार', 'में', 'प्रदर्शन', 'करने', 'के', 'लिए', 'निर्धारित', 'था।']**

FineLabels: [’O’, ’MusicalGRP’, ’MusicalGRP’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

7. **Fine Label**: **Politician**

English

Tokens: [’he’, ’started’, ’his’, ’national’, ’political’, ’career’, ’as’, ’a’, ’supporter’, ’of’, ’zia-ul-haq’, ’.’]

FineLabels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’Politician’, ’O’]

Hindi

Tokens: **['उन्होंने', 'अपना', 'राष्ट्रीय', 'राजनीतिक', 'करियर', 'जिया-उल-हक', 'के', 'समर्थक', 'के', 'रूप', 'में', 'शुरू', 'किया।']**

FineLabels: [’O’, ’O’, ’O’, ’O’, ’O’, ’Politician’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

8. **Fine Label**: **Software**

English

Tokens: ['firefox', 'for', 'ios', 'a', 'project', 'for', 'ios', 'smartphones', 'and', 'tablets']

FineLabels: ['Software', 'Software', 'Software', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

Hindi

Tokens: **['आईओएस', 'के', 'लिए', 'फायरफॉक्स', 'एक', 'प्रोजेक्ट', 'है', 'जो', 'आईओएस', 'स्मार्टफोन', 'और', 'टैबलेट्स', 'के', 'लिए', 'है।']**

FineLabels: ['Software', 'Software', 'Software', 'Software', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

9. **Fine Label**: **AnatomicalStructure** *English*

Tokens: ['the', 'sternohyoid', 'and', 'sternothyroid', 'muscles', 'stretch', 'along', 'its', 'length', '.']

FineLabels: ['O', 'AnatomicalStructure', 'O', 'AnatomicalStructure', 'AnatomicalStructure', 'O', 'O', 'O', 'O', 'O']

Hindi

Tokens: **['स्टर्नोहायोइड', 'और', 'स्टर्नोथायरॉयड', 'पेशियाँ', 'इसके', 'लंबाई', 'के', 'साथ', 'खींची', 'जाती', 'हैं।']**

FineLabels: ['AnatomicalStructure', 'O', 'AnatomicalStructure', 'AnatomicalStructure', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

10. **Fine Label**: **CarManufacturer**

English

Tokens: ['with', 'his', 'family', 'he', 'lived', 'in', 'a', 'cottage', 'and', 'used', 'the', 'horch', 'car', 'with', 'a', 'sailor', 'as', 'a', 'driver', '.']

FineLabels: ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'CarManufacturer', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

Hindi

Tokens: **['अपनी', 'परिवार', 'के', 'साथ', 'वह', 'एक', 'कुटिया', 'में', 'रहते', 'थे', 'और', 'हॉर्च', 'कार', 'का',**

'उपयोग', 'एक', 'नाविक', 'को', 'चालक', 'के', 'रूप', 'में', 'करते', 'थे।']

FineLabels: ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'CarManufacturer', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

11. **Fine Label**: **Athlete**

English

Tokens: ['he', 'eventually', 'took', 'two', 'seconds', 'as', 'ben', 'swift', '(', ')', 'snatched', 'the', 'three', 'bonus', 'seconds', '.']

FineLabels: ['O', 'O', 'O', 'O', 'O', 'O', 'Athlete', 'Athlete', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

Hindi

Tokens: ['अंततः', 'उन्होंने', 'दो', 'सेकंड', 'लिए', 'जब', 'बेन', 'स्विफ्ट', '(', ')', 'ने', 'तीन', 'बोनस', 'सेकंड', 'छीन', 'लिए।']

FineLabels: ['O', 'O', 'O', 'O', 'O', 'O', 'Athlete', 'Athlete', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

12. **Fine Label**: **ArtWork**

English

Tokens: ['she', 'produced', 'miniature', 'paintings', 'and', 'pastels', 'during', 'her', 'career', '.']

FineLabels: ['O', 'O', 'ArtWork', 'ArtWork', 'O', 'O', 'O', 'O', 'O', 'O']

Hindi

Tokens: ['उसने', 'अपने', 'करियर', 'के', 'दौरान', 'सूक्ष्म', 'चित्रकला', 'और', 'पेस्टल्स', 'निर्मित', 'किए।']

FineLabels: ['O', 'O', 'O', 'O', 'O', 'ArtWork', 'ArtWork', 'O', 'O', 'O', 'O']

13. **Fine Label**: **Vehicle**

English Tokens: ['the', 'ship', 'could', 'reach', 'a', 'maximum', 'speed', 'of', '25', 'knots', 'among', 'the', 'fastest', 'of', 'its', 'time', 'but', 'still', 'slower', 'than', 'the', 'mauretania', '.']

FineLabels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’Vehicle’, ’O’]

Hindi

Tokens: **['जहाज', 'अपने', 'समय', 'के', 'सबसे', 'तेज', 'में', 'से', '25', 'नॉट्स', 'की', 'अधिकतम', 'गति', 'तक', 'पहुंच', 'सकता', 'था', 'लेकिन', 'फिर', 'भी', 'मॉरिटेनिया', 'से', 'धीमा', 'था।']**

FineLabels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’Vehicle’, ’O’, ’O’, ’O’]

14. **Fine Label**: **OtherLOC**

English

Tokens: [’martha’, ’was’, ’inducted’, ’into’, ’the’, ’maryland’, ’women’, ”’s”, ’hall’, ’of’, ’fame’, ’in’, ’1988’, ’.’]

FineLabels: [’O’, ’O’, ’O’, ’O’, ’O’, ’OtherLOC’, ’OtherLOC’, ’OtherLOC’, ’OtherLOC’, ’OtherLOC’, ’OtherLOC’, ’O’, ’O’, ’O’]

Hindi

Tokens: **['मार्था', 'को', '1988', 'में', 'मैरीलैंड', 'महिला', 'हॉल', 'ऑफ़', 'फेम', 'में', 'सम्मिलित', 'किया', 'गया।']**

FineLabels: [’O’, ’O’, ’O’, ’O’, ’OtherLOC’, ’OtherLOC’, ’OtherLOC’, ’OtherLOC’, ’OtherLOC’, ’O’, ’O’, ’O’, ’O’]

15. **Fine Label**: **Facility**

English

Tokens: [’the’, ’artesian’, ’well’, ’at’, ’artesian’, ’commons’, ’park’, ’a’, ’former’, ’parking’, ’lot’, ’is’, ’active’, ’.’]

FineLabels: [’O’, ’O’, ’O’, ’O’, ’Facility’, ’Facility’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

Hindi

Tokens: **['आर्टेशियन', 'वेल', 'आर्टेशियन', 'कॉमन्स', 'पार्क', 'में', 'एक', 'पूर्व', 'पार्किंग', 'लॉट', 'है', 'जो', 'सक्रिय', 'है।']**

FineLabels: [’O’, ’O’, ’Facility’, ’Facility’, ’Facility’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

16. **Fine Label**: **OtherPROD**

English

Tokens: [’he’, ’began’, ’his’, ’business’, ’career’, ’as’, ’a’, ’stationery’, ’salesman’, ’.’]

FineLabels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’OtherPROD’, ’O’, ’O’]

Hindi

Tokens: **['उन्होंने', 'अपना', 'व्यावासिक', 'करियर', 'एक', 'स्टेशनरी', 'विक्रेता', 'के', 'रूप', 'में', 'शुरू', 'किया।']**

FineLabels: [’O’, ’O’, ’O’, ’O’, ’O’, ’OtherPROD’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

17. **Fine Label**: **MusicalWork**

English

Tokens: [’yoko’, ’wrote’, ’about’, ’the’, ’song’, ’when’, ’it’, ’was’, ’included’, ’on’, ’her’, ’1992’, ’boxset’, ’onobox’, ’:’]

FineLabels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’MusicalWork’, ’O’]

Hindi

Tokens: **['योको', 'ने', 'उस', 'गाने', 'के', 'बारे', 'में', 'लिखा', 'जब', 'वह', 'उनके', '1992', 'बॉक्ससेट', 'ओनो-बॉक्स', 'में', 'शामिल', 'था', ':']**

FineLabels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’MusicalWork’, ’O’, ’O’, ’O’, ’O’]

18. **Fine Label**: **AerospaceManufacturer**

English

Tokens: [’the’, ’school’, ’trained’, ’volunteers’, ’from’, ’the’, ’local’, ’territorial’, ’units’, ’using’, ’luscombe’, ’seaplanes’, ’.’]

FineLabels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’AerospaceManufacturer’, ’O’, ’O’]

Hindi

Tokens: **['स्कूल', 'ने', 'स्थानीय', 'क्षेत्रीय', 'यूनिट्स', 'से', 'स्वयंसेवकों', 'को', 'लुसकोम्ब', 'सीप्लेन', 'का', 'उपयोग', 'करके', 'प्रशिक्षित', 'किया।']**

FineLabels: ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'AerospaceManufacturer', 'O', 'O', 'O', 'O']

19. **Fine Label**: **Medication/Vaccine**

English

Tokens: ['the', 'caffeine', 'level', 'is', 'normal', 'for', 'green', 'tea', 'and', 'it', 'can', 'be', 'drunk', 'throughout', 'the', 'day', '.']

FineLabels: ['O', 'Medication/Vaccine', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

Hindi

Tokens: **['कैफीन', 'का', 'स्तर', 'हरी', 'चाय', 'के', 'लिए', 'सामान्य', 'है', 'और', 'इसे', 'पूरा', 'दिन', 'पीया', 'जा', 'सकता', 'है।']**

FineLabels: ['Medication/Vaccine', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

20. **Fine Label**: **Scientist**

English

Tokens: ['the', 'fracture', 'strength', '(', 'or', 'micro', 'crack', 'nucleation', 'stress', ')', 'of', 'a', 'material', 'was', 'first', 'theoretically', 'estimated', 'by', 'alan', 'arnold', 'griffith', 'in', '1921', ':']

FineLabels: ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'Scientist', 'Scientist', 'Scientist', 'O', 'O', 'O']

Hindi

Tokens: **['एक', 'सामग्री', 'की', 'दरार', 'की', 'मजबूती', '(', 'या', 'सूक्ष्म', 'दरार', 'न्यूक्लियेशन', 'तनाव', ')', 'का', 'सिद्धांतिक', 'अनुमान', 'पहली', 'बार', 'एलेन', 'अर्नोल्ड', 'ग्रिफिथ', 'द्वारा', '1921', 'में', 'लगा', ':']**

FineLabels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’Scientist’, ’Scientist’, ’Scientist’, ’O’, ’O’, ’O’, ’O’, ’O’]

21. **Fine Label**: **SportsManager**

English

Tokens: [’rams’, ’assistant’, ’vic’, ’rapp’, ’was’, ’brought’, ’in’, ’as’, ’the’, ’running’, ’backs’, ’coach’, ’.’]

FineLabels: [’O’, ’O’, ’SportsManager’, ’SportsManager’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

Hindi

Tokens: **['रैम्स', 'के', 'मुख्य', 'कोच', 'विक', 'रैप', 'को', 'रनिंग', 'बैक्स', 'कोच', 'के', 'रूप', 'में', 'नियुक्त', 'किया', 'गया।']**

FineLabels: [’O’, ’O’, ’O’, ’O’, ’SportsManager’, ’SportsManager’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

22. **Fine Label**: **Clothing**

English

Tokens: [’the’, ’riders’, ’in’, ’the’, ’team’, ’that’, ’led’, ’this’, ’classification’, ’wore’, ’yellow’, ’caps’, ’.’]

FineLabels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’Clothing’, ’O’]

Hindi

Tokens: **['टीम', 'के', 'सवार', 'जिन्होंने', 'इस', 'वर्गीकरण', 'का', 'नेतृत्व', 'किया', ',', 'पीली', 'टोपी', 'पहनी।']**

FineLabels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’Clothing’, ’O’]

23. **Fine Label**: **ORG**

English

Tokens: [’irish’, ’legislators’, ’began’, ’to’, ’comment’, ’publicly’, ’from’, ’2003’, ’some’, ’tentatively’, ’suggesting’, ’legislation’, ’and’, ’some’, ’referring’, ’to’, ’catholic’, ’teachings’, ’.’]

FineLabels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’ORG’, ’O’, ’O’]

Hindi

Tokens: **['आयरिश', 'विधायक', '2003', 'से', 'सार्वजनिक', 'रूप', 'से', 'टिप्पणी', 'करने', 'लगे', ',', 'कुछ', 'संकोचपूर्वक', 'विधेयक', 'का', 'सुझाव', 'देते', 'हुए', 'और', 'कुछ', 'कैथोलिक', 'शिक्षाओं', 'का', 'संदर्भ', 'देते', 'हुए।']**

FineLabels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’ORG’, ’O’, ’O’, ’O’, ’O’, ’O’]

24. **Fine Label**: **MedicalProcedure**

English

Tokens: [’one’, ’form’, ’of’, ’treatment’, ’is’, ’cognitive’, ’behavioral’, ’therapy’, ’which’, ’promotes’, ’desensitization’, ’methods’, ’.’]

FineLabels: [’O’, ’O’, ’O’, ’O’, ’O’, ’MedicalProcedure’, ’MedicalProcedure’, ’MedicalProcedure’, ’O’, ’O’, ’O’, ’O’, ’O’]

Hindi

Tokens: **['उपचार', 'के', 'एक', 'रूप', 'में', 'संज्ञानात्मक', 'व्यवहार', 'चिकित्सा', 'शामिल', 'है', ',', 'जो', 'संवेदनहीनता', 'के', 'तरीकों', 'को', 'प्रोत्साहित', 'करती', 'है।']**

FineLabels: [’O’, ’O’, ’O’, ’O’, ’O’, ’MedicalProcedure’, ’MedicalProcedure’, ’MedicalProcedure’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

25. **Fine Label**: **Cleric**

English

Tokens: [’his’, ’method’, ’and’, ’that’, ’of’, ’sulpice’, ’have’, ’helped’, ’spread’, ’the’, ’habit’, ’of’, ’meditating’, ’beyond’, ’the’, ’cloister’, ’.’]

FineLabels: [’O’, ’O’, ’O’, ’O’, ’O’, ’Cleric’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

Hindi

Tokens: **['उनकी', 'पद्धति', 'और', 'सुल्पाइस', 'की', 'पद्धति', 'ने', 'ध्यान', 'करने', 'की', 'आदत', 'को', 'मठ', 'की', 'सीमाओं', 'से', 'परे', 'फैलाने', 'में', 'मदद', 'की', 'है।']**

FineLabels: ['O', 'O', 'O', 'Cleric', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

26. **Fine Label**: **WrittenWork**

English

Tokens: ['in', 'most', 'of', 'these', 'cases', 'local', 'newspapers', 'such', 'as', 'the', 'chicago', 'tribune', 'reported', 'on', 'the', 'arrests', 'and', 'subsequent', 'court', 'cases', '.']

FineLabels: ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'WrittenWork', 'WrittenWork', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

Hindi

Tokens: **['इन', 'अधिकांश', 'मामलों', 'में', 'स्थानीय', 'समाचारपत्रों', 'जैसे', 'कि', 'शिकागो', 'ट्रिब्यून', 'ने', 'गिरफ्तारियों', 'और', 'उसके', 'बाद', 'के', 'अदालती', 'मामलों', 'की', 'रिपोर्टिंग', 'की।']**

FineLabels: ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'WrittenWork', 'WrittenWork', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

27. **Fine Label**: **VisualWork**

English

Tokens: ['tidying', 'up', 'with', 'marie', 'kondo', 'and', 'love', 'is', 'blind', 'received', 'nominations', 'in', '2019', 'and', '2020', 'respectively', '.']

FineLabels: ['VisualWork', 'VisualWork', 'VisualWork', 'VisualWork', 'VisualWork', 'O', 'VisualWork', 'VisualWork', 'VisualWork', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

Hindi

Tokens: **['टाइडिंग', 'अप', 'विद', 'मैरी', 'कोंडो', 'और', 'लव', 'इज', 'ब्लाइंड', 'को', 'क्रमशः', '2019', 'और', '2020', 'में', 'नामांकन', 'प्राप्त', 'हुआ।']**

FineLabels: ['VisualWork', 'VisualWork', 'VisualWork', 'VisualWork', 'VisualWork',

'O', 'VisualWork', 'VisualWork', 'VisualWork', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

28. **Fine Label**: **Symptom**

English

Tokens: ['in', '2000s', 'although', 'he', 'played', 'many', 'matches', 'he', 'suffered', 'from', 'occupational', 'burnout', 'in', '2005', 'and', 'chronic', 'fatigue', 'syndrome', 'in', '2009', '.']

FineLabels: ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'Symptom', 'Symptom', 'O', 'O', 'O', 'Symptom', 'Symptom', 'Symptom', 'O', 'O', 'O']

Hindi

Tokens: ['2000', 'के', 'दशक', 'में', 'हालांकि', 'उन्होंने', 'कई', 'मैच', 'खेलें', ',', 'वह', '2005', 'में', 'व्यावसायिक', 'थकावट', 'और', '2009', 'में', 'दीर्घकालिक', 'थकान', 'सिंड्रोम', 'से', 'पीड़ित', 'रहे।']

FineLabels: ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'Symptom', 'Symptom', 'O', 'O', 'O', 'Symptom', 'Symptom', 'Symptom', 'O', 'O', 'O']

29. **Fine Label**: **OtherPER**

English

Tokens: ['thomas', 'heaphy', 'the', 'elder', '(', '1775', '–', '1835', ')', 'watercolourist', 'and', 'portrait', 'painter']

FineLabels: ['OtherPER', 'OtherPER', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

Hindi

Tokens: ['थॉमस', 'हीफी', 'द', 'एल्डर', '(', '1775', '–', '1835', ')', 'जलरंग', 'कलाकार', 'और', 'चित्रकार']

FineLabels: ['OtherPER', 'OtherPER', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

30. **Fine Label**: **Disease**

English

Tokens: [’the’, ’pills’, ’have’, ’been’, ’used’, ’to’, ’treat’, ’ulcerative’, ’colitis’, ’.’]

FineLabels: [’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’Disease’, ’Disease’, ’O’]

Hindi

Tokens: **['ये', 'गोलियाँ', 'अल्सरेटिव', 'कोलाइटिस', 'का', 'इलाज', 'करने', 'के', 'लिए', 'इस्तेमाल', 'की', 'गई', 'हैं।']**

FineLabels: [’O’, ’O’, ’Disease’, ’Disease’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’, ’O’]

31. **Fine Label**: **Drink**

English

Tokens: [’the’, ’national’, ’alcoholic’, ’drink’, ’is’, ’beer’, ’.’]

FineLabels: [’O’, ’O’, ’O’, ’O’, ’O’, ’Drink’, ’O’]

Hindi

Tokens: **['राष्ट्रीय', 'मादक', 'पेय', 'बीयर', 'है।']**

FineLabels: [’O’, ’O’, ’O’, ’Drink’, ’O’]