Fine-tuning Language Models for Triple Extraction with Data Augmentation

Yujia Zhang, Tyler Sadler, Mohammad Reza Taesiri, Wenjie Xu, Marek Z. Reformat
University of Alberta
{yujia10, tsadle, taesiri, wx4, reformat}@ualberta.ca

Abstract

Advanced language models with impressive capabilities to process textual information can more effectively extract high-quality triples, which are the building blocks of knowledge graphs. Our work examines language models' abilities to extract entities and the relationships between them. We use a diverse data augmentation process to fine-tune large language models to extract triples from the text. Fine-tuning is performed using a mix of trainers from HuggingFace and five public datasets, such as different variations of the WebNLG, SKE, DocRed, FewRel, and KELM. Evaluation involves comparing model output with test set triples based on several criteria, such as type, partial, exact, and strict accuracy. The obtained results outperform ChatGPT and even match or exceed the performance of GPT-4.

1 Introduction

Knowledge graphs (KGs) represent knowledge in a semantically rich and intuitive way, enabling one to better understand and utilize gathered information. A KG is a data structure representing real-world entities and the relationships between them in the format of a triple, e.g., ⟨*head entity, relation, tail entity* ⟩ or ⟨ *subject, predicate, object* ⟩ (Ji et al., 2021).

The majority of available knowledge is composed of unstructured textual data. The need to 'convert it' into a structured format via extracting entities and relationships between them drives the construction of KGs. Large language models, like ChatGPT or GPT-4, have a remarkable capacity for understanding and generating text. It makes them useful tools for automating the process of knowledge extraction from textual sources. They can capture nuances and complexities of language, allowing for a deeper comprehension of the text's meaning. Therefore, they can be employed to create KGs that accurately and fully capture complicated semantic relations and the meaning of texts.

Extracting triples from texts poses several challenges (Hofer et al., 2023). Finding accurate and comprehensive entities and representative relationships from the text can be difficult, especially with various language usage, implicit references, and context-dependent interpretations. Additionally, processing and analyzing enormous quantities of text can be computationally demanding and resource-intensive. Therefore, methods for capturing reliable contextual information are paramount for KG's growth and development. Advanced context-aware techniques must be developed to identify and separate contextual references, capture relationships, and identify implicit connections.

This work aims to tune large language models (LLMs) to perform triple extraction from text. We have conducted several experiments using various models and datasets of different quality and sizes. The construction of triples adhering to the DBpedia ontology format has been particularly interesting. The WebNLG dataset (Castro Ferreira et al., 2020), predominantly using the DBpedia vocabulary for its entities and properties or emulating its ontological style, serves as the basis for our training data.

We have introduced a set of procedures to generate various prompts, instructing models about different processes related to triple extraction and understanding. This has led to the augmentation of the original WebNLG data and the creation of various versions of training datasets.

Eleven models, each with seven billion parameters, have been trained. Their efficacy has been evaluated in comparison with GPT-3.5 and GPT-4 on WebNLG. Additionally, we have preliminary assessed larger models with thirteen, thirty, and thirty-three billion parameters and trained them similarly.

The ultimate objective is to propose and illustrate a training methodology capable of elevating domain-specific models to or beyond the proficiency of leading-edge models.

Proceedings of the 1st Workshop on Knowledge Graphs and Large Language Models (KaLLM 2024), pages 116–124

The findings of the work that constitute our contributions are:

- the reasonably sized large language models, such as ones with seven billion parameters, can be successfully tuned to extract triples from text;
- the proposed procedures to build a variety of prompts lead to the generation of enlarged and enhanced (enriched with information that improves training) datasets;
- small, fine-tuned models can outperform the baselines set up by GPT family models: ChatGPT and GPT-4.
- high-quality data is essential for the triple generation task; many datasets in the triple extraction space focus on extracting only specific relationships from text rather than all possible relationships or do not follow particular vocabulary, like DBpedia ontology.

2 Related Work

In the field of triple extraction, LSTM is a conventional technique to explore. Seq2Rdf (Liu et al., 2018) employs an LSTM-based sequence-to-sequence model to map natural language text to RDF triples in one step, using pre-trained word and knowledge graph embeddings for initialization. However, it is limited to extracting single triples and cannot handle multi-triple extraction. The ChatIE framework (Wei et al., 2023) achieves zero-shot information extraction by promoting ChatGPT, without requiring any labeled data for training. It allows interactively querying the model to extract structured information piece by piece in a multi-turn conversational format. The ChatIE relies on LLM like ChatGPT which is not open source. The performance depends heavily on how well the prompts are engineered and provides many details.

The Head to Tail benchmark (Sun et al., 2023) provides a systematic way to evaluate how knowledgeable LLM are about facts in diverse domains(movies, books, academics). The benchmark is still limited in size and diversity compared to the vast world knowledge, 18k QA pairs may not comprehensively cover all entity types, relationships, and knowledge domains. Few-shot learning with GPT-3 (Wadhwa et al., 2023) achieves state-of-the-art performance on standard relation extraction

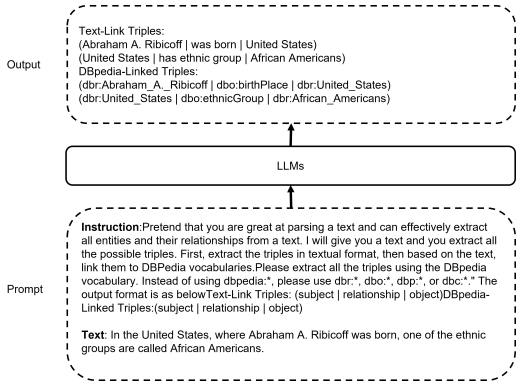

Figure 1: Example of triple extraction prompt workflow

datasets, surpassing existing fully supervised models. Fine-tuning Flan-T5 on explanations generated by GPT-3 further enhances performance. Treating relation extraction as a text-generation task provides flexibility in expressing entities and relations. However, GPT-3 is opaque, not open source, and significantly costly.

3 Problem and Experimental Setup

The paper focuses on extracting information from plain text. It is the task of building triples of the form $\langle$*subject, predicate, object*$\rangle$ based on the content of a sentence. Triple extraction is a domain-independent task. Two entities of a triple, i.e., subject and object, appear in the text, while a relation between these two entities is often deduced by 'understanding the meaning' of the sentence. All the components of a triple are extracted at the same time.

Here is a more formal description of the task. Given a set of sentences $D := \{w_1, w_2, ..., w_n\}$, we want to obtain a set of facts built from and based on these sentences. Let this set be $Facts := \{fact_1, fact_2, ..., fact_n\}$, and each fact is denoted as $\langle s, p, o \rangle, s \in S, p \in P, o \in O$, where S, P, O are sets of subjects, predicates, and objects respectively.

These triples are the basic units of knowledge graphs, resulting from the development of the Semantic Web concept. The classes (types of entities) and properties (relationships and attributes) used to describe triples' components are defined using ontologies. One of the most well-known ontologies is the one used by DBpedia (Lehmann et al., 2015).

Within the DBpedia dataset, triples are generated and represented using the DBpedia ontology as the schema. This ontology consists of 320 classes organized into a subsumption hierarchy and 1650 dis-

Llama2	You are an AI assistant who is an expert in knowledge graphs. You will be given an instruction and text. Generate a response to appropriately complete the instruction's request. {**instruction**}{**input**}{**output**}
LLongOrca	Below is an instruction that describes a task, paired with an input that provides further context. Write a output that appropriately completes the request. {**instruction**}{**input**}{**output**}
other models	### Instruction:{**instruction**} ### Input: {**input**} {**output**}

Table 1: Generic prompt template for different models.

tinct properties describing relations between them. The subsumption hierarchy is purposefully maintained relatively shallow, with a maximum depth of five to accommodate use cases where the ontology is traversed or visualized. Online browsing of the entire DBpedia ontology is available at [1].

3.1 Datasets

WebNLG The WebNLG corpus (Castro Ferreira et al., 2020) is made up of sets of triplets describing facts (entities and their relationships) and the matching facts expressed in natural language, in other words, text from which the triples are extracted. It includes 13,211 training data and 2,155 test data.

FewREL Few-Shot Relation Classification Dataset (FewRel)(Han et al., 2018) composes 70,000 instances from Wikipedia and 100 relations. The dataset is divided into three subsets: training set (64 relations), validation set (16 relations), and test set (20 relations).

DocRED Document-Level Relation Extraction Dataset (DocRED) (Yao et al., 2019) is created from Wikipedia and Wikidata in relation extraction data. Annotated on 5,053 Wikipedia articles, DocRED comprises 132,375 entities and 56,354 relational facts. The collection offers large-scale distantly supervised data over 101,873 documents in addition to the human-annotated data.

KELM The English Wikidata KG and the corresponding natural text sentences make up the large-scale synthetic corpus known as KELM(Agarwal et al., 2020). It has roughly 15 million artificially generated sentences produced by a refined T5 model. A list of triples of the format [subject, relation, object] is contained in each linearized KG graph in KELM. A subset of KELM, named KELM-sub, is used which contains 400,000/5,000 samples as train/test set.

SKE Baidu has released a Chinese dataset called SKE2019. The train set contains 194,747 sentences, whereas the validated set contains 21,639 sentences. SKE21 (Xie et al., 2021) has been released by manually labeling 1150 sentences from the test set with 2765 annotated triples. It contains 194,747 training data, 21,639 validation data, and 1,150 testing data. [2]

3.2 Large Language Models

LLMs like ChatGPT and GPT-4, pre-trained on a large-scale corpus, are composed of decoder modules based on the Transformer design, which incorporates a self-attention mechanism. However, it is difficult to conduct further research due to the close-source nature of models. Then, open-source decoder-only LLMs like Alpaca and Vicuna are released, which are fine-tuned based on LLaMA (Touvron et al., 2023a) and achieve competitive performance with ChatGPT and GPT-4.

ChatGPT-3.5 and GPT-4 Human-like conversations are the main purpose of ChatGPT, an advanced LLM created by OpenAI. To improve ChatGPT's alignment with human tastes and values, it uses RLHF (Christiano et al., 2017) during the fine-tuning process. GPT-4, an advanced big language model created by OpenAI, is expanding on the achievements of its forerunners, such as GPT-3 and ChatGPT.

Vicuna-13B (Chiang et al., 2023),Wizard (Xu et al., 2023), Orca (Mukherjee et al., 2023), **LLaMA** (Touvron et al., 2023a)(Touvron et al., 2023b), LlongOrca (Lian et al., 2023), **SOLAR** 10.7B (Kim et al., 2023)**Mixtral** Mixtral[3], Mistral mode[4], **Platypus** Platypus-30B (Lee et al., 2023) is the open-source model we choose from HuggingFace.

[1] http://mappings.dbpedia.org/server/ontology/classes/

[2] http://ai.baidu.com/broad/download?dataset=sked

[3] https://mistral.ai/news/mixtral-of-experts/

[4] https://huggingface.co/ignos/Mistral-T5-7B-v1

Data Format			
Data Augmentation (name)	Parts of prompt		Response
	instruction	**input**	**output**
Text2triples	Think of yourself as efficient in deconstructing a text and precisely identifying all the entities and their interrelations. I'll furnish you with a text and your job is to gather all potential triples, adhering to the pattern: (subject|relationship|object).	Sentence	Triples
Explanation	"Assume you're highly competent in scrutinizing a piece of text and successfully distilling all its entities along with their connections. I'll provide a text, and you are to extract every possible triplet, following the convention: (subject|relationship| object). Detail the entire process systematically."	Sentence	To extract triplets from the given text, we need to identify the subject, predicate, and object. Subject: "Aarhus Airport" Predicate: "cityServed" Object: "Aarhus, Denmark" The property "cityServed" is derived from the context of the sentence, where it implies that the airport serves the city of Aarhus. Therefore, here is the answer in the correct format: Aarhus_Airport | cityServed| "Aarhus, Denmark")
Triples2text	Picture yourself as an expert in scrutinizing a text, effectively extracting all entities and their relationships and then constructing text based on the given triples. Once I supply you with triples in the (subject|relationship|object) format, your duty is to reexamine these triples and create text that imparts their semantic interpretation.	Triples	Sentence
Reflection	Picture yourself as being highly skilled in text dissection, with the ability to efficiently identify all entities and their ties. When provided a text along with triples in the (subject|relationship|object) format, you are to check these triples in light of the text and correct any inaccuracies.	Sentence Triples	Corrected triples

Table 2: The overall Data Augmentation Tricks

3.3 Prompt Engineering & Data Preparation

Prompt engineering is an in-context method for learning language models. In a nutshell, a prompt is a sequence of natural language inputs for a model, consisting of an instruction, context, and input text. The instruction guides the model to perform a specific task, while the context provides additional information; the input text is the text to be processed by the model. An example of the triple extraction prompt is shown in Figure 1.

In this work, we used different prompt formats for various models, ensuring that both fine-tuning and inference employed the same prompt format. The three types of prompts are detailed in Table 1. The components {**instruction**}, {**input**}, and {**output**} are replaced with information/data specific to the proposed Data Formats, Table 2.

The experiments have been conducted with the training datasets built with different versions of Data Formats. Such an approach allowed us to increase the size of training datasets by 3- and 4-fold. The process of building different datasets is illustrated at the top of Figure 2. Examples of data formats are included in Table 2. Each format has its style of the *instruction*, *input*, as well as *output*. The tasks associated with each Data Format differed from explaining the extraction process via reconstructing a sentence from triples to evaluating triples. The data formats were used to construct various Training Datasets, Table 3.

The first Training Dataset is called **WebNLG-combined dataset**. It contains 39,633 entries in three categories/subsets, each of 13,211 entries. The first subset includes *Test2triples*, i.e., sets of sentences together with the triples extracted from them. The second subset is the extension of the first one. We have added *Explanation* of the triple extraction process. The explanations were generated by prompting GPT-3.5 with the input text and the ground truth triples to elucidate the extraction process. The explanations comprise entity identification, property analysis, source derivation, entity relationships, and the resultant triples. The third is *Triple2text* subset. It sets the ground truth triples as the model input and the original text as the target output. The aim is to enhance reasoning capabilities and improve triple generation performance.

The second generated Training Data is named **WebNLG-combined-with-reflections** with 52,844 entries. We have extended the WebNLG-combined dataset with so-called *Reflection* data. These data were generated by a *Vicuna* model previously trained for the triple extraction task using *Test2triples* and *Explanation*. The model was fed with the text and triples generated from it, and the task was either amending the triples or confirming their correctness. The anticipated output was either

Training Dataset Name	Used Data Format(s)	Size
WebNLG (original)	*Text2triples*	N
WebNLG-combined	*Text2triples + Explanations + Triples2text*	3*N
WebNLG-combined-with-reflections	*Text2triples + Explanations + Triples2text + Reflection*	4*N
WebNLG-reflections-updated-instructions	*Text2triples + Explanations + Triples2text + Reflection* + new_instructions	4*N

Table 3: Variants of WebNLG training data

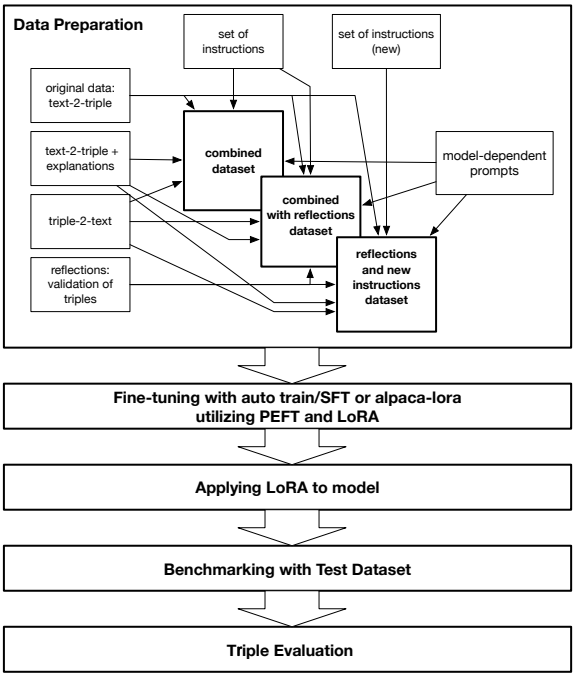

Figure 2: Experimental Workflow

a confirmation that a given input triple was accurate or its correct version.

For both datasets mentioned above, the **instruction** was randomly selected from the previously generated set of twenty distinct instructions. These instructions were a mix of human-authored instructions and variations generated by GPT-4 to enhance diversity. All instructions underwent thorough evaluation before they were used.

The **WebNLG-reflections-updated-instructions** dataset was the WebNLG-combined-with-reflections dataset when a new set of instructions was used. This time, there are eleven instructions: ten newly constructed and one from the original set. Again, this new set of instructions is a mixture of human-written and rephrased by GPT-4.

3.4 Overall Experiments Setup

The workflow of experimental steps and some details about the components forming different Training Datasets are shown in Figure 2. Once the datasets were prepared, the models have been tuned and benchmarked using the testing dataset. The final step was an evaluation of the results (for details, see next section).

To prepare models for the process of triple extraction, we utilized HuggingFace libraries to perform supervised finetuning utilizing Parameter-Efficient Finetuning (PEFT) (Liu et al., 2022) and Low-Rank Adaptation (LoRA) (Hu et al., 2021) on the WebNLG dataset. We used two prewritten trainers, *finetune script* from *alpaca-Lora* and *autotrain-advanced* from HuggingFace. The *finetune script* was slightly modified to change evaluation steps and to ensure the graphics processing unit (GPU) cache was cleared after all evaluations and checkpoints were saved. All models were trained using two Nvidia 3090 24GB GPUs and a cutoff length 1024, with varying configurations of packages and datasets based on the trainer used.

For the *finetune script*, we set an approximately 85:15 split between training and validation data. The validation set size is 6,000 for *WebNLG-combined* and 8,000 for *WebNLG-combined-with-reflections*.

For *autotrain-advanced*, Supervised Fine-tuning (SFT) Trainer is used from the Transformer Reinforcement Learning (TRL) package that is included as an option for training in autotrain-advanced (von Werra et al., 2020). The *WebNLG-reflections-updated-instructions* dataset is used. It contained different instructions for each training task, including additional details about formatting triples and better explaining the model's role.

We trained a collection of eleven models chosen based on relative performance on the HuggingFace LLM leaderboard, and compare their performance between each other and GPT-4. After training, the LoRA weights are combined with the base model to obtain our fine-tuned model output. These exported weights are used to run inference on the model.

4 Evaluation Procedure and Results

4.1 Evaluation Procedure

The evaluation framework comprises two phases: Inference, generating the model's output on the test set, and evaluation, comparing this output against

	Type			Partial			Exact			Strict		
Model	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
GPT-4 50 samples	0.706	0.729	0.714	0.684	0.707	0.692	0.651	0.668	0.657	0.640	0.652	0.645
GPT-4-0314	0.693	0.711	0.700	0.668	0.688	0.675	0.634	0.649	0.640	0.626	0.634	0.629
ChatGPT-3.5-2023	0.592	0.610	0.599	0.570	0.588	0.577	0.533	0.548	0.539	0.521	0.532	0.525
GPT-4 Full	0.567	0.624	0.587	0.536	0.580	0.552	0.478	0.506	0.488	0.455	0.482	0.465
Vicuna-7b	0.715	0.729	0.721	0.702	0.714	0.706	0.683	0.693	0.687	0.680	0.689	0.683
WizardLM-7b	0.700	0.715	0.706	0.688	0.701	0.693	0.671	0.682	0.675	0.667	0.677	0.671
Orca-mini-7b	0.683	0.700	0.690	0.670	0.686	0.677	0.652	0.664	0.657	0.647	0.658	0.652
Orca-mini-2-7b	0.711	0.726	0.717	0.698	0.710	0.703	0.681	0.690	0.684	0.677	0.687	0.681
Orca-mini-3-7b	**0.746**	**0.762**	**0.753**	**0.732**	**0.746**	**0.738**	**0.715**	**0.726**	**0.719**	**0.712**	**0.723**	**0.717**
Llama-2-7b	0.705	0.714	0.708	0.689	0.698	0.693	0.669	0.677	0.673	0.666	0.673	0.669
Llama-2-chat-7b	0.685	0.700	0.691	0.670	0.684	0.675	0.650	0.660	0.654	0.645	0.654	0.649
LlongOrca-7b	0.710	0.722	0.715	0.697	0.707	0.701	0.680	0.689	0.684	0.677	0.685	0.680
SOLAR-Instruct-10b	0.729	0.741	0.734	0.716	0.727	0.720	0.699	0.708	0.703	0.697	0.705	0.700
Mistral-t5-7b	0.731	0.746	0.738	0.716	0.729	**0.721**	0.697	0.708	0.702	0.695	0.704	0.698
Mixtral-8x7b	0.730	0.739	0.734	0.716	0.725	0.720	0.699	0.706	0.702	0.696	0.702	0.698
Vicuna-33b	**0.750**	**0.762**	**0.755**	**0.738**	**0.749**	**0.742**	**0.723**	**0.732**	**0.727**	**0.720**	**0.729**	**0.724**
Platypus-30b	**0.747**	**0.762**	**0.753**	**0.732**	**0.746**	**0.738**	**0.715**	**0.726**	**0.720**	**0.713**	**0.724**	**0.718**

Table 4: WebNLG-reflections-updated-instructions performance results

ground truth triples. All models were benchmarked with a maximum token limit of 1,024, and the output was generated without streaming. For evaluation, the numerical results such as precision, recall, and F1, and saved as the output file. The test set includes the same instructions in our training data and includes 2,155 instances of directly extracting triples from text.

The scores are calculated using the evaluate package (Segura-Bedmar et al., 2013). It calculates metrics based on four different criteria. First is *type evaluation (TE)* where only the tags must match to be considered correct. These tags are SUB, PRED, and OBJ for the subject, predicate, and object. *Partial evaluation (PE)* requires the triples to match partially or completely, irrespective of tag, to be considered partially or completely correct. *Exact evaluation (EE)* requires the triples to match exactly, irrespective of tag, to be considered correct. *Strict evaluation (SE)* requires both the triples and tag to match to be considered correct. Each evaluation type assigns a label of correct (COR), incorrect (InCOR), missed (MIS), or spurious (SPU), based on the triples and tags. Partial (PAR) is assigned only for the partial evaluation type. MIS and SPU are across all evaluation types, with MIS being assigned for each part of a reference triple when there is no matching candidate, and SPU assigned for each part of a candidate triple when there is no matching reference. The following formulas are calculations of precision (P), recall (R), and F1. The type and partial scores are calculated with the "partial" formulas and exact and strict scores are calculated with the "exact" formulas:

$$\begin{aligned} Possible &= COR + InCOR + PAR + MIS \\ &= TP + FN \\ Actual &= COR + InCOR + PAR + SPU \\ &= TP + FP \end{aligned}$$

$$P_{TE|PE} = \frac{COR + 0.5 * PAR}{Actual}$$

$$R_{TE|PE} = \frac{COR + 0.5 * PAR}{Possible}$$

$$\mathrm{P}_{EE|SE} = \frac{COR}{Actual} = \frac{COR}{COR + InCOR + SPU}$$

$$\mathrm{R}_{EE|SE} = \frac{COR}{Possible} = \frac{COR}{COR + InCOR + MIS}$$

4.2 Results

WebNLG Dataset. The obtained results for the fine-tuned models are included in Table 4. It can be observed that small 7b models *Orca-mini3-7b* and *Mistral-t5-7b* have the best performances even when compared with GPT-4. The *Orca-mini3-7b* model achieved the highest F1 scores for all evaluations, outperforming all 7b models in our comparative analysis.

Small variations have been observed between training methods and datasets. In general, models show slight improvement from WebNLG-combined to WebNLG-combined-with-reflections and then to WebNLG-reflections-updated-instructions. Additionally, modifying the instructions shows a decrease in training loss. GPT models had a bigger drop in performance going to the exact and strict metrics compared to our models, which resulted in our models performing relatively better on the exact and strict metrics.

Ablation Study. We performed ablation studies to evaluate the impact of different data augmentation strategies on the performance of these models. Figure 3 shows the effects of various data augmentation techniques on the mod-

els' performance. We show the performance results obtained for two models – *Orca* and *Vicuna* – and four Training Datasets: the original WebNLG dataset, the WebNLG-combined dataset, the WebNLG-combined-with-reflections dataset, and the WebNLG-reflections-updated-instructions dataset, Table 3. We report the precision, recall, and F1 values for the most demanding task of generating triples identical to those provided as the target. It is easily seen that the results obtained for the last Training Dataset are the best.

Other Datasets. Two models *Orca-mini-3* and *Llama-2-13b* have been finetuned on different datasets, Table 5. The best scores have been obtained for the **KELM** dataset. The *Llama-2-13b* finetuned on another dataset **DocRED** performed very poorly and was completely unable to learn proper formatting of triples.

The main issue with inference on other data is related to the type of triple properties and how many triples are extracted from a single sentence. For example, the analysis of the DocRED dataset revealed that it is focused mainly on such relations as *country* and *location* while ignoring any other relations. In DocRed, a few triples are extracted from paragraph sentences. There is much looping in the models' output; models do not efficiently learn triple formats. Some outputs were of the form (subject | predicate | object). Further, there are only about 3,000 entries in annotated training data. For yet another dataset – **FewRel** – the issue seems to be related to the model not knowing when to generate triples following the DBpedia and when using Wikidata formats.

5 Discussion and Limitations

The obtained results and their analysis have led to a few observations that confirmed known facts about tuning large models and allowed to draw some new ones. We can categorize them into three parts: data size, model selection, and interaction with a model (prompt and data preparation).

Size and Quality of Datasets. It is a well-known fact that larger datasets lead to better results. Such an obvious statement is also true for the triple extraction process. It is seen in Table 5. The results obtained for KELM data – 400,000 samples in the training set – confirm that. The model was tuned with a simple prompt containing text-2-triple and instructions. Comparing that with our primary focused data, WebNLG, which includes only 13,211 training datasets, shows a significant advantage of large datasets.

Once we collected results for the other two datasets – DocRED and FewRel - we investigated the content of the training datasets. It has become apparent that the reference triples that were supposed to be constructed from sentences were of poor quality: limited to a few relations, incoherent structure, a limited number of triples (quite often just one) form small paragraphs.

Model Selection/Multilingual Triples. In our experiments, one of the datasets – SKE – is a set of Chinese sentences and extracted from them triples. The difference in results obtained from *orca-mini-3-7b* and *llama-2-13b* is very large. A quick investigation revealed that the dataset used to train the *orca-mini-3-7* model contained a large amount of Chinese text. Again, it confirms a commonsense fact that if a language model is not exposed to a text in a given language, its performance, related to this language, is not satisfactory.

Prompt and Data Preparation. The most interesting and important observation coming from our experiments is a high significance of the creative approach to constructing prompts and 'augmentation' of the training datasets.

As indicated earlier, the task of extracting triples from WebNLG data involves the usage of DBpedia vocabulary. In particular, properties/relations of the extracted triples have/should be in the DBpedia format. The WebNLG dataset has been analyzed to ensure the training data is of high quality. DBpedia ontology has been used to determine if the triples/relations were consistent.

The consistent structure of triples is essential so the model can effectively learn how to form triples properly. Exposure to different properties is also of high importance. The properties seen in the training and testing sets overlapped, with thirty-six properties unique to the test set. All properties were checked to ensure they were present in DBpedia.

A small amount of training data, just 13,211, has forced us to generate larger datasets from the original set via setting different tasks related to processing and extraction of triples. Section 3 details how various versions of Training Datasets were created. We enhanced the data with explanations of triple generation processes generated by GPT-3.5 and previously tuned model, generation of sentences based on sets of triples, and simple evaluation of extracted triples. These activities have improved

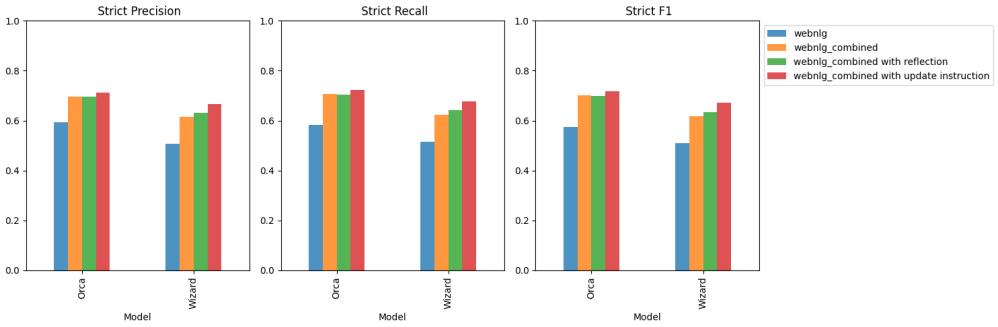

Figure 3: Results of ablation studies on two modes: Orca and Wizard

		Type			Partial			Exact			Strict		
Data	Model	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
SKE	orca-mini-3	0.828	0.828	0.828	0.829	0.829	0.829	0.829	0.829	0.829	0.827	0.827	0.827
	llama-2-13b	0.129	0.127	0.127	0.130	0.128	0.129	0.127	0.127	0.127	0.124	0.124	0.124
DocRED	orca-mini-3	0.057	0.054	0.052	0.050	0.050	0.048	0.031	0.031	0.030	0.024	0.025	0.024
	llama-2-13b	0.096	0.037	0.051	0.051	0.022	0.028	0.002	0.002	0.002	0.002	0.002	0.002
FewRel	orca-mini-3	0.314	0.402	0.342	0.354	0.425	0.376	0.312	0.362	0.327	0.240	0.286	0.254
	llama-2-13b	0.304	0.378	0.325	0.344	0.405	0.361	0.297	0.340	0.310	0.224	0.263	0.236
KELM	orca-mini-3-7b	0.867	0.899	0.879	0.848	0.873	0.857	0.823	0.841	0.830	0.820	0.837	0.826
	Llama-2-13b	0.861	0.865	0.852	0.825	0.836	0.825	0.779	0.796	0.785	0.769	0.786	0.776
raw_Webnlg	orca-mini-3-7b	0.618	0.638	0.626	0.598	0.615	0.605	0.574	0.588	0.579	0.593	0.583	0.575
	Llama-2-13b	0.62	0.637	0.626	0.602	0.618	0.608	0.581	0.593	0.586	0.577	0.588	0.581

Table 5: Performance on other datasets

our best model's performance, i.e., *orca-mini-3-7b*.

Limitations There are some limitations of fine-tuned models. They hallucinated on occasion, especially when they generated responses for more well-known topics, such as when we asked them to generate a response to the Jeff Bezos Wikipedia article. The models frequently hallucinated the birthplace of Bezos, providing false information about the location. Also, models had looping issues, where they would continually generate output until they reached the token limit.

6 Conclusion

The paper aims to investigate different scenarios of a triple extraction task. Various models and a few datasets have been used in the experiments. A prime contribution is the development of a procedure/methodology for augmenting the original dataset. The additions included several tasks indirectly related to the triple extraction process: explaining the extraction steps, reconstructing sentences from triples, and determining the correctness of extracted triples. It resulted in enlarged training datasets (3- or 4-fold). As an outcome, the performance of 7b tuned models is comparable to or even better than that of well-known models from the GPT family.

The applied procedures concentrated on generating triples containing elements compatible with a specific vocabulary, in our case, DBpedia. While our models suffer from occasional looping and hallucinations, they effectively extract triples following DBpedia ontology from sentences. The results demonstrate that achieving and exceeding GPT performance with fine-tuned models is possible without large datasets.

References

Oshin Agarwal, Heming Ge, Siamak Shakeri, and Rami Al-Rfou. 2020. Knowledge graph based synthetic corpus generation for knowledge-enhanced language model pre-training. *arXiv preprint arXiv:2010.12688*.

Thiago Castro Ferreira, Claire Gardent, Nikolai Ilinykh, Chris van der Lee, Simon Mille, Diego Moussallem, and Anastasia Shimorina. 2020. The 2020 bilingual, bi-directional WebNLG+ shared task: Overview and evaluation results (WebNLG+ 2020). In *3rd International Workshop on Natural Language Generation from the Semantic Web*, pages 55–76, Dublin, Ireland (Virtual). Association for Computational Linguistics.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep

reinforcement learning from human preferences. *Advances in neural information processing systems*, 30.

Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. Fewrel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. *arXiv preprint arXiv:1810.10147*.

Marvin Hofer, Daniel Obraczka, Alieh Saeedi, Hanna Köpcke, and Erhard Rahm. 2023. Construction of knowledge graphs: State and challenges. *arXiv preprint arXiv:2302.11509*.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *CoRR*.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and S Yu Philip. 2021. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE transactions on neural networks and learning systems*, 33(2):494–514.

Dahyun Kim, Chanjun Park, Sanghoon Kim, Wonsung Lee, Wonho Song, Yunsu Kim, Hyeonwoo Kim, Yungi Kim, Hyeonju Lee, Jihoo Kim, et al. 2023. Solar 10.7 b: Scaling large language models with simple yet effective depth up-scaling. *arXiv preprint arXiv:2312.15166*.

Ariel N Lee, Cole J Hunter, and Nataniel Ruiz. 2023. Platypus: Quick, cheap, and powerful refinement of llms. *arXiv preprint arXiv:2308.07317*.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195.

Wing Lian, Bleys Goodson, Guan Wang, Eugene Pentland, Austin Cook, Chanvichet Vong, and "Teknium". 2023. Llongorca7b: Llama2-7b model instruct-tuned for long context on filtered openorcav1 gpt-4 dataset. https://https://huggingface.co/Open-Orca/LlongOrca-7B-16k,.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. 2022. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *Advances in Neural Information Processing Systems*, 35:1950–1965.

Yue Liu, Tongtao Zhang, Zhicheng Liang, Heng Ji, and Deborah L. McGuinness. 2018. Seq2rdf: An end-to-end application for deriving triples from natural language text.

Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. 2023. Orca: Progressive learning from complex explanation traces of gpt-4.

Isabel Segura-Bedmar, Paloma Martínez Fernández, and María Herrero Zazo. 2013. Semeval-2013 task 9: Extraction of drug-drug interactions from biomedical texts (ddiextraction 2013). Association for Computational Linguistics.

Kai Sun, Yifan Ethan Xu, Hanwen Zha, Yue Liu, and Xin Luna Dong. 2023. Head-to-tail: How knowledgeable are large language models (llm)? a.k.a. will llms replace knowledge graphs?

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aur'elien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models. *ArXiv*, abs/2302.13971.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models, 2023. *URL https://arxiv.org/abs/2307.09288*.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, and Shengyi Huang. 2020. Trl: Transformer reinforcement learning. https://github.com/huggingface/trl.

Somin Wadhwa, Silvio Amir, and Byron C. Wallace. 2023. Revisiting relation extraction in the era of large language models.

Xiang Wei, Xingyu Cui, Ning Cheng, Xiaobin Wang, Xin Zhang, Shen Huang, Pengjun Xie, Jinan Xu, Yufeng Chen, Meishan Zhang, Yong Jiang, and Wenjuan Han. 2023. Zero-shot information extraction via chatting with chatgpt.

Chenhao Xie, Jiaqing Liang, Jingping Liu, Chengsong Huang, Wenhao Huang, and Yanghua Xiao. 2021. Revisiting the negative data of distantly supervised relation extraction. *arXiv preprint arXiv:2105.10158*.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. 2023. Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*.

Yuan Yao, Deming Ye, Peng Li, Xu Han, Yankai Lin, Zhenghao Liu, Zhiyuan Liu, Lixin Huang, Jie Zhou, and Maosong Sun. 2019. Docred: A large-scale document-level relation extraction dataset. *arXiv preprint arXiv:1906.06127*.