

ReqBrain: Task-Specific Instruction Tuning of LLMs for AI-Assisted Requirements Generation

Mohammad Kasra Habib, Daniel Graziotin, and Stefan Wagner

Abstract—Requirements elicitation and specification remains a labor-intensive, manual process prone to inconsistencies and gaps, presenting a significant challenge in modern software engineering. Emerging studies underscore the potential of employing large language models (LLMs) for automated requirements generation to support requirements elicitation and specification; however, it remains unclear how to implement this effectively. In this work, we introduce ReqBrain, an AI-assisted tool that employs a fine-tuned LLM to generate authentic and adequate software requirements. Software engineers can engage with ReqBrain through chat-based sessions to automatically generate software requirements and categorize them by type. We curated a high-quality dataset of ISO 29148-compliant requirements and fine-tuned five 7B-parameter LLMs to determine the most effective base model for ReqBrain. The top-performing model, Zephyr-7b-beta, achieved 89.30% F1 using the BERT score and a FRUGAL score of 91.20 in generating authentic and adequate requirements. Human evaluations further confirmed ReqBrain’s effectiveness in generating requirements. Our findings suggest that generative AI, when fine-tuned, has the potential to improve requirements elicitation and specification, paving the way for future extensions into areas such as defect identification, test case generation, and agile user story creation.

Index Terms—Requirements Elicitation, Requirements Specification, AI, AI-Assisted Requirements Elicitation, AI-assisted Requirements Generation, Large Language Model (LLM), Requirement Engineering, Deep Learning

I. INTRODUCTION

Requirements elicitation and specification is a continuous and fundamental activity in software development [1]. Despite its importance, the process is challenging, manual, and labor-intensive. A significant barrier is tacit knowledge – information held by a stakeholder but not explicitly shared with the requirements engineer [2]–[4] – which leads to incomplete requirements, a major pain in requirements engineering [5]. Clients often struggle to translate objectives into quantifiable requirements, resulting in misunderstandings that may cause deficient or missing critical requirements [2], [6] and ultimately produce a product lacking essential functionalities [7]. Natural language requirements lack structured syntax, leading to ambiguity, complexity, and vagueness, which makes understanding difficult for all stakeholders. [1], [8].

There is research leveraging AI to explore improving elicitation and specification activities. Earlier approaches, however,

were constrained by the limitations of traditional AI techniques: often narrow in focus, lacking cross-domain knowledge, required extensive and carefully structured input, and struggling to capture the complexities of human communication [9]. As a result, these tools provided only limited support in real-world scenarios [10]. Recent advances in AI enable the understanding of complex context, relationships, and domain knowledge, offering opportunities to proactively support requirements engineers.

In this study, we focus on an AI-assisted requirements generation approach and concentrate on the early phases of requirements engineering: elicitation and specification, with the potential for future work to expand into a broader range of requirements-related tasks. Whereas *elicitation* implies extracting unexpressed requirements, we define *generation* as the creation of requirements without prior confirmation of their alignment with stakeholder needs.

To support this approach, we propose using large language models (LLMs) to generate software requirements. LLMs trained on large datasets offer a broad cross-domain knowledge base that can support requirements elicitation and specification [11], [12]. However, general-purpose LLMs might require fine-tuning as they are not specifically designed to generate authentic and adequate requirements, which is essential for overcoming the labor-intensive manual process and ensuring adherence to established requirements engineering standards.

We consider generated requirements to be *authentic* if they are *indistinguishable* from those written by humans in terms of clarity, coherence, relevance, realism, and implementability. Furthermore, with *adequate*, we refer to four dimensions in AI-generated requirements: (1) *ISO 29148-compliant* [13], (2) *consistent* with, (3) *missing* from, and (4) *enhancing the overall completeness* of, a given requirements specification.

With that in mind, we introduce ReqBrain (**Requirements Brain**), a fine-tuned LLM and tool to generate authentic and adequate requirements to support the elicitation and specification phases of requirements engineering.

To achieve ReqBrain, we employ task-specific instruction tuning¹. We prefer fine-tuning over prompt engineering due to its ability to improve LLM performance on software engineering tasks and enhance context-specific performance [14]. It enables models to internalize task nuances, increasing usability for non-experts [15] and reducing computational overhead [16]. Moreover, it addresses limitations such as prompt length

M. K. Habib and S. Wagner are with the Chair of Software Engineering, Technical University of Munich, Heilbronn, Germany (e-mail: kasra.habib@tum.de; stefan.wagner@tum.de).
ORCID: M. K. Habib 0000-0002-1272-9873, S. Wagner 0000-0002-5256-8429.

D. Graziotin is with the Institute of Information Systems, University of Hohenheim, Stuttgart, Germany (e-mail: graziotin@uni-hohenheim.de).
ORCID: 0000-0002-9107-7681.

¹*Task-specific instruction tuning* is a supervised fine-tuning method (for more details, see Section II-A). In this paper, we use the term “*fine-tuning*” as a shorter alternative.

restrictions [16], the risk of knowledge conflict² [17], and reliance on advanced domain expertise [18] to generate requirements.

Our objective is to assess how fine-tuning affects large language models (LLMs) in generating authentic and adequate requirements. To achieve our objective for the potential of AI-assisted requirements generation employing ReqBrain, we explore the following research questions:

RQ1: How effectively does a fine-tuned large language model generate authentic requirements?

To address this research question, we split it into the following actionable sub-questions:

RQ1.1: Which fine-tuned large language model has the highest potential to generate authentic requirements?

We benchmark several LLMs, after fine-tuning, using automated NLP metrics. It is crucial to select the model with the highest potential for authentic requirements to reduce the need for exhaustive human evaluation across different models.

RQ1.2: Does fine-tuning improve the generation of authentic requirements?

We aim to understand whether fine-tuning can match or exceed the performance of untuned general-purpose commercial models, in particular ChatGPT-4o, for authentic requirements generation.

RQ1.3: Are generated requirements perceived as authentic by humans?

Human evaluators evaluate if ReqBrain generates requirements that are indistinguishable from those authored by humans. This is crucial because achieving human quality standards is fundamental for establishing trustworthiness, user confidence, and integration in development processes.

RQ2: How effectively does a fine-tuned LLM generate adequate requirements?

Human evaluators assess whether a fine-tuned LLM, ReqBrain, generates adequate requirements. Ensuring that ReqBrain meets the four dimensions of adequate requirements – ISO 29148-compliant, consistent with, missing from, and enhancing the overall completeness of a given requirements specification – is critical for generating initial high-quality requirements, saving structuring effort and time to specify requirements unambiguously, preventing costly development issues due to incomplete specifications or identifying potential gaps.

Our work contributes to advancing AI-assisted requirements generation by providing:

1. A novel method and tool for the generation of authentic and adequate requirements.
2. An open ‘instruct’³ dataset to support further development and evaluation.
3. Open-source fine-tuned LLMs that enable continual learning and domain adaptation.

²Knowledge conflict occurs when the model’s pre-existing training data causes it to interpret a provided instruction or concept differently than intended.

³‘Instruct’ refers to instructions, with each training instance comprising commands and the expected output, as detailed in Section IV-B.

Organization: The rest of the paper is organized as follows: Section II provides background, Section III related work, Section IV presents requirement generation with ReqBrain, Section V describes the evaluation methodology, Section VI presents results and a discussion, Section VII explores implications, Section VIII discusses threats to validity, and Section IX presents conclusions and future work.

II. BACKGROUND

This section defines key concepts and defines ISO 29148-compliant requirements relevant to our work.

A. Task-Specific Instruction Tuning

Task-specific instruction tuning enhances large language model (LLM) performance on specific tasks by using targeted instructions; in our study, these instructions are about writing requirements. A key benefit of this technique is its ability to reduce data and computational costs while maintaining or improving model effectiveness [19].

This supervised fine-tuning method represents the instruct dataset as $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$, where each pair (x_i, y_i) consists of an instruction x_i and its corresponding ground truth output y_i (also referred to as *completion*), with $x_i \in X$ and $y_i \in Y$. During the forward pass, the dataset D is input into the language model, producing predicted outputs $\hat{Y} = \{\hat{y}_1, \hat{y}_2, \dots, \hat{y}_n\}$. Then, each pair (y_i, \hat{y}_i) is compared, and gradients are iteratively computed to optimize the model and align its outputs with the ground truth in terms of syntax and semantics. AI-assisted requirements generation can benefit from this fine-tuning method for the automation of various requirements-related tasks.

In our case, x_i , y_i , and \hat{y}_i are text sequences. For example, y_i is a human-written requirement derived from a real-world project. Based on this requirement, the corresponding instruction x_i might be: “Write a functional requirement for a car’s ABS.” and \hat{y}_i is the model-generated requirement intended to match y_i .

B. ISO 29148-Compliant Requirements

ISO/IEC/IEEE 29148:2018 [13] provides guidelines for eliciting and specifying high-quality textual requirements in natural language for system and software engineering. We incorporate these guidelines to select high-quality, human-authored requirements to ensure that the training dataset reflects real-world language and the nuanced complexities of industry requirements.

The subject of ISO-29148-compliant requirements is expansive, and complete coverage of the standard is beyond the scope of this paper. Instead, we limit it to requirements that employ the below-recommended syntaxes and specific signaling keywords, namely, *shall*, *should*, *may*, and *will*.

SYNTAX-1 : [Subject][Action][Constraint]

SYNTAX-2 : [Condition][Subject][Action][Object][Constraint]

III. RELATED WORK

Studies that focus directly on the generation of requirements using LLMs are still limited, leaving a clear research gap: there is a lack of a systematic approach to internalize requirements-engineering-specific knowledge using fine-tuning LLMs to generate authentic and adequate requirements that support simultaneous elicitation and specification of user requirements by utilizing broad cross-domain knowledge. Additionally, there is a lack of proper human evaluations to validate the requirements generated by these models.

A related study by Arora, Grundy, and Abdelrazek [18] explores LLM generation across all requirements engineering stages, highlighting use cases in elicitation, specification, and validation, supported by a SWOT analysis and preliminary experiments. They emphasize that prompt design critically impacts output quality in prompt-based LLMs, often leading to inconsistent or overly generic requirements, a limitation experimentally demonstrated in other domains where fine-tuning yields more reliable outputs [20].

Similarly, Ronanki, Berger, and Horkoff [21] evaluate ChatGPT's potential to generate requirements through controlled experiments. They crafted six elicitation questions and presented them to ChatGPT and human experts. They then compared ChatGPT's outputs with human experts based on abstraction, atomicity, consistency, correctness, unambiguity, understandability, and feasibility. Results show that ChatGPT outperformed human experts in all aspects except unambiguity and feasibility. While LLMs are rich in knowledge, they lack the nuanced, domain-specific understanding needed for authentic and adequate requirement formulation, a limitation that an LLM can learn utilizing fine-tuning [22], [23].

While earlier studies have generally explored generating requirements with LLMs, the study by Voria et al. [24] introduces RECOVER, a pipeline that automatically generates system requirements from stakeholder conversations. The pipeline works by classifying parts of the conversation as requirements segments, cleaning the selected segments, connecting related ideas in conversation, and generating requirements using the LLaMA-2 model. Their results show vulnerability to hallucinations during generation, a known challenge in prompt-driven pipelines, resulting in knowledge conflict or fluent but unfaithful outputs without explicit domain knowledge using fine-tuning [22].

In contrast, AI-assisted requirements generation with ReqBrain addresses these limitations directly. ReqBrain is not tied to a specific technique, an initial set of requirements, stakeholder conversations, interviews, or pre-acquired data. When such data is available, ReqBrain can also be used to extract and generate requirements from it. We fine-tune ReqBrain to generate authentic and adequate requirements using its internal knowledge, and we are the first to investigate the effect of such tuning by employing a systematic approach. LLMs like ReqBrain can encourage dynamic, interactive engagement between requirements engineers and stakeholders, simulating stakeholder perspectives to generate missing requirements and address tacit knowledge gaps while simultaneously specifying the requirements. However, domain experts retain the final

decision on accepting, rejecting, or modifying the generated requirements to ensure they align with project-specific ethics, needs, and constraints.

Our contribution directly targets this research gap: the absence of a fine-tuned LLM, a systematic approach for authentic and adequate requirements generation, and the lack of systematic human evaluations to assess such LLMs' output quality.

IV. REQUIREMENTS GENERATION WITH REQRAIN

ReqBrain is a fine-tuned large language model (LLM) trained to generate authentic and adequate requirements using its internal knowledge to support the AI-assisted requirements generation approach.

Fig. 1 shows an overview of how a user – such as requirements engineers, project managers, end users, or customers – can generate the initial set of requirements through a simple instruction prompt. In this case, a user might start by specifying a target system (e.g., “Anti-lock brake system”) for which the user wants to generate initial requirements. Then, the user can prompt for further clarification about the generated requirements set, addressing the tacit knowledge by posing additional questions (e.g., “What is a CAN bus?”) and might continue by concentrating on generating specific requirement types (e.g., “safety requirements”) to expand the requirements set. This process can continue until the requirements set fully meet the user's objectives.

In addition to that scenario, a user might take a different approach to generate requirements for a target system by inputting a bug or ticket description and prompting ReqBrain to extract core requirements and generate additional ones. Similarly, users can input concrete requirements to ReqBrain and ask for missing ones or enable ReqBrain with retrieval-augmented generation (RAG) to connect it to an internal management system, such as Jira or local git deployments. In Fig. 2, we present an overview of ReqBrain's integration into the requirements engineering process based on the software engineering body of knowledge (SWEBOK) activities [25].

To support such capabilities, we incorporate a set of targeted training tasks into the training dataset to improve ReqBrain's performance in generating authentic and adequate requirements:

- *How-to? Instructions (INST)*: This task is designed to enhance the model's proficiency using ISO 29148-compliance syntax and signaling keywords to generate requirements aligning with industry standards.
- *RE-types INST*: This task builds on the previous one to broaden the model's understanding of different requirements classes, such as non-functional security, usability, or functional, to assist the model in accurately differentiating and generating various types of requirements when a user prompts the model.
- *Missing INST*: This task simulates scenarios where a set of requirements is incomplete, or a piece of information containing requirements is provided to the model, and the model is tasked to identify missing requirements or extract requirements from the provided information while considering the input information/data.

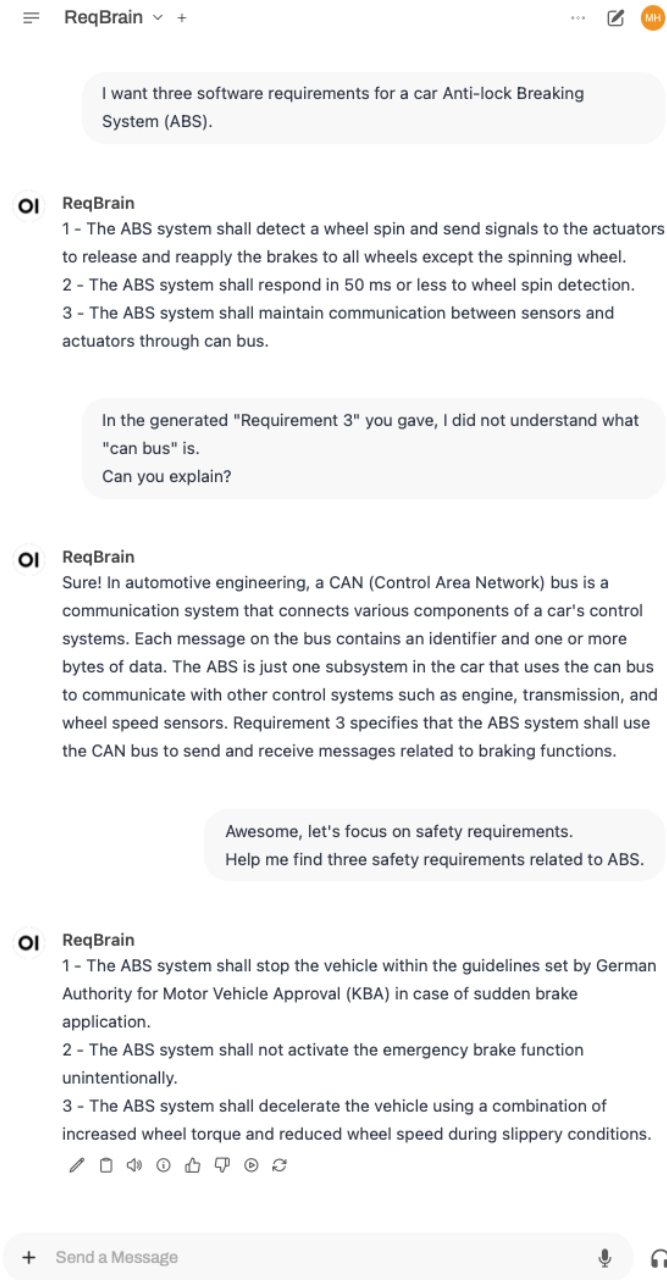


Fig. 1. AI-assisted generation of software requirements using ReqBrain.

A. Assisting Elicitation and Specification with Generation

SWEBOK [25] defines requirements development in four phases: elicitation, analysis, specification, and validation, which are performed in a sequence iteratively. Our focus is primarily on how generation can assist in the elicitation and specification phases.

As illustrated in Fig. 2, generation can be used to simultaneously produce requirements compliant to established specification standards, e.g., ISO 29148, during the elicitation phase, effectively merging elicitation with the specification. This reduces manual overhead, shortens iteration cycles, and ensures that early requirements are already in a usable form. Rather than treating elicited data as raw input that needs to

be transformed later, generation allows for real-time, semi-automated creation of clear and actionable requirements.

By reducing ambiguity early on, generation shifts the analysis from the second to the third phase, interpreting unclear inputs to make higher-level decisions, such as accepting, rejecting, or modifying generated requirements.

While other generative approaches are possible, LLMs provide promising support for requirements elicitation and specification. LLMs trained on large datasets offer a broad cross-domain knowledge base that supports requirements elicitation and specification [11], [12], where a lack of domain knowledge challenges requirements engineers. Furthermore, LLMs can process large volumes of domain-specific information, such as legacy documentation, Jira, or Git, to generate in-context requirements and save time and effort.

Additionally, LLMs can encourage requirements engineers and stakeholders to engage in a dynamic, interactive process to shape and refine authentic requirements [18]. They can also simulate stakeholder perspectives [18] to generate missing requirements and address tacit knowledge gaps.

Despite the potential for AI-assisted requirements generation in the elicitation and specification phases, human expert involvement and review remain essential during analysis and validation to ensure alignment with project goals, ethical considerations, emotional intelligence, and contextual understanding.

B. Training Dataset

To address the absence of a pre-existing dataset for requirements generation, we created an instruct dataset to fill that gap. As established in Section II-A, an instruct dataset comprises training records, each represented as a pair (x_i, y_i) , where x_i denotes an instruction and y_i its corresponding ground truth output (a human-authored requirement or set of requirements), also referred to as the *completion*. First, we describe the process of collecting and selecting the requirements (y_i) and the creation of the instructions (x_i) is discussed next.

1) *Requirements Selection*: We gathered requirements from the Software Requirements Dataset⁴ (SwaRD), which will soon be released as part of another study. SwaRD consolidates publicly disclosed software requirement artifacts from the internet along with non-disclosed requirements from our industry partners. It includes various types of requirements, such as user stories and acceptance criteria. For this study, we filtered the ISO 29148-compliant requirements from SwaRD.

Although these requirements are labeled as ISO 29148-compliant, we found gaps in their compliance upon closer inspection. The first author manually reviewed and selected the compliant requirements, as outlined in Section II-B. While this process is not fully replicable, the selected requirements will be made available within a replication package, allowing interested readers to load the dataset and assess their quality independently.

⁴The publicly available requirements datasets sourced from within the SwaRD are acknowledged for their contributions: [26]–[36].

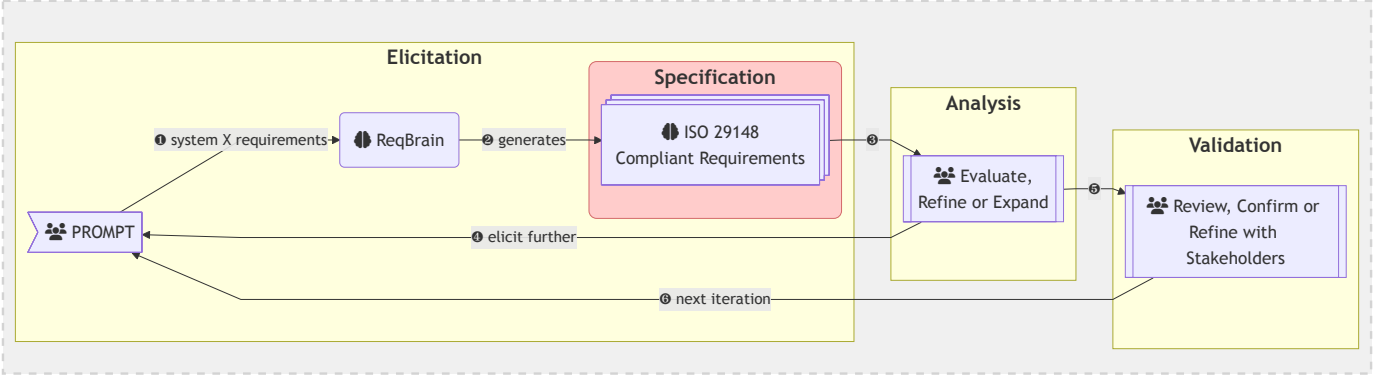


Fig. 2. AI-assisted requirements generation approach overview, integrating ReqBrain.

2) *Instruction Creation*: To create instructions (x_i), we followed established practices and guidelines from the documentation of Hugging Face and OpenAI. For each pair (x_i, y_i) , we reviewed the requirement (y_i) and created a context that includes supporting information about its intent, class (e.g., functional or non-functional), and ISO 29148-compliance (see Section II-B). We then incorporated this context to craft the instruction (x_i), paired it with y_i , and added the pair to our dataset, D . Our templates for writing instructions are given below:

Instruction Template-1 : [Context + Instruction]

Instruction Template-2 : [Instruction + Context]

To realize each of the three targeted training tasks, described earlier, we create a corresponding instruction category. First, for instructions in the *How-to? INST* task, we included syntax details such as the correct placement of constraints, conditions, subjects, or signaling keywords, enabling the model to learn the structure of requirements. Next, for instructions in the *RE-types INST* task, we added information about requirement classes using the relabeled PROMISE dataset [33], known for its quality and widespread use in requirement classification studies. Finally, for instructions in the *Missing INST* task, we grouped the selected requirements by their original software projects and split them into two groups: one used in the instruction to simulate an incomplete set of requirements, and the other serving as completion labels.

Table I draws a single instance from each task category from our dataset to illustrate their distinctions and provide an overall understanding.

3) *Training and Evaluation Sets*: Our instruction dataset comprises 166 training instances. Together, these instances cover a total of 242 individual requirements. Each instance is a training record that combines an instruction with its corresponding completion. A training record may include multiple requirements, as demonstrated by the *Missing INST* record in Table I.

The dataset is organized into columns containing metadata about the collected requirements, as outlined in Table II. Each column serves a specific purpose to ensure that all necessary components are available for efficient model training.

We applied a stratified split based on targeted task categories, allocating 80% of the dataset for training and 20% for

evaluation. This method ensures a balanced representation of instruction categories across both sets.

4) *Why not a larger training set?*: Creating an extensive training set manually is time-intensive; however, our fine-tuning approach reduces the need for large datasets. For example, Chen et al. [19] demonstrated that task-specific instruction-tuned models achieve significant performance with only a small fraction of a dataset.

Moreover, fine-tuning an LLM on a large dataset for a specific task can distort its pre-trained weights, leading to catastrophic forgetting and underperformance [37], [38]. Several studies, including the influential OpenAI paper on GPT-3 [39] and [40], suggest that LLMs require only a few high-quality examples to learn a new task. Therefore, we selected 242 high-quality requirements that best align with our definition of ISO 29148-compliant requirements from, Section II-B, instead of creating a large training set.

C. Fine-Tuning

Training large language models (LLMs) involves two main steps: pre-training and fine-tuning. During pre-training, models are exposed to vast text corpora without task-specific labels or annotations [11], [41], [42], enabling them to learn general linguistic patterns and structures unsupervised. However, pre-training is costly in terms of computational resources, time, and data requirements, making it less feasible for each new task. Fine-tuning refines a pre-trained model's representations for a downstream task by updating the pre-trained weights Φ_p to $\Phi_p + \Delta\Phi_p$, following gradients to maximize the conditional language modeling objective [43].

Fine-tuning is less costly than pre-training because it uses the pre-trained model as a base. However, it has a limitation: learning the complete set of parameters $\Delta\Phi_p$, whose dimension $|\Delta\Phi_p|$ equals $|\Phi_p|$.

Parameter-efficient fine-tuning methods, such as low-rank adaptation (LoRA) [43], address this limitation by reducing the number of parameters updated during training, thereby minimizing the risk of catastrophic forgetting [44] – the loss of pre-trained knowledge during full fine-tuning [45], [46]. Minimizing catastrophic forgetting is crucial for ReqBrain, as it preserves the base model's knowledge for use in chat/dialog

TABLE I
TRAINING PAIRS FROM THE INSTRUCT DATASET. *ELLIPSES ARE USED TO CONDENSE THE TEXT TO SAVE SPACE.

Task	Instructions	Completion
How to? INST	Write a requirement in compliance with ISO 29148 recommendations for a well-formed requirement. The desired functionality to be added is to print Construction Junction staff member cards, printing the staff name and unique number in both numeric and barcode formats using the existing company technology (constraints). This functionality must be operational only for registered employees (condition).	Once registered online, the system should allow construction junction staff to print out a member card via QuickBooks Point of Sale (POS) containing the member's name and unique member number in both numeric and bar code format.
RE- types INST	I want my system's customers to have the ability to both purchase and start watching a specific movie within a time frame of fewer than 3 minutes. Can you give me a usability (US) requirement?	Customers should be able to purchase and watch a pre-determined movie in under 3 minutes.
Missing INST	I want to build software for managing project requirements effectively. The requirements we have come up with are listed here: - The user is provided the functionality via "browse" mode to manually link low-level requirements to high-level requirements. *... - The user shall be able to activate the "save" function to save the current data and status of the project. Can you help me complete the missing software requirements?	Here are some more software requirements your software might need: 1 - The "show" function will provide the user with a display of the current file. *... 12 - The "save trace" function shall allow the user to save the trace result.

TABLE II
INSTRUCT DATASET STRUCTURE, COLUMNS AND DESCRIPTION

Columns	Description
REQID_ex	An identifier for each requirement
completion	Selected requirements from SwaRD (y_i).
instructions	Created instructions (x_i)
class	Requirements type information.
task	Task: <i>How-to?</i> <i>INST</i> ; <i>RE-types</i> <i>INST</i> ; <i>Missing</i> <i>INST</i>
text	Training records: x_i and y_i paired.

capabilities to inquire about various aspects of the generated requirements.

LoRA leverages the concept of intrinsic dimension – the minimum number of dimensions required to represent a matrix's essential features. In deep learning, training on the intrinsic dimension (i.e., partial training) means updating only a subset, r , of Φ_p for the downstream task [47]. LoRA achieves this by freezing the pre-trained weights Φ_p , training LoRA weights $\Delta\Phi_l$ for a weight matrix $\Phi_l \in \mathbb{R}^{A \times B}$, and decomposing the weight update matrix into two smaller matrices, as shown in equation 1.

$$\Delta\Phi_l = \Phi_A \times \Phi_B \quad (1)$$

Where, $\Phi_A \in \mathbb{R}^{A \times r}$ and $\Phi_B \in \mathbb{R}^{r \times B}$, with r representing the intrinsic dimension, a tunable parameter that effectively reduces the number of dimensions. For inference and evaluation, the LoRA weights are added to the original frozen weights Φ_p at the end of each training round, as shown in equation 2.

$$h = \Phi_p + \Delta\Phi_l = \Phi_p + AB \quad (2)$$

D. Selecting Models for Fine-Tuning

For fine-tuning, we focus on open-source models to enable reproducibility and to support organizations in hosting models on their own platforms for privacy. Although tuning commercial models is possible, our key contribution is sharing the dataset and open-source models to enable the models' continual learning, collaboration, and transparency, advancing the AI-assisted requirements generation.

Selecting pre-trained models requires balancing performance and computational resources, which is influenced by model size. Recent studies highlight the effectiveness of 7B LLMs in achieving this balance [48], [49]. To establish a baseline and identify the best variant for generating requirements, we initially fine-tuned and compared Falcon-7b-base and the instruct variant, with the instruct variant outperforming the others.

Instruct and chat models interact similarly by providing answers through chat, whereas base models are trained to acquire diverse features, serving as a robust foundation for various tasks. Therefore, we present results for four state-of-the-art open-source instruct or chat models: Llama-2-7b-chat-hf⁵ [50], Mistral-7B-Instruct-v0.2 [51], Zephyr-7b-beta [52], Falcon-7b-instruct [53], and one base model: Falcon-7b.

E. Fine-Tuning Hyperparameters Selection

Throughout our experiments, we used the Hugging Face API [54] and its models. For all models, we employed LoRA with $r = 64$, as supported by [43], which shows that a low-rank adaptation matrix with $r = 64$ effectively captures essential weight update information while ensuring competitive performance and computational efficiency. Based on [55], we opted for a learning rate of $2e - 4$ with a cosine scheduler, which balances stability and efficiency by gradually adjusting the learning rate to facilitate stable convergence and mitigate premature stagnation or divergence. For the remaining parameters, we used the original hyperparameters from the base model, as documented in the Hugging Face model's documentation.

V. EVALUATION

This section outlines the evaluation methodology and study design used to assess ReqBrain's performance.

⁵As of the training and evaluation period, LLaMA had no instruct-tuned version available.

A. Study Design

Our objective is to assess how fine-tuning affects large language models (LLMs) in generating authentic and adequate requirements. To achieve our objective, for RQ1.1 and RQ1.2, we used a standard NLP study design. We conducted a between-subjects study design for the remaining research questions to minimize biases such as carryover or learning effects [56], [57]. By exposing participants to only one condition, the design ensured independent evaluations free from the influence of prior conditions. This independence was crucial for nuanced judgments when comparing ReqBrain-generated requirements against its untuned baseline model and against human-authored requirements or assessing generated requirements for consistency under a single condition [56].

Where applicable, we followed the evaluation guidelines for empirical studies involving LLMs in software engineering by Wagner et al. [58], reporting model roles, versions, hyperparameters, and hosting details as recommended. Furthermore, all participants provided informed consent prior to their involvement, acknowledging their understanding of the study's purpose, the nature of their participation, and their right to privacy and anonymity.

Participants were asked to bring their laptops to the session, where the study objectives and background information (including knowledge refresher material from ISO 29148) were outlined. They were introduced to three evaluation datasets and their structure.

At the end of the session, an evaluation package containing essential information and assurances of privacy and anonymity was distributed. Before concluding, participants evaluated a few random requirements from each task to confirm their understanding of the process. Furthermore, all participants provided informed consent prior to their involvement, acknowledging their understanding of the study's purpose, the nature of their participation, and their right to privacy and anonymity.

B. Tasks

We address our research questions through the following tasks. All tasks, except Task A, are evaluated by human participants.

1) *Task A*: Within this task, we benchmark the performance of the five fine-tuned LLMs to identify the potential best-performing model in generating authentic requirements for RQ1.1 to reduce exhaustive human evaluation across various models for the subsequent questions. Then, we compare the selected model, ReqBrain, with the untuned ChatGPT-4o-latest⁶ to determine whether commercial models designed for general tasks can match or exceed the performance of our fine-tuned model for RQ1.2.

2) *Task B*: This task compares the requirements generated by ReqBrain with those produced by its untuned baseline model, addressing authenticity in RQ1.3 and the ISO 29148 compliance dimension of adequacy in RQ2. To determine authenticity, participants evaluated how indistinguishable the

generated requirements are from those written by humans, focusing on clarity, coherence, relevance, realism, and implementability. For adequacy, we considered the qualities defined for ISO 29148-compliant requirements in Section II-B. Human participants evaluated requirements from both models, knowing the set contained a mix of human-authored and AI-generated requirements. We assume a positive fine-tuning effect if participants frequently judge ReqBrain-generated requirements as human-authored.

3) *Task C*: Building on task B, participants assess authenticity in RQ1.3 and ISO 29148-compliant dimension of adequacy in RQ2 between ReqBrain-generated and human-authored requirements.

4) *Task D*: We focus on the three remaining dimensions of adequacy in RQ2. We input requirements specifications from real-world projects to ReqBrain and task participants to evaluate whether the generated requirements are consistent with, missing from, and enhance the overall completeness of the given requirements specification.

C. Evaluation Materials

A comprehensive evaluation set was created for each specific task to assess performance and ensure accurate measurement of outcomes.

1) *Benchmark Datasets*: For task A, we generated requirements for each fine-tuned LLM and untuned ChatGPT-4o by inputting the instructions corresponding to the human-authored requirements from our evaluation set detailed in Section IV-B3. Each human-authored requirement is paired with its corresponding LLM-generated requirement for each of the evaluation sets, resulting in a total of six benchmarking datasets.

2) *ReqBrain vs. Baseline Model Evaluation Dataset*: For task B, we input the instructions from our evaluation set to ReqBrain and its untuned baseline model to generate requirements. To ensure unbiased assessment, the authorship of all generated requirements was anonymized. The requirements were then combined and shuffled before being presented to participants for evaluation.

3) *ReqBrain vs. Human-Authored Evaluation Dataset*: For Task C, we combined the ReqBrain-generated requirements with their corresponding human-authored counterparts. The requirements were anonymized, shuffled, and presented in a stacked list format for direct comparison.

4) *ReqBrain Usability Evaluation Dataset*: For Task D, we developed a new evaluation set using requirements from three distinct software projects within KIB³ (Künstliche Intelligenz in die berufliche Bildung bringen). KIB³ is a German AI educational initiative aimed at developing innovative AI tools to support students in their studies. The selected requirements met the following criteria:

1. Formalized according to ISO 29148 guidelines
2. Elicited from diverse stakeholders, not derived from prior projects or the internet
3. Not published online, reducing the risk of inclusion in any model's training data
4. Created through a well-documented process

⁶This is the latest model release of ChatGPT as of 21 December 2024.

5. Open-source (KIB³), allowing requirements to be published for transparency and reproducibility

Three software projects were selected: students' self-evaluation software, adaptation software, and chatbot software. For each project, we created instructions incorporating its requirements and provided them to ReqBrain, which generated additional requirements. The generated requirements were paired with their corresponding instructions and presented to participants for evaluation. In this task, the authorship of the requirements was not concealed, enabling participants to evaluate the generated requirements in full context.

D. Hypotheses, Variables, and Operationalization

The variables used to measure the authentic and adequate constructs are provided in Table III. To assess how closely AI-generated text aligns with human-authored ground truth in semantics, fluency, coherence, factual accuracy, and originality, we use the established and automated Human Alignment_(HA) metrics BERT and FRUGAL. Although the FRUGAL Score is more powerful, BERT is more intuitive; therefore, we computed both. Furthermore, BERT and FRUGAL Scores are learned metrics [59], [60] that are preferred over traditional metrics such as BLEU, ROUGE, or TER [61]–[63], which emphasize surface-form similarity and are often not suitable for computing human alignment [64]–[66].

We distinguish comparisons (1) between requirements generated by ReqBrain and those produced by its untuned baseline model and (2) between human-authored requirements and those generated by ReqBrain. In the first comparison, failing to reject the null hypothesis indicates no fine-tuning effect. In the second comparison, it suggests a positive effect, as our goal is to achieve human-comparable qualities.

For RQ1.1 and RQ1.2, we used the Human Alignment_(HA) variable to evaluate the quality of AI-generated requirements relative to their corresponding human-authored counterparts.

For RQ1.3, we first compare participants' perceptions and success rates in identifying the requirements generated by ReqBrain against its untuned baseline model as human-authored using the Perceived Authorship_(PA) variable and formulate the following hypothesis:

■ Hypothesis:

$H_{0,1}$: The proportion of generated requirements identified as human-authored is independent of whether they were generated by ReqBrain (the fine-tuned model) or its untuned baseline model.

$H_{a,1}$: The proportion of generated requirements identified as human-authored is not independent of whether they were generated by ReqBrain (the fine-tuned model) or its untuned baseline, with ReqBrain producing a greater proportion.

Second, we compare human-authored and ReqBrain-generated requirements using the following hypothesis:

■ Hypothesis:

$H_{0,2}$: Humans do not reliably distinguish between human-authored and ReqBrain-generated requirements in terms of accuracy.

$H_{a,2}$: Humans reliably distinguish between human-authored and ReqBrain-generated requirements.

For the ISO 29148-compliant dimension of adequacy in RQ2, we used the variables Written Syntax Compliance_(WSC) and Signaling Keywords Compliance_(SKC) to collect participants' responses and formulated the following hypotheses between ReqBrain and its untuned baseline model:

■ Hypotheses

$H_{0,3}$: Requirements generated by ReqBrain do not show greater adherence to ISO 29148 syntax compared to those from its untuned baseline model.

$H_{a,3}$: Requirements generated by ReqBrain show greater adherence to ISO 29148 syntax compared to those from its untuned baseline model.

$H_{0,4}$: Requirements generated by ReqBrain do not show greater adherence to ISO 29148 signaling keywords compared to those from its untuned baseline model.

$H_{a,4}$: Requirements generated by ReqBrain show greater adherence to ISO 29148 signaling keywords compared to those from its untuned baseline model.

Next, we compare ReqBrain-generated with human-authored requirements using the following hypotheses:

■ Hypotheses:

$H_{0,5}$: Human-authored and ReqBrain-generated requirements do not differ in their adherence to ISO 29148 written syntax.

$H_{a,5}$: Human-authored and ReqBrain-generated requirements differ in their adherence to ISO 29148 written syntax.

$H_{0,6}$: Human-authored and ReqBrain-generated requirements do not differ in their adherence to ISO 29148 signaling keywords.

$H_{a,6}$: Human-authored and ReqBrain-generated requirements differ in their adherence to ISO 29148 signaling keywords.

For the remaining dimensions of adequacy in RQ2, we used the variables Consistent with Requirements Set_(CRS), Missing Requirements_(IMR), and Enhancing the Overall Completeness_(EOC) to collect participants responses. Responses ≤ 3 indicate a range from neutral to strongly disagree on our selected Likert scale.

TABLE III
USED VARIABLES FOR CONSTRUCT EVALUATION.

Constructs	Variable	Variable Description	Scale
Authentic	Human Alignment _(HA)	BERT Score: measures semantics, fluency, coherence, factual accuracy, and originality with human-authored requirements	Continuous (0–1)
	Human Alignment _(HA)	FRUGAL Score: measures semantics, fluency, coherence, factual accuracy, and originality with human-authored requirements	Continuous (0–100)
	Perceived Authorship _(PA)	Based on the style and content of the requirement, do you believe it was written by a human or generated by AI?	Dichotomous (Human/AI)
Adequate	Written Syntax Compliance _(WSC)	This requirement is well-structured according to the ISO 29148 recommended syntax.	Likert (1: Strongly Disagree, 5: Strongly Agree)
	Signaling Keywords Compliance _(SKC)	This use of signaling keywords to indicate the presence of a requirement is appropriate based on ISO 29148.	
	Consistent with Requirements Set _(CRS)	This requirement is consistent with other project requirements.	
	Identify Missing Requirements _(IMR)	This requirement accurately reflects the needs that were previously unstated, missing, or overlooked.	
	Enhancing the Overall Completeness _(EOC)	Including this requirement would lead to a more complete set of project specifications.	

Hypotheses:

$H_{0,7}$: The median rating (M) for the Consistent with Requirements Set_(CRS) is ≤ 3 .

$H_{a,7}$: The median rating for Consistent with Requirements Set_(CRS) is > 3 .

$H_{0,8}$: The median rating (M) for the Identify Missing Requirements_(IMR) is ≤ 3 .

$H_{a,8}$: The median rating for Identify Missing Requirements_(IMR) is > 3 .

$H_{0,9}$: The median rating (M) for the Enhancing the Overall Completeness_(EOC) is ≤ 3 .

$H_{a,9}$: The median rating for Enhancing the Overall Completeness_(EOC) is > 3 .

E. Analysis Procedure

First, we analyze RQ1.1 and RQ1.2 using NLP metrics to measure the similarity between requirements. Then, we outline the evaluation process for the remaining research questions.

1) *NLP Metrics Analysis Procedure*: We compute the pairwise similarity between the ReqBrain-generated and human-authored ground truth requirements (see Section V-C1 for the evaluation set setup) for RQ1.1 and RQ1.2.

a) *The BERT Score*: BERT score is a learned evaluation metric for text generation that utilizes contextualized embeddings from BERT [11]. It computes the cosine similarity between token embeddings of a human-authored reference $x = x_1, x_2, \dots, x_n$ and an AI-generated equivalent $\hat{x} = \hat{x}_1, \hat{x}_2, \dots, \hat{x}_m$. The precision and recall are computed as:

$$R = \frac{1}{|x|} \sum_{x_i \in x} \max_{\hat{x}_j \in \hat{x}} x_i^\top \hat{x}_j \text{ and } P = \frac{1}{|\hat{x}|} \sum_{\hat{x}_j \in \hat{x}} x_i \in x \max_{\hat{x}_i \in \hat{x}} \hat{x}_j^\top x_i \quad (3)$$

The F1 score is derived from these values. Unlike traditional n -gram metrics, contextualized embeddings capture word meaning, synonyms, context, and grammar [11].

b) *The FRUGAL Score*: The FRUGAL score is similar to the BERT score but is faster and lighter, and in some cases, outperforms the BERT score [60]. Its training involves generating a synthetic dataset by pairing sequences annotated with costly metrics aligned with human judgment, followed by pre-training a miniature language model⁷ on this dataset to learn the mapping of costly metrics and similarity functions.

2) *Human Evaluation Analysis Procedure*: For RQ1.3 and the four dimensions in RQ2, we used descriptive statistics to summarize sample characteristics and inferential statistics to test the hypotheses.

For all samples used in RQ2, we calculated the mean (\bar{x}), standard deviation (s), and median (M). Although we report \bar{x} and s , hypothesis testing relies on the median (M), which is more appropriate for ordinal data [56], [68].

Due to the nature of the data, non-parametric tests were employed to evaluate all the hypotheses. Non-parametric tests are robust for ordinal data and do not assume normality or equal intervals [56], [69], [70]. For all tests, a significance level of $\alpha = .05$ was used to determine statistical significance, and 95% confidence intervals were reported for effect size estimates.

For RQ1.3, descriptive statistics include success and failure counts and success proportions. A right-tailed Fisher's Exact test was used to test $H_{a,1}$, comparing the proportions of requirements identified as human-authored between ReqBrain and its untuned baseline model. An odds ratio was calculated as the effect size.

For identifying authorship between ReqBrain-generated and human-authored requirements, a contingency table and expected frequencies were computed for both samples employing

⁷Is a downscaled larger model that maintains the original performance or comes close to it [67].

the Chi-square test (χ^2) to evaluate $H_{a,2}$. Overall human precision in identifying authorship between ReqBrain-generated and human-authored requirements was also calculated with confidence intervals.

For the ISO 29148-compliant dimension of adequacy in RQ2, four hypotheses are tested. Two hypotheses ($H_{a,3}$ and $H_{a,4}$) compare ISO 29148-compliance between ReqBrain-generated and its untuned baseline model using right-tailed Mann-Whitney U tests. The remaining two ($H_{a,5}$ and $H_{a,6}$) compare ReqBrain-generated requirements with human-authored ones using two-tailed Mann-Whitney U tests to evaluate equivalence. The effect size for all Mann-Whitney U tests is quantified using Vargha and Delaney’s A-statistic [70].

For the remaining dimensions of adequacy in RQ2, three one-sample Wilcoxon signed-rank tests were conducted, one for each hypothesis ($H_{a,7}$, $H_{a,8}$, and $H_{a,9}$), with the rank biserial (r) effect size. Additionally, to account for multiple tests for RQ1.3 and each dimension in RQ2, we calculate and report adjusted p-values using the Holm-Bonferroni method [71].

F. Sample Size and Participants

We performed an a-priori power analysis to determine the sample size for different evaluations. Following Dybå et al. [72], we conducted power analysis for the non-parametric tests using their analogous parametric tests. We used the conventional $\alpha = 0.05$, power = 0.8, and a recommended effect size $\beta = 0.5$ for software engineering studies [72]. An optimal sample size of 64 requirements was calculated for two-tailed Mann-Whitney U tests and 51 for its one-tailed tests using a two-tailed t-test and one-tailed t-test, respectively. For one-sample, one-tailed Wilcoxon signed-rank tests, an optimal sample size of 26 was determined using a one-sample, one-tailed t-test, and for Fisher’s exact, we used Chi-square to calculate an optimal sample size of 32. The first part of our study design (see Section IV-B3) resulted in sample sizes two to three times larger.

Four experienced participants evaluated a total of 672 distinct requirements for different tasks. Two had 1–3 years of work experience, and the other had 4–6 years in software and requirements engineering. Each participant also held at least a bachelor’s degree in software engineering. In terms of familiarity with AI content, two were “Very familiar,” one was “Somewhat familiar,” and one was “Moderately familiar.” Regarding the use of generative AI tools like ChatGPT, two used them “Sometimes,” one answered “Yes,” and one responded “No.”

Table IV presents a comprehensive overview of the main points in this section.

VI. RESULTS AND DISCUSSION

We first present the results corresponding to each research question, followed by a brief discussion and then a summary of findings.

A. RQ1 Generating Authentic Requirements

1) *RQ1.1 Benchmarking the Fine-tuned Models:* Table V presents the performance metrics of five fine-tuned large language models. Zephyr-7b-beta outperforms all other models on both metrics.

Figure 3 illustrates the performance of the models across the three instruction categories described in Section IV-B using FRUGAL and BERT scores. Both scores show that Mistral-7B-Instruct-v0.2 performs slightly better than Zephyr-7b-beta in the *Missing INST* task. This may stem from its architectures and training data, which likely include a broader range of similar tasks.

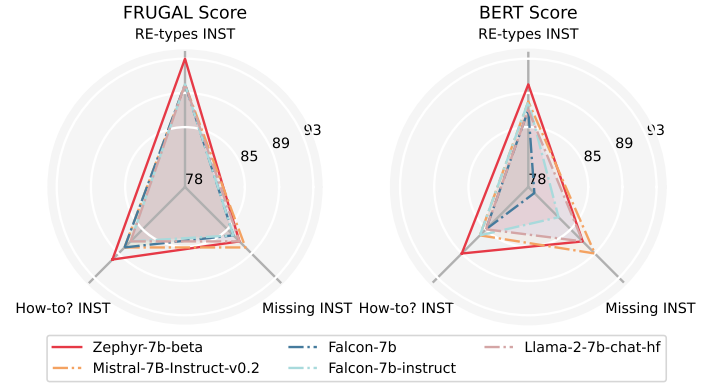


Fig. 3. Performance metrics across three task categories (see Section IV-B).

Nevertheless, Zephyr-7b-beta records higher aggregate scores across all three task categories. Hence, we identified Zephyr-7b-beta as the most effective model for generating authentic requirements.

2) *RQ1.2 Benchmarking ReqBrain against ChatGPT-4o:* Table VI summarizes performance metrics for ReqBrain, our best-performing fine-tuned model, and untuned ChatGPT-4o. The metrics show that ReqBrain outperforms the untuned ChatGPT-4o in generating authentic requirements.

Although the comparison with untuned ChatGPT-4o might seem unfair, it underscores the importance of fine-tuning LLMs for requirements elicitation tasks. ChatGPT-4o, with its larger parameter count, might surpass ReqBrain (which has 7 billion parameters) in performance if fine-tuned.

3) *RQ1.3 ReqBrain vs. its Untuned Baseline Model:* Table VII provides results to assess the perceived human authorship of requirements generated by ReqBrain and its untuned baseline model.

For ReqBrain, 47.8% of the generated requirements are identified as human-authored, compared to only 8.8% for the untuned baseline model. The right-tailed Fisher’s Exact test produced a p-value < 0.001 , providing strong evidence in favor of the alternative hypothesis ($H_{a,1}$). The odds ratio of 9.46 indicates that the odds of fine-tuned model outputs being perceived as authentic are approximately 9.5 times higher than those of the baseline model.

4) *RQ1.3 ReqBrain vs. Human Authors:* In Table VIII, we summarize the results for this comparison. The Chi-square test yielded $\chi^2(1; N = 272) = 0.01475694$ and $p = 0.90331$,

TABLE IV
RQ MAPPING TO HYPOTHESES, EVALUATION MATERIALS, VARIABLES, STATISTICAL TESTS, DIRECTIONALITY, AND COMPARED SAMPLES.
ABBREVIATION: M_h , HYPOTHEZED MEDIAN

RQ	Hypotheses	Evaluation Materials	Variables	Statistical Test	Directionality	Compared Samples
RQ1.1						Human vs. ReqBrain
RQ1.2	n/a	Benchmark Datasets	Human Alignment $_{(HA)}$	BERT Score; FRUGAL Scores	n/a	Human vs. ReqBrain Human vs. ChatGPT-4o
RQ1.3	$H_0, 1,$ $H_a, 1$	ReqBrain vs. its Baseline Model Evaluation Dataset	Perceived Authorship $_{(PA)}$	Fisher’s Exact	Right-tailed	ReqBrain vs. its Baseline Model
RQ1.3	$H_0, 2,$ $H_a, 2$	ReqBrain vs. Human-Authored Evaluation Dataset		Chi-square	n/a	Human vs. ReqBrain
RQ2	$H_0, 3 - 4,$ $H_a, 3 - 4$	ReqBrain vs. Baseline Model Evaluation Dataset	Written Syntax Compliance $_{(WSC)}$, Signaling Keywords Compliance $_{(SKC)}$	Mann-Whitney U	Right-tailed	ReqBrain vs. its Baseline Model
	$H_0, 5 - 6,$ $H_a, 5 - 6$	ReqBrain vs. Human-Authored Evaluation Dataset		Mann-Whitney U	Two-tailed	Human vs. ReqBrain
	$H_0, 7 - 9,$ $H_a, 7 - 9$	ReqBrain Usability Evaluation Dataset	Consistent with Requirements Set $_{(CRS)}$, Identify Missing Requirements $_{(IMR)}$, Enhancing the Overall Completeness $_{(EOC)}$	Wilcoxon Signed-rank	Right-tailed	ReqBrain vs. $M_h > 3$

TABLE V
Human Alignment $_{(HA)}$ RESULTS: PERFORMANCE METRICS FOR FIVE FINE-TUNED LLMs. ABBREVIATIONS: P, PRECISION; R, RECALL.

Models	BERT Score			FRUGAL Score
	P	R	F1	
Zephyr-7b-beta	0.89	0.89	0.89	0.91
Mistral-7B-Instruct-v0.2	0.84	0.89	0.86	0.88
Falcon-7b	0.80	0.82	0.81	0.88
Falcon-7b-instruct	0.85	0.88	0.86	0.88
Llama-2-7b-chat-hf	0.81	0.85	0.83	0.88

TABLE VI
Human Alignment $_{(HA)}$ RESULTS: PERFORMANCE METRICS FOR REQRAIN VS. CHATGPT-4O. ABBREVIATIONS: P, PRECISION; R, RECALL.

Models	BERT Score			FRUGAL Score
	P	R	F1	
ReqBrain	0.89	0.89	0.89	0.91
ChatGPT-4o	0.81	0.88	0.84	0.86

with an odds ratio of 1.06, providing no evidence to support the alternative hypothesis ($H_{a,2}$). Furthermore, the classification precision is 50.7%. The results suggest that ReqBrain-generated requirements are perceived as authentic by humans, as evaluators could not reliably distinguish between them and those authored by humans.

Findings RQ1:

Our study shows that fine-tuning LLMs effectively generates authentic requirements indistinguishable from those authored by humans. Human evaluators could not reliably differentiate between human-authored and fine-tuned LLM-generated requirements (ReqBrain), indicating that its outputs meet human quality standards.

B. RQ2 Generating Adequate Requirements

1) *Comparing ReqBrain and its Untuned Baseline Model on ISO 29148:* Table IX provides a summary of the results. For written syntax, ReqBrain achieved a median rating of 4

TABLE VII
Perceived Authorship $_{(PA)}$ RESULTS: PERCEIVED HUMAN-LIKENESS BETWEEN REQRAIN AND ITS UNTUNED BASELINE MODEL.

Attribute	ReqBrain	Untuned Baseline
Sample Size	n_1 : 136	n_2 : 136
Human-Classified Count	65	12
AI-Classified Count	71	124
Human-identified Proportions	0.47	0.08
Odds Ratio	9.46 (95% $CI = [4.79, 18.70]$)	
p -Value	< .001	
$Adj. p$	< .001	

TABLE VIII
Perceived Authorship $_{(PA)}$ RESULTS: HUMAN ABILITY TO DISTINGUISH REQRAIN-GENERATED REQUIREMENTS FROM HUMAN-AUTHORED REQUIREMENTS

Attribute	Human	ReqBrain
Sample Size	n_1 : 136	n_2 : 136
Contingency Frequencies	[[65, 71][63, 73]]	
Expected Frequencies	[[64, 72][64, 72]]	
χ^2 Statistics	0.01475694	
p -Value	0.90331	
$Adj. p$	0.90331	
Odds Ratio	1.06 (95% $CI = [0.64, 1.75]$)	
Human Identification Precision	50.7% (95% $CI = [44\%, 56\%]$)	

compared to 2 for the untuned baseline model. The right-tailed Mann-Whitney U test yielded $U = 14203.5$ and $p < .001$, with large effect size $A_{12} = 0.76$, indicating strong evidence in favor of the alternative hypothesis ($H_{a,3}$). For signaling keywords, ReqBrain also achieved a median rating of 4 compared to 2 for the untuned baseline model, with $U = 13766.0$ and $p < .001$, and large effect size $A_{12} = 0.74$, supporting the alternative hypothesis ($H_{a,4}$).

The results suggest that a fine-tuned LLM significantly

outperforms its untuned baseline model in generating ISO 29148-compliant requirements.

2) *Comparing ReqBrain and Humans on ISO 29148*: Table X provides a summary of the results. For written syntax, both groups achieved a median rating of 4. The Mann-Whitney U test revealed no significant difference with $U = 10118.5$ and $p = 0.15521$, with an effect size of $A_{12} = 0.54$, thereby providing no support for the alternative hypothesis ($H_{a,5}$). For signaling keywords, both groups recorded a median rating of 4. However, the Mann-Whitney U test revealed a statistically significant difference with $U = 10482.0$ and $p = 0.04068$, supporting the alternative hypothesis ($H_{a,6}$) and a small effect size $A_{12} = 0.56$. Nevertheless, observing its $Adj. p = 0.12204$ suggests rejecting the alternative hypothesis; note that for all other hypotheses, the adjusted p -values confirm the unadjusted p -values.

The findings suggest that fine-tuned LLM produces requirements that are comparable to those authored by humans in terms of appropriate use of syntax and signaling keyword usage.

3) *Evaluating ReqBrain on the Remaining Dimensions*: Table XI summarizes the results for these three dimensions. For all three dimensions, the median rating was $M = 4$.

For *consistent with* dimension, the right-tailed Wilcoxon Signed Rank test yielded a statistically significant median rating greater than 3 with $W = 7470.5$ and $p < .001$ with a large effect size $r = 0.82$ supporting the alternative hypothesis ($H_{a,7}$).

For *missing from* dimension, the right-tailed Wilcoxon Signed Rank test showed a statistically significant median rating greater than 3 with $W = 7035.0$ and $p < .001$, and a large effect size $r = 0.72$. These results support the alternative hypothesis ($H_{a,8}$).

For *enhancing the overall completeness* dimension, the right-tailed Wilcoxon Signed Rank test with $W = 6378.0$ and $p < .001$ confirms the rejection of the null hypothesis in favor of the alternative ($H_{a,9}$), with a large effect size $r = 0.58$.

In summary, all three dimensions confirm that ReqBrain is effective in generating requirements that are consistent with, missing from, and enhancing the overall completeness of a given specification.

Findings RQ2:

Our results indicate that a fine-tuned LLM generates adequate requirements, thereby validating its effectiveness in generating high-quality requirements.

VII. IMPLICATIONS FOR RESEARCH AND PRACTICE

Below, we discuss how ReqBrain contributes to both research and practice in requirements elicitation and specification.

A. Research Implications

ReqBrain contributes to requirements engineering research by demonstrating that fine-tuning LLMs can enhance the generation of high-quality requirements. It fills a gap in the largely

manual process of requirements elicitation and specification by generating authentic and adequate requirements. Empirical validation using BERT and FRUGAL scores, along with human evaluations, underscores the effectiveness of customized LLMs over untuned models. Future research may extend our approach to cover further requirements-related tasks to advance the AI-assisted requirements generation approach, supported by our open-source dataset and methodology for continued collaboration and advancement in the field.

B. Practice Implications

Integrating ReqBrain into the requirements elicitation phase has the potential to improve the efficiency and accuracy of collecting, categorizing, and documenting requirements. Automatically generating authentic and adequate requirements may reduce manual workload and enable software engineers to focus more on strategic decision-making and stakeholder engagement. ReqBrain can be deployed individually or in group sessions. As an open-source model, it can be hosted locally to ensure data privacy and extended with RAG to process large volumes of text from sources like JIRA, databases, project documents, and interviews.

Although our empirical validation demonstrates that ReqBrain is effective in generating authentic and adequate requirements, human expert review remains essential to ensure ethical considerations, emotional intelligence, and contextual understanding.

VIII. THREATS TO VALIDITY

In the following section, we will outline the various potential threats that could undermine the validity of our study.

A. Construct Validity

For the authentic construct, we used a two-step assessment. First, we deployed automated NLP-based metrics, then conducted a human evaluation. This order mitigates the limitations of automated NLP metrics in representing human alignment on clarity, coherence, relevance, realism, and implementability—key aspects of authenticity. Additionally, it helps reduce human evaluation across multiple low-quality models. Further, as an intermediary step, although this step is not replicable, between NLP-based metrics and human evaluators, the first author performed a manual review to confirm that these metrics identified the best-performing model. For the adequate construct, we employed human evaluations across four dimensions.

B. Internal Validity

Differences in participants' interpretations of evaluation criteria may affect validity. To mitigate this, we selected experienced participants, held an onboarding session with detailed guidelines and knowledge refresher materials, and allowed participants to evaluate a few instances from each task after onboarding and ask questions. Anonymizing requirements in tasks B and C further minimized bias.

Further, using the same ReqBrain-generated requirements in tasks B and C ensured that differences in evaluations were due

TABLE IX

ISO 29148-COMPLIANT DIMENSIONS OF ADEQUATE: REQBRAIN VS. ITS UNTUNED BASELINE MODEL ABBREVIATIONS: N , SAMPLE SIZE; M , MEDIAN; \tilde{x} , MEAN; p , P-VALUE; $Adj. p$, HOLM-BONFERRONI ADJUSTED P-VALUE; U , MANN-WHITNEY U; A_{12} , VARGHA AND DELANEY'S EFFECT SIZE.

Variables	Author	Hypotheses	N	M	\tilde{x}	s	p	$Adj. p$	U	A_{12}	95% A_{12} CI
Written Syntax Compliance _(WSC)	ReqBrain	$H_{0,3}, H_{a,3}$	n_1 : 136	4	3.78	1.26	< .001	< .001	14203.5	0.76	[0.71, 0.82]
	Untuned Baseline		n_2 : 136	2	2.316	1.45					
Signaling Keywords Compliance _(SKC)	ReqBrain	$H_{0,4}, H_{a,4}$	n_1 : 136	4	3.90	1.27	< .001	< .001	13766.0	0.74	[0.68, 0.80]
	Untuned Baseline		n_2 : 136	2	2.64	1.38					

TABLE X

ISO 29148-COMPLIANT DIMENSIONS OF ADEQUATE: REQBRAIN VS. HUMAN AUTHORS. ABBREVIATIONS: N , SAMPLE SIZE; M , MEDIAN; \tilde{x} , MEAN; p , P-VALUE; $Adj. p$, HOLM-BONFERRONI ADJUSTED P-VALUE; U , MANN-WHITNEY U; A_{12} , VARGHA AND DELANEY'S EFFECT SIZE.

Variables	Author	Hypotheses	N	M	\tilde{x}	s	p	$Adj. p$	U	A_{12}	95% A_{12} CI
Written Syntax Compliance _(WSC)	Human	$H_{0,5}, H_{a,5}$	n_1 : 136	4	3.80	1.20	0.15521	0.31043	10118.5	0.54	[0.48, 0.61]
	ReqBrain		n_2 : 136	4	3.58	1.30					
Signaling Keywords Compliance _(SKC)	Human	$H_{0,6}, H_{a,6}$	n_1 : 136	4	4.11	0.96	0.04068	0.12204	10482.0	0.56	[0.50, 0.63]
	ReqBrain		n_2 : 136	4	3.83	1.14					

TABLE XI

REQBRAIN EFFECTIVENESS ON REMAINING DIMENSIONS OF ADEQUATE. ABBREVIATIONS: n , SAMPLE SIZE; M , MEDIAN; \tilde{x} , MEAN; s , STANDARD DEVIATION; p , P-VALUE; $Adj. p$, HOLM-BONFERRONI ADJUSTED P-VALUE; r , RANK BISERIAL EFFECT SIZE.

Variables	Hypotheses	n	M	\tilde{x}	s	Wilcoxon Signed-Rank	p	$Adj. p$	r	95% r CI
Consistent with Requirements Set _(CRS)	$H_{0,7}, H_{a,7}$	128	4	4.0	1.02	7470.5	< .001	< .001	0.82	[0.72, 0.94]
Identify Missing Requirements _(IMR)	$H_{0,8}, H_{a,8}$	128	4	3.86	1.06	7035.0	< .001	< .001	0.71	[0.58, 0.85]
Enhancing the Overall Completeness _(EOC)	$H_{0,9}, H_{a,9}$	128	4	3.63	1.13	6378.0	< .001	< .001	0.57	[0.44, 0.74]

to output quality and fine-tuning effectiveness rather than data variations. Requirements were anonymized and shuffled, and participants were asked to complete tasks in different orders to minimize carryover effects.

C. External Validity

Our sample sizes were two to three times larger than those determined by our a-priori power analyses, ensuring robust statistical power.

We deem our four evaluators to have adequate job experience and education in the domain to provide a fair generalizability of their ratings. While this enhances the reliability of the insights, we acknowledge that four evaluators constitute a limited representation of cross-domain professionals dealing with requirements, which may limit generalizability.

The evaluation for consistent with, missing from, and enhancing the overall completeness of, a given requirements specification dimensions in RQ2 was based on three distinct software projects coming from the same domain, which may result in not sufficiently representing the full diversity of software development projects. The present study may not wholly represent the real-world utility of ReqBrain: a case study aligned with the ISO 9241-11 definition of usability is underway to further explore this aspect.

D. Conclusion Validity

Given the ordinal nature of Likert scale responses, we used the median over the mean. We employed non-parametric statistical tests that do not assume normality or equal intervals. These choices ensure our statistical analyses are valid and appropriately aligned with the characteristics of our data.

IX. CONCLUSION

We present ReqBrain, a fine-tuned large language model (LLM) and tool to generate authentic and adequate requirements supporting the AI-assisted requirements generation approach.

To develop ReqBrain, we created an instruct dataset by manually reviewing and incorporating high-quality human-written requirements that are ISO 29148-compliant, ensuring that the training dataset reflects the real-world language and the nuanced complexities of industry requirements. Further, we fine-tuned five 7-billion-parameter models on the created instruct dataset.

ReqBrain, our fine-tuned variant of Zephyr-7b-beta, demonstrated superior performance with an 89.30% F1 score using the BERT Score and 91.20 (out of 100) using the FRUGAL score, significantly outperforming other baseline fine-tuned models and the general-purpose ChatGPT-4o.

The human evaluation further confirmed that ReqBrain-generated requirements are authentic and indistinguishable from those written by humans, underscoring the model's ability to produce outputs that meet industry standards in terms of clarity, coherence, relevance, realism, and implementability. Furthermore, ReqBrain demonstrates significant potential in generating adequate requirements that are ISO 29148-compliant, consistent with, missing from, and enhancing the overall completeness of a given requirements specification.

ReqBrain can shorten the labor-intensive, manual process that is often prone to inconsistencies in traditional requirements development. It automates both the elicitation and specification phases by merging them into a single, efficient workflow, enabling the real-time, semi-automated creation of clear and actionable requirements instead of merely collecting raw data and then transforming it. This integration can reduce manual overhead and shorten iteration cycles while ensuring that the early requirements generated are authentic and adequate, thereby supporting requirements engineers in making strategic decisions such as accepting, rejecting, or modifying the generated requirements.

In summary, ReqBrain is a novel contribution, integrating fine-tuned LLMs into the requirements engineering process, marking a significant step toward an AI-assisted requirements generation approach.

Future work will expand ReqBrain's application in requirements engineering. We are currently conducting an industrial case study on ReqBrain's usability in terms of efficiency, effectiveness, satisfaction, and process integration. Simultaneously, we are developing a RAG-enabled version to process company data, including logs, legacy documentation, and project records, to generate contextually accurate, ISO-compliant requirements. Additional work may broaden the instructional dataset to include acceptance criteria, use cases, user stories, and scenarios.

REFERENCES

- [1] D. Zowghi and C. Coulin, "Requirements Elicitation: A Survey of Techniques, Approaches, and Tools," *Springer-Verlag*, pp. 19–46, 2005.
- [2] A. Ferrari, P. Spoleitini, and S. Gnesi, "Ambiguity as a resource to disclose tacit knowledge," in *2015 IEEE 23rd International Requirements Engineering Conference (RE)*, 2015, pp. 26–35.
- [3] V. e. a. Gervasi, "Unpacking Tacit Knowledge for Requirements Engineering," in *Managing Requirements Knowledge*. Springer Berlin Heidelberg, 2013, pp. 23–47.
- [4] A. Distanont, H. Haapasalo, M. Vaananen, and J. Lehto, "The engagement between knowledge transfer and requirements engineering," *International Journal of Management, Knowledge and Learning*, vol. 1, no. 2, pp. 131–156, 2012.
- [5] D. Méndez Fernández, S. Wagner, and et al., "Naming the pain in requirements engineering: Contemporary problems, causes, and effects in practice," *Empirical Software Engineering*, vol. 22, pp. 2298–2338, 2017.
- [6] A. Sutcliffe and P. Sawyer, "Requirements elicitation: Towards the unknown unknowns," in *2013 21st IEEE International Requirements Engineering Conference (RE)*, 2013, pp. 92–104.
- [7] M. K. Habib, S. Wagner, and D. Graziotin, "Detecting Requirements Smells With Deep Learning: Experiences, Challenges and Future Work," in *2021 IEEE 29th International Requirements Engineering Conference Workshops (REW)*, 2021, pp. 153–156.
- [8] A. Mavin, P. Wilkinson, A. Harwood, and M. Novak, "Easy Approach to Requirements Syntax (EARS)," in *2009 17th IEEE International Requirements Engineering Conference*, 2009, pp. 317–322.
- [9] D. Hovy and D. Yang, "The Importance of Modeling Social Factors of Language: Theory and Practice," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021, pp. 588–602.
- [10] Z. Shakeri Hossein Abad, V. Gervasi, D. Zowghi, and K. Barker, "ELICA: An Automated Tool for Dynamic Extraction of Requirements Relevant Information," in *2018 5th International Workshop on Artificial Intelligence for Requirements Engineering (AIRE)*, 2018, pp. 8–14.
- [11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, pp. 4171–4186.
- [12] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving Language Understanding by Generative Pre-Training," 2018.
- [13] "ISO/IEC/IEEE 29148:2018," <https://www.iso.org/standard/72089.html>, 2018, [Accessed: Dec. 26, 2023].
- [14] X. Hou, Y. Zhao, and et al., "Large Language Models for Software Engineering: A Systematic Literature Review," *ACM Trans. Softw. Eng. Methodol.*, vol. 33, no. 8, pp. 1–79, Nov. 2024.
- [15] J. Zhang and S. e. a. Kapse, "Prompt-MIL: Boosting Multi-Instance Learning Schemes via Task-specific Prompt Tuning," arXiv preprint arXiv:2303.12214, 2023, available: <http://arxiv.org/abs/2303.12214>.
- [16] Y. Gu, X. Han, Z. Liu, and M. Huang, "PPT: Pre-trained Prompt Tuning for Few-shot Learning," arXiv preprint arXiv:2109.04332, 2022, available: <http://arxiv.org/abs/2109.04332>.
- [17] W. Zhou, S. Zhang, H. Poon, and M. Chen, "Context-faithful Prompting for Large Language Models," arXiv preprint arXiv:2303.11315, 2023, available: <https://arxiv.org/abs/2303.11315>.
- [18] C. Arora, J. Grundy, and M. Abdelrazek, "Advancing Requirements Engineering through Generative AI: Assessing the Role of LLMs," arXiv preprint arXiv:2310.13976, 2023, available: <http://arxiv.org/abs/2310.13976>.
- [19] H. Chen, Y. Zhang, Q. Zhang, H. Yang, X. Hu, X. Ma, Y. Yanggong, and J. Zhao, "Maybe Only 0.5% Data is Needed: A Preliminary Exploration of Low Training Data Instruction Tuning," arXiv preprint arXiv:2305.09246, 2023, available: <http://arxiv.org/abs/2305.09246>.
- [20] F. Trad and A. Chehab, "Prompt Engineering or Fine-Tuning? A Case Study on Phishing Detection with Large Language Models," *Machine Learning and Knowledge Extraction*, vol. 6, no. 1, pp. 367–384, Mar. 2024.
- [21] K. Ronanki, C. Berger, and J. Horkoff, "Investigating ChatGPT's Potential to Assist in Requirements Elicitation Processes," in *2023 49th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, 2023, pp. 354–361.
- [22] M. Hu, B. He, Y. Wang, L. Li, C. Ma, and I. King, "Mitigating Large Language Model Hallucination with Faithful Finetuning," arXiv preprint arXiv:2406.11267, 2024, available: <http://arxiv.org/abs/2406.11267>.
- [23] Y. Wang and S. e. a. Si, "Two-stage LLM Fine-tuning with Less Specialization and More Generalization," arXiv preprint arXiv:2211.00635, 2024, available: <http://arxiv.org/abs/2211.00635>.
- [24] G. Voria, F. Casillo, C. Gravino, G. Catolino, and F. Palomba, "RECOVER: Toward the Automatic Requirements Generation from Stakeholders' Conversations," arXiv preprint arXiv:2411.19552, 2024, available: <http://arxiv.org/abs/2411.19552>.
- [25] H. Washizaki, *Guide to the Software Engineering Body of Knowledge (SWEBOK Guide)*, 4th ed. Los Alamitos, CA, USA: IEEE Computer Society, 2024.
- [26] C. of Excellence for Software & Systems Traceability, "Coest," <http://sarec.nd.edu/coest/datasets.html>, 2002.
- [27] S. Abualhaija, "frieden84/nlp4re-reveal: Second release," *Zenodo*, 2021. [Online]. Available: <https://doi.org/10.5281/zenodo.4471411>
- [28] A. Ferrari, G. O. Spagnolo, and S. Gnesi, "Pure: A dataset of public requirements documents," in *2017 IEEE 25th International Requirements Engineering Conference (RE)*, 2017, pp. 502–505.
- [29] V. Ivanov, A. Sadovykh, A. Naumchev, A. Bagnato, and K. Yakovlev, "Extracting Software Requirements from Unstructured Documents," in *Recent Trends in Analysis of Images, Social Networks and Texts: 10th International Conference, AIST 2021, Tbilisi, Georgia, December 16–18, 2021, Revised Selected Papers*, 2022, pp. 17–29.
- [30] E. Knauss, S. H. Houmb, S. Islam, J. Jürjens, and K. Schneider, "Secreq," *Zenodo*, 2021. [Online]. Available: <https://doi.org/10.5281/zenodo.4530183>
- [31] Z. shaukat, R. Naseem, and M. Zubair, "Software requirement risk prediction dataset," *Zenodo*, 2018. [Online]. Available: <https://doi.org/10.5281/zenodo.1209601>

- [32] S. o. V. Green Mountain Care Board, "Rfp attachment #1: Functional and nonfunctional requirements for vhcures 3.0," 2018. [Online]. Available: <https://gmcboard.vermont.gov/content/RFP-Attachment1-functional-and-nonfunctional-requirements-vhcures-30-rfp>
- [33] F. Dalpiaz, D. Dell'Anna, F. B. Aydemir, and S. Çevikol, "Requirements Classification with Interpretable Machine Learning and Dependency Parsing," in *2019 IEEE 27th International Requirements Engineering Conference (RE)*, 2019, pp. 142–152.
- [34] J. H. Hayes, "The requirements tracing on target (retro).net dataset," *Zenodo*, 2018. [Online]. Available: <https://doi.org/10.5281/zenodo.1223649>
- [35] O. Karras, "Eye tracking experiments data set - linking use cases and associated requirements: On the impact of linking variants on reading behavior," *Zenodo*, 2021. [Online]. Available: <https://doi.org/10.5281/zenodo.4778899>
- [36] Eclipse Foundation, "Eclipse public license-v2.0," 2017. [Online]. Available: <https://www.eclipse.org/legal/epl-2.0/>
- [37] A. Kumar, A. Raghunathan, R. Jones, T. Ma, and P. Liang, "Fine-Tuning can Distort Pretrained Features and Underperform Out-of-Distribution," arXiv preprint arXiv:2202.10054, 2022, available: <https://arxiv.org/abs/2202.10054>.
- [38] K. Gupta, B. Thérien, and et al., "Continual Pre-Training of Large Language Models: How to (re)warm your model?" in *International Conference on Machine Learning*, 2023, p. 1.
- [39] T. Brown, B. Mann, Ryder, and et al., "Language Models are Few-Shot Learners," in *Advances in Neural Information Processing Systems*, Vancouver, Canada, 2020, pp. 1877–1901.
- [40] "Efficient Few-Shot Learning Without Prompts," Tunstall, Lewis and Reimers, Nils and Jo, Unso Eun Seo and Bates, Luke and Korat, Daniel and Wasserblat, Moshe and Pereg, Oren, 2022, available: <http://arxiv.org/abs/2209.11055>.
- [41] H. Wang, "Pre-Trained Language Models and Their Applications," *Engineering*, vol. 25, pp. 51–65, Jun. 2023.
- [42] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding with unsupervised learning," 2018.
- [43] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "LoRA: Low-Rank Adaptation of Large Language Models," arXiv preprint arXiv:2106.09685, 2021, available: <http://arxiv.org/abs/2106.09685>.
- [44] J. Kirkpatrick and R. e. a. Pascanu, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the National Academy of Sciences*, vol. 114, no. 13, pp. 3521–3526, mar 2017.
- [45] R. Chitale, A. Vaidya, A. Kane, and A. Ghotkar, "Task Arithmetic with LoRA for Continual Learning," arXiv preprint arXiv:2311.02428, 2023, available: <http://arxiv.org/abs/2311.02428>.
- [46] Y. Zhai, S. Tong, X. Li, M. Cai, Q. Qu, Y. J. Lee, and Y. Ma, "Investigating the Catastrophic Forgetting in Multimodal Large Language Models," arXiv preprint arXiv:2309.10313, 2023, available: <http://arxiv.org/abs/2309.10313>.
- [47] A. Aghajanyan, L. Zettlemoyer, and S. Gupta, "Intrinsic Dimensionality Explains the Effectiveness of Language Model Fine-Tuning," arXiv preprint arXiv:2012.13255, 2020, available: <http://arxiv.org/abs/2012.13255>.
- [48] A. Creswell, M. Shanahan, and I. Higgins, "Selection-Inference: Exploiting Large Language Models for Interpretable Logical Reasoning," arXiv preprint arXiv:2205.09712, 2022, available: <https://arxiv.org/abs/2205.09712>.
- [49] M. T. R. Laskar, X.-Y. Fu, C. Chen, and S. B. TN, "Building Real-World Meeting Summarization Systems using Large Language Models: A Practical Perspective," arXiv preprint arXiv:2310.19233, 2023, available: <https://arxiv.org/abs/2310.19233>.
- [50] M. Facebook, "Llama 2: Open Foundation and Fine-Tuned Chat Models | Research - AI at Meta," 2023. [Online]. Available: <https://ai.meta.com/research/publications/llama-2-open-foundation-and-fine-tuned-chat-models/>
- [51] A. Q. Jiang, A. Sablayrolles, and A. e. a. Mensch, "Mistral 7B," arXiv preprint arXiv:2310.06825, 2023, available: <http://arxiv.org/abs/2310.06825>.
- [52] L. Tunstall, E. Beeching, and et al., "Zephyr: Direct Distillation of LM Alignment," arXiv preprint arXiv:2310.16944, 2023, available: <http://arxiv.org/abs/2310.16944>.
- [53] E. Almazrouei, H. Alobeidli, and et al., "Falcon-40B: an open large language model with state-of-the-art performance," 2023.
- [54] H. Face, "Hugging face," 2024. [Online]. Available: <https://huggingface.co/docs>
- [55] J. Hoffmann and e. a. Borgeaud, "Training Compute-Optimal Large Language Models," arXiv preprint arXiv:2203.15556, 2022, available: <http://arxiv.org/abs/2203.15556>.
- [56] H. Schuff, L. Vanderlyn, H. Adel, and N. T. Vu, "How to do human evaluation: A brief introduction to user studies in NLP," *Natural Language Engineering*, vol. 29, no. 5, pp. 1199–1222, 2023.
- [57] A. G. Greenwald, "Within-subjects designs: To use or not to use?" *Psychological Bulletin*, vol. 83, no. 2, pp. 314–320, 1976.
- [58] S. Wagner, M. M. Baron, D. Falesi, and S. Baltes, "Towards Evaluation Guidelines for Empirical Studies involving LLMs," arXiv preprint arXiv:2411.07668, 2025, available: <http://arxiv.org/abs/2411.07668>.
- [59] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "BERTScore: Evaluating Text Generation with BERT," arXiv preprint arXiv:1904.09675, 2020, available: <http://arxiv.org/abs/1904.09675>.
- [60] M. K. Eddine, G. Shang, A. J.-P. Tixier, and M. Vazirgiannis, "FragalScore: Learning Cheaper, Lighter and Faster Evaluation Metrics for Automatic Text Generation," arXiv preprint arXiv:2110.08559, 2021, available: <http://arxiv.org/abs/2110.08559>.
- [61] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a Method for Automatic Evaluation of Machine Translation," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, 2002, pp. 311–318.
- [62] C.-Y. Lin, "ROUGE: A Package for Automatic Evaluation of Summaries," in *Text Summarization Branches Out*, 2004, pp. 74–81.
- [63] M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul, "A Study of Translation Edit Rate with Targeted Human Annotation," in *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers*, 2006, pp. 223–231.
- [64] F. Liu and Y. Liu, "Exploring Correlation Between ROUGE and Human Evaluation on Meeting Summaries," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 1, pp. 187–196, jan 2010.
- [65] D. Elliott and F. Keller, "Comparing Automatic Evaluation Measures for Image Description," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2014, pp. 452–457.
- [66] J. Novikova, O. Dušek, A. C. Curry, and V. Rieser, "Why We Need New Evaluation Metrics for NLG," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 2231–2240.
- [67] I. Turc, M.-W. Chang, K. Lee, and K. Toutanova, "Well-Read Students Learn Better: On the Importance of Pre-training Compact Models," arXiv preprint arXiv:1908.08962, 2019, available: <http://arxiv.org/abs/1908.08962>.
- [68] N. L. Leech and A. J. Onwuegbuzie, "A call for greater use of nonparametric statistics," *Annual Meeting of the Mid-South Educational Research Association*, 2002.
- [69] A. Arcuri and L. Briand, "A practical guide for using statistical tests to assess randomized algorithms in software engineering," in *Proceedings of the 33rd International Conference on Software Engineering*. ACM, 2011, pp. 1–10.
- [70] A. Vargha and H. D. Delaney, "A Critique and Improvement of the CL Common Language Effect Size Statistics of McGraw and Wong," *Journal of Educational and Behavioral Statistics*, vol. 25, no. 2, pp. 101–132, Jun. 2000.
- [71] S. Holm, "A Simple Sequentially Rejective Multiple Test Procedure," *Scandinavian Journal of Statistics*, vol. 6, no. 2, pp. 65–70, 1979.
- [72] T. Dyba, V. B. Kampenes, and D. I. K. Sjøberg, "A systematic review of statistical power in software engineering experiments," *Information and Software Technology*, vol. 48, no. 8, pp. 745–755, Aug. 2006.