Generation of test datasets using LLM - Quality Assurance Perspective

Jose Leandro Sousa, Cristian Souza, Raiza Hanada, Diogo Nascimento, Eliane Collins
{jose.sousa,cristian.souza,raiza.hanada,diogo.nascimento,eliane.collins}@indt.org.br
Instituto de Desenvolvimento Tecnológico
Manaus, Amazonas, Brazil

ABSTRACT

Domain relevant data and an adequate number of samples are necessary to properly evaluate the robustness of the Machine Learning (ML) models. This is the case for ML models used in the software localization translation task. In general, Neural Machine Translation (NMT) models are used in software localization by automating the translation process of textual content to consider specific linguistic aspects and culture. However, unlike general machine translation which can easily utilize translation corpus for model training and testing, domain-specific machine translation faces a major obstacle due to the scarcity of domain-specific translation data. In the absence of adequate data, this paper first presents a method to generate test samples based on a text generation Large Language Model (LLM) approach. Based on the generated samples, we run tests to assess the robustness of an NMT translation model. The evaluation indicates that human judgment is important to check if the generated text is robust and coherent under different conditions. The evaluation also demonstrates that the generated samples were crucial to show some limitations related to the model's effectiveness in software localization translation. Basically we discuss issues in specific situations such as date, time formats, numeric representations and measurement units.

CCS CONCEPTS

• **Software and its engineering** → **Acceptance testing**; • **Computing methodologies** → *Machine translation.*

KEYWORDS

dataset creation, LLMs evaluation, text generation, neural machine translation, software translation

1 INTRODUCTION

The exponential growth of digital data has provided Machine Learning (ML) algorithms with large and diverse datasets for training [3]. This abundance allows models to learn more complex patterns and improve their predictive capabilities.

However ML models often require specific datasets to perform well. It is important that data used for training a model aligns with the real-world scenarios or problems the model is intended to address [7]. For example, in Natural Language Processing (NLP), a dataset containing medical texts is considered relevant to the healthcare domain, as it captures the language and terminology commonly used in medical contexts. Other instance where we need domain-relevant data is in the software technology domain, specifically in the software localization task.

Software localization is the process of adapting a software application, including its user interface and content, to suit the linguistic, cultural, and functional requirements of a specific target audience or locale [15]. The goal of software localization is to make the software appear as though it was originally created for the target market, ensuring a seamless and culturally appropriate user experience. The ability to adapt the software main language to the language and culture of the end user is essential for reaching global markets and customers [9].

The translation step is a crucial part in software localization [27]. The translation step is labor-intensive and often requires a significant amount of effort from the development teams [5]. A possible solution is to use Neural Machine Translation (NMT) approaches. In fact, NMT is is the current state-of-the-art approach for generating the initial "draft" for the translations [16].

A problem arises when insufficient data is available in the NMT process test. Although there are several translation datasets in the literature, there are a lack of datasets specifically focused on the software domain [27]. Moreover, the test dataset must be representative to ensure that there are enough samples to evaluate the performance of a translation model for specific test cases. In our work, we present an innovative approach that leverages a LLM to generate a test dataset in the software domain. This approach enables the evaluation of translation model performance for specific scenarios, accommodating rules that vary by language, including formatting and cultural aspects relevant to the country where the software will be applied. An example is the decimal separator symbol, which is the decimal point (.) for English-speaking countries, and is a comma (,) in many European countries, South America, and parts of Asia.

In this paper, we present a process on data generation for NMT testing. Specifically, we use the Large Language Model (LLM) LLaMa 2-Chat versions [21] to generate test samples and boost the number of test samples available. We discuss aspects related to prompt engineering [13] and then in the quality control step we point out limitations. To prove its utility for software localization, we used the newly generated test samples to evaluate the NMT models' robustness.

Our results reveal that LLM test sample generation faces some issues and still requires human validation. In this work we report some issues such as duplicates and similar samples, out of context samples or samples that do not fit our requirements. In addition, the generated test samples proved to be crucial in pointing out some limitations related to the translation task. Basically they revealed translation issues related to date, time formats, numeric representations and measurement units.

According to our literature searches, several works have been proposed using LLMs in software testing (test case generation, test input generation, debugging and other tasks). In addition, we observe recent emergence of studies that uses LLMs to create test datasets for evaluating ML models, which points to the innovative nature of our work. This paper still makes the following contributions:

- we present an approach to generate samples based on a LLM and then we used the generated test dataset to evaluate a NMT model. In particular, we report the process and metrics.
- we present a quality protocol to evaluate the quality, relevance, and appropriateness of the generated samples for the software localization purpose.
- we used traditional software testing principles to evaluate an ML model, specifically a Neural Machine Translation (NMT) model.

The remainder of the paper is structured as follows. Section 2 gives an overview of the text generation, LLMs and the principles of prompt engineering used in this work. The subsequent section 3 describes a test data generation approach based on a LLM. Section 4 describes and evaluates (i) the generated test samples and (ii) tests to evaluate the translation model. Section 5 concludes the paper and provides future directions for continuing this study.

2 BACKGROUND AND RELATED WORKS

2.1 Text Generation and LLMs

Text data generation involves the creation of new textual content using Artificial Intelligence (AI) or NLP techniques[22]. In literature, there are various approaches to text data generation. Basic methods include Rule-based Generation, where text is created based on predefined rules and templates, and Statistical Language Models, like n-gram models, which analyze statistical relationships between words in a dataset [17].

Advanced approaches encompass ML models, particularly those utilizing deep learning techniques such as Recurrent Neural Networks (RNNs) [19], Long Short-Term Memory (LSTM) networks [23], and Transformers [24].

Pre-trained Language Models (PLMs) [25] have gained prominence in recent years. The fundamental concept involves pre-training models on large-scale unsupervised *corpora* and then fine-tuning them in downstream supervised tasks, achieving state-of-the-art performance. With the advent of Transformer architecture and increased computational power, PLMs have evolved from shallow to deeper architectures such as the large-scale pre-trained LLMs, including BERT [10], OpenAI GPT [2], and LLaMa [21].

Since their release, LLMs have been tested into a variety of downstream tasks and achieved impressive results by providing a few examples as instructions, rather than collecting large-scale task-specific data or tuning model parameter [6].

This technique, commonly called *few-shot learning*, helped to overcome issues related to low-quality and low-quantity data, which often leads to variance in the quality of the generated text, and a bias towards the over-represented registers, by carefully selecting a small set of high-quality data as demonstrations [11].

Studies are being made to understand the actual range of current LLM models' effectiveness in providing trustworthy annotation for unlabeled data in different tasks, such as *creation of a taxonomy, shortening text, writing a short story* [12], *user query and keyword relevance assessment* [28], disambiguation of word senses through binary classification of sentence pairs [28], *question-answering* [28], generation of queries and query variants from a description [4].

In software testing domain, the LLM's have been used for several tasks, such as *test case generation,test input generation*, *bug analysis*, *debug*, and *program repair* [26]. We also have found studies that have used LLMs to generate data for training and testing ML models [1], [8].

In [27], the authors described a large-scale human-translated bilingual sentence pairs dataset from different Android applications, which are crawled from the Google Play Store. The training corpus consists of millions of aligned dual-language sentence pairs (e.g., English-Chinese) extracted from language packs of Android applications. This dataset is instrumental for translating text in mobile apps for app localization. However, to the best of our knowledge, we did not find a test dataset based on an LLM that focuses on specific test cases for software localization.

With LLMs model advancements, concerns about their limitations and risks have grown. It's been observed that these models can produce nonsensical or inaccurate text, termed as "*hallucinations*" by researchers [18].

Within the NML field, LLMs are being used to create new *State-of-the-art* Translation models, either by using few-shot prompts composed of synthetic translations made by the LLM itself to improve the dataset and provide high-quality translations [11, 14].

2.2 Prompt Engineering

The text used as input to a language model is called the *prompt*. The work of developing and improving the prompt is known as *prompt engineering* [13]. In practice, it is common for the model not to produce the desired result on the first attempt. It was therefore necessary to revise the language of the prompt or the way it is written several times to make the model behave in the desired way. A powerful strategy for making the model produce better results is to include examples of the task you want the model to perform in the prompt. Providing examples within the context window is called *context learning*. With contextual learning, we direct LLMs to learn more about the requested task by including examples or additional data in the prompt.

As well as the *prompt engineering*, there are some other parameters that can be modified to make the LLM *model's output* work as the user wants to. The *temperature* parameter controls the randomness and creativity in the generated outputs. The temperature parameter influences the probability distribution of the next word or token in the generated sequence.

In simpler terms, a higher temperature encourages the model to take more risks and explore different possibilities, leading to more varied and creative outputs. On the other hand, a lower temperature makes the model more conservative, favoring more probable choices and generating more controlled and deterministic outputs. The choice of temperature depends on the desired characteristics of the generated text.

2.3 Neural Machine Translators

Machine Translator (MT) is a classical subfield of NLP (Natural Language Process) with the aim of automatically translating a written text or speech from one natural language to another [29]. In general, the task of translation consists in processing a source sequence of symbols belonging to a source language into a target sequence belonging to a target language.

The first MT models were based on rules and followed the idea that every word in the source language had an equivalent word with the same meaning in the target language, so the translation process was treated as word replacement in the source sentence. The MT evolved to Statistical Machine Translation which used bilingual corpora for statistics to find words (or phrases) that have the same meaning [29]. The most recent models, known as Neural Machine Translators (NMT), use a neural network to train all parts of the neural translation model jointly to find the correct target sequence given the source sequence [20]. This way, the NMT tries to map both source and target into a single semantic space.

There are many practical applications for NMTs, ranging from tourism and education to business and communication. Currently, NMT models are deployed in various systems by Google, Microsoft, Facebook, Amazon, among others [20].

3 PROPOSED SOLUTION

In this section we present our approach to generate test samples based on LLMs for software localization testing. Here's our approach step by step including methodology information:

- LLM choice: first we had to select a large pre-trained language model. Examples include GPT-3, BERT, or other state-of-the-art models. Given that the task of software localization is more exploited in industry, we chose to use an Open Source model. Thus, we choose to use the LLaMa 2 - chat version (7B and 13B variants). In the model's loading step, we used a 4-bit quantization process that makes them lighter without impacting the results quality. We used a Google Collaboratory notebook using a T4 execution environment. The main goal here is to compare the 2 variants and check if a specific one generates better results than the other.
- Test Cases/Suites definition: here we clearly define specific types of samples that we need. In the Software localization, some key aspects are involved such as Date and Time Formats, Measurement Units and language variants.
- Test Samples Generation: we used the LLaMa 2 (7b and 13B) to generate test samples based on the defined format. We defined a prompt to the model and let it generate text accordingly.
- Diversify and Augment Samples: to enhance the diversity of our test samples, we vary the input prompts. We also experiment with different parameters, such as temperature (controlling randomness in the generated output), to get a range of samples.
- Quality Control: in this step we review and evaluate the generated test samples to ensure they meet its quality standards. We run a human evaluation to check for coherence, relevance, and correctness. And then we manually filter the samples to eliminate any that do not align with our requirements.

In the test Cases/suites definition step, we choose the following cases: *Date format*, *Hour format*, *Thousand separator*, *Measure units* and *Questions*. Basically we choose test cases related to date and time formats, as well as numeric representations, to match the conventions of the target locale. This ensures that the software reflects the customary way in which dates, times, and numerical values are presented in the local culture. We also include test cases related to Measurement Units that involves converting measurement units (e.g., metric or imperial) to those commonly used in the target region. This ensures that measurement-related information is presented in a familiar and understandable manner.

Test case	Qnt	Example
Thousand Marks	2	The device has 1,000GB of storage space.
Date Format	6	The sale ends on 12/10/2024.
Measure Units	-	The laptop weighs 4.5lbs.
Hour Format	4	The store will open at 9 AM.
Questions	-	Can you confirm that you want to log out?

Table 1: Test cases information

Table 1 shows more information about the test Cases used in this study. As we can see from the table, in all test cases we have insufficient data in the original dataset.

In the Test Samples Generation, we have to test and define first a system prompt to generate the samples. We defined the following system prompt [1] as input to the LLaMa 2 models. As discussed, we clearly and concisely define the test cases and include examples.

You are a linguistics specialist, that generates phrases in a software context following these test cases:

- *Thousand Marks: Phrases with thousand marks on numbers (ex.: You can store up to 20,000 songs from your library in the cloud)*
- *Date Format: Phrases with date format like this month/day/year (ex.: "The software update will be released on 01/24/2023", "The maintenance window is from 03/11/2019 to 10/27/2020.")*
- *Measure Units: Phrases with American measurement units like "lbs","inches","miles","ounce","yard","feet".(ex.: "The UI will be scaled to 15 inches or larger")*
- *Hour Format: Phrases with hour specifically in the 12 hour clock format (ex.: "It's late! It's already 09:00 PM","Sent at 11:04 AM on Friday")*
- *Questions: Questions using a direct style (i.e., implying "do you want"). Example: "Do you want to delete this picture?","Would you like to save this picture?"*

Then we can use instruction prompts to generate the test samples. For example *"Write N sentences based on the test case thousand marks"*. These test samples are used as input on the NMT and can be used to evaluate the translation of any target language as long as its localized format is known. Considering the target language is Portuguese, from each test case, we will be able to assess whether the NMT model is capable of inverting the order of the day and month for the *Date format*, changing the *Hour format* to a 24-hour format, changing the *Thousand separator* of the input samples from

[1]Note that initially other prompts were tested, including more detailed ones, until we arrived at the choice of prompts used in the solution

a comma to a period, converting the *Measure units* to the metric system (i.e. meters, kilograms, liters, etc.) and creating more direct *Questions*.

In the Diversify and Augment Samples step, the goal is to ensure diversity in order to have as many representative samples as possible. Thus, we run an experiment using different temperature values. We used 3 *temperature* values on LLaMa 2. Table 2 shows examples from each temperature setup. We noticed that when using the temperature level 1, the results are very similar to each other. On the other side, a higher temperature as 3 returns some unpredictable results that are not consistent to the requirements specified in the System Prompt. The temperature level as 2 returns the best representative and variate samples according to the system prompt specifications and then we adopted it in our experiments.

Temp.	Samples
1	The application will be available on 04/01/2023.
	The application will be available on 04/14/2023.
	The project will be completed on 09/30/2023.
	The project will be started on 03/01/2023.
2	Deadline is on 05/22/2023.
	Please submit your report by 03/31/2023.
	The conference call is on 12/15/2023 at 11:00 AM.
	The maintenance window is from 03/11/2019 to 10/27/2020.
3	The application can sync your calendar data between May 15, 22:05 - Jun 15, 04:35 AM
	The event occurs on the fifth of February 22:30 of the current year
	The wedding is on March twenty fourth 05. At half ten, past four
	Our summer holidays began the third day past December 8, on Wed, March 4 at exactly 2019 at noon

Table 2: Samples generated using 13B *temperature* 1, 2 and 3.

Then we generated for each test case and model variation (7B and 13B) at least 50 samples.

And finally, in the Quality Control step, we ran a qualitative assessment over the generated samples. We noticed inconsistent samples that should be removed. We identified 4 exclusion criteria:

- Duplicates: Samples that appeared more than once in the testset
- Out of requested: Samples that did not cover the test case we want to evaluate
- Similar: Samples that were very similar to other already existing in the testset
- No sense: Samples that were not consistent with reality

We considered similar samples basically situations we notice small difference between the samples such as ("You can expect the new version of the software to be released on 08/25/2023." ,"You can expect the new version of the software to be released on 08/30/2023.") and ("The new software feature will be available on 04/27/2023." , "The next software update will be available on 05/12/2023.").

In addition, *No sense* samples are basically out of context sentences. For example, "The server will be rebooted in 5 miles or less" in which a sample referring to time is related a unit of length and "You have 07:00 AM to complete the task before the deadline", where a wrong time formatting was inserted. We removed these samples once they have a high potential to confuse the model translation.

In the Quality Control step, 3 authors evaluated each sample in a blind review process. Once a test set with the 250 correct samples (50 for each test case) was obtained, we ran the translation model and we compared its performance.

4 EVALUATION

The evaluation was divided in two phases: in the first phase we evaluate the ability of the LLM in creating a valid testset; in the second one we use the generated testset to evaluate the NMT model. Below we show the main findings for each phase.

4.1 Generation of test samples

Following the exclusion criteria, we removed more samples when using LLaMa7B compared to LLaMa13B, especially for the *Hour format* and *Measure units* cases. From 250 samples (50 from each test case), we removed 80 samples generated by LLaMa7B (32%) and 29 samples generated by LLaMa13B (11%) as we can see in Figure 1. These numbers can be seen in more details in Figure 2.

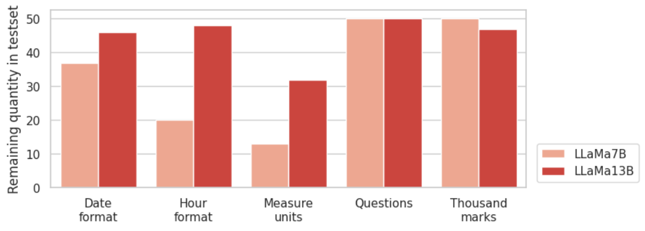

Figure 1: Final quantity remaining for each test case.

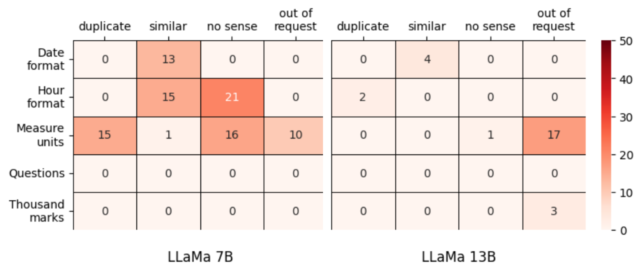

Figure 2: Number of samples discarded according to each exclusion criteria.

We can observe that LLaMa7B repeated the generation of the same samples more than LLaMa13B. In fact, LLaMa7B was not very inclined to diversity of examples. We can also see that LLaMa7B produced many more similar samples (29 samples) than LLaMa13B (4 samples). Both created similar samples for *Date format*, however

LLaMa7B produced more than three times as many similar samples as LLaMa13B. Likewise, LLaMa7B also produced several cases that were still similar but not enough to be removed by the exclusion criteria such as those in Table 3. Table 3 also shows the most similar samples produced by LLaMa13B among those that were not excluded. We also can find a huge amount of similar samples produced to *Hour format*.

Model	Sample
LLaMa 7B	"The software patch will be available on 07/25/2023."
	"The software patch will be released on 02/28/2023."
	"The software update will be available on 10/15/2023."
	"The software update will be released on 03/15/2023."
LLaMa 13B	The meeting is on 11/17/2023 at 2:00 PM.
	The meeting is scheduled for 02/20/2023 at 10:00 AM.
	The meeting will be held on 09/22/2022 at 2:00 PM.
	The meeting will be on 05/17/2025.

Table 3: Example of similar samples produced by LLaMa7B and LLaMa13B for *Date format* that was kept in the testset.

LLaMa7B also generated more samples out of context. Table 4 shows some removed samples produced by LLaMa 7B according to the *No sense* criteria. In the samples generated by LLaMa7b, we can still observe the similarity in the samples for both *Hour format* and *Measure units*. For *Hour format* the main mistakes made were the use of erroneous time format and the use of instructions that were very different from what anyone would expect from a mobile software. The same happened to *Measure units*.

Test case	Sample
Hour format	The team meeting will take place at 03:00 PM and last for 01:00 PM.
	The team will meet at 05:00 PM to discuss the project and last for 02:00 PM.
	The team will meet at 09:00 AM to brainstorm ideas and last for 02:00 PM.
	The camera app can take pictures 7 feet away at 7:00 PM.
Measure units	The issue will be fixed in 1 pound or less.
	The update will be deployed in 3 pounds or less.
	The software will be deployed in 5 feet or less.

Table 4: Samples generated by LLaMa 7B and removed according to the *No sense* criteria.

Regarding the samples that were different than requested by the prompt, LLaMa13B generated more samples than LLaMa7B (10 samples). Table 5 shows some errors for *Measure units*. In general, LLaMa7B model created samples using units of measurement belonging to the International System (instead of using the American system), related to units of distance and weight, as we expected. In contrast, the erroneous samples generated by LLaMa13B referred to different types of measurement units. This result is probably related to the ability of LLaMa13B in creating more diverse samples.

Model	Sample
LLaMa7B	The application can handle volumes ranging from 10 cubic centimeters to 100 cubic meters.
	The application can handle weights ranging from 10 grams to 100 kilograms.
LLaMa13B	The smartphone's battery lasts up to 12 hours.
	The smart TV's screen resolution is 4k (3840 x 2160).
	The smart TV's refresh rate is 144Hz

Table 5: Samples removed for being *Out of requested* by the prompt.

LLaMa13B still made mistakes in the creation of samples for *Thousand marks*, generating numbers that were not in the thousands.

When observing each test case individually, the *Measure units* was the one that both models had most difficulty generating valid test cases for. This generation has been difficult, especially for LLaMa7B that remained with only 13 samples out of the 50 generated. LLaMa7B is also struggling in generating samples for the *Hour format* case.

Given the lower effort required to obtain a valid dataset using LLaMa13B, it was therefore decided to explore the LLaMa13B version. We populate the dataset until we have 50 samples for each test case. Note that for *Measure units* we still had to create 11 samples in addition to the 20 required that were removed by the exclusion criteria. All the other cases did not require extra samples.

4.2 Applying on the machine translator

After having a valid test dataset, we finally evaluate the machine translation model for the target language, Portuguese. Table 6 shows the accuracy obtained for each test case in the original testset with few cases and the one generated by LLaMa13B.

Testset	Date format	Hour format	Thous. marks	Quest.	Measure units
Original	0.33	1.00	0.00	-	-
LLaMa13B	0.18	0.52	0.66	0.98	0.06

Table 6: Accuracy of the machine translator for each test case

An important contribution of the test dataset generation was the possibility of evaluating the *Questions* and *Measure units* test cases that had no samples before. We can see that the model translated the questions as required by the rule, but it fails badly to convert the measure units from the American system to the International System of Units. The generated test set also reaffirmed the model's inefficiency in translating *Date formats* (6 samples in the original dataset), showed several errors produced when translating *Hour format* (4 samples in the original dataset), and pointed out that the model was able to hit more cases of the *Thousand mark* (2 samples in the original dataset) than expected.

The qualitative analysis of the translations was even more revealing. From 9 samples in which the model managed to put the correct *Date format*, in 66% it produced hallucinations as in the translation of “Please submit your report by 03/31/2023” to “Envie o relatório até 30/03/2023”. For the *Hour format*, the model made confusions between *am.* and *pm.* as we can see in the translation of “Meeting will start at 10:00 AM and end at 11:00 AM” by “"A reunião terá início às 22:00 e fim às 11:00”. For *Thousand marks*, the major mistake was using a dot instead of non-break space as a thousand separator. In a few cases the model used a separator for numbers below 10,000 or wrote the number extensively (e.g. it wrote “mil” instead of the cardinal number “1 000”). As for *Measure units*, the main error found was the translation of the measurement into Portuguese instead of the conversion from the system (e. g. “miles” for “milhas”). We also noticed that in 88% of the cases in which the measure unit was abbreviated, the model kept the English form.

These findings are particularly important for the development team to be able to assess how to improve the model or correct any failure with post-processing.

5 CONCLUSION

We address the use of LLMs to generate test samples in the software domain for the evaluation of a NMT model in a localization task. For Quality Assurance (QA) point of view, our experiments shows the testing ability of LLMs in creating new test sets for specific tests. Another gain was the possibility to assess the limitations of the NMT model when using the generated test dataset. Knowing these limitations is important in the process of creating more robust NMT models. In turn, more robust models allow a technology manufacturing company to focus more precisely on producing content in a single scope or language, reducing the costs related to human labor.

Two variants of LLaMa 2 - chat version were used (7B and 13B) for evaluating five test cases: *Hour format*, *Date format*, *Measure units* conversion, *Questions* and *Thousand marks*. In general, we observed that LLaMa7B required greater human validation. The high amount of repeated and similar samples shows its inefficiency in diversity, an important requirement for test dataset creation. LLaMa7B also generated several out-of-context samples. In turn, LLaMa13b generated more different cases than requested by the prompt.

Some limitations we identify are the lack of deep investigation of the impact of different parameters on the sample generation, such as prompt, temperature and the use of other models. Although we have carried out a variety of tests on these parameters, a more in-depth study on the impact of each of them on the generation of test cases would be interesting. For instance, the greater difficulty in generating cases for *Measure units* can indicate the parameters used for its case might not have been adequate. Besides, we can observe a high number of similar samples. To bypass these limitations, in the future we intend to work on prompt engineering to explore how prompt modifications can improve the creation of more valid samples. Other localization test cases could also be evaluated, such as currency, decimal markers or translation of specific cases as expressions and slangs. Moreover, to address the problem of diversity of examples, we intend to make an analysis of temperature parameters. We also intend to test other LLMs as BERT, T5 and GPT. In addition, as we understand that human intervention can be reduced by using rules and regular expressions to automatically remove samples with errors, we intend to investigate how well these techniques can reduce human effort.

ACKNOWLEDGMENTS

This work was partially supported by SUFRAMA (Law No. 8387/1991).

REFERENCES

[1] Arkadeep Acharya, Brijraj Singh, and Naoyuki Onoe. 2023. LLM Based Generation of Item-Description for Recommendation System. In *Proceedings of the 17th ACM Conference on Recommender Systems* (Singapore, Singapore) *(RecSys '23)*. Association for Computing Machinery, New York, NY, USA, 1204–1207. https://doi.org/10.1145/3604915.3610647

[2] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774* (2023).

[3] Karan Aggarwal, Maad M Mijwil, Abdel-Hameed Al-Mistarehi, Safwan Alomari, Murat Gök, Anas M Zein Alaabdin, Safaa H Abdulrhman, et al. 2022. Has the future started? The current growth of artificial intelligence, machine learning, and deep learning. *Iraqi Journal for Computer Science and Mathematics* 3, 1 (2022), 115–123.

[4] Marwah Alaofi, Luke Gallagher, Mark Sanderson, Falk Scholer, and Paul Thomas. 2023. Can Generative LLMs Create Query Variants for Test Collections? An Exploratory Study. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '23)*. Association for Computing Machinery, New York, NY, USA, 1869–1873. https://doi.org/10.1145/3539618.3591960

[5] Zeyad Alshaikh, Shaikh Mostafa, Xiaoyin Wang, and Sen He. 2015. A Empirical Study on the Status of Software Localization in Open Source Projects.. In *SEKE*. 692–695.

[6] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. arXiv:2005.14165 [cs.CL]

[7] Lukas Budach, Moritz Feuerpfeil, Nina Ihde, Andrea Nathansen, Nele Noack, Hendrik Patzlaff, Felix Naumann, and Hazar Harmouch. 2022. The effects of data quality on machine learning performance. *arXiv preprint arXiv:2207.14529* (2022).

[8] John Chung, Ece Kamar, and Saleema Amershi. 2023. Increasing Diversity While Maintaining Accuracy: Text Data Generation with Large Language Models and Human Interventions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (Eds.). Association for Computational Linguistics, Toronto, Canada, 575–593. https://doi.org/10.18653/v1/2023.acl-long.34

[9] Rosann Webb Collins. 2001. Software localization: Issues and methods. (2001).

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR* abs/1810.04805 (2018). arXiv:1810.04805 http://arxiv.org/abs/1810.04805

[11] Xavier Garcia, Yamini Bansal, Colin Cherry, George Foster, Maxim Krikun, Melvin Johnson, and Orhan Firat. 2023. The Unreasonable Effectiveness of Few-shot Learning for Machine Translation. In *Proceedings of the 40th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 202)*, Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (Eds.). PMLR, 10867–10878. https://proceedings.mlr.press/v202/garcia23a.html

[12] Madeleine Grunde-McLaughlin, Michelle S. Lam, Ranjay Krishna, Daniel S. Weld, and Jeffrey Heer. 2023. Designing LLM Chains by Adapting Techniques from Crowdsourcing Workflows. arXiv:2312.11681 [cs.HC]

[13] Jindong Gu, Zhen Han, Shuo Chen, Ahmad Beirami, Bailan He, Gengyuan Zhang, Ruotong Liao, Yao Qin, Volker Tresp, and Philip Torr. 2023. A systematic survey of prompt engineering on vision-language foundation models. *arXiv preprint arXiv:2307.12980* (2023).

[14] Jesse Michael Han, Igor Babuschkin, Harrison Edwards, Arvind Neelakantan, Tao Xu, Stanislas Polu, Alex Ray, Pranav Shyam, Aditya Ramesh, Alec Radford, and Ilya Sutskever. 2021. Unsupervised Neural Machine Translation with Generative Language Models Only. arXiv:2110.05448 [cs.CL]

[15] Kenneth Keniston. 1997. *Software localization: Notes on technology and culture.* Program in Science, Technology, and Society, Massachusetts Institute of Technology.

[16] Sai Koneru, Matthias Huck, Miriam Exel, and Jan Niehues. 2023. Analyzing Challenges in Neural Machine Translation for Software Localization. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*. 2434–2446.

[17] Junyi Li, Tianyi Tang, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2022. Pretrained language models for text generation: A survey. *arXiv preprint arXiv:2201.05273* (2022).

[18] Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. *arXiv preprint arXiv:2005.00661* (2020).

[19] Fathi M Salem. 2022. *Recurrent Neural Networks*. Springer.

[20] Felix Stahlberg. 2020. Neural Machine Translation: A Review and Survey. arXiv:1912.02047 [cs.CL]

[21] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288* (2023).

[22] Sowmya Vajjala, Bodhisattwa Majumder, Anuj Gupta, and Harshit Surana. 2020. *Practical natural language processing: A comprehensive guide to building real-world NLP systems*. O'Reilly Media.

[23] Greg Van Houdt, Carlos Mosquera, and Gonzalo Nápoles. 2020. A review on the long short-term memory model. *Artificial Intelligence Review* 53 (2020), 5929–5955.

[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[25] Haifeng Wang, Jiwei Li, Hua Wu, Eduard Hovy, and Yu Sun. 2022. Pre-trained language models and their applications. *Engineering* (2022).

[26] Junjie Wang, Yuchao Huang, Chunyang Chen, Zhe Liu, Song Wang, and Qing Wang. 2024. Software Testing with Large Language Models: Survey, Landscape, and Vision. arXiv:2307.07221 [cs.SE]

[27] Xu Wang, Chunyang Chen, and Zhenchang Xing. 2019. Domain-specific machine translation with recurrent neural network for software localization. *Empirical Software Engineering* 24 (2019), 3514–3545.

[28] Tongshuang Wu, Haiyi Zhu, Maya Albayrak, Alexis Axon, Amanda Bertsch, Wenxing Deng, Ziqi Ding, Bill Guo, Sireesh Gururaja, Tzu-Sheng Kuo, Jenny T. Liang, Ryan Liu, Ihita Mandal, Jeremiah Milbauer, Xiaolin Ni, Namrata Padmanabhan, Subhashini Ramkumar, Alexis Sudjianto, Jordan Taylor, Ying-Jui Tseng, Patricia Vaidos, Zhijin Wu, Wei Wu, and Chenyang Yang. 2023. LLMs as Workers in Human-Computational Algorithms? Replicating Crowdsourcing Pipelines with LLMs. arXiv:2307.10168 [cs.CL]

[29] Shuoheng Yang, Yuxin Wang, and Xiaowen Chu. 2020. A Survey of Deep Learning Techniques for Neural Machine Translation. arXiv:2002.07526 [cs.CL]