A GPT-based method of Automated Compliance Checking through prompt engineering

Xiaoyu Liu, Haijiang Li*, Xiaofeng Zhu
Cardiff University, United Kingdom
LiuX133@cardiff.ac.uk

Abstract: Automated Compliance Checking (ACC) in the Architecture, Engineering, and Construction (AEC) industry can significantly improve project delivery efficiency. This research introduces an early application of Generative Pre-trained Transformer (GPT) models for ACC, requiring no additional domain knowledge or term explanation. Our method involves direct input of building design specifications and corresponding codes into the model, guided by a task-specific prompt. The GPT models then generate compliance results. Initial tests on an artificially generated dataset yielded up to 91% accuracy, demonstrating the model's effectiveness. As an early adopter of applying Large Language Models (LLMs) to AEC challenges, our work offers a practical workflow and dataset for others seeking to leverage GPT in this field. The full paper will discuss potential limitations and challenges of this application.

Introduction:

Compliance checking happens consistently during the entire project execution in the Architecture, Engineering, and Construction (AEC) industry. The traditional manual compliance checking method is time-consuming, unstable, and costly, which is critical to improve performance including efficiency, accuracy, ease to use, and generalization during the process (Malsane et al., 2015; Zhang et al., 2023). The Automated Compliance Checking (ACC) process is recognised as an effective method to solve the issues of the manual compliance checking process (Beach et al., 2015; Soliman-Junior et al., 2021).

Five years of research related to ACC are reviewed in this study. With the advantages of interoperability and flexibility, hard coding and ontology under Building Information Modelling (BIM) environment become common ways to reach the ACC process (Choi et al., 2014; Melzner et al., 2013; Tan et al., 2010; Zhang & El-Gohary, 2017). However, the existing methods have limitations on automation as the extracting of logic or semantic representation from the text information is highly dependent on manual work, which causes issues in that the semantic web is built with very limited generalisation for each different representation generated from different projects, companies, institutions, or individuals that require the manual processing of semantic representation converting (Bloch & Sacks, 2018; Xu & Cai, 2020; Zhang et al., 2022; Zhong et al., 2019). This process could cause duplication of work and conflict among different semantic webs from parties.

In order to further enhance the degree of automation, LLMs are applied to execute ACC tasks. In this research, an LLMs-based method of ACC for building design specifications is developed which can automatically generate logic by Prompt Engineering (PE); an artificially generated domain knowledge of building design specification compliance checking prompt dataset is built for driving LLMs; the state-of-art GPT-3 and GPT-3.5 LLMs are applied and evaluated regarding on this scenario.

Rather than extracts text information from models through specific forms of technology, the workflow ACC process in this research starts from processing prepared text dataset as in the most forms of BIM program, there are common ways to automatically convert building design specifications from typical drawings of CAD or information models into pure text format including ".txt", ".doc", ".xls", and ".csv" etc. The LLMs have the reliable capability of generalization which can be applied to process pure text information in most of the forms.

Section 2 describes the methodology, section 3 explains the design of experiments, section 4 concludes the results and the research.

1. Methodology

This research adopts 2 types of GPT-based LLMs to evaluate the core function and the best capabilities during the task implementation. In general, the core function of the LLMs is text generation according to the given text so-called "Prompt".

1.1 General design

Figure 1 presents the general design of implementing research. According to the architecture of the transformer, the basic scenario of the GPT-based models could be considered as sentence prediction from context. So the first step to drive the LLMs is generating appropriate prompts based on the target scenario. In general, there are 3 types of prompts for LLMs, zero-shot learning, one-shot learning, and few shots learning(Saravia, 2022), the details of the prompt design are explained in Session 4. Experiment. When the prompts have been prepared, they are fed in LLMs separately in 2 different ways, one way is directly feeding in complete models, and the other way is applying fine-tuning process before the test. Both of the models produce completions (or results) based on a prompt. According to the qualities of completions, the models' performance can be assessed. Furthermore, both performance of prompt and LLMs can be analyzed through variables controlled during the experiment. Finally, according to the analyses of the prompts and models, the prompts can be modified, and correspondingly the models with the best performance are applied for further research.

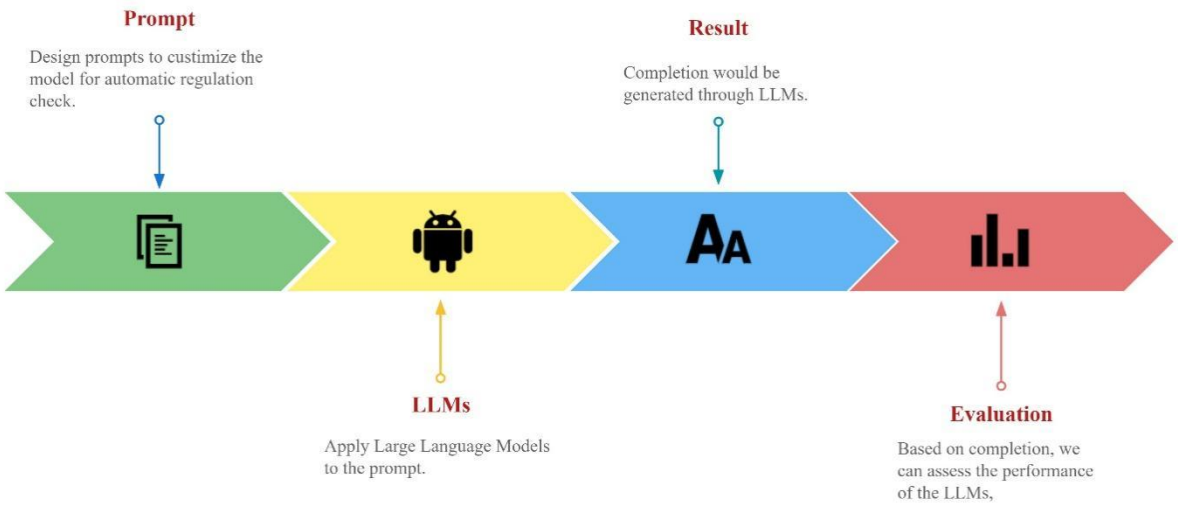

Figure 1: the design of the research implementation.

Figure 2 presents the detailed processing procedure for realizing LLMs-based ACC through prompt engineering. According to the figure, 2 types of models are applied, GPT-3-based fine-tuned models and GPT-3.5-based complete models. As reinforcement learning methods are integrated, GPT-3.5-based models present a better performance in multi-tasks. Consequently, they can realize deep analysis to large complex (maximum 4096 tokens in GPT-3.5 based models) contexts without fine-tuning. However, GPT-3 base models don't have such powerful language processing performance in general, but the models can be boosted through the fine-tuning process to reach the same level of performance.

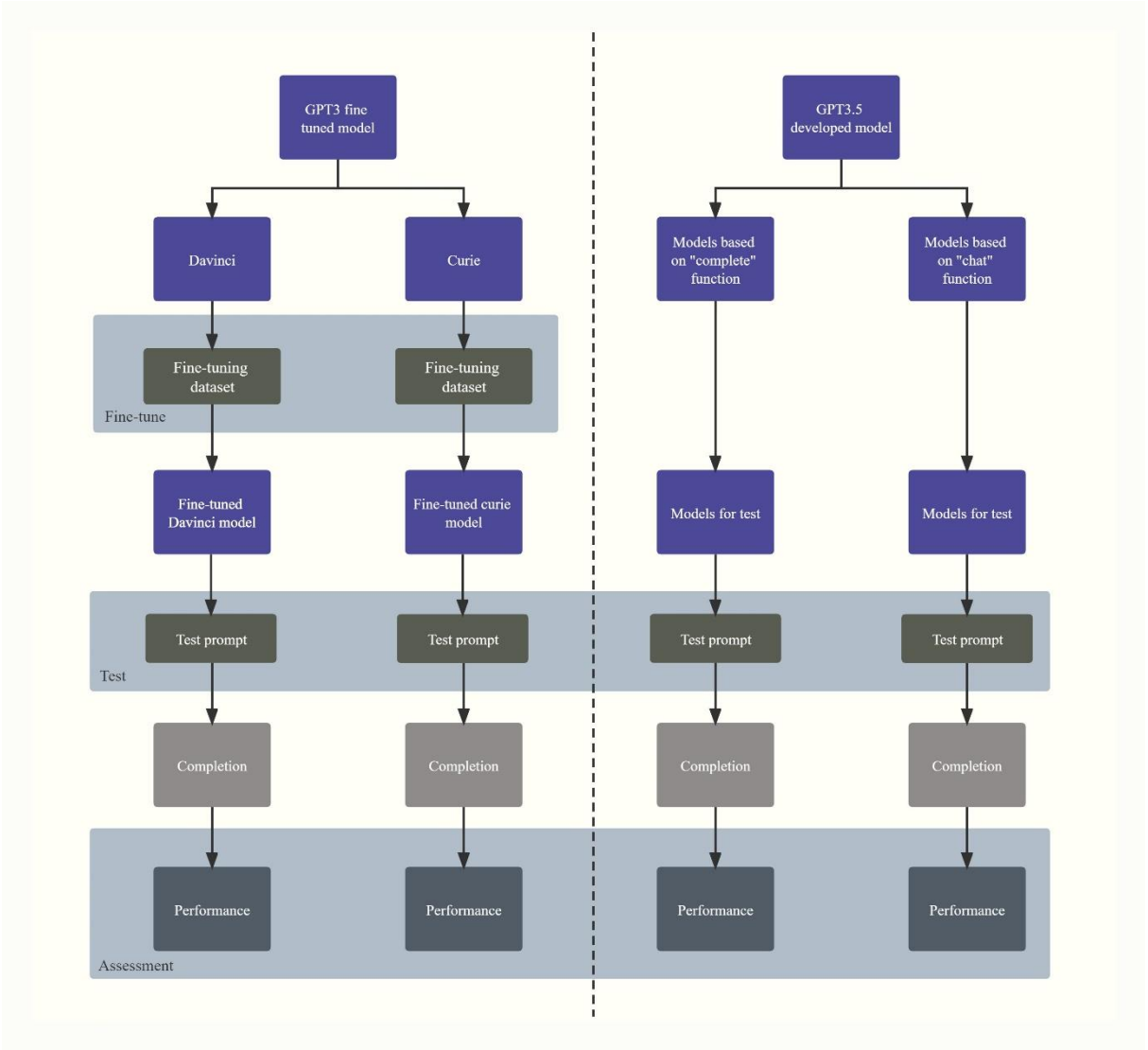

Figure 2: the flow chart of automatic compliance checks through LLMs

For the two main types of models:

GPT-3 based models: The dataset of prompt includes hundreds of samples with organised structure pairs: "prompt with samples of instructions, building design regulations' clause, and completion with samples of checking results".

GPT-3.5 prepared models: This type of model is capable of implementing the tasks proposed through prompt, which provides detailed instructions, and several precise samples to limit the form of generated results.

As both types of models complete essential preparation, their performance of the specific task can be evaluated through unified prompt tests. There are 2 different ways to evaluate the models from different perspectives through prompt designing.

The first is to build simple, organized, structured prompts which only have only one sentence of the instruction following no example or few examples. This task evaluates LLMs' basic capacities of learning from context and tries to figure out the boosting capacities of fine-tuning process. Consequently, the organized structure of the test prompts are the same as the fine-tuning prompt. This task is assumed to be completed by most of the tasks hence batch processing is implemented in this task. The generated results are divided into 2 types, and accordingly, a confusion matrix is built. Finally, a comprehensive quantitative performance evaluation in this scenario is provided.

The second is to build large, complex, naturally structured prompts which have a more similar form to general design documents. This task generally requires LLMs to learn more complicated internal logical connections from instructions and examples and applies them to completion generating. The capacities of LLMs can be claimed if the task can be realized precisely.

There are 2 types of experimental forms in this research, for complex large, naturally structured contexts, the "playground" from OpenAI official website is applied for the test environment. And API of ChatGPT is applied for batch processing in Python. The generated results are recorded in an extra column of original data to be converted into a confusion matrix. Due to space limitations, all of the codes, datasets and row data of the experimental results can be found at the following link: https://github.com/xiaoyuliu822/GPT-based-ACC.git.

2. Experiment

2.1 Prompt engineering

According to the definitions, "Prompt" refers to the input of the LLMs (Zhou et al., 2022). A proper prompt can clearly describe the tasks, including provide clear instructions, give necessary examples, references, require exact output form etc. In order to realize the ACC process to building design specifications, a general form of prompt needs to be built for batch processing through Python.

In this section, the details of the prompt design are explained comprehensively, including tasks, structures, and contents. As previous introducing, tasks include fine-tuning and unified tests, structures include natural and organised 2 types, and contents are extracted from HTM 05-02 fire safety codes.

3.1.1 Prompt for fine-tuning

In fine-tuning task, prompts are built into a dataset which contains hundreds of independent samples, Figure 4 presents the fine-tuning prompt structure of the dataset. These samples comprise all of the examples of compliance checking to build design specifications under HTM 05-02 fire safety codes. This process is designed to enhance the capability of GPT-3 based models learning, inference and judgment during implementing compliance checks. The dataset is divided into the fine-tuning set, validating set, and testing set in a ratio of 8:1:1.

Fine-tuned models are pre-tested on validating set before the unified test. Figure 5 presents the fine-tuned models' performance on validating set. There is a confusion matrix and 4

indices for each of the fine-tuned models. According to the figure, there are 3 basic statuses in this task, “negative” means “the requirement is not met”, “positive” means “the requirement is met”, and “task fail” means “the model produces meaningless results or the model doesn’t understand the task”. Each row of the matrix represents the statuses of the true value which are provided by the dataset, there are only 2 rows, “negative” and “positive”, as no meaningless tasks are provided, which can be seen through the vertical axis. Each column of the matrix represents the statuses of the predicted value generated by models, the same values as predictions.

Four main evaluation indices are calculated in validating process, and Chart 1 explains each element of a confusion matrix:

Accuracy: the proportion of correct predictions out of the total predictions:

$$A = \frac{(TP+TN)}{N} \tag{1}$$

Precision: the proportion of true positives (“the requirement is met”) out of all the positive predictions (“the requirement is met”):

$$P = \frac{TP}{(TP+FP)} \tag{2}$$

Recall: the proportion of true positives out of all the actual positive values:

$$R = \frac{TP}{(TP+FN)} \tag{3}$$

F1 score: the harmonic mean of precision and recall, it is useful when dealing with imbalanced datasets:

$$F1 = \frac{2}{(\frac{1}{P}+\frac{1}{R})} \tag{4}$$

(Goutte & Gaussier, 2005).

TN: True Negative, the true values are negative, and the predicted values are negative.

FN: False Negative, the true values are negative, and the predicted values are positive.

FP: False Positive, the true values are positive, and the predicted values are negative.

TP: True Positive, the true values are positive, and the predicted values are positive.

TF: Task failure, the tasks failed during implementation.

TN01	FN02	TF03
FP11	TP12	TF13
TF21	TF22	TF23

Table 1: Confusion matrix.

The performance of 2 fine-tuned models is presented in Figure 3. A mini validation dataset is applied, there are 21 samples in the validation dataset, 14 of them are positive, and 7 are

negative, fine-tuned curie model has better performance than fine-tuned davinci model in this dataset, and both of the models provide acceptable performance. The fine-tuned models are prepared after the validating process is complete, and they are provided for further unified tests with other GPT-3.5 based models.

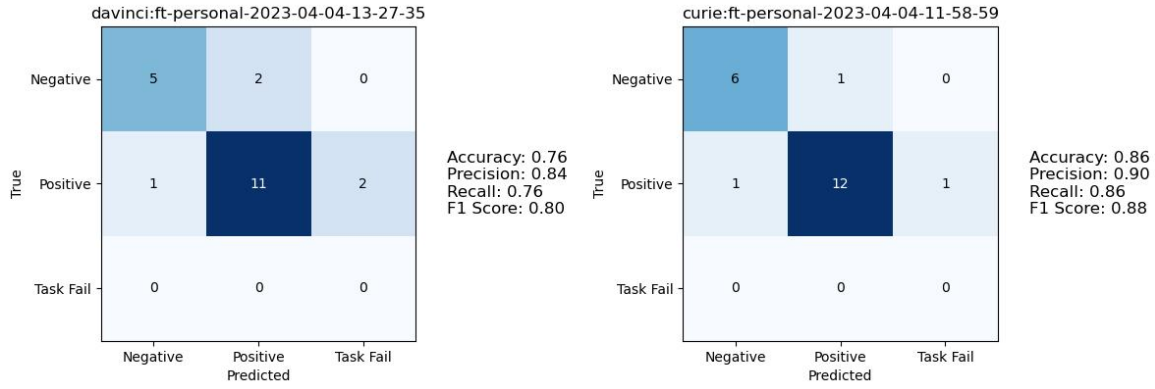

Figure 3: The confusion matrices of fine-tuning models on the validation dataset.

3.1.2 Prompt for testing

In the testing task, 4 prompts in 2 types are built for different capabilities, as figure 4 presenting, the first 2 prompts adopt an organized structure context, and the second 2 prompts adopt a naturally structured context.

Test 1: test 1 is the most simplified prompt in this research, it adopts the same organized structure as the prompts in fine-tuning process, the prompt is divided into several parts with symbolled separators, and completion is required to produce results from LLMs, no examples are given, true results are provided for comparison after generating. The LLMs need to produce results directly. This prompt design is applied mainly for quantifying models' performance. Through a series of evaluation indices, including accuracy, precision, recall, F1 score and confusion matrices, the model's performance can be precisely evaluated and visualized. This would directly prove that the GPT-3 models' capability can be boosted closely to a GPT-3.5 model.

Test 2: the prompt of test 2 adopts a similar organized structure, this is an extension of test 1, in which all the symbolled separators are cancelled in prompts and examples are provided to boost the generalization of the tested models instead. The examples are built as clauses pairs of fire safety regulations and building design specifications, the true results are given for learning. The test clauses pair from regulations and specifications are provided with the following, which requires LLMs to produce the results based on their learning results. The organized structures of the test prompts are designed to evaluate the GPT-3 based models considering the models have less capacity compared with the GPT-3.5 models. The capability of retrieving is not integrated directly into the GPT-3 models. Hence pre-processing operations are required before implementing the compliance check.

Test 3: the prompt of test 3 adopts a natural structure, though, in this test, no examples are given. The regulation clauses and corresponding building design specifications are listed separately, and the LLMs are required to produce the results directly. This task is designed to simulate the general ACC process in the deployment environment, the LLMs should match every regulation clause pair from whole documents of regulations and specifications, then implement the checking process and produce the results.

Test 4: this prompt of test 4 adopts a natural structure, which means the contexts are designed closely to the general documents. The instructions are detailed and precise to describe the scope of compliance checking, in this research, several fire safety regulation clauses and corresponding building design specifications are listed together in prompts as examples which helps LLMs to learn the compliance checking process from internal logic connections within the clauses and specifications pairs. The specified forms of generated completions can help LLMs to provide required results representations like "0" stands for false, and "1" for true. Finally, 2 test building design specifications are given to let the LLMs implement the ACC process. In this test, the LLMs should match the regulation clauses with building design specifications first, then implement the checking process and produce the results.

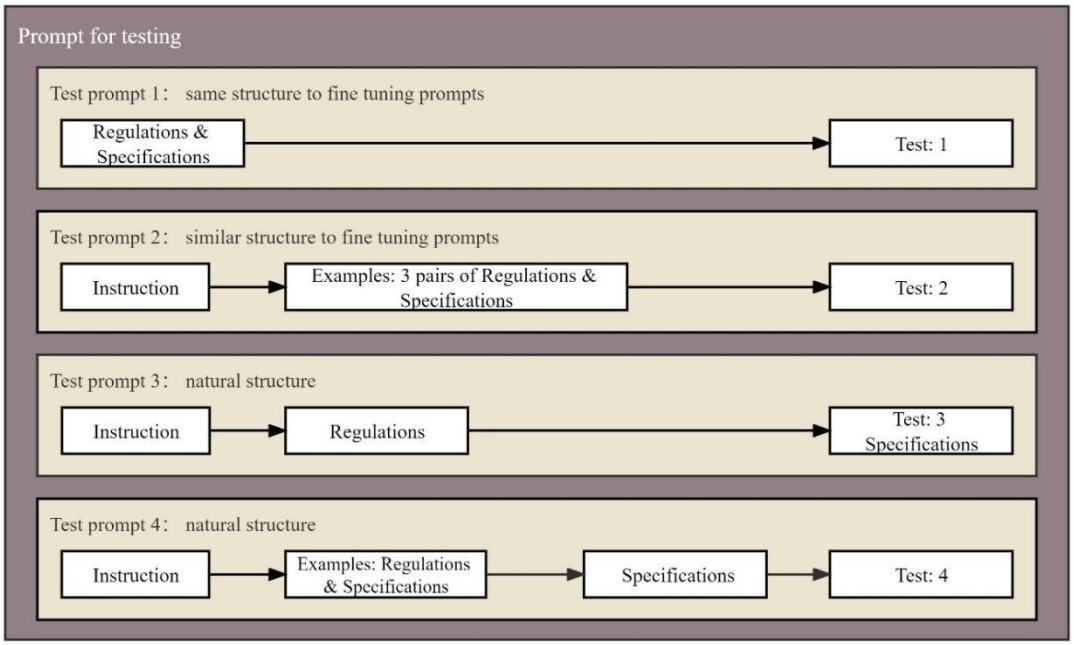

Figure 4: The prompt for testing.

2.2 Contents

In this research, clauses of fire safety regulations and responding building design specifications. Fire safety codes are extracted from *Health Technical Memorandum 05-02: Firecode, fire safety in the design of Healthcare premises (HTM 05-02) (Health, 2015)*. The responding building design specifications are generated by the author based on general design principles from various virtual projects.

2.3 Results

3.3.1 General results

The initial results of the experiments are recorded in Table 2. This is a general review of the tests' results, in the chart, "Fail" means the model can't understand the task and generate meaningless content, "1" means the model can understand the task and generate results, though the results may not be correct. All the GPT-3 models represent fine-tuned models, and GPT-3.5 are prepared models. The tasks of tests 1 and 2 are highly simplified, so the results can be quantified and evaluated by confusion matrices. Hence the performance of tests 1 and

2 are presented in Figure 7 and Figure 8. On the contrary, the tasks of tests 3 and 4 are highly integrated. Moreover, each prompt is required to process multi-tasks. Therefore, tests 3 and 4 cannot be simply identified as classification tasks.

Table 2: Results of the tests

	GPT-3.5-turbo	GPT-3.5-text-davinci-003	GPT-3.5-text-curie-001	GPT-3.5-text-babbge-001	GPT-3.5-text-ada-001	GPT-3-Curie	GPT-3-Davinci
Test 1	Fail	Pass	Pass	Pass	Pass	Pass	Pass
Test 2	Pass	Pass	Pass	Pass	Pass	Pass	Fail
Test 3	Pass	Pass	Fail	Fail	Fail	Fail	Fail
Test 4	Pass	Pass	Fail	Fail	Fail	Fail	Fail
Test 1: Prompt structure: organized, together; Instructions: required; examples: 0							
Test 2: Prompt structure: organized, together; Instructions: required; examples: 3.							
Test 3: Prompt structure: natural, separate; Instructions: required; Examples: 2.							
Test 4: Prompt structure: natural, together; Instructions: required; Examples: 3.							

According to Table 2, several essential results can be proved:

- The GPT-3 models (Curie and Davinci) could provide the same level of capabilities as the GPT-3.5 models when the GPT-3 models are fine-tuned for the specific tasks. However, the performance of the GPT-3 models is highly dependent on prompt engineering.
- In the general case, the most capable GPT-3.5 models (GPT-3.5-turbo, GPT-3.5-text-davinci-003) consistently provide the best performance among GPT models, which indicates that the inherent improvements in GPT-3.5 models may be more impactful than the fine-tuning models.
- The GPT-3.5 models show capabilities of generalization, with some models (i.e. GPT-3.5-text-davinci-003) performing better than GPT-3 models in most tests. This suggests that the fine-tuning process applied to the GPT-3 models may not be sufficient to outperform all GPT-3.5 models in building design compliance checking scenarios.

3.3.2 Quantified results

In tests 1 and 2, as the tasks are simplified to multiclassification scenarios, the performance of the models can be quantified and visualized. Figure 5 and Figure 6 present the confusion matrices of 6 models' performance in Test 1 and 2, the matrix of GPT-3.5-turbo can't be generated as the OpenAI doesn't provide GPT-3.5-turbo API reference.

As shown in Figure 5, the models with the best performance are GPT-3.5-text-davinci-003, GPT-3-curie, and GPT-3-davinci, which provide acceptable accuracy, precision, recall and F1 score in Test 1. Other models (text-curie-001, text-babbage-001, text-ada-001) present low accuracy and a high ratio of task failures which reveal these models can't implement the tasks of compliance checking.

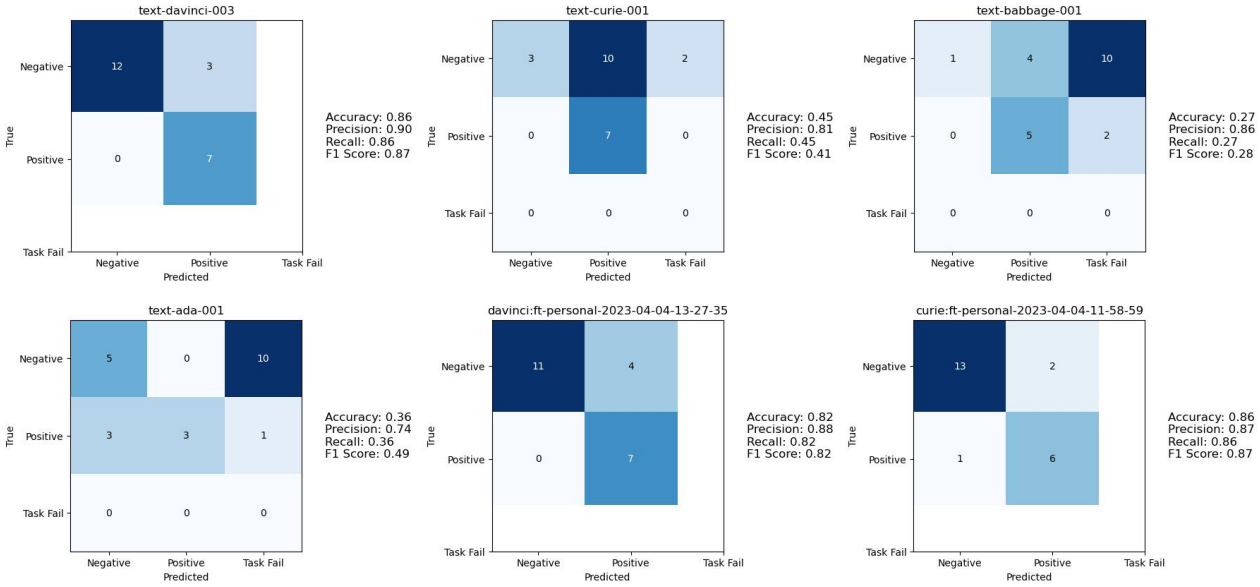

Figure 5: Confusion matrices of Test 1

In Figure 6, the davinci series model performance (GPT-3.5-text-davinci-003 and fine-tuned GPT-3-davinci) deteriorated compared with Test 1. On the contrary, other models' performance, including fine-tuned GPT-3-curie, GPT-3.5-text-curie-001, GPT-3.5-text-babbage-001, and GPT-3.5-ada-001 are improved, which proves most of the GPT-based models have a certain degree of generalization ability when the prompts structures are similar.

3.3.3 Deep analysis

Prompts of tests 3 and 4 are more challenging as LLMs are required to analyse documents which have similar structures to project documents, most of the models failed in these tests, only GPT-3.5-turbo and GPT-3.5-text-davinci-003 implemented the tasks, and the tests are not designed to be multiclassification tasks. In that case, the evaluations of the model's performance are only based on the generated completions.

In test 3, only GPT-3.5-turbo and GPT-3.5-text-davinci-003 models implement the tasks which require models to process 3 building design specifications at once, in addition, the models need to retrieve and match the regulation clauses corresponding to the design specification.

Based on test 3, test 4 requires models to generate completions in specific forms and give their explanations, moreover, unnecessary and pointless terms are added to the prompt, and the building design specification clause no longer corresponds to a single regulation term. These tasks test the comprehensive capability and robustness of the model. Only GPT-3.5-turbo and GPT-3.5-text-davinci-003 implement the tasks, though other models don't understand the task or generate unsatisfied completions.

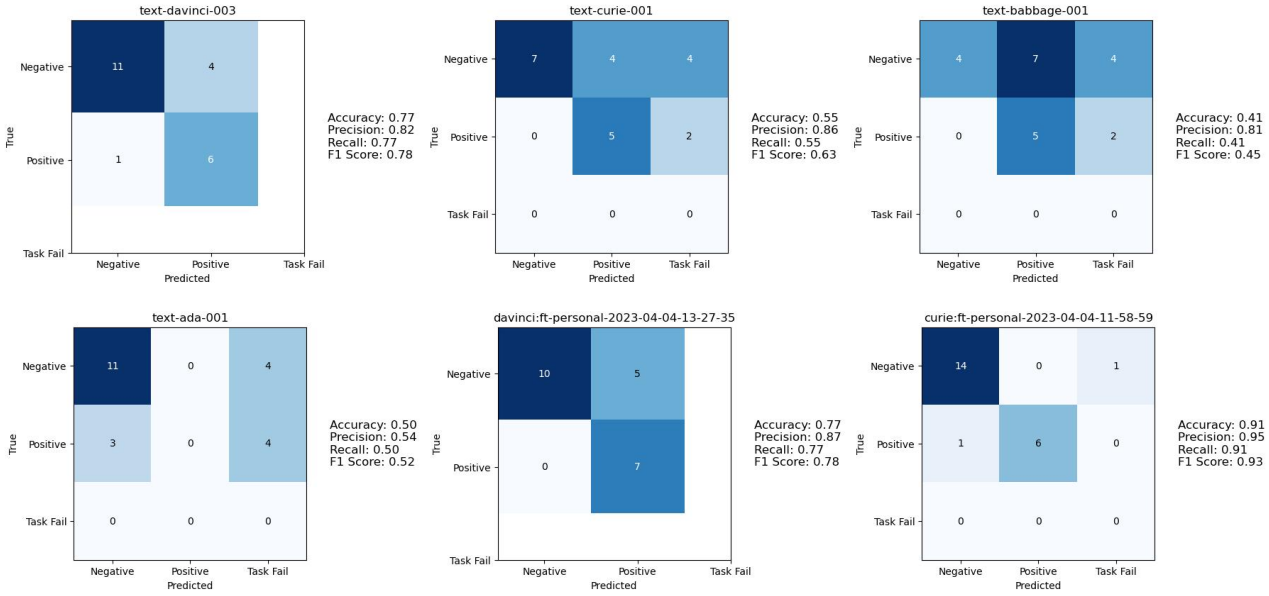

Figure 6: Confusion matrices of Test 2.

3. Conclusion

Large Language Models (LLMs) have demonstrated significant potential in Automated Compliance Checking (ACC) for building design specifications. Their generalization capabilities, flexibility, and ease of use can greatly enhance language processing performance in analyzing complex text files. Through fine-tuning, LLMs can integrate domain knowledge and provide customization, thus improving ACC automation by directly processing text documents in natural language, a process known as Prompt Engineering.

Despite their promise, GPT-based models in ACC still have limitations. For instance, fine-tuning, crucial for enhancing early versions of GPT models, is not applicable to recent models like GPT-3.5 and later versions. Also, the capacity of current LLMs is limited to approximately 4000 tokens, requiring careful prompt organization to execute the ACC process in one go. Moreover, current models can only process single-modality information, primarily text, highlighting a need for multi-modality models that can process blueprints, drawings, and models for complete ACC.

This research provides a practical methodology for implementing ACC using LLMs and offers valuable insights for future LLM deployment in the AEC industry. However, further developments are needed to address these limitations and fully realize the potential of LLMs in ACC.

References

Beach, T. H., Rezgui, Y., Li, H., & Kasim, T. (2015). A rule-based semantic approach for automated regulatory compliance in the construction sector [Article]. *Expert systems with applications*, *42*(12), 5219-5231. https://doi.org/10.1016/j.eswa.2015.02.029

Bloch, T., & Sacks, R. (2018). Comparing machine learning and rule-based inferencing for semantic enrichment of BIM models [Article]. *Automation in construction*, *91*, 256-272. https://doi.org/10.1016/j.autcon.2018.03.018

Choi, J., Choi, J., & Kim, I. (2014). Development of BIM-based evacuation regulation checking system for high-rise and complex buildings [Article]. *Automation in construction*, *46*, 38-49. https://doi.org/10.1016/j.autcon.2013.12.005

Goutte, C., & Gaussier, E. (2005). A probabilistic interpretation of precision, recall and F-score, with implication for evaluation. Advances in Information Retrieval: 27th European Conference on IR Research, ECIR 2005, Santiago de Compostela, Spain, March 21-23, 2005. Proceedings 27,

Health, D. o. (2015). *Health Technical Memorandum 05-02: Firecode, fire safety in the design of healthcare premises*. Department of Health Retrieved from https://www.england.nhs.uk/wp-content/uploads/2021/05/HTM_05-02_2015.pdf

Malsane, S., Matthews, J., Lockley, S., Love, P. E. D., & Greenwood, D. (2015). Development of an object model for automated compliance checking [Article]. *Automation in construction*, *49*(PA), 51-58. https://doi.org/10.1016/j.autcon.2014.10.004

Melzner, J., Zhang, S., Teizer, J., & Bargstädt, H. J. (2013). A case study on automated safety compliance checking to assist fall protection design and planning in building information models [Article]. *Construction Management and Economics*, *31*(6), 661-674. https://doi.org/10.1080/01446193.2013.780662

Saravia, E. (2022). Prompt Engineering Guide. https://github.com/dair-ai/Prompt-Engineering-Guide

Soliman-Junior, J., Tzortzopoulos, P., Baldauf, J. P., Pedo, B., Kagioglou, M., Formoso, C. T., & Humphreys, J. (2021). Automated compliance checking in healthcare building design. *Automation in Construction*, *129*, 103822. https://doi.org/https://doi.org/10.1016/j.autcon.2021.103822

Tan, X., Hammad, A., & Fazio, P. (2010). Automated code compliance checking for building envelope design [Article]. *Journal of Computing in Civil Engineering*, *24*(2), 203-211. https://doi.org/10.1061/(ASCE)0887-3801(2010)24:2(203)

Xu, X., & Cai, H. (2020). Semantic approach to compliance checking of underground utilities. *Automation in construction*, *109*, 103006. https://doi.org/https://doi.org/10.1016/j.autcon.2019.103006

Zhang, J., & El-Gohary, N. M. (2017). Integrating semantic NLP and logic reasoning into a unified system for fully-automated code checking [Article]. *Automation in construction*, *73*, 45-57. https://doi.org/10.1016/j.autcon.2016.08.027

Zhang, Z., Ma, L., & Broyd, T. (2022). Towards fully-automated code compliance checking of building regulations: challenges for rule interpretation and representation.

Zhang, Z., Nisbet, N., Ma, L., & Broyd, T. (2023). Capabilities of rule representations for automated compliance checking in healthcare buildings. *Automation in Construction*, *146*, 104688.

Zhong, B., Wu, H., Li, H., Sepasgozar, S., Luo, H., & He, L. (2019). A scientometric analysis and critical review of construction related ontology research [Review]. *Automation in construction*, *101*, 17-31. https://doi.org/10.1016/j.autcon.2018.12.013

Zhou, Y., Muresanu, A. I., Han, Z., Paster, K., Pitis, S., Chan, H., & Ba, J. (2022). Large language models are human-level prompt engineers. *arXiv preprint arXiv:2211.01910*.