**國 立 中 央 大 學**

資 訊 管 理 學 系

碩 士 論 文

應用Llama 2模型於資訊產品(?)生產製造測試案例異常事件之根本原因預測分析－以桌上型工作站電腦產品為例

Applying Llama 2 Model to Predictive Analysis of Root Causes for Test Case Anomalies in Manufacturing - A Case Study of Workstation Desktop Computer Product

研 究 生：施佳妏

指導教授：陳仲儼 博士

中 華 民 國 年 月

### 摘要

近年來，為了因應電腦需求市場需求的快速變化，在產品研發階段進行測試和驗證時能夠快速解決問題，以及在有限的時間內確保所有的設計都符合規格至關重要。然而，企業現有之問題追蹤系統雖然可提供使用者追蹤異常測試案例狀態，但查詢時僅能使用關鍵字，且搜尋出來的海量結果需人工逐筆過濾與判斷，以及大量的領域知識和經驗才能找出根本原因，導致效率低下且耗費大量時間。

在快速發展的人工智慧技術中，生成式AI工具在近幾年因其可以透過資料的收集和分析，進行問題回答、總結文件，在文本生成領域被快速採用，處理複雜語句的能力與一般AI相比，展現出更高的語言理解和創造力。(再想想看怎麼樣更吸引人)。其中大型語言模型(LLM)的技術被應用於各種領域的任務上，例如教育、醫療及法律領域，而本研究專注於應用在電腦製造業，針對測試案例中所產生的異常事件，將利用大型語言模型，改善現有之問題追蹤系統之檢索功能，建置一個預測問題根本原因系統，該系統可以讓使用者以自然語言提問，系統將提供可能的根本原因與說明，提供測試人員未知或未注意到的觀點和思路，達到組織內的知識共享、更快速地識別和解決問題，減少測試人員分析根本原因所需的時間，並提高產品交付的速度和品質。

**關鍵詞:**大型語言模型、生成式AI、測試案例、異常事件、根本原因

# 一、 緒論

## 1.1研究背景

在現今的高科技產業環境中，電腦製造業正面臨同業和市場競爭以及技術的挑戰，因此快速且有效地解決產品開發過程中被測試出來的問題(Issue)，成為一個重要的議題，例如：執行測試案例時出現藍白當機畫面(Blue Screen of Death, BSOD)、裝置管理員出現黃色驚嘆號(Yellow Bang)等異常狀況，需要去釐清是哪些元件(Component)造成的問題，甚至有時也需要進行多次的測試，來確認無法再次重現該異常狀況，或是確認是否為僅在單一機台上發生的問題。有效處理這些異常狀況不僅可以加速產品開發過程，還有助於提高產品的品質與可靠性。且為了增加市場佔有率，確保產品符合規格標準，需要做好品質控制，在製造過程中測試產品功能是否良好運作，防止產品缺陷的發生(Jozsef & Blaga, 2014)。

測試階段是製造開發過程中保證產品品質的關鍵任務，其目的是根據大量測試案例的規格要求，驗證正在開發的產品並檢測出異常事件及錯誤。在開發階段，測試人員每日都會收到數十甚至上百條的異常問題報告，需要依賴其領域知識來釐清和診斷出根本原因，進行問題排除。因此，診斷及故障的檢測和識別，在產品開發階段中是非常重要的任務，確保系統可以正常運作(Wotawa & Kaufmann, 2022)。

通常，測試人員會透過關鍵字檢索先前的異常事件，根據先前發生過之異常測試案例作為參考，人工比對類似的測試案例，加上自身經驗與知識的累積來判斷與找尋出根本原因及解決方案。(此新增段落為個案的介紹)由於電腦製造業的範圍相當廣泛，都面臨著複雜規格和配置的生產需求的挑戰，而本研究選擇聚焦桌上型工作站電腦產品作為個案研究對象，因其主要客戶群體為繪圖或設計專業人士，使得產品測試過程中異常事件的識別和解決尤為關鍵(我覺得為何關鍵的原因可以再加強)。隨著生成式AI(Generative AI)模型的快速發展(如OpenAI的ChatGPT)，這樣的大型語言模型(Large Language Models, LLM)的出現已經改變了人們與文本互動和檢索答案的方式。

## 1.2研究動機

為了能追蹤桌上型工作站電腦產品在生產階段測試功能時發現的問題，企業現有之問題追蹤系統(Issue Tracking System, ITS)可以記錄異常測試事件的相關資訊，包含異常事件描述、異常事件類別、系統環境配置、測試方法與重現步驟、預期的測試結果，以及整個解決問題過程的思路在系統上都會被保存下來，並且以協作的方式來編輯與紀錄這些資訊，長期累積下來，建立了龐大測試相關的知識文件，然而，雖然該系統在協助測試人員追蹤異常事件時發揮作用，但有關異常事件發生之根本原因(Root Cause)的識別，例如：出現藍白當機畫面，需要透過自身知識及經驗來釐清是BIOS或是顯示卡的問題，這些嘗試及解決問題的過程和思路都會以文字記錄在現有之問題追蹤系統中，但因為每位測試人員的敘述表達方式不盡相同，例如：不同的測試人員會將黃色驚嘆號描述成"Yellow Bang"或是"YB"，而當一位工作專業知識與經驗都尚有不足的新進員工，不明白YB亦代表黃色驚嘆號時，將難以依賴現有系統的關鍵字檢索功能找尋到所有相關案例。而若是過去從未發生過的案例，測試人員也僅能試圖從傳統的問題追蹤系統中做關鍵字搜尋，且仍然需要仰賴測試人員專業的判斷，在面對海量的測試案例和複雜難解的問題時往往顯得力不從心。

雖然現有之問題追蹤系統可以透過關鍵字搜尋找尋到包含該關鍵字的案例，但仍需逐筆經過人工篩選與判斷來找尋出相關案例，在面對不同測試人員對不同的名詞的描述方式各不相同的情況時，更需要有知識及經驗的累積才能夠透過現有之問題追蹤系統找尋到相關案例。另外，現有之問題追蹤系統在透過關鍵字搜尋後會顯示出所有包含該關鍵字的案例，這些大量案例會使測試人員在找尋解決方案時效率低下，常常耗費精力與時間過濾出相關的異常事件，並且需判斷真正可參考並且具有價值的描述，以找出問題根本原因，進而導致專案進度的延遲。因此有效地管理和分析異常事件之根本原因對於縮短產品開發階段和提高品質至關重要，為此，需要透過人工智慧(Artificial Intelligence, AI)技術，透過將問題追蹤系統存放之案例描述作為文本，分析過去的知識與經驗來協助生成出一個可能的根本原因與建議的解決方案，優化現有問題追蹤系統之檢索功能，進而達到組織內部的知識共享與傳承。

近年來發展的大型語言模型是具有複雜架構和大量參數的人工智慧模型，例如生成式預訓練 (Generative Pre-trained Transformer, GPT) Transformer (GPT-3)(這個只有trasnformer不對吧,應該可以刪除)(Floridi & Chiriatti, 2020)，在自然語言處理(Natural Language Processing, NLP)領域取得了顯著的進展。這些模型經過大量文字資料的訓練，改變以往資訊檢索的模式，不再僅回傳基於相關性的排名列表，而是對資訊進行深入理解和推理，直接產生對使用者有價值的解釋或結果(Ai et al., 2023)，具備接受和處理人類輸入複雜的句子的能力，並且以類似人類的方式生成文字、回答問題，進行知識推理，完成各種與語言相關的任務。這種AI技術展現了驚人的創造力，不僅僅是挖掘出過去已知的資訊，而是能夠深入地去分析數據，發掘新的模式和關聯，提出精煉過後的摘要或結論。對於本個案的測試人員來說，則可以提供測試人員們過去未知或未注意到的觀點或思路，從而更有效地識別和解決測試過程中出現的異常事件，快速精確地找到問題的根本原因。(再加上能一定程度理解某領域專業知識)。

而OpenAI在2022年底推出的開放後，ChatGPT迅速成為有史以來增長最快的應用程式，而其他公司也紛紛提供了自己所開發的LLM應用程式(Rillig et al., 2023)。因此，大型語言模型儼然已成為當前趨勢之一(前面這段再讀一下,感覺原因結果不是很連貫)，本研究將專注於產品製造開發階段中發生之異常問題，嘗試使用大型語言模型Llama2，透過過去類似的異常事件來建置一個預測根本原因的系統，提供測試人員建議的解決方案。

綜上所述，本研究之動機條列如下：

1. 改善現有問題追蹤系統之檢索功能，來解決關鍵字檢索的侷限性與使用者不一致的描述方式，以及需人工逐筆過濾眾多查詢結果的問題，從而大幅提升問題解決的時間與效率。
2. 結合過去積累的大量案例作為資料集，透過該問答系統對文本進行分析，讓測試人員更容易使用和檢索得到答案，從而促進組織內部的知識共享與傳承。
3. 有沒有什麼過去研究沒做到的, 你這次研究的亮點

## 1.3研究目的

個案公司之問題追蹤系統主要是紀錄產品在生產階段所有測試驗證的問題，協助測試人員追蹤問題狀態，透過其專業的知識與經驗，來釐清問題之根本原因並更新至系統上，藉此確保每一個異常事件都有準確地被解決。因此，本研究希望透過大型語言模型，來設計一個根本原因分析預測系統，能夠透過分析過去之異常事件，找出過去類似事件之根本原因，並以自然語言的方式回應使用者的提問，提供一個建議的解決方案，改善測試人員在找尋相關案例時，僅能透過現有之問題追蹤系統執行關鍵字搜尋的不便性，以及讓這些測試的工作知識與經驗可以更容易被重複利用，且不會受限於關鍵字的或描述方式或文字表達邏輯的不同而找不到相關測試異常的案例。

隨著大型語言模型應用的發展，在各種不同領域的用途也不斷地擴大與創新(Kaddour et al., 2023)，但在桌上型工作站電腦產業的領域中，特別是在製造開發階段的測試，尚缺乏有大型語言模型的應用。因此本研究嘗試導入大型語言模型技術，專注於桌上型工作站電腦產品測試案例中的異常問題，協助組織改善現有之問題追蹤系統之檢索功能，並利用該系統所記錄的大量異常問題作為資料集，透過大型語言模型建置一個問答系統，讓測試人員可以更直觀的透過自然語言直接陳述問題，並且從系統中獲得相應的解答或建議，從而提高問題解決的效率和準確性，進而降低測試時程延遲的風險。

綜上所述，本研究之目的條列如下：

1. 應用大型語言模型Llama2建置一個問答系統，透過其對複雜與靈活表達語句的理解，讓測試人員可容易地進行知識的重用和共享。
2. 系統可以根據使用者的提問，透過理解和分析自然語言，從過去的異常事件中找出類似的案例，讓測試人員可以得到一套異常事件的建議解決方案，加快開發階段的速度，降低專案延遲的風險。

## 1.4研究範圍

在電腦產品開發的流程中，從市場調查、提出設計規格、產品設計與開發、樣品製造與測試等階段，皆存在有許多因素和風險會導致產品最終的成果，例如：市場研究不充分、成本控管不當、品質控管不足、存在有技術上的限制或挑戰等。本研究主要針對個案公司之工作站桌上型電腦產品，並且專注在測試階段時，執行測試案例後出現之異常狀況作為探討對象，將嘗試以大型語言模型分析過去存放於現有問題追蹤系統中之異常事件資料集，目的在幫助測試人員更快地找到問題的根本原因，從而減少因錯誤診斷或遺漏而產生專案延遲問題，影響到產品最終品質，如此可進而提升產品品質與整體測試流程的效率。

## 1.5研究架構(尚待更新)

圖 1為本研究之論文架構圖。第一章緒論，針對個案(為何用個案這個名詞)之研究背景及重要性進行描述，再藉由研究動機與問題帶出研究目的。第二章文獻探討，對現有之根本原因分析方法、問答系統、生成式預訓練轉換器技術的相關文獻進行說明和探討。第三章系統設計，詳細說明本研究之個案系統內容、資料集、大型語言模型的運作流程。第四章系統實作與展示，將展示該系統之問答功能與成果。第五章系統成果與討論，針對本研究設計之系統進行回答內容的準確度驗證及使用者訪談。第六章結論與未來研究方向，對本研究進行總結，並說明研究貢獻與限制以及未來的研究發展方向。(寫法比較像套用一個公式, 想想看怎麼寫會比較像介紹你這個研究)

結論與未來研究方向

系統成果與討論

系統實作與展示

系統設計

文獻探討

緒論

圖 1論文架構圖

# 二、 文獻探討

## 2.1問題追蹤系統(Issue Tracking System, ITS)

(我覺得從問答系統講起範圍有點太大了,是不是聚焦在測試案例異常事件查詢系統就好🡪第一章有不少描述有關ITS的關鍵字檢索功能，思考應該可以寫這小節，而不寫問答系統 我覺得ITS比較好,但不用花太多篇幅介紹ITS本身是什麼,大概最多2段就好, 可以介紹多一點相關研究怎麼做的)

問題追蹤系統(Issue Tracking System, ITS)又稱缺陷追蹤系統(Bug Tracking System, BTS)(Janák, 2009)，在軟體開發和資訊科技產業已被廣泛使用(Lessel et al., 2015)，經常使用問題追蹤系統來支援問題管理流程，包含報告、指派、追蹤、解決和歸檔，在產品生命週期的後期階段追蹤問題報告來控制產品的品質(Tran et al., 2009)。異常檢測在管理電腦系統中扮演著重要角色，電腦系統越複雜，問題的檢測就越困難，系統人員會觀察和分析異常事件，然後建立並提交一份問題報告到問題追蹤系統以待解決(Tran et al., 2019)。問題追蹤系統提供一個網路互動式平台，供使用者來追蹤問題報告和進度，從而促進組織內資訊的透明與即時共享，通常包含以下幾個資訊(Singh & Chaturvedi, 2011)：

1. 標題(Title)：問題報告的標題。
2. 描述(Description)：問題的詳細描述，異常發生的具體情況，包含在操作過程中出現的資訊和預期結果。
3. 版本(Version)：專案的版本。
4. 元件(Component)：特定的元件。
5. 截圖/附件(Screenshot/Attachment)：上傳與實際操作或結果相關的截圖。
6. 優先度(Priority)：可以根據緊急程度指派優先順序。
7. 嚴重性(Severity)：對系統影響的程度。
8. 狀態(Status)：異常狀況目前的狀態，包含新建立(New)、開啟(Open)、已確認(Confirmed)、關閉(Closed)等等。
9. 建立者(Created by)：回報該異常狀況的人員。
10. 指派對象(Assigned to)：將該異常狀況分配給負責產生解決方案的人員。
11. 修訂歷史記錄(Revision History)：歷史變更紀錄。
12. 預估時間(Estimated time)：可指定預估時間。
13. 評論(Comments)：有助於識別異常狀況的任何其他資訊。

問題追蹤系統存放的資料由一系列結構化資料(如優先度、嚴重性、嚴重性)和非結構化的自由格式文本組成(Do et al., 2021)，然而，這種非結構化資料被認為是一個挑戰，在Lewis & Dreo(1993)的研究中就建議盡可能避免自由形式的文字描述，因為它們會導致處理數據時遇到困難。雖然從程式設計師的角度來看這個建議是有意義的，但顯然不符合一般喜歡撰寫自由格式文字的使用者的要求，因此，如何從這些異常問題報告複雜的描述中準確獲取有用的資訊，已成為一個嚴重的議題。為了幫助使用者有效搜尋，通常問題追蹤系統都設置了搜尋引擎，當使用者輸入問題時，搜尋引擎會使用關鍵字匹配技術進行檢索，但使用者需人工篩選所有的結果，並選擇可能與他們需要的資訊相符合的答案，且開發者的檢索意圖往往複雜且特殊，搜尋的結果通常會回傳過多的錯誤資訊，或是僅與部分關鍵字匹配的無效資訊(Lu et al., 2021)。因此，這種傳統的搜尋引擎僅能提供有限的幫助，無法正確地理解使用者的意圖，自然也無法返回滿足其需求的答案。

## 2.2生成式預訓練轉換器(Generative Pre-trained Transformer, GPT)

## 2.3大型語言模型(Large Language Model, LLM)

# 參考文獻

Ai, Q., Bai, T., Cao, Z., Chang, Y., Chen, J., Chen, Z., Cheng, Z., Dong, S., Dou, Z., Feng, F., Gao, S., Guo, J., He, X., Lan, Y., Li, C., Liu, Y., Lyu, Z., Ma, W., Ma, J., … Zhu, X. (2023). Information Retrieval meets Large Language Models: A strategic report from Chinese IR community. *AI Open*, *4*, 80–90. https://doi.org/10.1016/j.aiopen.2023.08.001

Do, T. T. H., Dobler, M., & Kühl, N. (2021). *What to Prioritize? Natural Language Processing for the Development of a Modern Bug Tracking Solution in Hardware Development* (arXiv:2109.13825). arXiv. https://doi.org/10.48550/arXiv.2109.13825

Floridi, L., & Chiriatti, M. (2020). GPT-3: Its Nature, Scope, Limits, and Consequences. *Minds and Machines*, *30*(4), 681–694. https://doi.org/10.1007/s11023-020-09548-1

Janák, J. (n.d.). *Issue Tracking Systems*.

Jozsef, B., & Blaga, P. (2014). Production Quality Control in the Process of Coating in an Electrostatic Field. *Procedia Technology*, *12*, 476–482. https://doi.org/10.1016/j.protcy.2013.12.517

Kaddour, J., Harris, J., Mozes, M., Bradley, H., Raileanu, R., & McHardy, R. (2023). *Challenges and Applications of Large Language Models* (arXiv:2307.10169). arXiv. http://arxiv.org/abs/2307.10169

Lessel, P., Müller, M., & Krüger, A. (2015). Towards a Novel Issue Tracking System for “Industry 4.0” Environments. *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, 1809–1814. https://doi.org/10.1145/2702613.2732720

Lewis, L., & Dreo, G. (1993). Extending trouble ticket systems to fault diagnostics. *IEEE Network*, *7*(6), 44–51. https://doi.org/10.1109/65.244793

Lu, J., Sun, X., Li, B., Bo, L., & Zhang, T. (2021). BEAT: Considering question types for bug question answering via templates. *Knowledge-Based Systems*, *225*, 107098. https://doi.org/10.1016/j.knosys.2021.107098

Rillig, M. C., Ågerstrand, M., Bi, M., Gould, K. A., & Sauerland, U. (2023). Risks and Benefits of Large Language Models for the Environment. *Environmental Science & Technology*, *57*(9), 3464–3466. https://doi.org/10.1021/acs.est.3c01106

Singh, V., & Chaturvedi, K. (2011). Bug Tracking and Reliability Assessment System (BTRAS). *International Journal of Software Engineering and Its Applications*, *5*.

Tran, H. M., Lange, C., Chulkov, G., Schönwälder, J., & Kohlhase, M. (2009). Applying Semantic Techniques to Search and Analyze Bug Tracking Data. *Journal of Network and Systems Management*, *17*(3), 285–308. https://doi.org/10.1007/s10922-009-9134-4

Tran, H. M., Le, S. T., Nguyen, S. V., & Ho, P. T. (2019). An Analysis of Software Bug Reports Using Machine Learning Techniques. *SN Computer Science*, *1*(1), 4. https://doi.org/10.1007/s42979-019-0004-1

Wotawa, F., & Kaufmann, D. (2022). Model-based reasoning using answer set programming. *Applied Intelligence*, *52*(15), 16993–17011. https://doi.org/10.1007/s10489-022-03272-2