國立政治大學資訊科學系
Department of Computer Science
National Chengchi University

碩士論文
Master's Thesis

基於注意力機制語言模型之財務風險文章偵測與實體辨識

Financial Risk-related News Detection and Named Entity Recognition via Transformer-based Language Models

指導教授：蔡銘峰 博士

研究生：盧佳妤 撰

中華民國 一百一十年八月
August 2021

DOI:10.6814/NCCU202101564

致謝

光陰似箭，歲月如梭，轉眼間充滿挑戰的研究所學業已圓滿完成。回顧這兩年的碩士生活，雖然過程中充滿著各種困難與壓力，既忙碌且辛苦，但當碩士論文經幾番修改完成後，不僅頗有成就感且覺得一切努力和辛勞都是值得的。能有如此豐碩的收穫，要感謝求學中每階段的老師們、同學們、朋友們和家人，讓我有機會跨領域再學習，如願獲得碩士學位，邁向人生旅途的另一階段。

首先，感謝我的碩班指導教授　蔡銘峰老師，不嫌棄對於資訊領域認知尚淺跨領域學習的學生，接受我加入實驗室，更有幸參與玉山產學合作計畫，在此計畫中不僅學到資料處理及機器學習的技術外，老師也教導我們做人處事的道理及分享業界的寶貴經驗，讓我受益良多。而在研究知識圖應用於推薦系統專案上，要感謝中研院的王釧茹老師，在王老師用心耐心的教導下，讓我們充分了解了設計實驗的邏輯及嚴謹性，受益匪淺。

我也要謝謝實驗室的學長們，感謝志明、喻能及昇芳學長的幫助，在入學前告知就讀研究所可能會遭遇的問題，避免重蹈覆轍，此外，當我的專案遇到難題時學長也會給予專業的技術指導和建議。在玉山專案中要感謝中研院 CFDA 的 RA 小白學長，因為有他的帶領，使我們快速順利地進入狀況，藉由花時間跟我們討論專案進度，讓我們面對專案不再徬徨緊張，而是安心的跟著學長的步調穩紮穩打的進行實驗研究。還要謝謝我的同窗先灝、均捷、寶鈞及韋勝，不管是課業切磋還是專案分工合作，都是好夥伴，願友誼長存。

最後謝謝家人的支持和鼓勵，讓我無後顧之憂完成碩士學位。尤其要感謝我的精神糧食親愛的媽媽，以及我腦袋當機時的救星，感謝有您們。總之，要感恩的人不勝枚舉，除了銘記在心外，更要祝福您們健康平安！

盧佳妤

國立政治大學資訊科學系

August 2021

DOI:10.6814/NCCU202101564

摘要

本研究利用注意力機制模型偵測財務文章之風險事件及抽取潛在金融犯罪名單，建構自動化模型以降低人力標記成本及提升預測速度。我們分析不同模型架構及訓練方法之優缺點，並比較傳統神經網路方法與 Transformer Based 模型的差異。模型架構分為兩階段，第一階段判斷目標文章是否包含金融風險事件，而第二階段則在這些文章中抽取高危險的名單。我們提出聯合訓練方法同時訓練兩階段的模型，透過實驗證明可在不損失正確性的情況提升訓練及預測速度，並得以提升模型穩定性。我們亦針對注意力機制模型內部的 Attention Weight 做視覺化分析，顯示模型能在不提供標注的情況自動關注金融風險詞彙。另外我們針對缺乏風險人名標記的訓練資料之情況，利用以上 Attention Weight 分析設計特殊的規則，達到一定程度的效果提升。最後我們額外在一個 Wikipedia 上的英文資料集做測試，說明此研究結果亦可應用於不同領域及不同語言的任務。

關鍵字：注意力機制模型、聯合訓練、實體辨識、自然語言處理

DOI:10.6814/NCCU202101564

Abstract

This thesis uses transformer-based models to detect risk events from financial articles and extract potential financial criminals. With such automated models, we can reduce human costs on labeling and increase prediction performance. In this thesis, we analyze the advantages and disadvantages of different approaches and compare the differences between traditional neural networks and Transformer-based models. The proposed method contains two stages: the first stage determines whether the target news contains financial risk events, and the second stage extracts high-risk entities from the news. We propose a joint-training method to train these two stages at the same time. Experimental results show that the proposed joint-training method improves prediction accuracy and enhances the stability of the training process. We also visualize the attention weights of the attention mechanism model, showing that the model automatically pays attention to financial risk vocabularies without providing annotations. In addition, we use the above attention weight scheme to design special rules, achieving a certain degree of effect improvement for the case that lacks risk-name-annotation. Finally, further experiments conducted on a dataset from English Wikipedia confirm that the proposed method can also apply to different domains and languages.

Keywords: Transformer, Attention Mechanism, Joint Training, Named-Entity Recognization, Natual Language Processing

DOI:10.6814/NCCU202101564

目錄

DOI:10.6814/NCCU202101564

DOI:10.6814/NCCU202101564

圖目錄

DOI:10.6814/NCCU202101564

表目錄

DOI:10.6814/NCCU202101564

第一章　緒論

1.1　前言

近年來，由於國際洗錢和資助恐怖活動的活動頻繁發生，洗錢是指通過各種手段隱匿、隱瞞犯罪所得，讓犯罪所得合法化的行為。為了預防犯罪集團侵入金融企業、政府機構等，國際於西元 1988 年維也納會議時，訂定《聯合國禁止非法販運麻醉藥品及精神藥物公約》（United Nations Convention Against Illicit Traffic in Narcotic Drugs and Psychotropic Substances）要求締約國立法懲罰與毒品有關的洗錢活動[1]。

1989 年，七大工業國發現，涉毒的洗錢犯罪對銀行體系和金融機構造成嚴重威脅，故在高峰會議中決議設置 Financial Action Task Force on Money Laundering，以發展和改進國際對於洗錢犯罪之對應。

隨著洗錢犯罪和打擊資本恐怖主義的概念自國際逐漸傳入我國，我國政府在 1985 年 10 月制定《洗錢防制法》，並於 86 年依〈法務部調查局洗錢防制中心設置要點〉成立「洗錢防制中心」[2]，以執行金融情報中心及涉及反洗錢的相關業務，來帶動政府及各行業致力於洗錢防制（Anti Money Laundering，AML）工作。

然而，該政策並未得到持續推進，公眾對反洗錢觀念的認識也薄弱，直到 2018 年，亞太防制洗錢組織（Asia / Pacific Group on Money Laundering）來台進行反洗錢相關檢查[3]，其結果攸關台灣金融業在海外市場的競爭力和國際形象。但當時的台灣在法律制度、監管或執法方面都遠遠落後於國際標準，加上美國金融服務署開始對國內外知名銀行因洗錢防制工作疏忽開出巨額罰款，引發各界批評和關注，故政府成立行政院洗錢防制辦公室，以凝聚和推進反洗錢和打擊資恐之風氣。

銀行業為金錢頻繁流動之處，故其角色在防洗錢工作上重要性極大，所以銀

[1] https://www.mjib.gov.tw/EditPage/?PageID=e21658e0-bde6-475f-884d-f2415a446d02

[2] https://www.amlo.moj.gov.tw/1461/1462/1463/28852/post

[3] https://www.amlo.moj.gov.tw/media/10178/8111220442162.pdf?mediaDL=true

DOI:10.6814/NCCU202101564

行為因應監管機關法規，除了考慮其業務的多樣性和洗錢風險的程度外，還需要對現有客戶進行客戶審查，並在考慮所獲得的信息有充分性後，在適當的時候審查現有的客戶關係。故額外的洗錢調查或身份驗證過程使銀行人員工作量大幅增加，且由於目前許多銀行沒有導入系統或者系統判斷正確率極低，需要以人力的方式閱讀相關調查資料，讓監控效率不佳，也怠慢了原先的工作。而系統的誤判率高會對銀行業務造成負面影響，因為銀行需要致電及花更多時間調查和確認客戶資料細節，以進一步確定客戶是否為高洗錢風險族群，可能導致客戶體驗不佳，進而讓客戶流失率上升。因此，綜合考慮人力成本和客戶體驗滿意度，如何遵守法律法規，降低被罰款的風險，是當今金融機構需要面對的問題。

一般來說，客戶在與金融機構溝通時，銀行需要立刻確認客戶身份，並通過系統對客戶進行比對，看其是否被列入反洗錢重點名單中。為了生成以上系統所需的判斷資料，銀行人員了花大量的時間閱讀各個報社的新聞，並依據新聞報導內容標記哪些人與洗錢風險高度相關，這些被標記的人就會放入反洗錢黑名單中。因此如果能夠在人工智慧的輔助下定期更新反洗錢重點人員名單，並進行自動比對，銀行執行反洗錢業務的人力和時間成本將大大降低。

AML 系統的自動化需仰賴自然語言處理的各種技術。自然語言處理（Nature Language Processing，NLP）[1] 為機器學習領域的一門重要的分支，主要處理文字語言等相關問題，其命名「自然語言」指的便是人類所使用的各種語言，使用「自然」這個詞用以跟程式語言等人工發明的語言（又稱人工語言）做區別。與人工語言不同，自然語言並沒有完美且邏輯化的文法與單詞，原因在於其發展是基於人類社會文化及歷史發展，因此沒有一個完美的邏輯規則可以直接處理。自然語言處理最開始的發展是透過統計等技術，而近年來由於機器學習的發展，許多應用於自然語言處理的機器學習技術也相繼發展，包含文本分類（Document Classification）、實體辨識（Entity-Recognition）、機器問答（Question Answering）、機器翻譯（Machine Translation）等技術。

自然語言處理與其他領域有個特別的差異，在於其輸入資料通常為非結構化的文本資訊，因此沒有固定的長度及模式。循環神經網路（Recurrent neural network，RNN）[2]、長短期記憶（Long Short-Term Memory，LSTM）[3] 模型的發展得以理解自然語言任務的文本，並解決文本長度不一致的問題。而注意力機制模型更進一步打破 RNN 系列模型的限制，讓模型能夠更精確的抓住重要的部分。Transformers [4] 系列模型基於注意力機制架構，充分利用現今硬體的發展，在許多領域及任務上達到了最好的成果，成為是近年來最著名的自然語言模型之一。基於此架構的 BERT [5]、GPT [6] 等模型利用大量的資料訓練出逼近人類表現的語言模型，充分的理解人類語言的本質，使得研究者能夠輕易的將語言模型應用在任意的自然語言任務上。

DOI:10.6814/NCCU202101564

1.2 研究目的與貢獻

本研究希望利用自然語言的技術，以及不同的機器學習模型架構，自動化的應用於反洗錢偵測系統，透過觀察大量的新聞文本，從中抽取出高風險金融犯罪事件，並歸納出異常的人名與組織名，進而幫助銀行人員能快速地確認要優先調查的對象。透過比較 Transformers 系列模型，及其基礎架構之注意力機制模型，以及傳統機器學習方法的 LSTM 模型，我們希望能夠找出最適合此任務的模型，並且歸納出它們間的優缺點，提供未來實際應用上的參考。

為了處理本研究的兩個主要目標 — 找尋高風險金融犯罪事件及歸納出異常的人名與組織名，我們將研究分為兩個階段：第一階段為文本分類任務，以判斷文章是否含有風險事件；第二階段為實體辨識任務，目的為抽取金融風險名單。另外，我們亦將兩個階段的任務組合在一起，建構出聯合訓練（Joint Training）架構，來達到任務間的資訊共享及訓練速度提升的效果，並比較此訓練方法與兩階段訓練的差異性。

本研究將以上模型於真實世界應用在反洗錢系統的資料集測試，以顯示我們所提出的模型對反洗錢系統自動化的實際效果。另外為了測試以上方法在不同領域及語言上的表現，我們亦從網路上透過 Wikipedia 資料庫中抽取符合本研究的資料特性而建構出虛假資料集，並比較在不同資料之上模型的表現差異。

本研究希望利用機器學習技術偵測財務文章之風險事件及抽取潛在金融犯罪名單，建構自動化模型以降低人力標記成本及提升預測速度。本研究的貢獻為以下幾點：

1. 建構不同的模型架構，並測驗不同的參數對模型的效果。
2. 建構不同的訓練方法，比較兩階段訓練架構及聯合訓練（Joint Training）架構之優缺點。
3. 使用兩個不同領域的資料集做實驗，證明實驗方法的實用性及泛用性。
4. 針對 Attention Weight 做視覺化分析，並針對沒有風險名單訓練資料的情況，設計特殊的模型架構及規則。

本文架構如下：第 2 章針對自然語言處理模型及相關技術之文獻做探討；第 3 章定義問題並說明模型架構；第 4 章說明資料集、實驗設定及各種模型與參數實驗結果分析；第 5 章給出研究的結論及未來研究方向。

DOI:10.6814/NCCU202101564

第二章　相關文獻探討

本研究所使用的資料為非結構性文本資料，屬於自然語言處理（Nature Language Processing，NLP）範疇，故在接下來針對處理文本資訊的自然語言處理模型之相關文獻做探討。

2.1　自然語言中的文字表示法

自然語言處理旨在讓電腦可以讀懂人類的語言，所以模型中會需要先將詞轉換成詞向量（Word Vectors）。傳統的方法有 One-Hot Encoding [7] 及 Word Embedding，前者使用與字典大小相同的向量表示，每個維度對應到一個詞，只有對應該詞之位元為 1。然而 One-Hot Encoding 無法表達詞之間的關係，另外所使用的向量維度隨著字典增長而變大，因此在模型使用上效果較差且耗費記憶體。

Word Embedding 等方法透過較低維度的向量表達各個詞，並使用相似的向量來表達類似的詞，來解決以上 One-Hot Encoding 的缺點。Word2Vec 是學習 Word Embedding 的一個經典模型，這個模型在 2013 年由 Google 的 Tomas Mikolov 等人提出 [8]，分為兩種 Word2Vec 模型：Continuous Bag-of-Words（CBOW）和 Skip-Gram，前者利用上下文預測當前單詞；而後者則是使用中間詞來訓練上下文各詞的機率。

在 NLP 任務中，會使用透過大量的文本來訓練 Word Embedding 做下游任務的輸入，好處在於可以在下游任務中更快的學習到語意，使模型訓練更快速且穩定。

延續這個想法，發展出 Contextual Embedding [9]，不同於傳統的 Word Embedding 技術，它通過考慮文檔中所有單詞的順序來學習整句話的語義，這樣的技術可基於多義詞的上下文來學習多義詞的不同表示形式，像是在 2018 年 Google 提出的 BERT 預訓練模型（Pretrained Language Model）[5] 以及其他 RoBERTa、ALBERT、ELMo ……等模型使用 Contextual Embedding 技術，使得自然語言處理領域有重大的進步。

DOI:10.6814/NCCU202101564

2.2 中文斷詞

語言處理的模型大多需要先分辨文本中的詞才能進行進一步的訓練，然而與英文不同，中文無法透過空白來切割各個詞，因此需要透過中文斷詞工具達到斷詞目的。

目前在中文上著名的斷詞工具有結巴[1]和中研院資訊所詞庫小組所開發的CkipTagger[2]系統，這兩者在繁體中文上的效果 CkipTagger 更勝一籌，原因在於 CkipTagger 大量訓練在繁體中文文本上，加上文本除了 Wikipedia 資料外也有新聞，所以模型有一定程度上理解新聞用語。由於本研究使用繁體中文的新聞資料當作訓練文本，因此將選擇 CkipTagger 工具來做文本的斷詞工具。

2.3 循環神經網路

循環神經網路（Recurrent neural network，RNN）[2] 是一種使用序列數據或時間序列數據的人工神經網絡，這類模型之深度學習算法通常用於順序或時間問題，例如：語言翻譯、自然語言處理、影像處理……等。以自然語言處理為例，RNN 除了根據輸入文字來判輸出結果之外，亦會參考前文的資訊，因此可以表達文字順序的關係。

然而 RNN 會受到梯度消失問題的影響，在句子長度過長的時候無法將最前面的資訊傳達到後面，因此發展出長短期記憶（Long Short-Term Memory，LSTM）[3] 模型解決以上的問題。LSTM 透過多個 Gate 控管記憶的傳遞，因此相較於傳統的 RNN 更能處理長期記憶。

然而 RNN 與 LSTM 都只有參考前文的資訊，但在許多的自然語言任務下，亦需參考後文的資訊才能夠正確地理解整個句子，因此可使用雙向的 LSTM（Bi-LSTM），輸出由這兩個 LSTM 的狀態共同決定，以達到同時理解上下文的目的。

2.4 Transformer-Based 模型

2.4.1 注意力機制模型

現今最熱門的做 NLP 的模型 BERT ，其架構主要由 Transformer 組成，而它的源頭之一便是 Attention is All You Need [4] 這篇論文。

[1] https://github.com/fxsjy/jieba
[2] https://github.com/ckiplab/ckiptagger

DOI:10.6814/NCCU202101564

注意力機制（Attention）模型的核心想法為找出哪個字相對於其他字較為重要，由於其架構設計，使得無論兩個字的距離多遠皆可直接計算兩者間的關係，因此解決了 RNN 模型輸入過長句子導致容易丟失訊息使模型效果不佳的問題。另外由於 Attention 機制中詞之間的重要度是互相獨立，因此相較於 RNN 須按照時間或文字先後順序計算，Attention 模型可平行運算以提升運算速度。

Attention 模型的細節如下：

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V \tag{2.1}$$

輸入的每個字皆對應一個 Query，一個 Key 及一個 Value，而透過計算兩個字的 Query 及 Key 的相似度決定這兩個字的重要度，並利用這個值作為權重對 Value 做加權。

Multi-Head Attention 模型為 Attention 的拓展，平行使用多個 Attention 模型，將輸入的文字映射至不同的空間向量，每個 Attention 可專注於不同的文字特徵，讓模型可以從不同的方面理解輸入資訊。

2.4.2 Transformer

Google 基於 Attention 架構設計出了一個全新的 Seq2Seq 模型 — Transformer，不同於傳統基於 RNN 的 Seq2Seq 模型，Transformer 利用多層 Multi-Head Attention 做語意理解及文字生成。其架構由 Encoder-Decoder 組成，Encoder 負責理解輸入句子的語義，而 Decoder 則用 Encoder 所學到的資訊來輸出目標的文字。以英翻中的翻譯任務為例，Encoder 將輸入的英文句子 Encode 成 Transformer 可理解的資訊，而 Decode 根據 Encoder 所理解的資訊翻譯成對應的中文句子。

而為了讓 Attention 模型理解文字間上下文的關係，Transformer 亦在模型中的 Word Embedding 加上 Position Embedding，讓模型除了文字的語意資訊之外，亦可理解該字在句中的位置訊息。

2.4.3 BERT

根據 Transformer 架構，Google 在 2018 年發表 BERT（Bidirectional Encoder Representations from Transformers）[5]，可以學習文本中單詞之間的上下文關係。BERT 的目標是語意理解的語言模型，所以只應用了 Transformer 的 Encoder 部分。

BERT 的訓練分為兩大步驟：Pre-Training 與 Fine-Tuning（如圖 2.1），在 Pre-Training 階段，Google 使用大量文本資料，以非監督式學習的方式訓練模型，為了一次性預測出文字，加上 Masked 技術，在將單詞序列輸入 BERT 之前，每個序

DOI:10.6814/NCCU202101564

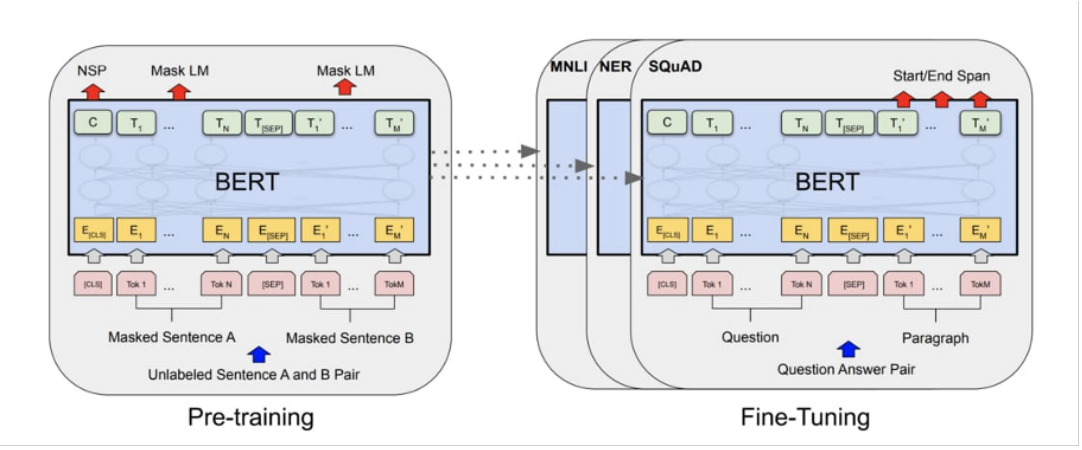


圖 2.1: BERT 模型圖[3]

列中 15% 的單詞被替換為 [MASK] 標記，嘗試根據序列中其他非屏蔽詞提供的上下文來預測屏蔽詞。

而在 Fine-Tuning 階段，利用前階段 Pre-Training 的模型作為起始，針對不同的下游任務訓練（如文章分類、實體辨識、問答題等任務），對模型進行微調。由於在 Pre-Training 階段，模型已經經由大量文本學到了該語言的文法及語意資訊，因此可大幅減少下游任務的訓練時間。

BERT 的字典中有個特殊 Token — [CLS]，當作整個輸入句子的 Aggregated Embedding，故在下游任務中通常用於做整個句子的預測任務，例如文章分類等任務。

2.5 命名實體識別

命名實體識別（Named Entity Recognition，NER）[10]，是指辨識文章中專有名詞及其出現位置的技術，主要包括人名、地名、組織名、時間、金錢等，或是指定領域中的特定名詞（如醫學上的疾病名稱或新聞中的罪犯姓名等），因此可透過 NER 模型讓機器自動找尋文本中我們感興趣的實體，並加以分析，達到資訊抽取（Information Extraction）的目的。

NER 標記方式主要有 BIO 和 BIOES 兩種 [11]，前著用 B（Beginning）表示實體名的開始，I（Inside）表示實體名的中間及結尾，O（Other）不屬於任何實體的其他字。在多類別的 NER 任務中，每個類別議會有自己的 B 與 I 標記，利如用 B-PERSON、I-PERSON 表示人名、用 B-LOC、I-LOC 表示地名等。而 BIOES 是 BIO 的擴充，增加 E（Ending）標記代表實體名的結尾，S（Single）代表只有一個字／詞的實體名。

[3]圖片來源：https://arxiv.org/pdf/1810.04805.pdf

DOI:10.6814/NCCU202101564

NER 最早期的方法為基於規則或字典的方法，接下來發展基於機率模型的傳統機器學習方法。隱藏式馬可夫模型（Hidden Markov Model，HMM）[12] 的架構由 Markov Model 衍伸而來，它用來描述一個含有隱含未知參數的馬可夫過程，假設每個 NER Tag 的出現機率僅與它對應的文字及前一個 Tag 有關。條件隨機場（Conditional Random Field，CRF）[13] 則是以較全面的觀點，計算整句文字與所有的 NER Tag 組合間的機率關係，因此相較於 HMM 能更好的建模。隨著深度機器學習發展蓬勃，後人更將機率模型與深度學習方法結合，例如：RNN-CRF、CNN-CRF，到近年來提出更進階的 NER 模型，如：注意力模型、遷移學習、半監督學習…等方法。

DOI:10.6814/NCCU202101564

第三章　研究方法

3.1　問題定義

以金融黑名單偵測系統情境說明，銀行人員每天會花許多時間看上百篇新聞資料，從新聞中標記出與金融犯罪相關的人名與組織名列入反洗錢黑名單當中，範例如圖 3.1，紅匡圈出的部分為標記的目標（慶富造船集團、陳慶男），而文本中不僅有這些金融犯罪相關的人名與組織名，還會有與之相同屬性但非金融犯罪相關的實體，例：公家機關、記者名、檢察官名字等，像是圖 3.1 的高雄地院。

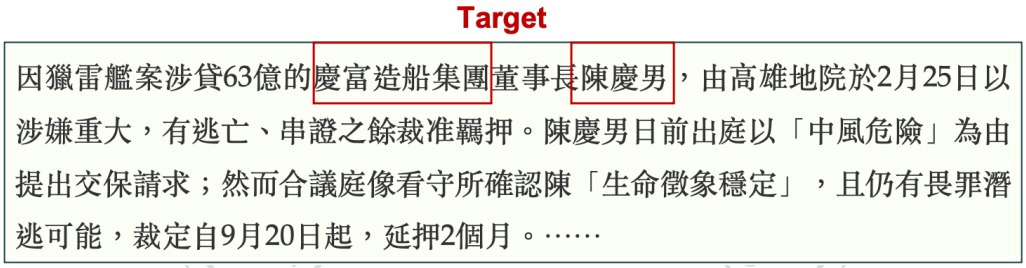


圖 3.1: 新聞標記範例

根據上述情境描述，給定一篇新聞文本，其中的文字為 $x = (x_1, x_2, \ldots, x_n)$，每個字對應各對應一個向量 $(v_1, v_2, \ldots, v_n)$，其中 $v_i \in \mathbb{R}^{d_v}$，輸出為有犯罪風險之實體 $\{e_1, e_2, \ldots, e_m\}$。資料分為兩種資料標記格式，其一，Document Level 標記新聞為 $y \in \{0, 1\}$，1 表示新聞含有金融犯罪風險內容，0 則反之；其二，Entity Level 為使用 BIOES 格式標記每個文字 x_i 之 NER Tag t_i，分為 Person 和 Other Person 兩種類別，Person 為有涉及金融犯罪的人及組織，Other Person 則為其他的人及組織，所以共有 9 種 Tag — $t_i \in$ {B-PERSON, I-PERSON, E-PERSON, S-PERSON, B-O-PERSON, I-O-PERSON, E-O-PERSON, S-O-PERSON, O}，故模型架構為輸入以 BIOES 標記實體的新聞文本，經過模型後，萃取出目標實體 — 有金融犯罪風險的組織與人名。詳細符號列表請參考表 3.1。

DOI:10.6814/NCCU202101564

符號	描述
n	輸入文章的文字數量
m	輸入文章的目標實體數量
$x = (x_1, \ldots, x_n)$	輸入文章的每個文字
$v = (v_1, \ldots, v_n), v_i \in \mathbb{R}^{d_v}$	輸入文章每個文字對應的詞向量
$t = (t_1, \ldots, t_n)$	輸入文章每個文字對應的 NER 標記（BIOES 標記）
$e = \{e_1, \ldots, e_m\}$	輸入文章的目標實體集合
$y \in \{0, 1\}$	輸入文章的分類（含有金融犯罪風險內容與否）
$\hat{t} = (\hat{t}_1, \ldots, \hat{t}_n), \hat{t}_i \in \mathbb{R}^9$	模型預測之 NER 標記（BIOES 標記）
$\hat{y} \in (0, 1)$	模型所預測之文章分類
$h^* \in \mathbb{R}^{d_h}$	CLASS 模型的輸出，代表文章分類的 Hidden Vector
$h = (h_1, \ldots, h_n), h_i \in \mathbb{R}^{d_h}$	NER 模型的輸出，每個文字的 Hidden Vector
k	Attn+Generic NER 方法中取 Attention Weight 前 k 高的詞
$v = \text{Embedding}(x)$	Embedding Layer，將輸入文字 x 轉換成詞向量 v
$h^* = \text{Model}_{\text{CLASS}}(v)$	CLASS 模型的通稱
$h = \text{Model}_{\text{NER}}(v)$	NER 模型的通稱
$(h^*; h) = \text{Model}_{\text{Joint}}(v)$	Joint 模型的通稱
$\hat{y} = \text{Linear}_{\text{CLASS}}(h^*)$	CLASS 任務所使用的 Feed-Forward Layer，將 h^* 降維至 $\hat{y}$
$\hat{t} = \text{Linear}_{\text{NER}}(h)$	NER 任務所使用的 Feed-Forward Layer，將 h 降維至 $\hat{t}$
Bi-LSTM$(\cdot)$, LSTM$(\cdot)$	雙向 LSTM Layer 及單向 LSTM Layer
Attention$(\cdot, \cdot, \cdot)$	Multi-Head Attention Layer
BERT$(\cdot)$	BERT Layer（包含其 Embedding Layer）
$\sigma(\cdot)$	Sigmoid Activation Function
Softmax$(\cdot)$	Softmax Activation Function
BinaryCrossEntropy$(y, \hat{y})$	Binary Cross Entropy Loss
CrossEntropy$(t_i, \hat{t}_i)$	Cross Entropy Loss
$\mathcal{L}_{\text{CLASS}}$	CLASS 任務之 Loss Function
$\mathcal{L}_{\text{NER}}$	NER 任務之 Loss Function
$\cdot \Vert \cdot$	Concatenate Operator，將兩個向量連接起來

表 3.1: 數學符號列表

3.2 模型簡介

本篇論文之研究目的在於提出自動化標記新聞中有金融犯罪風險的人名與組織名的模型。研究主要會分為兩個面向探討：首先是依照資料的精煉程度建構兩階段訓練，並對各個階段進行不同模型及參數設定實驗；第二，研究是否可建構出一個 End-to-End 模型，將兩階段方法合併，來提升模型效果及速度。

兩種訓練方法均有使用 LSTM、Attention、LSTM+Attention、BERT 架構來建構模型做實驗，前三種方法統稱 Basic 架構。此外，依據 Attention 架構特性，另外針對缺乏 NER 訓練標記的情況，我們提出了 Generic NER 加規則之方法。以下簡單介紹兩種訓練方法。

DOI:10.6814/NCCU202101564

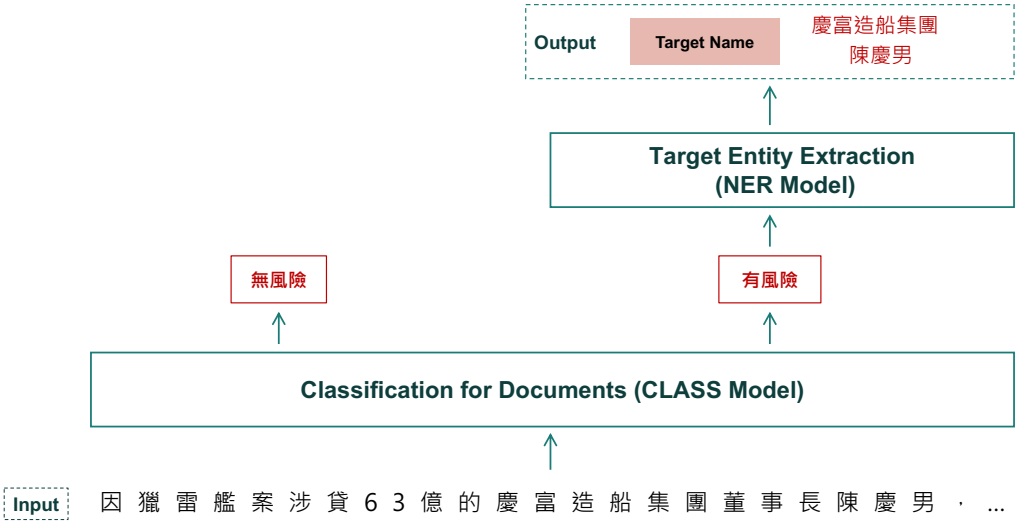


圖 3.2: 兩階段訓練

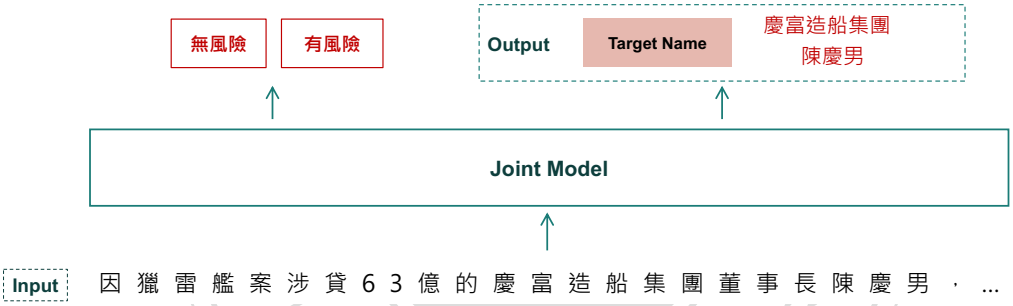


圖 3.3: 聯合訓練

兩階段訓練（Two Stages Training） 依據研究資料的特性，我們將模型分為兩階段，第一階段為新聞文本分類任務（Classification，CLASS），為每篇新聞賦予一個標記 y，判斷新聞是否具有風險事件。找出有風險的新聞後，將這些文本輸入至下一階段任務 — 實體辨識任務（Named-Entity Recognition，NER），此階段透過預測的 NER Tag t_i，萃取出有金融犯罪風險的人名與組織名 e_j，如圖 3.2。

聯合訓練（Joint Training） 兩階段訓練缺點在於需要花較多的時間訓練與較繁複的資料轉換，而一階段模型可以整合輸入的資料格式，將 CLASS 和 NER 任務中的模型合併成一個模型，稱作 Joint 模型，如圖 3.3。

故根據上述，提出之兩種訓練方法中可分為三個模型元件 — CLASS、NER 及 Joint 模型，以下先做統合的介紹，在 3.3 和 3.4 節會根據每種技術的模型構成進行詳細的說明。

DOI:10.6814/NCCU202101564

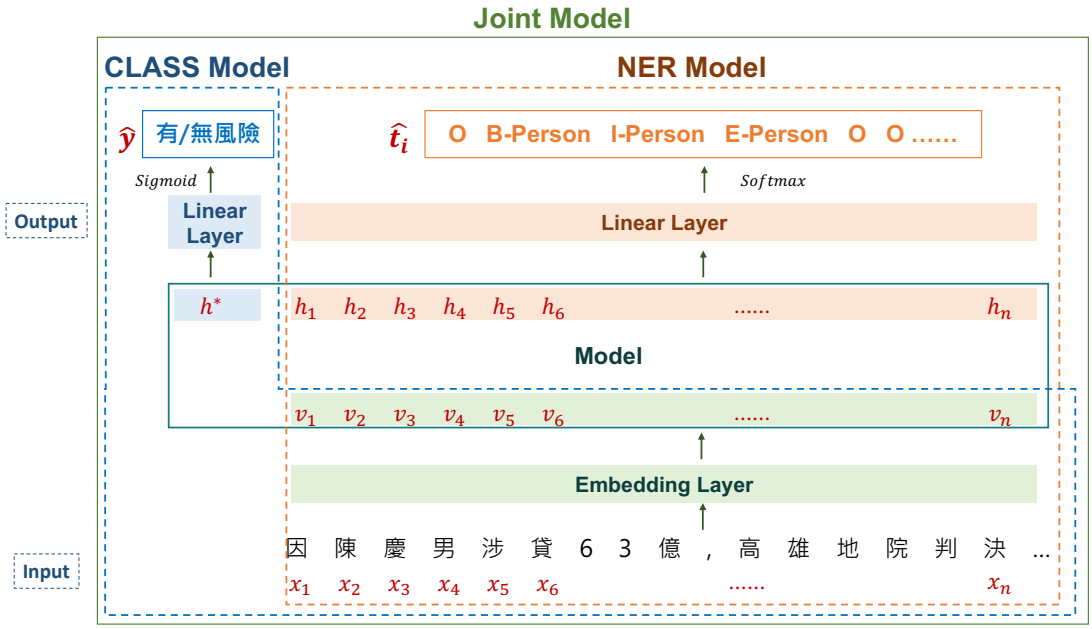


圖 3.4: 模型概圖

3.2.1 新聞文本分類任務 — CLASS 任務

此任務為兩階段訓練之前半部分，目的在於對新聞文本分類，判斷內容是否有犯罪金融風險，預期利用此模型過濾出目標新聞，以減少下一個階段 NER 任務的輸入資料量，增加結果的準確率及提升預測速度。本研究在 CLASS 任務中探討四種架構及其差異性 — LSTM、Attention、LSTM+Attention、BERT。

各個模型所需的 Embedding 皆使用他人利用大量文本資料訓練出的 Pretrained Embedding，在 Basic 方法中使用的是用 Word2Vec 系列訓練出來的 Embedding，BERT 模型使用以 BERT 模型訓練出的 Pretraining Embedding。

CLASS 任務雛形方法為將新聞文本 $(x_1, \ldots, x_n)$ 轉成 Embedding，得到文字向量 $(v_1, \ldots, v_n)$，經過 CLASS 模型後，得到 Hidden State h^*，再過一層 Linear Layer 降維，並使用 Sigmoid σ 作為 Activation Function，最終輸出 $\hat{y}$，數值大於 0.5 則預測為有風險的文章，Loss Function 使用 Binary Cross Entropy，如式子 (3.1) 及圖 3.4 中藍色虛線圈起之部分。

$$\begin{aligned}(v_1, \ldots, v_n) &= \text{Embedding}(x_1, \ldots, x_n) \\ h^* &= \text{Model}_{\text{CLASS}}(v_1, \ldots, v_n) \\ \hat{y} &= \sigma(\text{Linear}_{\text{CLASS}}(h^*)) \\ \mathcal{L}_{\text{CLASS}} &= \text{BinaryCrossEntropy}(y, \hat{y})\end{aligned} \tag{3.1}$$

DOI:10.6814/NCCU202101564

3.2.2 實體辨識任務 — NER 任務

此任務為兩階段訓練之後半部分，目的為將針對新聞文本中的每一個字 x_i，賦予一個標記 t_i，並依此找出文章中與金融犯罪風險相關之人名或組織名。與 CLASS 任務相同，NER 任務亦可利用 LSTM、Attention、LSTM+Attention、BERT 四種架構實作。

NER 任務雛形方法亦是將新聞文本轉成 Embedding，這邊使用與前述 CLASS 任務相同的 Pretrained Embedding 得到文字向量 $(v_1, v_2, \ldots, v_n)$，經過 NER 模型後，得到每個文字 x_i 的 Hidden State h_i，再將這些 Hidden State 過 Linear Layer 和 Softmax，最後輸出 $\hat{t}_i$，其中 $\hat{t}_i \in \mathbb{R}^9$ 為該文字 x_i 預測之每個 NER Tag 機率，而最終每個字的 NER 預測結果即為 $\hat{t}_i$ 中最大的值對應的 NER Tag。Loss Function 使用 Cross Entropy，如式子 (3.2) 和圖 3.4 中橘色虛線圈起之部分。

$$\begin{aligned}
(v_1, \ldots, v_n) &= \text{Embedding}(x_1, \ldots, x_n) \\
(h_1, \ldots, h_n) &= \text{Model}_{\text{NER}}(v_1, \ldots, v_n) \\
\hat{t}_i &= \text{Softmax}(\text{Linear}_{\text{NER}}(h_i)) \qquad \text{for } i = 1, \ldots, n \\
\mathcal{L}_{\text{NER}} &= \sum_i \text{CrossEntropy}(t_i, \hat{t}_i)
\end{aligned} \tag{3.2}$$

為了在模型訓練時強調 NER Tag 之間的關係（如 Begining Tag 後面只能接 Inside Tag 或 Ending Tag 等關係），我們亦嘗試在 NER 任務中引入 CRF 技術，增加 NER 模型的穩定性，此方法實作為在前述 NER 任務中 $\text{Linear}_{\text{NER}}$ 後面再加一層 CRF Layer，而 Loss Function 改用 Log-Likelihood 算法。

3.2.3 聯合訓練 — Joint Training

在聯合訓練的情況，我們將以上的 CLASS 模型與 NER 模型合併為 Joint 模型，目的為同時訓練 CLASS 及 NER 兩種任務，並希望可以透過文章分類的資訊幫助 NER 的訓練，使結果變好，並節省訓練所需的時間。同樣地，我們也會在接下來的章節探討聯合訓練在 LSTM、Attention、LSTM+Attention、BERT 四種架構中的差異性。

聯合訓練結合以上 CLASS 及 NER 任務的架構，使用相同的 Pretrained Embedding 及核心模型架構，將文字向量 $(v_1, \ldots, v_n)$ 經過 Joint 模型後，分別得到整體文章 Hidden State h^* 及每個文字的 Hidden State $(h_1, \ldots, h_n)$，h^* 的部分再經過一層 Linear Layer 降維，並使用 Sigmoid σ 作為 Activation Function，最終輸出 $\hat{y}$；$(h_1, \ldots, h_n)$ 則透過另一個 Linear Layer 和 Softmax，最後輸出 $\hat{t}_i$。聯合訓練所使用的 Loss Function 為 Binary Cross Entropy 及 Cross Entropy 相加（也就是加總 CLASS 及 NER 任務中的 Loss Function），式子如 (3.3) 及圖 3.4 中綠色實線圈起之部分。

DOI:10.6814/NCCU202101564

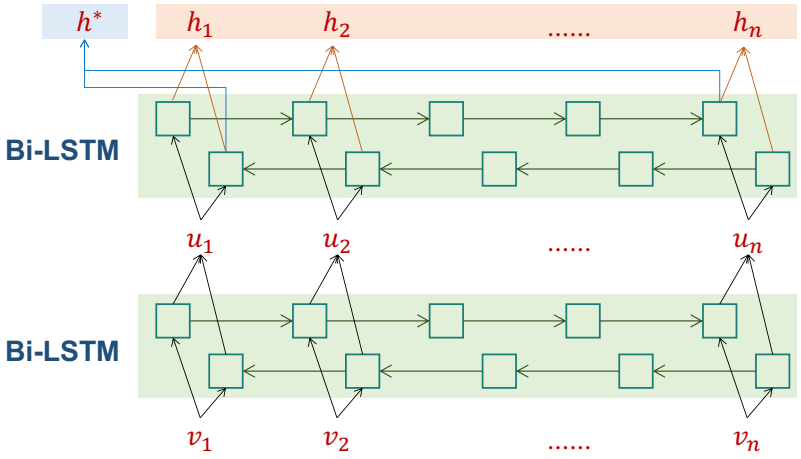


圖 3.5: LSTM 模型

$$
\begin{aligned}
(h^*; h_1, \ldots, h_n) &= \text{Model}_{\text{Joint}}(v_1, \ldots, v_n) \\
\hat{y} &= \sigma(\text{Linear}_{\text{CLASS}}(h^*)) \\
\hat{t}_i &= \text{Softmax}(\text{Linear}_{\text{NER}}(h_i)) \qquad \text{for } i = 1, \ldots, n \\
\mathcal{L}_{\text{Joint}} &= \mathcal{L}_{\text{CLASS}} + \mathcal{L}_{\text{NER}} \\
&= \text{BinaryCrossEntropy}(y, \hat{y}) + \sum_i \text{CrossEntropy}(t_i, \hat{t}_i)
\end{aligned}
\tag{3.3}
$$

3.3 各種架構之模型實作方式

此節說明本實驗中使用的四種架構 — LSTM、Attention、LSTM+Attention、BERT，在上述任務中所提之式子中的 $\text{Model}_{\text{CLASS}}$、$\text{Model}_{\text{NER}}$ 及 $\text{Model}_{\text{Joint}}$ 如何實作。

3.3.1 LSTM 架構實作

首先我們使用 RNN 系列模型中的 LSTM 模型，這裡採用 Bi-LSTM 模型，原因在於此模型相比於一般的 LSTM，可以透過雙向的訓練讓文字看到上下文，進而更能容易理解語意，故效果會比一般的 LSTM 好。綜合上述，在 LSTM 架構中，使用 Bi-LSTM[1] 模型，模型架構如式子 (3.4) 與圖 3.5。

在此架構中，輸入文字向量 $(v_1, \ldots, v_n)$ 經過兩層 Bi-LSTM Layer 後，得到兩個方向的 Hidden State $\overrightarrow{h_i}, \overleftarrow{h_i}$，$\text{Model}_{\text{CLASS}}$ 的實作使用兩個方向最後的 Hidden State

[1]雙向 LSTM（Bi-LSTM）定義為 $\text{Bi-LSTM}(v) = \overrightarrow{h} \| \overleftarrow{h}$，其中 $\overrightarrow{h} = \overrightarrow{\text{LSTM}}(v)$ 與 $\overleftarrow{h} = \overleftarrow{\text{LSTM}}(v)$ 分別為順向及反向的 LSTM 輸出。

DOI:10.6814/NCCU202101564

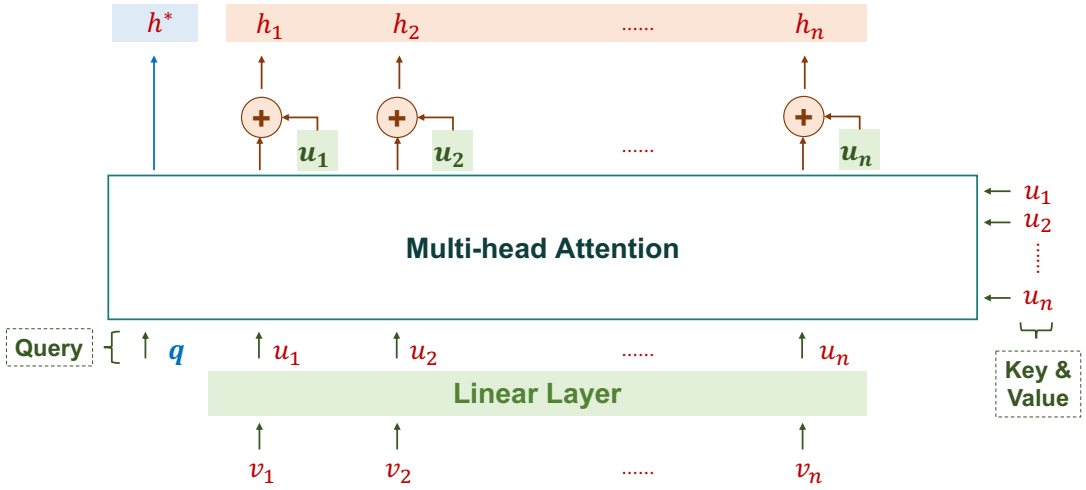


圖 3.6: Attention 模型

並連接起來為 h^*；而 $\text{Model}_{\text{NER}}$ 的實作則是將每個文字的雙向之 Hidden State 連接起來為 h_i；而 $\text{Model}_{\text{Joint}}$ 將前兩種結合同時拿到 h^* 和 h_i 做計算。

$$\begin{aligned}
(u_1, \ldots, u_n) &= \text{Bi-LSTM}(v_1, \ldots, v_n) \\
(\overrightarrow{h_1}, \ldots, \overrightarrow{h_n}) &= \overrightarrow{\text{LSTM}}(u_1, \ldots, u_n) \\
(\overleftarrow{h_1}, \ldots, \overleftarrow{h_n}) &= \overleftarrow{\text{LSTM}}(u_1, \ldots, u_n) \\
h^* &= \overrightarrow{h_n} \| \overleftarrow{h_1} \\
h_i &= \overrightarrow{h_i} \| \overleftarrow{h_i} \qquad \text{for } i = 1, \ldots, n
\end{aligned} \tag{3.4}$$

3.3.2 Attention 架構實作

Attention 架構的實作在 $\text{Model}_{\text{CLASS}}$ 與 $\text{Model}_{\text{NER}}$ 有些許不同。首先透過一個 Linear Layer 將輸入向量降維成 $u_i \in \mathbb{R}^{d_h}$，在 $\text{Model}_{\text{CLASS}}$ 上，使用一個共用的向量 q 作為 Query 使用，並將文章中的每個文字向量 v_i 作為 Key 與 Value 使用，透過 Attention Layer 得到 h^*；而 $\text{Model}_{\text{NER}}$ 中則是使用 Self-Attention 架構，將 u_i 同時作為 Query、Key、Value 使用，並且在通過 Attention Layer 取得 Hidden 向量後，將其與輸入向量 u_i 相加得到向量 h_i；而 $\text{Model}_{\text{Joint}}$ 為將前兩種方法結合同時得到 h^* 和 h_i，如式子 (3.5) 與圖 3.6。此架構在往後的內容中縮寫為 Attn。

$$\begin{aligned}
u_i &= \text{Linear}_{\text{input}}(v_i) && \text{for } i = 1, \ldots, n \\
h^* &= \text{Attention}(q, u, u) \\
h_i &= \text{Attention}(u_i, u, u) + u_i && \text{for } i = 1, \ldots, n
\end{aligned} \tag{3.5}$$

DOI:10.6814/NCCU202101564

因獵雷艦案涉貸63億的**慶富造船集團**董事長**陳慶男**，由高雄地院於2月25日以涉嫌重大，有逃亡、串證之餘裁准羈押。陳慶男日前出庭以「中風危險」為由提出交保請求；然而合議庭像看守所確認陳「生命徵象穩定」，且仍有畏罪潛逃可能，裁定自9月20日起，延押2個月。

圖 3.7: 新聞範例一

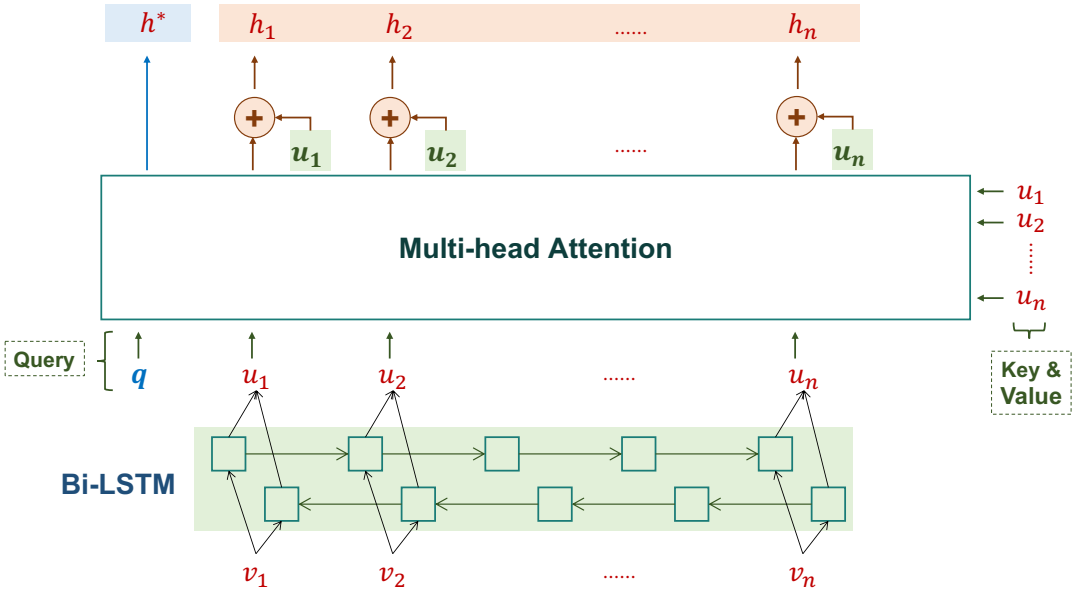


圖 3.8: LSTM+Attntion 模型

另外針對 Attention 架構在 CLASS 任務上的實驗，根據我們對新聞文本資料的觀察，當一篇新聞要被標記成有風險時，判斷標準為新聞中出現跟犯罪有關的詞，例如圖 3.7 為有風險之新聞片段出現涉貸、涉嫌之犯罪相關詞，而 Attention 模型的特性便是透過訓練學習到哪些詞相對重要，進而判斷文章分類，與我們分類新聞的思考邏輯是一致，而在實驗中（詳見第 4.6 章）確實證明此特性。

3.3.3 混合使用 LSTM 與 Attention 架構

由於 Attention 架構中沒有文字序列資訊，所以在做非常看重文字序列資訊的 NER 任務上分數會較差，故將有序列資訊的 LSTM 架構與 Attention 架構結合，希望綜合兩個架構的優勢，提升模型的效果。

建構方式如式子 (3.6) 及圖 3.8，使用 Bi-LSTM 取代 (3.5) 中的 $\text{Linear}_{\text{input}}$，得到之向量在 $\text{Model}_{\text{CLASS}}$ 做為 Attention Layer 的 Key 和 Value，算出而在 Hidden 向量 h^*，而在 $\text{Model}_{\text{NER}}$ 當作 Attention 模型的 Query、Key、Value 計算 h_i；$\text{Model}_{\text{Joint}}$ 則為將前兩種方法結合同時得到 h^* 和 h_i。此架構在往後的內容中縮寫為 LSTM+Attn。

DOI:10.6814/NCCU202101564

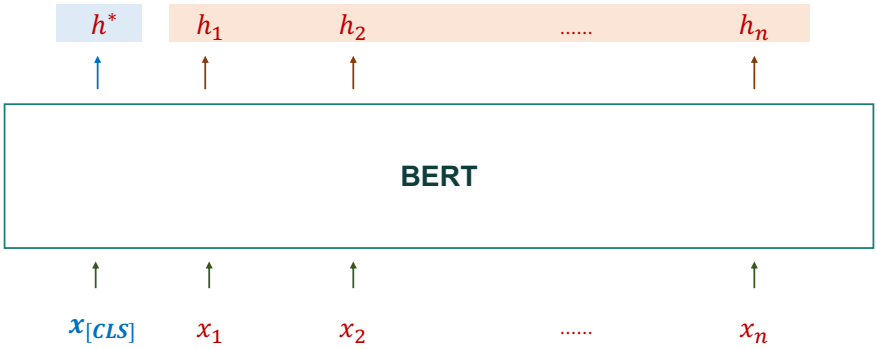


圖 3.9: BERT 模型

$$
\begin{aligned}
(u_1, \ldots, u_n) &= \text{Bi-LSTM}(v_1, \ldots, v_n) \\
h^* &= \text{Attention}(q, u, u) \\
h_i &= \text{Attention}(u_i, u, u) + u_i \quad \text{for } i = 1, \ldots, n
\end{aligned} \tag{3.6}
$$

3.3.4 BERT 架構實作

BERT 架構實作為式子 (3.7)及圖 3.9，輸入文字 $(x_1, \ldots, x_n)$，並且在句子最前面加上 [CLS] Token，此 Token 可用於整篇文章的分類，含有整個輸入文本的資訊，因此使用它來做文章分類任務。經過 BERT 模型後，得到各 Token 對應個 Hidden 向量 $(h_{[\text{CLS}]}, h_1, \ldots, h_n)$，其中 $h_{[\text{CLS}]}$ 的功能與前述之 h^* 相同，作為 $\text{Model}_{\text{CLASS}}$ 之輸出，；而 $\text{Model}_{\text{NER}}$ 的部分使用其餘每個文字 Hidden 向量 $(h_1, \ldots, h_n)$ 作爲模型輸出；$\text{Model}_{\text{Joint}}$ 則為將前兩種方法結合同時得到 $h_{[\text{CLS}]}$ 和 h_i。

$$
(h_{[\text{CLS}]}, h_1, \ldots, h_n) = \text{BERT}(x_{[\text{CLS}]}, x_1, \ldots, x_n) \tag{3.7}
$$

3.4 利用 Attention Weight 配合通用 NER 工具之實作

標記文章中高風險的實體需耗費大量人力才能完成，然而判斷一篇文章是否含有金融犯罪事件則相對簡單許多。因此我們額外提出了一個方法，用於缺乏 NER 標記（僅有文章分類標記）的資料集上，透過現有的工具來達成目標實體抽取之任務。

在缺乏 NER 標記的情況下，我們僅能仰賴現有的通用 NER 工具（這邊為了與以上的 NER 任務做區別，將此類工具稱為 Generic NER），從新聞文本中直接抓出人名及組織名。但是這個方法有個嚴重的問題，由於不是每個在有風險新聞

DOI:10.6814/NCCU202101564

因獵雷艦案涉貸63億的慶富造船集團董事長陳慶男，由高雄地院於2月25日以涉嫌重大，有逃亡、串證之餘裁准羈押。
陳慶男日前出庭以「中風危險」為由提出交保請求；然而合議庭像看守所確認陳「生命徵象穩定」，且仍有畏罪潛逃可能，裁定自9月20日起，延押2個月。台北地院檢察官王大平針對這件事情做出了以下的評論……

圖 3.10: 新聞範例二

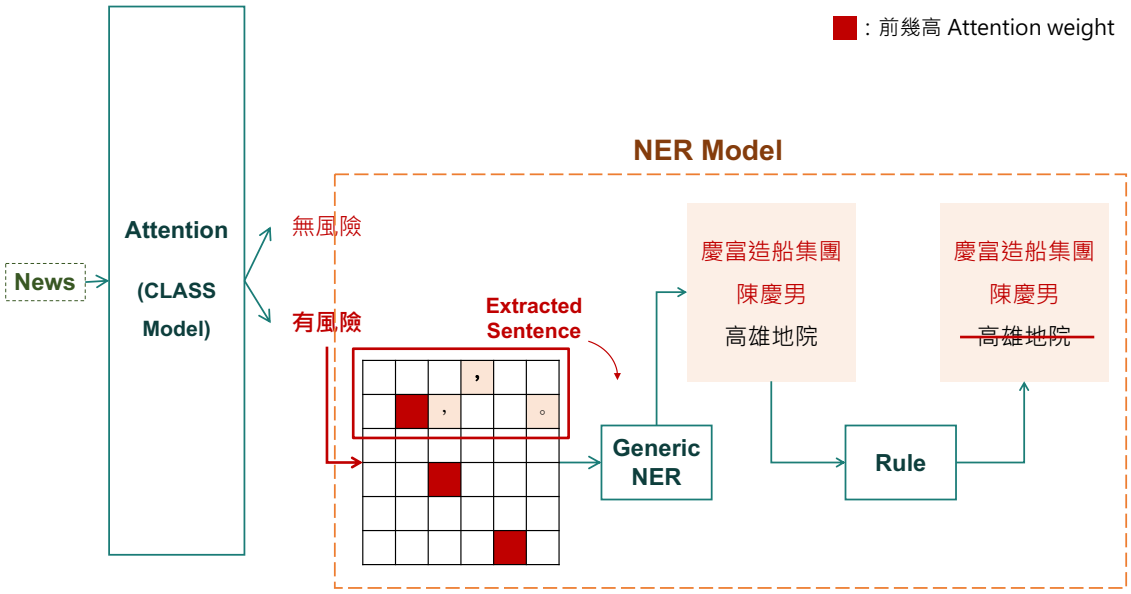


圖 3.11: Attn CLASS 模型和 Generic NER & 規則

裡的人名或組織名都是潛在罪犯，如記者、法官名……等名稱皆不會是我們任務的目標（如圖 3.10 中藍色字體）。不過經由觀察發現，通常有金融犯罪的人和組織名（圖 3.10 紅色字體）會出現在犯罪詞彙（圖 3.10 橘色字體）的附近，因此根據 3.3.2 節所述 Attention 之特性，透過 CLASS 任務中學習到的 Attention 定位犯罪詞彙，排除不合理的人名及組織名。

我們將此方法稱之為 Attn + Generic NER，如圖 3.11 所示，首先將新聞文本丟入 Attn CLASS 模型後，抽取出高風險新聞，並使用通用 NER 工具抽出所有的人名及組織名。接著在 Attn CLASS 模型中，取 Attention Weight 前 k 高的詞，圈選出這些詞所在的句子及其前後各一句（將這些句子稱為 Extracted Sentence），並濾掉所有不屬於這些句子中的實體。根據以上的假設，這個方法可以過濾掉不少的非犯罪風險的實體。

但在 Extracted Sentence 中，還是會有些非犯罪風險的實體，如圖 3.10 範例之高雄地院（紫色字體），我們額外設計了一些規則（Rule），篩掉一些常見的非犯罪的實體。詳細條件為刪除行政機關（如：五院及其底下組織……等）、組織名包含府、院、地院、局、法院、派出所、檢方……等的名詞，新聞開頭的記者名（如「○○○報導」）、詞長度為一……等的實體。

DOI:10.6814/NCCU202101564

第四章　實驗結果

4.1　資料說明

本研究目標在於萃取出文本資料中的目標實體。不同於傳統之通用 NER 任務，本研究注重於較細緻的實體類別，如：從眾多人名、地名等找出與任務目標「有關」的人名、地名實體。根據本研究所需之資料特性，找到兩份資料集 — Wiki 資料集與 News 資料集，前者為從 Wikipedia 資料庫爬取資料構成之虛擬資料集，後者則是真實的金融犯罪之新聞資料集，用以證明方法的可行及實用性。

4.1.1　Wikipedia 資料（Wiki 資料集）

欲模擬金融黑名單系統環境，我們從 Wikipedia[1] 上抽取兩種性質的文章 — 電影和小說的介紹文。此資料集的任務目標為從文章中標記出演員，故在文章層次之分類為電影及非電影（小說的文章）；實體層次標記類別分為演員和非演員，其中非演員包含電影文本裡的導演、編劇、故事角色名及小說文章裡的作家、故事角色名等人名。統計數據如表 4.1，共爬取 7140 篇文章，包含電影 2997 篇、小說 4143 篇，4325 個演員實體。

抽取方式為使用 Wikidata[2] 提供的 Query API 找出所需的 Wikipedia 頁面網址後，再用爬蟲技術將網址文章擷取下來。而兩種文章因資料特質不同，所以使用的 Query 參數及步驟也有所不同，電影的部分，先找出屬於電影的 Item，再抽取出 Item 的屬性為 Cast Member 裡所有演員名字，為了得到更多文章，再將這些名字丟入搜尋，抽取出他們所飾演的電影名單，最終得到的電影名單再去查詢出對應 Wikipedia 頁面網址。而在小說上，與電影介紹文不同，文章中會提到名字的小說介紹文比例較少，因此先抽取 Item 性質為文學角色，再搜尋這些文學角色所登場的作品，最終得到的小說作品名單再去查詢對應之 Wikipedia 頁面網址。

[1] http://en.wikipedia.org
[2] https://www.wikidata.org

DOI:10.6814/NCCU202101564

	News	*Wiki*
文章總數	8022	7140
目標文章數	670	2997
目標 NER 數	1450	4325
文章類別	risk / non-risk	movie / non-movie
實體標記類別	criminal / other-person	actor / other-person

表 4.1: 資料集統計數據[3]

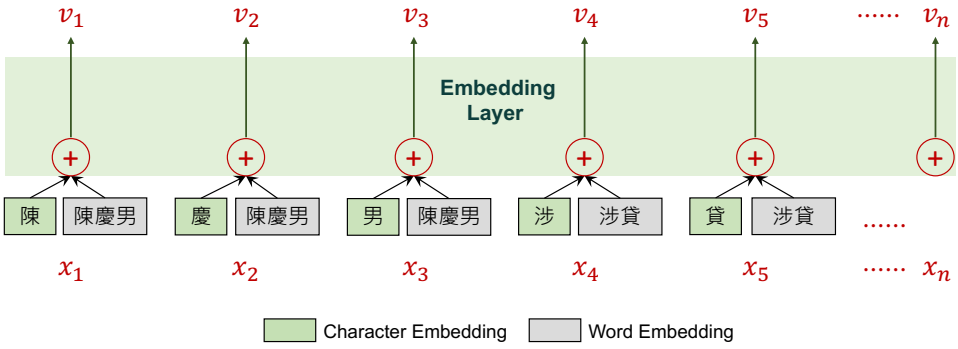


圖 4.1: News 資料集之輸入向量示意圖

4.1.2 新聞資料 (News 資料集)

為了將模型實驗在真實世界的資料上，測試方法的可行及實用性，我們使用網上公開的比賽資料[4]做為新聞文本。此資料集的任務目標為判斷新聞內文是否含有反洗錢相關焦點人物，並擷取出焦點人物名單。在文章層次的我們將新聞分類為風險新聞及無風險新聞，也就是說，當新聞中出現與金融犯罪事件，則新聞標為有風險；而實體層次標記類別分為與金融犯罪相關之實體（人名或組織名）與其他非相關的名字。統計數據如表 4.1，風險文章 670 篇、非風險文章 7352 篇，共 8022 篇，焦點人物名單共 1450 個。

4.2 實驗設定

本研究在以上兩個資料集上進行實驗。在模型輸入之文字 Embedding 設定上，所有方法模型之 Pre-Trained Embedding 會隨著訓練過程變動。在 Basic 架構[5]上，News 資料集使用 CkipTagger 提供之預訓練向量，將 Word 和 Character Embedding 相加作為輸入向量（維度為 $d_v = 300$），如圖 4.1；而 Wiki 資料集

[3]資料集下載連結：https://drive.google.com/drive/folders/1jaPWII_VqtPZA65cZglckVMDxBCqtU0J?usp=sharing

[4]https://tbrain.trendmicro.com.tw/Competitions/Details/11

[5]為 LSTM、Attention 及 LSTM+Attention 三種方法

DOI:10.6814/NCCU202101564

	#Parameters
LSTM	1085K
Attn	452K
LSTM+Attn	1446K
BERT	102M

表 4.2: 模型之參數數量

使用 Google 用 Word2Vec 在 News 上 Pre-Trained 的 Embedding[6]，模型輸入特徵只使用 Word Embedding（維度為 $d_v = 300$）。在 BERT 模型中，News 資料使用 `bert-base-chinese` Pre-Trained 模型，Wiki 資料則用 `bert-base-cased` 為 Pre-Trained 模型。

程式碼實作使用開源的 Python 套件 — PyTorch [14]，而 Transformer-Based 模型則用 Hugging Face 團隊所開發的 Transformers 套件 [15] 進行實作，CRF Layer 則使用 pytorch-crf 套件[7]。在 Generic NER 的部分，News 資料使用 CkipTagger 的 NER 工具，Wiki 資料則是使用 Nltk[8] 工具抽取 NER。

此外在參數設定方面，Basic 架構的 Optimizer 為 Adam [16]，Learning Rate 為 0.001，Max Length 為 512 個字，Hidden Size d_h 為 300；而 BERT 模型設定為：Optimizer 為 AdamW [17]，Learning Rate 為 0.00005，Max Length 為 512 個字，Hidden Size d_h 為 768 維度；Attn+Generic NER 方法在 News 資料集中 $k = 10$、在 Wiki 資料集中 $k = 3$。超過 Max Length 的文章內容會被切割成多個 Chunk 分別送入模型。表 4.2 為各個模型的參數數量。

4.3 實驗結果分析

本研究在以下小節對 Wiki 及 News 資料集，分別針對兩階段和聯合訓練方法做文章分類、實體辨識及人名抽取結果實驗分析。其中文章分類實驗主要為評斷各模型在 CLASS 任務上的表現；實體辨識實驗主要為評斷各模型在 NER 任務上的表現；而人名抽取實驗則是評斷整個完整模型及 Pipeline 的表現。

4.3.1 文章分類結果

此實驗目的為評估 CLASS 任務中個模型的分類文章的能力，在兩階段訓練的方法是使用 CLASS 任務之模型，聯合訓練則是使用 CLASS Output 做測試，Wiki 資料集為分辨文章是否為電影相關文章；News 資料為分辨新聞文本是否包含金

[6] `https://code.google.com/archive/p/word2vec/`
[7] `https://github.com/kmkurn/pytorch-crf`
[8] `https://www.nltk.org`

DOI:10.6814/NCCU202101564

		Two Stages				Joint			
		Acc.	*Prec.*	*Rec.*	*F1*	*Acc.*	*Prec.*	*Rec.*	*F1*
LSTM		95.73	91.91	96.43	94.12	94.02	93.21	91.96	92.58
Attn		94.65	92.51	93.75	93.13	83.53	84.08	75.45	79.53
LSTM+Attn		93.13	93.09	90.18	91.61	92.59	93.43	88.84	91.08
LSTM	**+CRF**	-	-	-	-	94.73	94.12	92.86	93.48
Attn	**+CRF**	-	-	-	-	88.65	87.50	84.38	85.91
LSTM+Attn	**+CRF**	-	-	-	-	94.47	93.27	92.86	93.06
BERT		96.37	95.91	94.20	**95.05**	96.37	94.69	95.54	**95.11**

表 4.3: Wiki 文章分類結果（%）

		Two Stages				Joint			
		Acc.	*Prec.*	*Rec.*	*F1*	*Acc.*	*Prec.*	*Rec.*	*F1*
LSTM		88.21	91.37	77.09	83.62	87.11	90.36	74.96	81.94
Attn		89.37	95.91	79.04	86.66	88.17	88.41	77.26	82.46
LSTM+Attn		92.79	97.38	85.79	**91.22**	89.68	88.80	80.28	84.33
LSTM	**+CRF**	-	-	-	-	83.79	93.19	68.03	78.64
Attn	**+CRF**	-	-	-	-	90.89	90.12	82.59	86.19
LSTM+Attn	**+CRF**	-	-	-	-	89.14	94.66	78.69	85.94
BERT		98.49	94.58	86.86	90.56	98.67	97.02	86.68	**91.56**

表 4.4: News 文章分類結果（%）

融犯罪相關的內容。每篇文章中只要有任何一個 Chunk 被標記為目標類別便預測該文章為電影文章／金融風險新聞。本任務使用 scikit-learn 套件[9]計算以下指標：Accuracy（Acc.）、Precision（Prec.）、Recall（Rec.）及 F1 分數[10]。

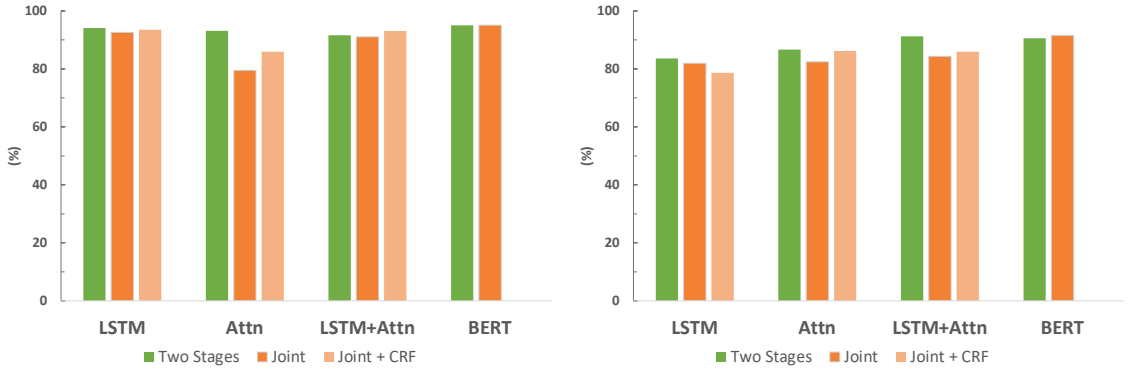
從表 4.3、4.4 及圖 4.2 觀察可知，兩個資料集在 BERT 架構上表現最好，且 BERT 架構的兩階段訓練與聯合訓練的效果皆差不多。而在 Basic 架構中，兩階段訓練表現比聯合訓練稍微好一些，推測是因爲 NER 任務較為複雜，故聯合訓練需要學習的資訊較多，使得整體表現略差一些。Basic 架構中的 Joint 模型，大致上加 CRF 比沒有加的好，表示雖然 Joint 模型會受到 NER 的拖累而影響，但依然可以利用 CRF 學習到的資訊幫助文章分類任務。

Basic 架構之三種架構比較，發現在 Wiki 資料集上 Attention 模型比 LSTM 模型差，News 資料上則相反，推測原因為兩種資料集的差異導致，表示在 Wiki 資料集除了關注到的詞重要之外，上下文也是主要的判斷依據，因為 Attention 模型在學習過程中沒有上下文資訊，但 LSTM 模型有，故 LSTM 的結果相對較好；反

[9] `https://scikit-learn.org`

[10] Accuracy = (TP+TN)/(TP+TN+FP+FN)、Precision = TP/(TP+FP)、Recall = TP/(TP+FN)、F1 Score = 2/(1/Precision + (1/Recall))。其中 TP 為預測為真且實際為真的樣本數、FP 為預測為真且實際為假的樣本數、FN 為預測為假且實際為真的樣本數、TN 為預測為假且實際為假的樣本數。

DOI:10.6814/NCCU202101564



(a) Wiki 資料　　(b) News 資料

圖 4.2: 文章分類結果之 F1 分數

之，在 News 資料集上，關注到的詞比上下文還要重要。在 LSTM+Attention 的模型表現上，因為最後是由 Attention Layer 輸出結果，因此模型表現會與 Attention 模型表現較一致。也因為如此，LSTM+Attention 模型在 Wiki 資料上表現最差，反之，在 News 資料上模型效果最好。

另外，在 News 資料上，出乎意料的 LSTM+Attention 模型效果比 BERT 兩階段訓練好，推測原因為 BERT 模型其架構比 LSTM+Attention 模型複雜，所以需訓練學習的參數很多，而文章分類任務較為簡單，故當使用複雜參數來模擬簡單情境可能導致模型效果較差。

4.3.2 實體辨識結果

此實驗目地為評估 NER 模型使用抽取實體的能力，僅有合乎順序的 NER 標記才會被視為實體（也就是說如 B-PERSON 接 E-O-PERSON 這樣不合法的標記在此任務中會被移除）。在兩階段的方法是使用 NER 模型，Joint 模型則是使用 NER Output 做測試。Generic NER 方法因不是使用 BIOES 標記，故不列入分數中。此任務使用 seqeval 套件[11]計算以下指標：Precision（Prec.）、Recall（Rec.）及 F1 分數，並使用 Macro 平均[12]兩個類別（Person 與 Other Person）。因為 NER 任務中兩個類別比例相對懸殊，故使用 Macro 平均以避免結果過度偏向某個指標。

從及表 4.5、4.6 及圖 4.3 觀察可知，兩個資料集的 BERT 模型表現最好。然而同樣使用 Attention 機制的 Attention 模型分數卻最差，原因在於上下文資訊之於 NER 任務非常重要，但 Attention 模型並沒有此文字先後順序的資訊，亦沒有使用像 BERT 中的 Position Encoding，所以導致此模型在此實驗表現差。在其他 Basic 架構上，也因為上下文資訊重要，所以有序列資訊的 LSTM 模型結果比 Attention

[11] https://github.com/chakki-works/seqeval

[12] Micro 平均是直接計算整份資料集的指標分數，Macro 則是分別計算每個類別的指標再進行平均。

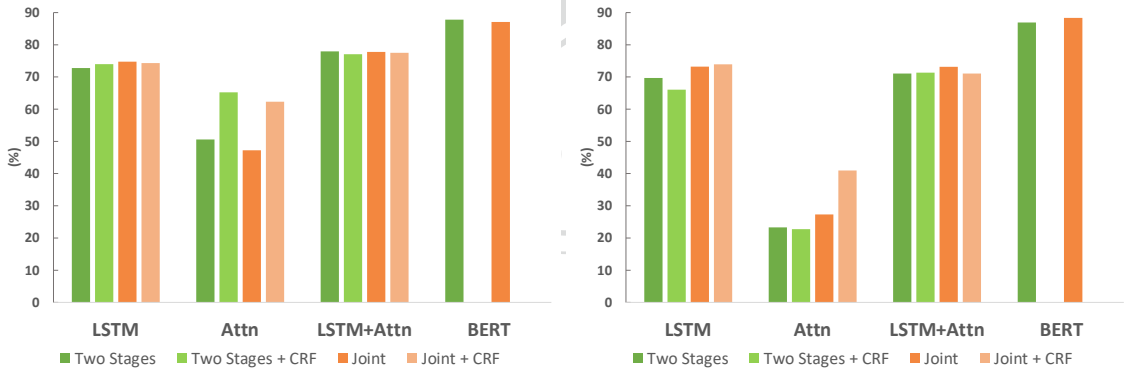
DOI:10.6814/NCCU202101564

		Two Stages			Joint		
		Prec.	*Rec.*	*F1*	*Prec.*	*Rec.*	*F1*
LSTM		74.83	71.32	72.79	77.29	72.50	74.74
Attn		74.11	39.48	50.62	76.42	35.92	47.24
LSTM+Attn		79.12	76.87	77.97	79.78	75.93	77.79
LSTM	**+CRF**	73.55	74.53	74.01	75.78	73.10	74.34
Attn	**+CRF**	71.51	60.32	65.23	73.28	55.01	62.34
LSTM+Attn	**+CRF**	76.81	77.36	77.06	78.25	76.80	77.52
BERT		87.73	87.90	**87.82**	86.91	87.28	**87.10**

表 4.5: Wiki 實體辨識結果（%）

		Two Stages			Joint		
		Prec.	*Rec.*	*F1*	*Prec.*	*Rec.*	*F1*
LSTM		70.56	68.92	69.68	79.90	67.62	73.19
Attn		41.66	16.31	23.29	49.45	19.02	27.34
LSTM+Attn		73.65	68.67	71.05	81.99	66.63	73.14
LSTM	**+CRF**	69.20	64.87	66.04	79.42	69.48	73.92
Attn	**+CRF**	47.95	16.51	22.75	74.06	30.87	40.94
LSTM+Attn	**+CRF**	79.30	64.98	71.32	78.77	65.56	71.07
BERT		89.07	84.90	**86.93**	89.98	86.79	**88.36**

表 4.6: News 實體辨識結果（%）



(a) Wiki 資料　　(b) News 資料

圖 4.3: 實體辨識結果之 F1 分數

模型好，不過當 LSTM 再加上 Attention 模型分數有再提升，表示增加 Attention 模型資訊幫助 NER 任務的學習。

在 Attention 模型上，加了 CRF 大致上可以使效果變好，代表 CRF 模型補足 Attention 模型沒有序列概念的問題。然而在 News 資料的兩階段之 Attention 模型增加 CRF 並沒有提升，可能原因在於單 Attention 模型學出來的資訊就已經很差，

DOI:10.6814/NCCU202101564

以至於 CRF 模型無法從 Attention 提供的資訊學習到 NER Tag 之間的關係。

在兩階段訓練和聯合訓練的比較上，兩種資料集表現結果不同：對於 Wiki 資料集，兩階段訓練和聯合訓練結果差不多，顯示 NER 模型在 Wiki 資料上已經學習的夠好了，透過 Joint 模型學習文章分類的資訊對於 NER 任務並沒有額外的幫助。雖然如此，但 Joint 模型訓練時間比兩階段訓練快，也不需要繁複的模型間資料轉換，訓練上效率較高。而在 News 資料上，大部分模型聯合訓練分數比兩階段訓練高，推測可能原因在於相對於英文語義，中文較難學習，故在將兩種任務一起訓練時，模型可以利用文章分類資訊幫助 NER 任務的學習，這是兩階段訓練無法辦到。

4.3.3 目標實體抽取結果

此實驗為本研究之最終任務目標 — 計算文章中正確抽出的目標實體，若文本出現多次相同名字，只要有任何一個人名被標記為 Person 即列入目標名單內。Wiki 資料集的目標為抽出演員名，News 資料集的目標為抽出的反洗錢黑名單之實體。主要比較結果「有／無用 CLASS 模型過濾」，「有」用 CLASS 模型過濾表示利用文章分類模型過濾出目標文章，再將這些文章輸入至下一個階段 NER 模型，希望增加結果的準確率。其中 Generic NER 方法為將測試資料丟入 NER 工具直接計算結果，沒有使用其他模型。此任務計算以每篇文章中目標名單的 Precision（Prec.）、Recall（Rec.）及 F1 分數[13]。

從表 4.7、4.8 及圖 4.4、 4.5 可觀察到，兩個資料集的 BERT 模型表現最好。在有使用 Generic NER 的所有方法上，透過 CLASS 模型過濾後效果都有提升，原因在於 Generic NER 使用的 NER Tag 為廣義的標記，故所抓出的實體均為廣義的人名，會在非目標文章上拿到很多錯誤的實體，因此能透過 CLASS 模型，將那些非目標文章過濾掉，達到在非目標文章上的錯誤實體被過濾掉的目的，進而提升效果。

在 Wiki 資料上，機器學習架構（Basic 架構與 BERT 架構）整體而言無論是否使用 CLASS 模型過濾並無太大影響，與一開始設想的「有」用分類模型過濾的效果應較好有差異，推測可能原因為此資料集較為簡單，NER 任務學得不錯，因此少有在非目標文章上抓到目標實體，導致 CLASS 模型過濾作用不大。

在 News 資料上，在兩階段訓練使用 CLASS 模型過濾有顯著的提升，因為 NER 模型的預測不是完全正確，而且表示目標實體的萃取除了需要 NER 資訊外，還需要看整體資訊，而 CLASS 模型的文章分類模型包涵了整體資訊，因此可透過使用 CLASS 模型模型過濾來增加額外資訊，達到表現變好的結果。而因整體資訊的重要性，在聯合訓練上，可透過將文章分類模型和 NER 模型一起學習，讓

[13]這邊的 Precision 定義為整份資料集中「標記正確的目標實體總數」與「模型預測的目標實體總數」之比例；Recall 則為整份資料集中「標記正確的目標實體總數」與「真實的目標實體總數」之比例。

DOI:10.6814/NCCU202101564

			無用 CLASS 過濾			有用 CLASS 過濾		
			Prec.	*Rec.*	*F1*	*Prec.*	*Rec.*	*F1*
Two Stages	**LSTM**		74.78	63.67	68.78	76.64	61.42	68.19
	LSTM+Attn		77.14	74.16	75.62	79.00	67.17	72.60
	LSTM	**+CRF**	71.05	70.16	70.60	74.42	67.54	70.81
	LSTM+Attn	**+CRF**	72.67	77.03	74.79	76.42	70.41	73.29
Joint	**LSTM**		75.39	66.92	70.90	76.15	64.17	69.65
	LSTM+Attn		78.64	73.53	76.00	78.39	71.54	74.80
	LSTM	**+CRF**	74.28	67.79	70.89	76.35	63.67	69.43
	LSTM+Attn	**+CRF**	76.11	74.78	75.44	76.57	74.66	75.60
Two Stages	**BERT**		82.25	89.64	85.78	85.50	86.89	**86.19**
Joint	**BERT**		82.96	87.52	85.18	83.53	87.39	**85.42**
Generic NER			22.65	92.76	36.41	41.27	85.89	55.75
Attn + Generic NER			22.90	92.76	36.74	41.65	85.89	56.09

表 4.7: Wiki 目標實體抽取結果（%）

			無用 CLASS 過濾			有用 CLASS 過濾		
			Prec.	*Rec.*	*F1*	*Prec.*	*Rec.*	*F1*
Two Stages	**LSTM**		53.35	81.65	64.53	74.97	80.43	77.60
	LSTM+Attn		56.65	74.76	64.46	83.41	73.28	78.02
	LSTM	**+CRF**	62.86	67.61	65.15	80.26	66.40	72.67
	LSTM+Attn	**+CRF**	66.19	69.23	67.68	82.49	68.02	74.56
Joint	**LSTM**		71.46	79.76	75.38	77.20	78.14	77.67
	LSTM+Attn		76.50	72.47	74.43	76.50	72.47	74.43
	LSTM	**+CRF**	72.37	81.65	76.73	79.09	77.60	78.34
	LSTM+Attn	**+CRF**	67.34	67.07	67.21	67.34	67.07	67.21
Two Stages	**BERT**		76.44	92.85	83.85	86.32	91.09	**88.64**
Joint	**BERT**		84.98	93.93	89.23	86.09	93.52	**89.65**
Generic NER			02.52	98.65	04.91	18.71	93.66	31.18
Attn + Generic NER			03.38	75.03	06.47	22.81	71.39	34.58
Attn + Generic NER + Rule			03.98	75.03	07.56	33.29	71.39	45.41

表 4.8: News 目標實體抽取結果（%）

NER 模型可以得到文章分類模型的資訊當作輔助資訊，因此就算不使用 CLASS 模型過濾，也能達到一樣好的效果。

最後我們比較 News 資料集上 Generic NER 系列方法。單純使用 Generic NER 的 Recall 分數很高，高達 98.65%，表示 CkipTagger 工具 NER 做得很好，但因爲此工具適合使用在廣義的 NER 上，故 Precision 很低。然而，Attn + Generic NER 模型雖然 Recall 有所下降，但 Precision 卻大幅的提高，這是因為 Attention 模型傾向於關注與犯罪相關的關鍵字，而且透過資料觀察亦發現，目標實體通常位於這

DOI:10.6814/NCCU202101564

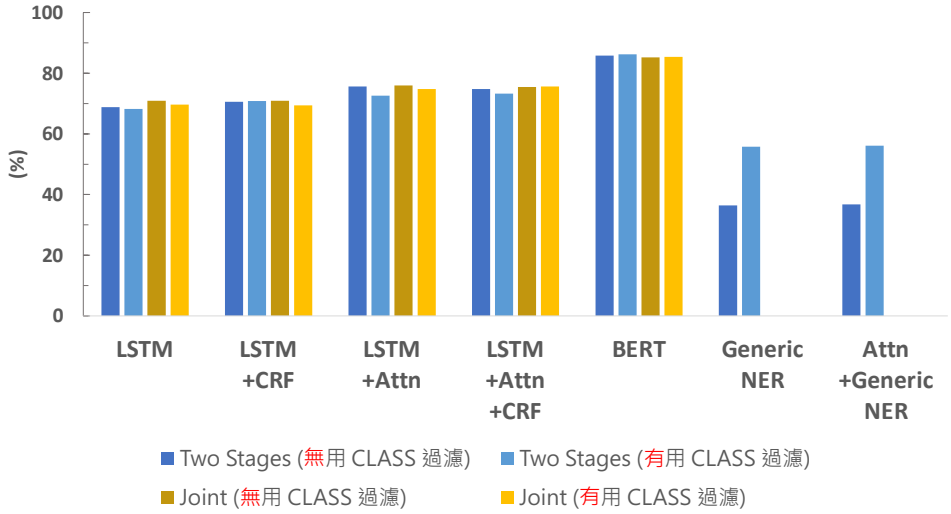


圖 4.4: Wiki 資料目標實體抽取結果之 F1 分數

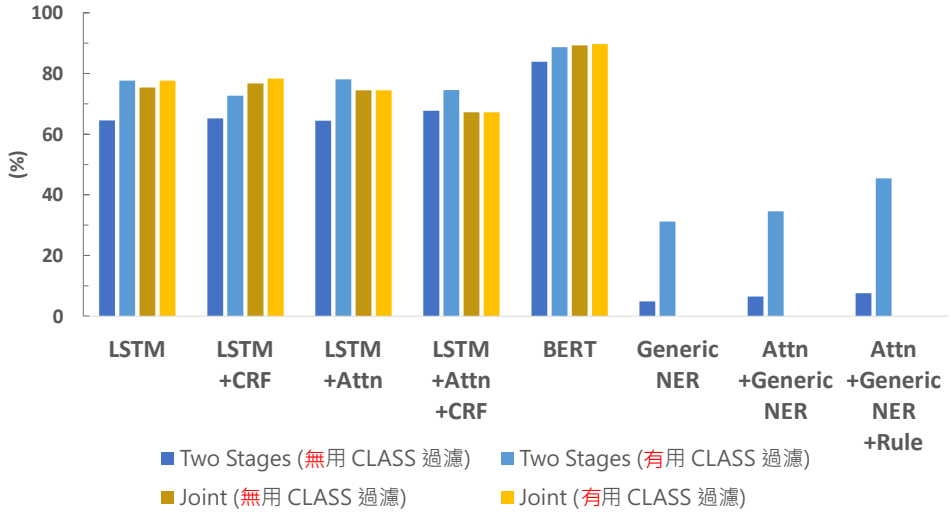


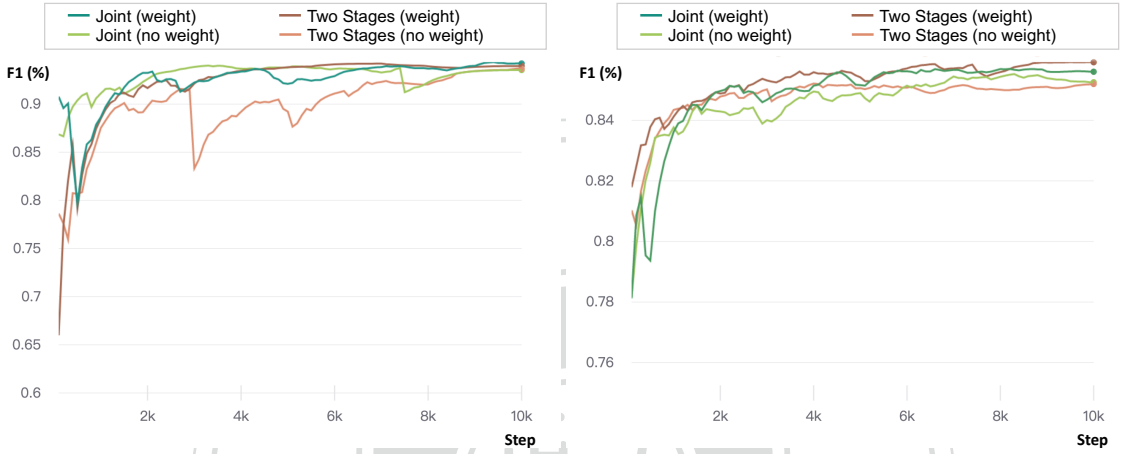
圖 4.5: News 資料目標實體抽取結果之 F1 分數

些關鍵字的前後句，所以距離關鍵字過遠的實體較不可能是任務的目標，因此透過 Attention Weight 的過濾可將這些實體過濾掉，因而達到較高的 Precision。最後觀察規則的部分，發現加上規則可在不改變 Recall 的情況下，使 Precision 有明顯上升，代表實驗定義的規則嚴謹，沒有過濾不該拔除的實體。

DOI:10.6814/NCCU202101564

		Wiki 資料			News 資料		
		Prec.	*Rec.*	*F1*	*Prec.*	*Rec.*	*F1*
Two Stages	**No Weight**	85.50	86.89	86.19	86.32	91.09	88.64
	Weight	84.17	87.64	85.87	86.46	93.93	**90.04**
Joint	**No Weight**	83.53	87.39	85.42	86.09	93.52	89.65
	Weight	82.78	87.64	85.14	86.02	96.36	**90.90**

表 4.9: 依資料比例調整 Loss 權重之目標實體抽取結果（%）



(a) CLASS 模型　　(b) NER 模型

圖 4.6: News 資料 — 依資料比例訓練模型在測試集上之 F1 分數趨勢圖

4.4 模型訓練參數分析

4.4.1 依 Positive 及 Negative 資料比例調整 Loss 權重之分析

此實驗根據訓練資料中目標文章（Positive）與其他文章（Negtive）的樣本比例，如 (4.1) 所示調整 Loss Function 中的對應權重，並分析此權重對於模型效果的影響。從表 4.9 發現，在 News 資料的實驗中，透過權重調整（表中 Weight 那列）的模型效果較好，但是在 Wiki 資料表現卻沒有明顯提升，原因在於 News 資料集中的資料比例（670：7532）與 Wiki 資料（2997：4143）相比較為懸殊，因此可透過權重調整改善模型學習的過程。

$$\begin{aligned}\text{Total Loss} = &\frac{\#\text{Positive} + \#\text{Negative}}{2\#\text{Positive}} \sum_{\text{sample}\in\text{Positive}} \text{Loss(sample)} \\ &+ \frac{\#\text{Positive} + \#\text{Negative}}{2\#\text{Negative}} \sum_{\text{sample}\in\text{Negative}} \text{Loss(sample)}\end{aligned} \tag{4.1}$$

權重調整除了可以提升模型的準確度，在模型訓練的穩定度上也有幫助。以

DOI:10.6814/NCCU202101564

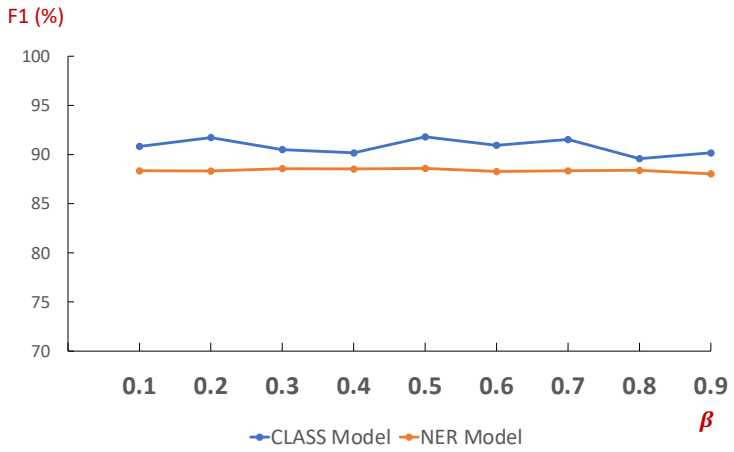


圖 4.7: News 資料 — Joint 模型兩任務 Loss 權重（β）之 F1 分數

News 資料舉例，圖 4.6 所示，橫軸為訓練時的迭代次數，縱軸為當前模型的 F1 分數（百分比制），在 CLASS 任務上（圖 4.6a），聯合訓練有加權重的穩定度較高（圖中深綠色與淺綠色比較），兩階段訓練亦是如此（圖中深和淺咖啡色比較）。有趣的是，在 NER 任務上（圖 4.6b），對於是否加權重並沒有太大的影響，分析是因為 NER 任務是使用文字的 Tag 訓練，不管是否為目標文章其內文 Tag 會有許多的「O」，因此整體文章的分佈比例不會對 NER 任務結果有很大的影響。

綜合上述，根據訓練資料的 Positive 及 Negative 比例調整在訓練模型中的 Loss 權重對資料比例越懸殊的效果越好，且較穩定。

4.4.2 Joint 模型兩任務 Loss 之權重比較

聯合訓練的 Loss 為將 CLASS 任務及 NER 任務之 Loss 相加。此實驗測試這兩項 Loss 權重不同對模型的影響，如 (4.2) 所示，測試 $\beta \in 0.1, 0.2, ...0.9$ 對於模型效果之影響。

$$\mathcal{L}_{\text{Joint}} = 2\beta\mathcal{L}_{\text{CLASS}} + 2(1-\beta)\mathcal{L}_{\text{NER}} \tag{4.2}$$

以 News 資料舉例，圖 4.7 所示，橫軸為 β，縱軸為模型的 F1 分數，可發現不管是 CLASS 模型（藍色線）還是 NER 模型（橘色線）皆不受 β 值改變所影響，表示由這兩種任務所組成的聯合訓練可以透過模型訓練共享資訊來達到相輔相成的結果，兩邊任務不會形成衝突。

DOI:10.6814/NCCU202101564

		Prec.	*Rec.*	*F1*
LSTM	**9 tag**	70.56	68.92	69.68
	3 tag	80.77	81.79	**81.20**
Attn	**9 tag**	41.66	16.31	23.29
	3 tag	72.62	74.58	**72.97**

表 4.10: News 資料 — 不同 Tag 之 NER 模型結果（%）

Before	the	Revolution	(	)	is	a	1964
Italian	romantic	drama	film	directed	by	Mery	White
.	It	stars	Adriana	Asti	and	Francesco	Barilli
and	is	centred	on	'	political	and	romantic
uncertainty	among	the	youth	of	Parma	'	.

圖 4.8: Wiki 資料 Attention Weight 圖示

4.5 Attention NER 模型學習到之資訊分析

根據 4.3.2 節實體辨識實驗結果發現 Attention 架構在 NER 任務上效果不好，推測原因為 Attention 模型無法學習到 NER Tag 的序列資訊，但還是有學習到 NER Tag 的類別訊息，故此實驗目的為證明 Attention 架構在 NER 任務是否有學到 PERSON 及 O-PERSON 類別資訊，但學不到 BIES Tag 的序列資訊。作法為將同類別的 BIES 合成為一類，例如：B-PERSON、I-PERSON、E-PERSON 及 S-PERSON 均改為 PERSON 標記，所以更改後，從原本的 9 種標記變為 3 種（PERSON、O-PERSON 和 O）。

以 News 資料為例，從表 4.10 發現，Attention 模型從 9 種標記變為 3 種標記的結果有大幅增加，表示此方法在 NER 任務上雖然沒有辦法學到 NER Tag 間的關係，但可以學到 Tag 類別的位置。與 LSTM 模型相比，LSTM 因為本身有序列資訊，所以改為 3 種標記的結果，變化幅度小於 Attention 模型很多。

4.6 Attention Weight 分析

此實驗為分析使用 Attention 方法做為文章分類模型時，Attention Weight 所關注的字詞，以圖 4.8 及 4.9 所示，圖中顏色越紅代表 Attention Weight 越大，Attention 模型越注重這個字。

在 Wiki 資料上，從圖 4.8 可看到如 film、star 與電影相關的字 Attention Weight

DOI:10.6814/NCCU202101564

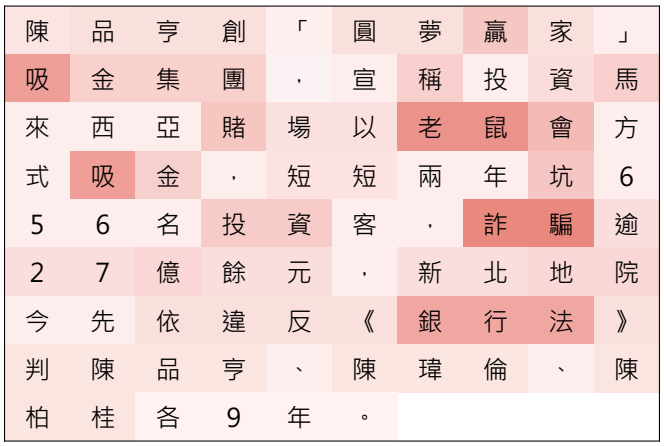


圖 4.9: News 資料 Atention Weight 圖示

較高，亦是說明 Attention 模型在做分類時很好的學習出這些與電影相關的字之重要性，進而判斷是否為電影類別。

而 News 資料上，從圖 4.9 發現如吸金、詐騙、銀行法等與犯罪相關字詞的 Attention Weight 較高，亦是說明 Attention 模型在做分類時很好的學習出這些與犯罪相關字之重要性，進而判斷新聞是否含有風險事件。

DOI:10.6814/NCCU202101564

第五章　結論

本研究透過機器學習方法從財務新聞文本資料中萃取出與金融犯罪相關的人名與組織名，並在最好的模型上可達到 90.15% 的 F1 分數。

由於此任務中人名及組織名的目標與傳統實體辨識任務不同，故我們依據資料特性提出模型，將任務分為兩個部分：其中 CLASS 任務負責分類新聞文本，找出含有金融犯罪事件的文章，減少下一階段任務的輸入資料量，以提升模型預測速度；而 NER 任務根據前一階段所抽取的新聞，萃取出目標實體。另外我們提出聯合訓練（Joint Training）方法，將兩階段訓練合併，使用共用的 Encode 模型架構同時訓練兩個任務。從實驗結果發現，聯合訓練可達到與兩階段訓練相同的正確率，並且在訓練及預測上的速度皆有提升。另外，使用聯合訓練所訓練出的 NER 模型在不使用 CLASS 任務過濾非金融犯罪事件的文章的情況下亦可以達到差不多的結果，顯示聯合訓練成功讓模型同時學到兩個任務的資訊，進一步提升預測速度。

我們使用 LSTM、Attention、混合 LSTM 及 Attention、BERT 四種不同的機器學習架構實作本實驗的各個模型，顯示 BERT 模型在各個任務上皆有最好的表現。而透過比較 LSTM 與 Attention 架構，顯示 NER 任務對於文字序列的依賴性極強，故 LSTM 架構表現比 Attention 來的好。在 Attention 架構的部分，透過圖形化分析 Attention Weight，我們發現模型會自動關注犯罪金融風險詞彙，顯示 Attention 架構模型與人類有相似的分析文章方式。

除了使用財務新聞資料做實驗外，也從英文的 Wikipedia 上爬取文章建構出相同結構的資料，試將同一套方法使用在不同領域且不同語言的資料集上，測試研究提出之方法的可行性及實用性。從實驗結果顯示，與新聞的資料集相同 BERT 架構各個任務上皆有最好的表現，並主要任務上可達到 85.87% 的 F1 分數。

我們特別針對沒有 NER 訓練資料的情況，設計了以下的模型，希望可以節省 NER 標記的人力及模型訓練成本。此模型利用了 Attention Weight 的分佈資訊，並使用的通用 NER 工具及一些特別設計的規則，雖然不及使用機器學習方法所得模型，但依然可以達到 45.41% 的 F1 分數。

在未來研究方向上，除了犯罪實體的抽取之外，我們亦打算訓練模型以抽取他們犯的罪名及其時間、地點、金額，延續本研究之方法，找出跟金融犯罪有關

DOI:10.6814/NCCU202101564

的實體及其對應的相關資訊。也希望透過收集更多的訓練資料，提升模型的效果及比較更多不同的模型設置及訓練方法的優缺點。最後希望可以將本研究提出的方法應用在更多不同領域的任務上，提供一個新的解決方案。

DOI:10.6814/NCCU202101564

參考文獻

[1] D. W. Otter, J. R. Medina, and J. K. Kalita, "A survey of the usages of deep learning for natural language processing," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 2, pp. 604–624, 2020.

[2] R. Jozefowicz, W. Zaremba, and I. Sutskever, "An empirical exploration of recurrent network architectures," in *International conference on machine learning*. PMLR, 2015, pp. 2342–2350.

[3] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *arXiv preprint arXiv:1706.03762*, 2017.

[5] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[6] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[7] K. Potdar, T. S. Pardawala, and C. D. Pai, "A comparative study of categorical variable encoding techniques for neural network classifiers," *International journal of computer applications*, vol. 175, no. 4, pp. 7–9, 2017.

[8] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[9] Q. Liu, M. J. Kusner, and P. Blunsom, "A survey on contextual embeddings," *arXiv preprint arXiv:2003.07278*, 2020.

[10] J. Li, A. Sun, J. Han, and C. Li, "A survey on deep learning for named entity recognition," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, 2020.

[11] V. Krishnan and V. Ganapathy, "Named entity recognition," *Stanford Lecture CS229*, 2005.

DOI:10.6814/NCCU202101564

[12] S. R. Eddy, "Hidden markov models," *Current opinion in structural biology*, vol. 6, no. 3, pp. 361–365, 1996.

[13] J. Lafferty, A. McCallum, and F. C. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," 2001.

[14] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[15] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. [Online]. Available: https://www.aclweb.org/anthology/2020.emnlp-demos.6

[16] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[17] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

DOI:10.6814/NCCU202101564