**Group G 期末書面報告**

**一、主題發想:**

社群媒體對任何產業具有重大影響。訊息、評論和反饋能快速傳達給大眾，直接或間接影響產品/品牌形象。故情感分析在社群媒體分析中至關重要。若有快速貼文情感分類模型，可做更多分析。如想瞭解社群對特定關鍵字態度，用情緒辨識技術對貼文分類，了解風向。此外，情緒分析也可協助企業瞭解顧客對產品/服務評價，改善品質，提升滿意度。故選擇情感分析技術作為主題。

**二、資料預處理:**

1. 資料合併:

因為我們資料分成多個檔案，還有分成繁體與簡體的資料，因此需要合併資料，並手動處理資料讀取的問題。

1. 資料清理:

我們是做自然語言處理，資料清理的步驟如下:

* 基本清理: 選擇欄位(content & sentiment)、去除空值與重複值、篩選有效的Label (sentiment欄位中會有一些文字跑進去的錯誤，需要清除)、調整Label (將N、M、P 換成0、1、2)
* 去除與情感無關字元，用正則化消除文本中的換行符號、HTML標籤、url、數字、標點符號以及空格，此步驟可以加強模型精準度，也可以減少運算資源的浪費。

1. 資料平衡:

未平衡前資料筆數(包括簡體與繁體):

正面資料筆數: 60870

負面資料筆數: 49897

中性資料筆數: 140028

由於中性資料數量過大，我們只挑出中性資料的其中 49466 筆使用。最終採用資料量60793+49847+49466=160106。

1. 資料分割:

我們將資料分配成訓練用和測試用，各為獨立資料集，詳細資料筆數如下表:

|  | 訓練用(training) | 測試用(testing) |
| --- | --- | --- |
| 正面 | 41038 | 19755 |
| 負面 | 32297 | 17550 |
| 中性 | 30000 | 19466 |
| 總計 | 103335 | 56771 |

**三、模型預訓練**

我們使用Hugging Face提供的 Transformer 模型，Transformer模型透過引入self-attention layer來解決傳統循環神經網路（RNN）在處理長序列時的限制。self-attention可以同時考慮輸入序列的所有位置，並捕捉全域的上下文資訊，使得模型能夠更好地理解序列之間的關係。

另外，我們使用的是Transformer中的Chinese-BERT-wwm，提供給中文自然語言處理的模型，我們取用其中的兩個要素，BertModel和 BertTokenizer，BertMode為神經網絡模型的核心，輸入向量化的文本後，輸出文本的embedding，即已經了解每個詞語上下文關係的向量。BertTokenizer將文本進行分詞並把文本轉換為模型可理解的輸入格式(PyTorch tensors)，並使用encoding\_plus函數，進行向量化與加入BERT獨有的特殊字元，以便做訓練。

**四、模型Fine-Tuning(調整參數前)**

因為我們是要做情感分析，所以我們需要調整模型至適合做分類學習。使用以下2個步驟:

1. **使用Pytorch提供套件建立DataLoader**，此DataLoader從原始資料集中將feature和label取出後，自動將資料集根據設定的batch\_size將資料分割，接著對feature使用tokenizer，將每筆資料向量化，同時設定向量的最大維度，並可以從中取用訓練所需的input\_ids、attention\_mask。詳細步驟可參考fine\_tuning\_for\_sentiment\_analysis.ipynb
2. **建立SentimentClassifier:**

* 初始化設定:

- 使用Pytorch中的nn.Module初始化神經網絡的基本架構

- 加載預訓練的BertModel

- 建立一个 Dropout 層，隨機丟棄部分特徵，以減少過擬合的風險。設置丟棄率為 0.3。

- 使用nn.Linear()將BERT 模型的輸出特徵维度(bert.config.hidden\_size) 映射到情感別的數量

* 向前傳播學習方法:

- 將每一批資料的input\_ids和attention\_mask輸入bert模型

- 取出模型輸出中的pooler\_output

- 透過dropout層丟棄部分特徵

- 回傳用linear進行分類的結果，即是每一筆資料每個類別的預測分數

**五、訓練模型**

1. **optimizer設定**，經過幾篇文章搜尋，我們使用Hugging Face提供的AdamW優化器，並設定學習率。AdamW修正了權重衰減（weight decay），因此與原始論文中的方法類似。另外我們還使用沒有Warmup步驟的線性學習率調整器（scheduler）。
2. **loss function設定**，使用nn.CrossEntropyLoss()當作損失函數，因為它是一個適合計算多分類模型損失的函數，用以計算實際標籤與模型輸出之間的差距。
3. **訓練函數:**

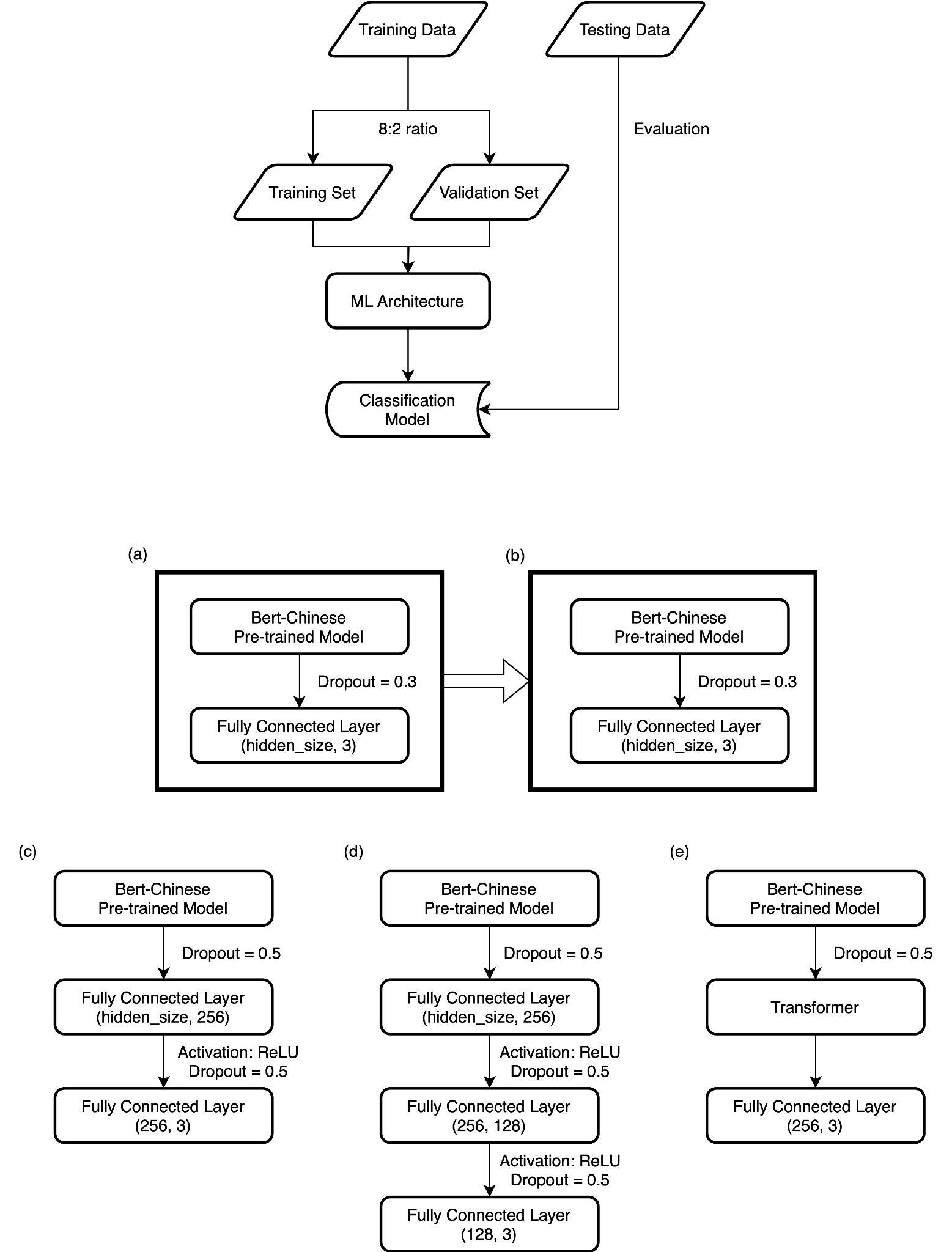
* 將模型調整到訓練模式
* 將批次的資料迭代輸入模型與紀錄輸出
* 找輸出中的最大值索引，即是該預測結果
* 用損失函數計算模型輸出與實際標籤之間的差異
* 執行反向傳播，計算梯度並更新模型的參數(詳細解釋請參閱source code)
* 紀錄並回傳該批次準確度與平均誤差值

1. **評估函數:**

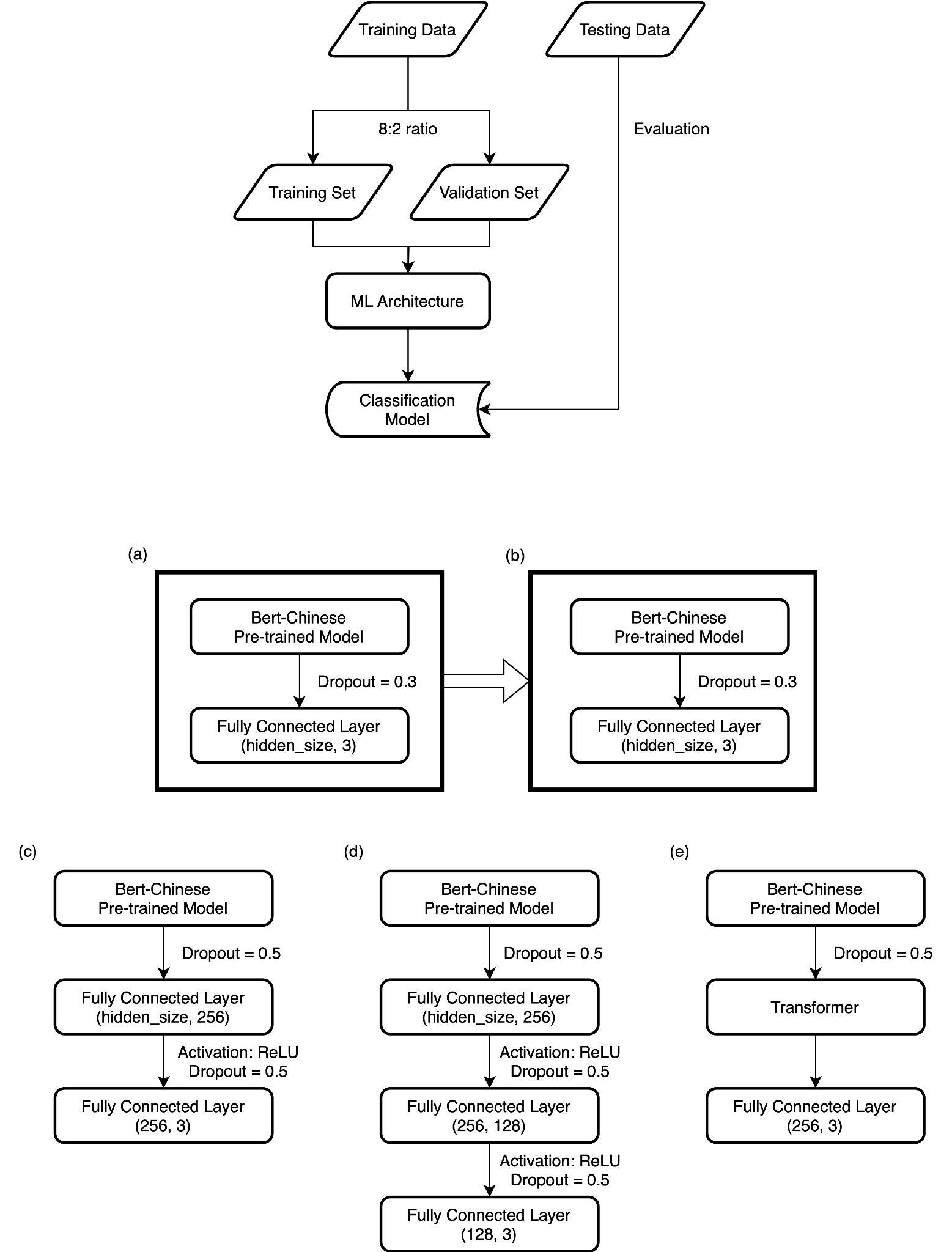
在訓練的過程中同時準備一份evaluation的資料，用來看每一輪epoch的準確度與損失函數是否提升，確認學習的效果，寫函數時，與訓練函數不同在於，評估函數不需要去使用反向傳播更新參數，只需要使用此次epoch訓練出來的模型預測，並計算準確度。

1. **開始訓練**

設定好我們要幾次epoch後，開始訓練模型，並記錄每一輪epoch的準確度與損失函數結果，若是評估函數的準確度有上升，那就保留最佳的模型。最後就會儲存最佳的模型。



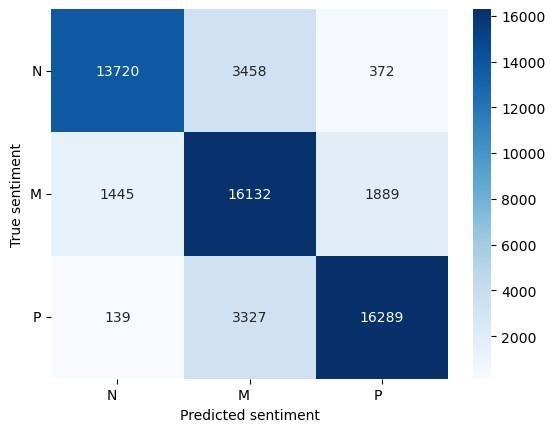
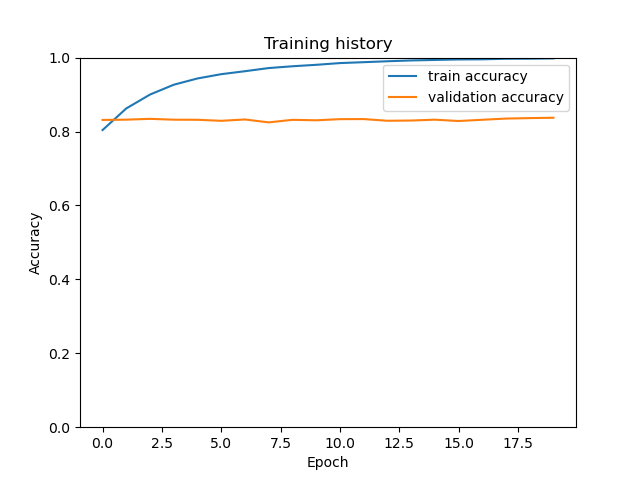
**六、測試與調整模型**

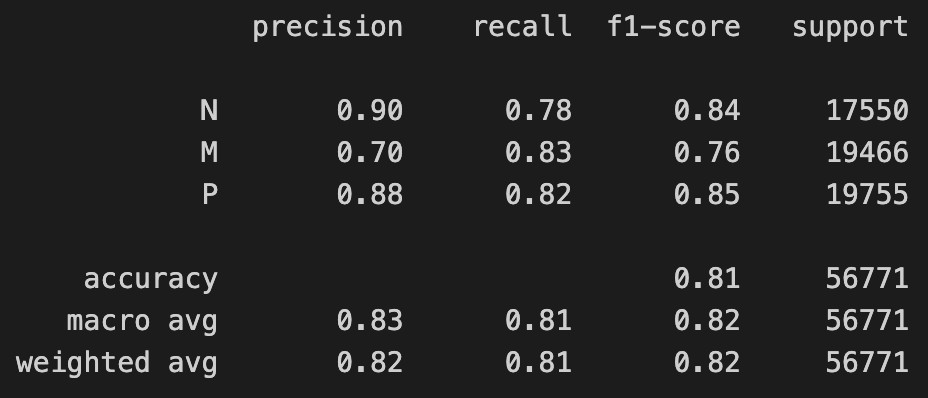
****

**ML Architectures**

1. single layer with 20 epochs

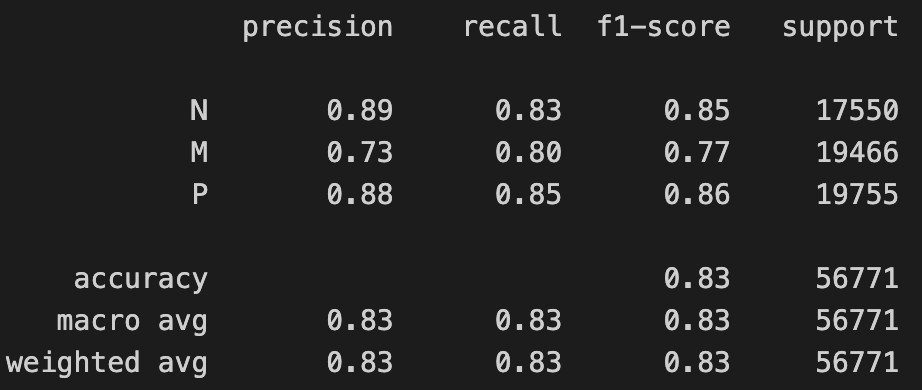
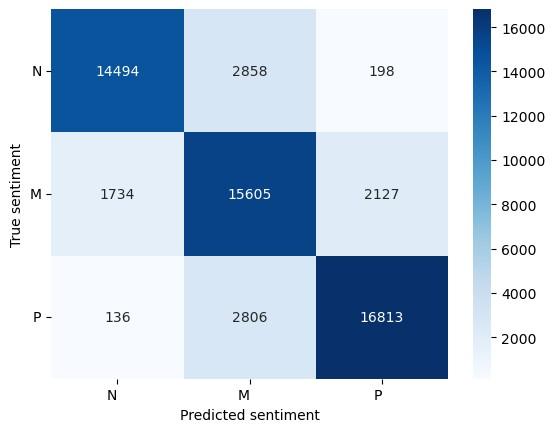
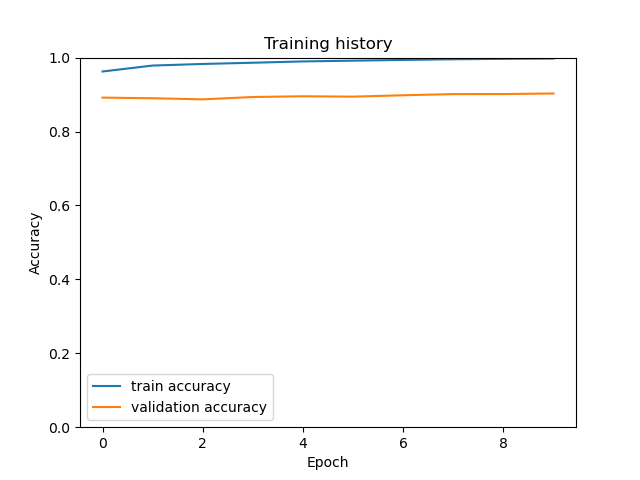
從預處理模型後只接了全連結層做分類，對於Validation沒有明顯上升





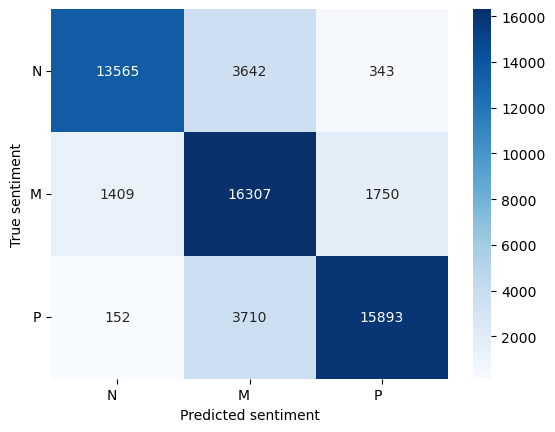
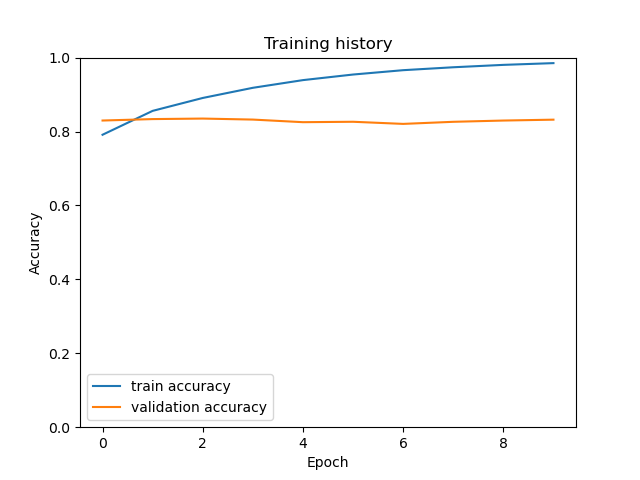
1. transfer learning-liked method based on the first model with 10 epochs \*

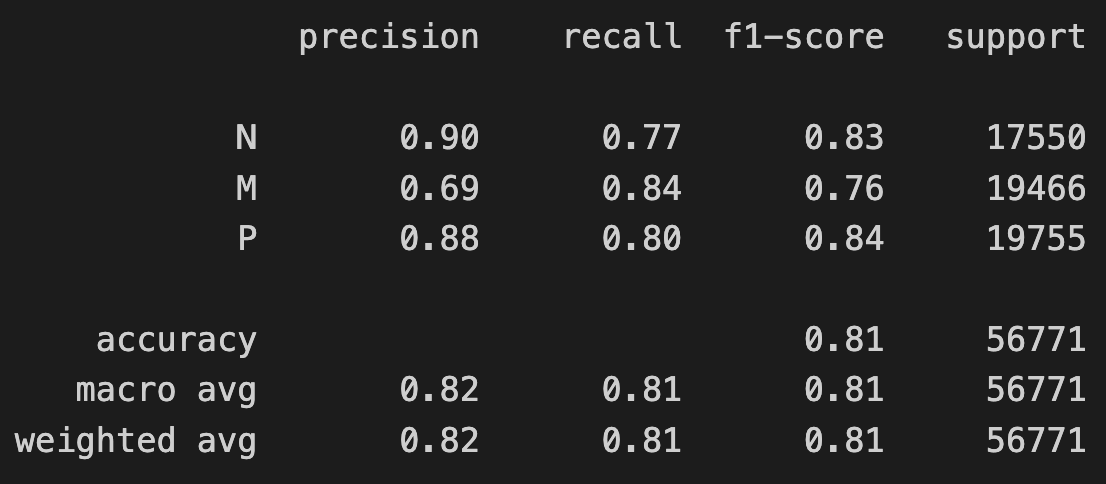
我們嘗試將第一個模型的權重輸入，並且多訓練10個epochs。transfer learning的概念中雖然要導入新的資料集進行fine tuning，但我們只是重新分配原本的training set和validation set，讓我們所有的訓練資料能夠更好的活用。



1. 2 layers with 10 epochs

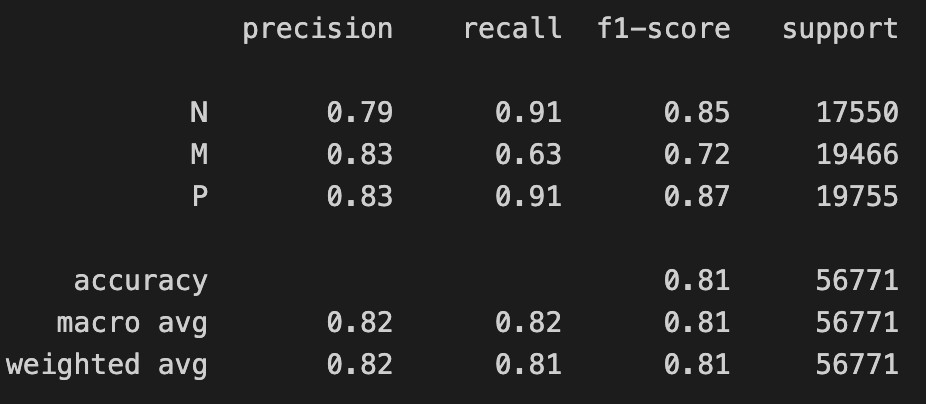
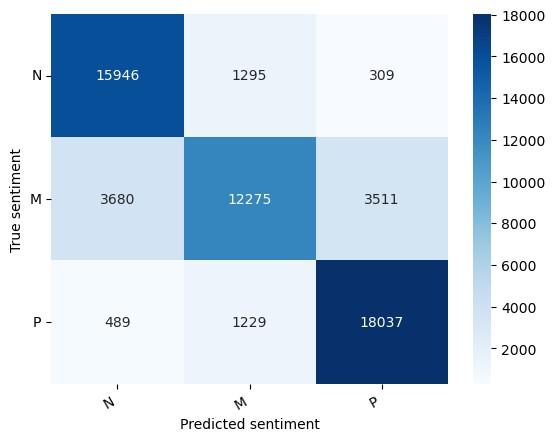
我們嘗試新增一個全連接層，並且提高Droupout，希望能對training的效果有所提升





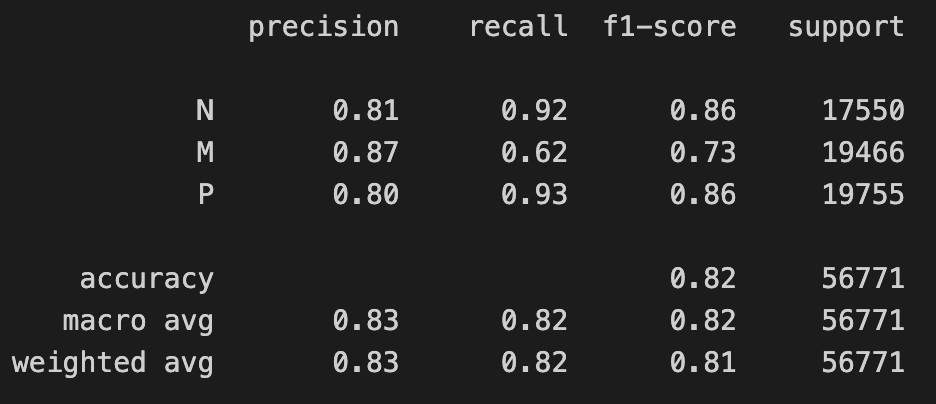
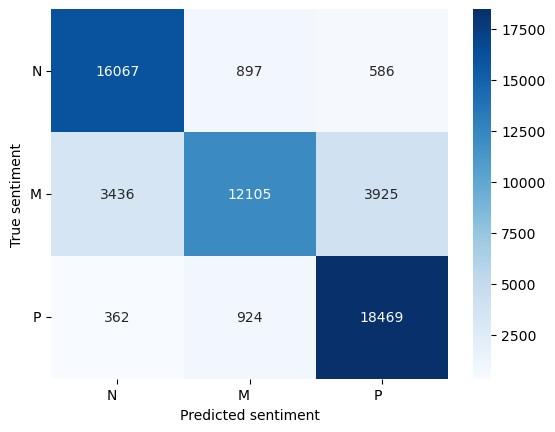
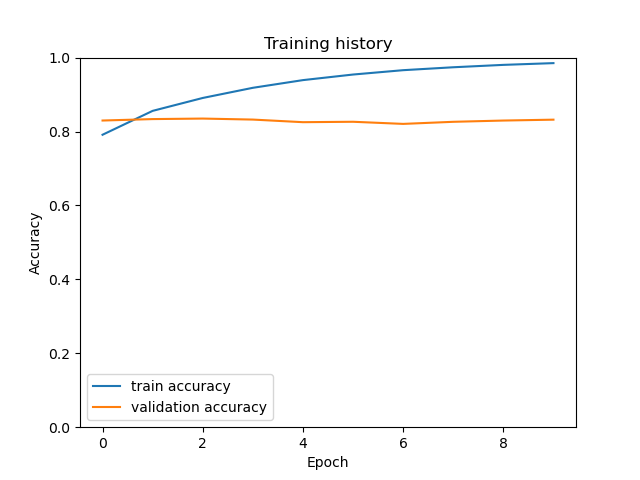
1. 3 layers with 10 epochs

再多新增一個全連接層



1. Transformer with 10 epochs

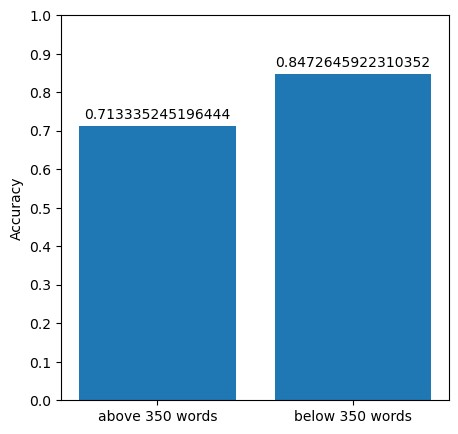
雖然經過Bert預處理後，文字已經被向量化，但在字數多的狀況下，



(\* Represents our best model)

**七、模型成果**

Bert在文字處理的預處理模型其實就能夠有十分好的成果，我們只需要做最後的分類就能得到80%左右的準確度。我們也使用較少的模型進行超參數（Hyperparameter）的Grid Search，選擇一組我們認為合適的數據再進行模型的更改。模型的架構中，我們嘗試了各種方法，並在最後能得到83%的準確率（Accuracy）。其中，我們根據最高準確率並且預測結果能在的精確率（Precision）中數值總和最高當作我們最好的模型。理由是因為有許多長文本中，其實混合了正反面的子段落，造成文意的判斷困難，細節會於困難與挑戰中討論。我們也發現在短文本的預測上其實是有更高度的準確性的，我們用切分向量最高限度的350字當作區隔（350字以下，每個字一個向量，350字以上則平均分配，共350個向量），結果如下圖。



**八、困難與挑戰**

1. **不使用CNN:**

我們在期中的報告中有提及想要使用CNN來將BERT轉成向量的輸出進行分類預測，但在經過研究與閱讀文獻之後，我們決定不這麼做，原因如下:

- CNN適合用在表示圖像的二維空間資料，而我們的資料是連續的文本，是屬於序列型向量，在"文字"本身的情感分析中，我們認為也不需要考慮空間或位置的層面(資料清理的時候甚至會把換行符號去掉)，所以CNN並不適合用在此。

- Transformer的神經網絡是通過self-attention去提取文本的上下文資訊和語義特徵，是全面性的，但是CNN擅長的是捕捉局部特徵，因此我們認為混用會造成學習效果不佳。

決定不使用CNN後，我們仍希望透過神經網絡的架構來增加學習效果。於是使用以下的解決辦法:

- 續用Transformer，增加新的layer，在現有的架構中，嘗試添加新的self-attention layer、linear layer、encoder等，並從中調整參數，嘗試透過增加神經網絡的複雜度提升學習效果。

1. **未執行停用詞刪除**

在期中我們也提及要刪除停用詞只留下對情感分析有幫助的文字，但因為中文不像英文本來就以空格分隔，中文文本進行停用詞刪除前，需要先進行斷詞操作，才能準確識別哪些詞語是停用詞。然而，經過實際嘗試後，發現這種方法的效果並不理想。由於中文的特點，例如詞語之間沒有明確的分隔符，以及一詞多義等問題，文本的複雜性導致斷詞結果不夠精準(如圖)，進而影響了停用詞的準確性。因此，我們決定不執行這一步驟。



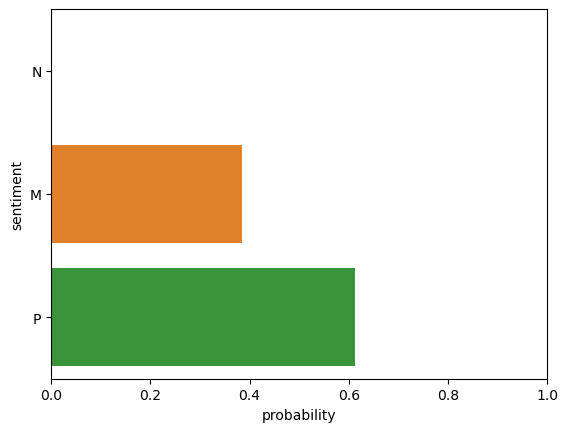
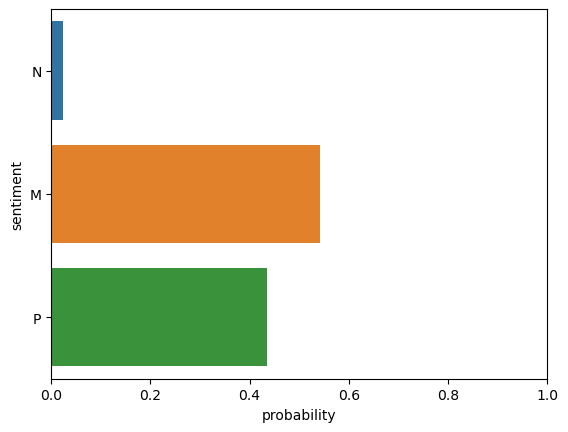
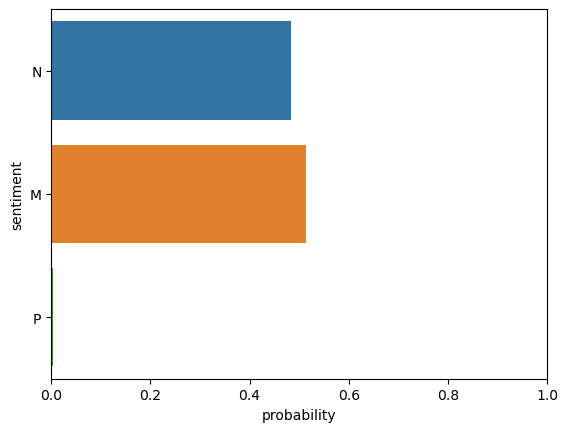
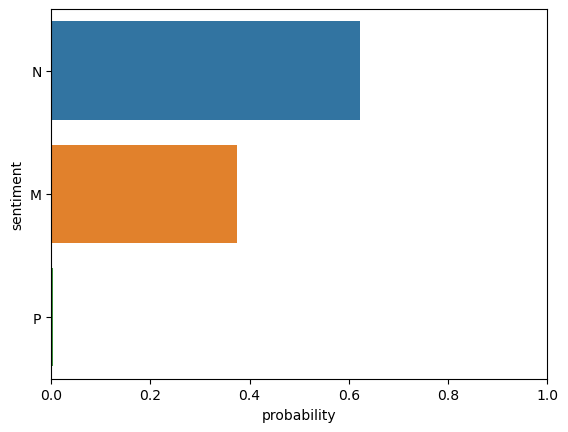
儘管停用詞刪除在一些情況下可以提高文本處理的效果，但根據我們的情況，斷詞結果的不精確性影響了停用詞的識別，因此決定放棄這一步驟。嘗試其他的文本預處理方法或調整參數來改善模型的效果。

1. **運算資源缺乏**

因為我們資料量很多(總計約20萬)，又是使用深度學習，googlecolab免費的GPU不夠用，只夠使用比例極低的資料去測試pipeline是否正常運作，且每次測試仍非常費時，加上要花時間除錯，運算資源確實是讓人困擾。幸好正式的訓練有組員實驗室的RTX4090 24G支援，才可以順利訓練模型，但是仍然有上限，像是batch size不能調太高（out of memory）、epoch無法太多（佔用GPU hour）。不過，為了解決這個問題，我們試驗了數次的調整與重新訓練，我們也發現batch size 與epoch的數量並不會對模型表現有重大影響，因此也算是順利解決。

1. **語意判斷困難**

在許多文章段落中，透露的語氣會因每個人的詮釋有所不同。本次的資料集雖然豐富，但我們並沒有餘力去針對所有的標籤進行檢查，確認語意資訊是否正確。另外，本次的分類條件除了正負面之外也包含了中性言論，隨然為我們的專題增加了創新度和豐富度，但也增加了分類上的困難。我們絕大多數的判斷錯誤都是對於中性言論和正、反面混淆，這部分可以在混淆矩陣中發現。另外，我們抓出判斷錯誤的部分來討論，其實他在兩個機率十分接近（如下圖）。



**九、心得**

這學期的課程是一段充滿挑戰的過程。一開始，我們多數人對機器學習和深度學習的概念只是模糊的了解，並且對要如何應用於中文情感分析這樣的實際問題毫無頭緒。

隨著課程的進展，還有因為這堂課的契機，大量學習相關的知識和文章，才逐漸進入了這個領域。從學習基礎的機器學習算法開始，我逐步理解各種基礎模型的原理和應用。但接觸到深度學習時，還是有些不知所措。深度神經網絡、卷積神經網絡和自注意力機制等概念需要更多的時間學習。

實踐是最好的老師，在實作專案中，遇到了各種困難和挑戰。文本的前處理、特徵提取、模型選擇和參數調整等步驟都需要不斷地嘗試和調整。分析錯誤的原因，並進行改進。我發現模型的性能不僅取決於算法本身，還取決於資料的質量和特徵的選擇。這項專案教會了我如何在實際應用中解決問題，並且對於中文文本處理和情感分析有了更深入的理解。

最終，當看到中文情感分析模型取得了不錯的結果時，非常有成就感，這次實做專案和學習的過程讓我深刻體會到機器學習的魅力和應用的廣泛性。我明白到這只是學習的開始，還有很多需要探索和學習的領域。在這個時代努力跟上知識翻新的速度並不容易。期待未來繼續探索。