國立嘉義大學資訊工程學系

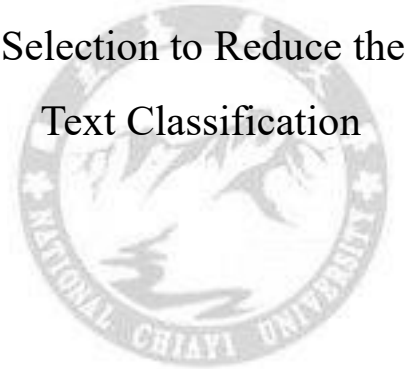
碩士論文

Department of Computer Science and Information Engineering
National Chiayi University

Master Thesis

以重複樣本選擇降低噪音資料對文章分類的影響

Using Repeated Instance Selection to Reduce the Impact of Noise Data on Text Classification



研 究 生： 孫立洋

Li-Yang Sun

指導教授： 陳耀輝 博士

Yaw-Huei Chen Ph. D.

中華民國 110 年 4 月

April 2021

以重複樣本選擇降低噪音資料對文章分類的影響

指導教授：陳耀輝 博士　　研究生：孫立洋

國立嘉義大學資訊工程學系

摘要

訓練一個深度學習的模型需要大量的訓練資料，但實際收集到的大量資料中不免有噪音資料，這些訓練資料中的噪音會嚴重影響模型分類的正確性。我們提出了使用多階段的樣本選擇方法來增強模型抵抗噪音訓練資料的能力，這個方法在每個階段都要挑選上階段資料標籤與預測標籤一致和不一致的兩種資料當成下階段的訓練資料。挑選標籤一致的資料可以留下大部分的乾淨資料，而挑選少數上階段標籤不一致的資料可以增加訓練資料的數量與多樣性。實驗結果顯示我們的方法在含有不同比例噪音的四個資料集中都可以增加模型的強健性，進而達到降低噪音訓練資料對模型影響的目的。

關鍵字：文章分類；噪音資料；強健訓練；樣本選擇

Using Repeated Instance Selection to Reduce the Impact of Noise Data on Text Classification

Advisor: Yaw-Huei Chen, Ph.D. Student: Li-Yang Sun
Department of Computer Science and Information Engineering
National Chiayi University

Abstract

Training a deep learning model requires a large amount of training data, but the actual collected data may contain noisy data, and the noise in these training data can seriously affect the classification correctness of the model. We propose a multi-stage instance selection method to enhance the model's ability to resist noise training data. At each stage, we select instances whose labels are consistent with the predicted labels as well as a portion of instances whose labels are inconsistent with the predicted labels as the training data for the next stage. Picking the consistent labeled data preserves most of the clean data, while selecting a small portion of inconsistent labeled data from the previous stage can increase the number and diversity of training data. The experimental results show that our method can increase the robustness of the models with various noise ratios in four data sets, and thus reduce the influence of noisy training data on the models.
Keywords: text classification; noise data; robust training; instance selection.

誌謝

研究所的求學期間讓我學習到很多事情，不只是課業方面，在待人處事方面也受益良多。能完成這篇論文，首先要感謝的是我的指導教授陳耀輝老師，不管是在課業、研究與生活方面，老師總能不厭其煩的教導我幫助我解決問題，每次在與老師開會討論的過程中我總能從中學習到許多東西，讓我學習到面對問題需要抱著嚴謹、細心與認真的態度，以及從不同的面向來研究問題。在求學期間的上台報告也讓不善言辭的我藉由老師的教導讓我學習到如何表達，以及閱讀與整理論文。感謝口試委員: 蔡正發教授、邱志義教授與方文杰教授提出對本論文非常寶貴的意見，使得本論文得以更加完整的呈現。也感謝實驗室裡一起努力的學長姐與學弟妹等，還有其他好朋友們，在我有困難時總能伸出援手給予我協助，因為有你們讓我的研究生生活更加地豐富精采。最後我要感謝我的父母以及家人們，謝謝你們的支持與鼓勵還有體諒，讓我得以完成碩士學位。

孫立洋

國立嘉義大學 知識與資訊研究室

中華民國一一〇年四月二十八日星期二

目錄

摘要 i
Abstract ii
誌謝 iii
目錄 iv
圖目錄 vi
表目錄 viii
第一章 緒論 1
1.1. 研究背景 1
1.2. 研究目的 1
1.3. 研究貢獻 1
1.4. 論文架構 2
第二章 相關研究 3
2.1. 資料前處理 3
2.2. 強健訓練 (robust training) 5
第三章 研究方法 16
3.1. 方法概念 16
3.2. 方法流程 16
第四章 實驗 26
4.1. 資料集 26
4.2. 實驗環境 28
4.3. 實驗設計 28
4.3.1. 評量公式 40
4.3.2. 噪音加入方法 40
4.3.3. 訓練 41

4.3.4. 測試.....44
4.4. 實驗結果.....44
4.5. 實驗討論.....51
第五章 結論與展望.....54
參考文獻.....56

圖目錄

圖 2.1: 自動編碼器架構示意圖 4
圖 2.2: 自動編碼器偵測噪音資料示意圖 4
圖 2.3: 噪音適應層架構示意圖 7
圖 2.4: Decoupling 示意圖 9
圖 2.5: Co-teaching 方法示意圖 10
圖 2.6: Co-teaching+方法示意圖 12
圖 3.1: 二分類演算法步驟 22
圖 3.2: 二分類演算法流程圖 22
圖 3.3: 多分類演算法步驟 25
圖 3.4: 多分類演算法流程圖 25
圖 4.1: 傳統類神經網路參數數量 30
圖 4.2: AG's News 中傳統類神經網路參數數量 30
圖 4.3: Sogou 中傳統類神經網路參數數量 30
圖 4.4: 傳統類神經網路在 IMDb 與 Elec 架構圖 31
圖 4.5: 傳統類神經網路在 AG's News 架構圖 32
圖 4.6: 傳統類神經網路在 Sogou 架構圖 33
圖 4.7: IMDb 與 Elec 中我們方法的網路參數數量 33
圖 4.8: AG's News 中我們方法的網路參數數量 34
圖 4.9: Sogou 中我們方法的網路參數數量 34
圖 4.10: 我們方法使用的網路在 IMDb 與 Elec 架構圖 35
圖 4.11: 我們方法使用的網路在 AG's News 架構圖 36
圖 4.12: 我們方法使用的網路在 Sogou 架構圖 37
圖 4.13: Co-teaching 在 IMDb 與 Elec 中網路參數數量 38

圖 4.14: Co-teaching 在 AG's News 中網路參數數量 38
圖 4.15: Co-teaching 在 Sogou 網路參數數量 38
圖 4.16: Co-teaching 在 IMDb 與 Elec 網路架構圖 39
圖 4.17: Co-teaching 在 AG's News 網路架構圖 39
圖 4.18: Co-teaching 在 Sogou 網路架構圖 40
圖 4.19: 準確率公式 40
圖 4.20 : IMDb 在不同噪音率下測試資料準確率的折線圖 45
圖 4.21: Elec 在不同噪音率下測試資料準確率的折線圖 46
圖 4.22: AG's News 在不同噪音率下測試資料準確率的折線圖 47
圖 4.23: Sogou 在不同噪音率下測試資料準確率的折線圖 48

表目錄

表 2.1: 我們方法與其他兩種方法挑選差異.....12
表 2.2: 相關研究差異比較.....15
表 3.1: 假設 10 筆二分類資料全為乾淨或噪音的分類結果.....17
表 3.2: 有 40%噪音的 23000 筆二分類資料經過 80%預測正確率模型 M 分類.....18
表 3.3: 假設 10 筆多分類資料全為乾淨或噪音的分類結果.....19
表 3.4: 有 40%噪音的 23000 筆多分類資料經過 80%預測正確率模型 N 分類.....20
表 3.5: 二分類二次挑選範例說明.....21
表 3.6: 多分類二次挑選範例說明.....24
表 4.1: 使用的資料集資訊.....27
表 4.2: 資料集中訓練、驗證與測試資料數量.....28
表 4.3: 五類別資料集在 40%噪音率標籤更改情形.....41
表 4.4: 我們方法與另外兩種方法的表示法與網路.....42
表 4.5: IMDb 在不同噪音資料下測試資料準確率.....49
表 4.6: Elec 在不同噪音資料下測試資料準確率.....49
表 4.7: AG's News 在不同噪音資料下測試資料準確率.....50
表 4.8: Sogou 在不同噪音資料下測試資料準確率.....50

第一章 緒論

1.1. 研究背景

現實中的資料集不會像人工產生的資料集那樣完美，所以實際用於訓練機器學習模型的資料不會全是乾淨 (clean) 的資料，其中不免包含有噪音 (noise) 的資料。這裡的乾淨資料就是指有正確標籤的資料，而噪音資料則是指有錯誤標籤的資料。標籤則是每筆資料的一個註記，例如電影評論就會有正評與負評，正評表示覺得這部電影很好，負評表示這部電影不好，這裡的正評與負評就稱作資料的標籤，噪音資料的意思我們也可以用電影評論來說明，例如在需要將評論分成正負評兩類的資料集中，某篇評論的內容應該是正評，但是它的標籤卻被標成了負評；在多分類的資料集中，本來是一篇關於運動的報導，但是該標籤被標成經濟，這樣的話這些資料就成為了噪音資料。噪音資料對訓練出來的模型分類結果影響很大，會造成模型分類結果的正確率變差。我們想要能從這些有噪音的資料集中訓練出一個不太會被噪音影響分類結果的模型。

1.2. 研究目的

我們的研究動機如同我們在前面所說，因為噪音資料對訓練出來的模型分類結果影響很大，會造成模型表現變差。大部分噪音資料對模型來說分類不會很困難，但因為這些資料的標籤是錯誤的，模型預測出來的結果與實際的標籤不一致時就會讓模型更難找出最好的權重。模型認為這些資料應該是歸類於某一類，但是結果卻告訴模型分類錯誤了，模型就會被這些噪音資料所誤導，導致分類結果變差。我們的研究目的是想要能從這些有噪音的資料集中訓練出一個不太會被噪音影響分類結果的模型。

處理噪音資料一直以來就是個很重要的問題，隨著網際網路的發展，資料取得更加容

易，相對的噪音資料的比例也會提高很多，許多學者提出方法希望能在這些有噪音的資料集下訓練出不受噪音資料影響的模型。以前的學者針對處理噪音資料的方法分成兩大類：資料前處理 (data preprocessing) [3][8][10][20] 和強健訓練 (robust training) [12][13][14][25][30][34]。資料前處理希望透過將訓練資料進行前處理之後減少訓練資料中的噪音資料的比例，讓模型盡可能的使用乾淨資料進行訓練。強健訓練則是注重模型的強健性 (robustness)，讓模型不會受到噪音資料的影響。強健訓練常見的有針對損失函數去進行調整的方法 [25]，還有修改模型來讓模型學習噪音的樣貌[12]，後來表現較好的強健訓練是樣本選擇 (instance selection) [14]，其概念是在訓練過程中持續挑選乾淨的資料，並使用這些乾淨的資料來訓練模型，最新的方法則是將噪音資料視為無標籤資料的半監督式學習 (Semi-supervised Learning) [30]。

1.3. 研究貢獻

本論文使用樣本選擇的概念。以往的樣本選擇方法都是參考損失的大小來判斷是否為噪音資料，但我們使用的是從預測結果是否一致來判斷是否為噪音資料。噪音資料在資料中的百分比稱為噪音率，我們發現保留預測標籤與資料標籤一致的資料可以有效地降低噪音資料的比例。在訓練過程中會先挑選預測標籤與資料標籤一致的資料當成下一次的訓練資料。但是為了確保訓練資料的數量以及多樣性，我們也會從預測標籤與資料標籤不一致的資料中挑選一些資料加入至下一次的訓練資料。在加入預測標籤與資料標籤不一致資料的過程中會逐漸減少加入預測標籤與資料標籤不一致資料的數量，以避免模型在訓練後期會被過多的噪音資料所影響。

為了檢驗我們重複挑選的方法是否真的能讓模型更強健，我們使用四個不同的資料集驗證我們的方法：IMDb、Elec、AG's News、Sogou。其中前兩個為二分類資料集，後兩個為多分類資料集。我們在實驗的訓練資料中加入了不同比例的噪音 (0%、10%、20%、30%、

40%)，以觀察結果並進行比較。比較的對象有 Co-teaching [14]及傳統類神經網路。實驗結果顯示我們的方法在實驗中表現較好，例如在 IMDb 資料集中 Co-teaching 方法在 40%噪音率的實驗中有 80.79%的準確率，而我們的方法在 40%噪音率的實驗中可以達到 83.16%的準確率，Elec 資料集中 Co-teaching 在 40%噪音率有 78.60%的準確率，我們的方法可以達到 81.76%的準確率，AG's News 資料集中 Co-teaching 在 40%噪音率有 88.58%的準確率，我們可以達到 89.21%，在最後一個 Sogou 資料集中 Co-teaching 在 40%噪音率有 93.27%的準確率，我們可以達到 93.37%的準確率。實驗結果也顯示，在沒有藉由外部資料且只使用本身包含噪音資料的資料集中，我們提出的方法可以訓練出比現有方法更加不被噪音資料影響的模型。

1.4. 論文架構

本論文的架構如下：第二章為相關研究，說明資料前處理和強健訓練的相關研究；第三章為研究方法，會介紹我們的方法概念和說明方法流程；第四章為實驗，會介紹我們實驗資料、實驗環境、實驗設計、實驗結果與實驗討論；第五章為結論與未來展望。

第二章 相關研究

針對噪音資料處理的方法可以分成兩大類，第一類是資料前處理，第二類是強健訓練。資料前處理是訓練模型之前使用一些方法去除資料中的噪音資料，使模型接收到的訓練資料中沒有噪音的資料。強健訓練則是針對模型或訓練過程去做調整，使模型就算使用有噪音的訓練資料也能訓練出好的結果。

2.1. 資料前處理

有些傳統的模型像是 SVM 並不適合用於有噪音資料的狀況，但可以透過資料前處理的方法移除噪音資料。例如在進行 SVM 之前使用權重調整的估計方法先行找出並移除噪音資料，以避免噪音資料進入模型影響結果[3][10]。這種方法在資料進入 SVM 訓練之前先建立一個核心矩陣 (kernel matrix)，接著輸入的資料會經過核心矩陣得到一個權重，若權重大於某個標準則判定為噪音資料，最後用小於標準的其它資料訓練 SVM [3]。

自動編碼器 (Autoencoder) [20]是一個對稱的網路架構，也是一種非監督式學習 (Unsupervised Learning) 的方法。自動編碼器可以分成前半層、中間層和後半層，前半層的部分是編碼層 (encoder)，中間層是隱藏層 (hidden layer)，後半層是解碼層 (decoder)，圖 2.1 為自動編碼器架構示意圖。自動編碼器的原理是將輸入資料透過編碼層降維至隱藏層，而隱藏層中的資料經過解碼之後要盡量和輸入資料相同。自動編碼器用於偵測噪音時會先使用多數的乾淨資料對自動編碼器進行訓練，以致乾淨資料經過解碼和編碼之後的差異較小，而噪音資料在解碼之後可能會和輸入時差異較大。透過這個特性，我們就可以使用自動編碼器偵測噪音資料[8]。判斷是否為噪音資料的方法是以歐幾里得距離來計算輸入值與輸出值之間的距離，計算出來的數值即為噪音分數，分數較高的資料就比較可能為噪音資料。圖 2.2 為自動編碼器偵測異常資料的示意圖，假設有兩筆資料一筆為乾淨另一筆為噪

音，這兩筆資料的三個數值為三維空間中的位置，將兩筆資料透過編碼與解碼之後計算其歐幾里得距離，可以發現資料二計算出來的數值較大，所以和資料一相比資料二為噪音資料的可能性比較高。

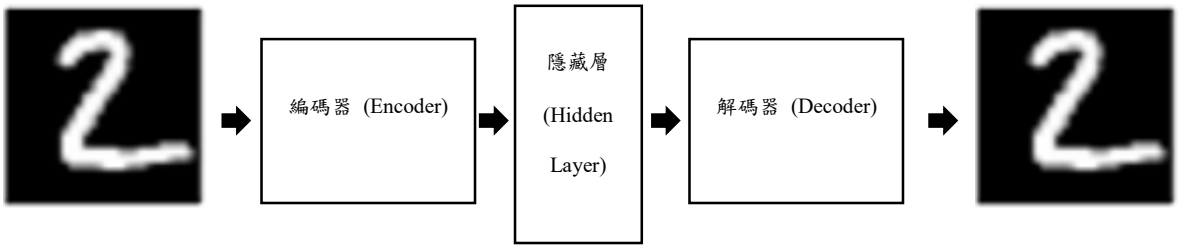


圖 2.1: 自動編碼器架構示意圖

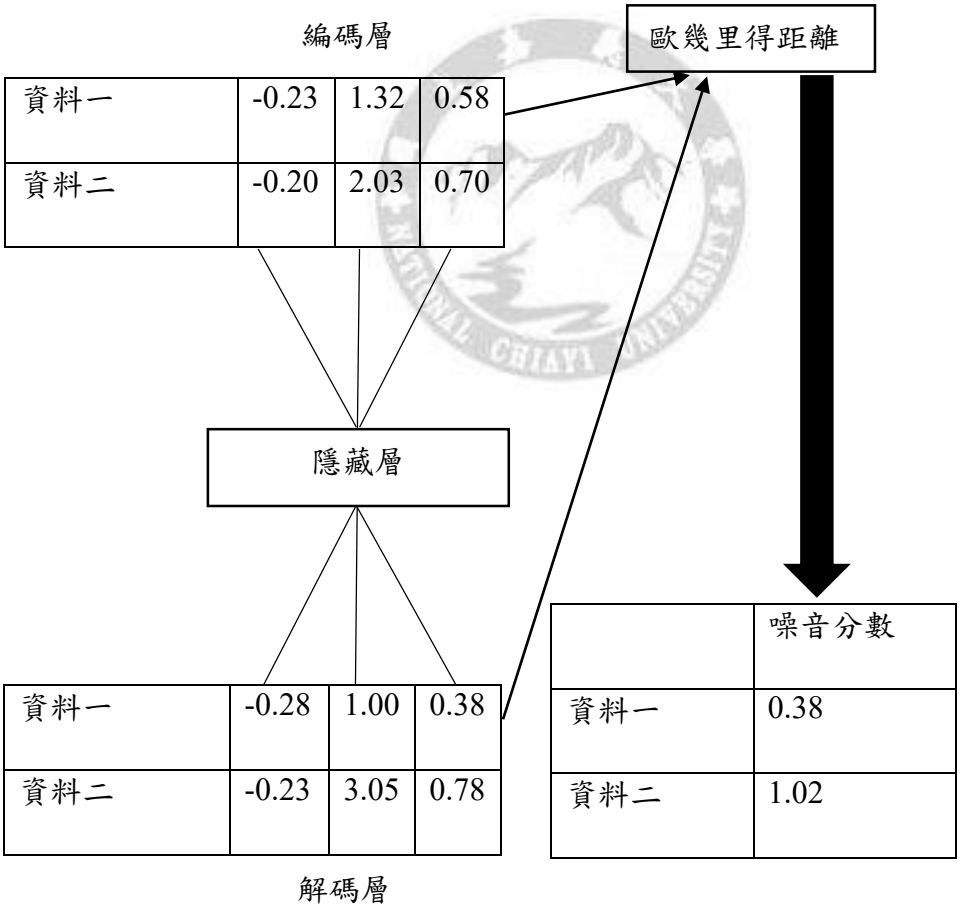


圖 2.2: 自動編碼器偵測噪音資料示意圖

資料前處理中傳統的方法都是在訓練之前先找出噪音資料，目的在於希望能盡可能地將資料中的噪音資料去除並保留下乾淨資料，但缺點在於無法精確的判斷有多少個噪音資

料。在現實中的資料集都是不會先知道噪音資料的數量，數量有可能多也有可能少，但部分乾淨資料在經過資料前處理之後都有可能被誤判為噪音資料，這些乾淨資料就不會被使用於訓練，噪音資料也有可能被誤判為乾淨資料，這樣的話訓練起來也會影響模型分類的結果。

自動編碼器的方法則是需要先使用乾淨的資料集訓練編碼器，接著編碼器就可以從混合噪音資料的資料集中找出噪音資料。這個方法的缺點是需要先有乾淨的資料來訓練編碼器，現實中的資料集較少有這種狀況，所以並不適用在現實中已經混和噪音資料的資料集。

2.2. 強健訓練 (robust training)

集成學習是較基本的強健訓練方法，其目的是希望藉由多個分類器或多次的分類結果來調整權重並讓模型自己挑出噪音資料。Bagging [5]、Boosting [4]、Stacking [6]都是集成學習中比較常見的方法。Bagging 是將多個分類器的分類結果進行投票藉此來找出噪音資料；Boosting 會進行多次的分類，每一次會將分類結果錯誤的資料權重提高，增加在下一次的分類被選中的機率；Stacking 和 Bagging 相似，但 Stacking 會把分類器的分類結果當成特徵並結合成新的分類器，此分類器的結果才是最後的結果。

代理人損失函數 (surrogate loss function) 是使用自定義的損失函數去替代本來的損失函數，目的是即使訓練資料中有噪音資料，藉由自定義的損失函數在計算損失的時候能將對乾淨資料的影響降低，讓模型更強健。模型訓練的過程中會藉由計算出來的損失來調整模型的參數，希望調整之後的參數能讓模型計算出來的損失越來越小。但是噪音資料因為標籤本身就是錯誤的，若模型使用乾淨資料訓練出來的參數來計算噪音資料的損失會變大，模型反而會為了學習這些噪音資料的特徵，導致了乾淨資料計算出來的損失更大，這樣的話模型會因為沒辦法同時降低乾淨資料與噪音資料的損失導致總損失無法下降。這個方法的另外一個缺點是計算成本會比原來高很多，所以不適用於多分類問題，只適用於二分類

問題[25][26]。後來有學者提出了能用於多分類的方法[11][38]，但這些方法只能用於類別有限的多類別問題，所以在類別很多的資料集中就不適用，而且自定義的損失函數會導致模型需要較長的時間才可以收斂[38]。

有另外一類調整損失的方法是要減少噪音資料的影響，目的是在更新網路的參數之前調整全部資料的損失來降低噪音資料的不良影響。有一種調整損失的方法是修改損失，這種方法不會直接使用模型的輸出來計算損失，會將模型的輸出乘以標籤轉換機率 (label transition probability)得到新的輸出之後再使用此輸出來計算損失，藉此來降低噪音資料的影響[27]。另一種是對損失進行權重上的調整，這種方法的目的是將資料中的噪音資料給予較低的權重，乾淨資料則給予較高的權重[7][32]。最後一種則是直接對資料標籤進行翻新，這種方法會將資料的標籤與模型輸出的標籤進行標籤可信度上的調整之後再使用翻新過的標籤來計算損失。標籤可信度一般使用模型的分類準確率來當作標準 [1][29]，公式 2-1 為標籤翻新的公式，其中標籤可信度為$\alpha \in [0,1]$，$\tilde{y}$為資料的標籤，$\hat{y}$為模型預測的標籤，翻新的標籤$\bar{y}$會由此公式進行調整。常見的有 Bootstrapping [29]，Dynamic-Bootstrapping [1]。Bootstrapping 的概念為先使用一個小型且乾淨的資料集訓練出一個分類器，並使用此分類器來對另一個大型但混合噪音資料的資料集進行訓練，並使用標籤可信度來將大型資料集的標籤與模型預測出來的標籤進行標籤調整，Dynamic-Bootstrapping 則是將標籤可信度變成可以動態地進行調整，傳統的 Bootstrapping 的標籤可信度則為固定的。標籤翻新的方法有一個缺點是一開始需要準確率較高的模型才可以得到較好的標籤可信度，所以才需要先以乾淨的小型資料集進行訓練，並不能一開始就使用有大量噪音資料的資料集進行訓練，現實中不一定每一種狀況都能有小型的乾淨資料集，所以能應用的範圍可能有限。

$$\bar{y} = \alpha\tilde{y} + (1-\alpha)\hat{y} \tag{2-1}$$

為了改善以上的問題，有些學者針對模型架構去修改，目的是不需更動損失函數或調整損失，只調整模型的架構，這樣的話計算成本會相對地下降，模型也會比較容易收斂。這類架構上的修改是要針對噪音資料集建立一個標籤轉換矩陣 (label transition matrix)，其中包含在模型中加入一層噪音適應層 (noise adaption layer) [12][16][18]或是直接在模型架構進行調整[13][17][34][36]。噪音適應層的目的是要去模仿噪音資料的樣貌。圖 2.3 為噪音適應層架構的示意圖，我們一開始有資料 x 與標籤 y，我們將標籤 y 更改成噪音標籤$\tilde{y}$，資料 x 經過模型之後會得到輸出 M(x)，M(x)為模型的輸出並不是預測出來的標籤，接著將 M(x)經過噪音適應層的轉換後得到$\widehat{M(x)}$，此時將$\widehat{M(x)}$與$\tilde{y}$計算損失並用此損失來更新網路。

圖 2.3: 噪音適應層架構示意圖

噪音適應層的缺點是對噪音類型的假設太過強烈，當模型遇上過於複雜的噪音資料將沒辦法很好的模仿噪音的樣貌[34]。為了解決這個問題有些學者提出了一些方法，這些方法目的是使用標籤轉換機率 (label transition probability) 找出乾淨資料與噪音資料之間的關係，但這些方法主要是要提高標籤轉換機率的可靠性，以處理更加複雜的現實噪音資料[13][17][34][36]。這裡的標籤轉換機率與上面代理人損失函數中提到的轉換機率是一樣的，但此處的差別是這裡的標籤轉換機率是在模型中一起學習出來的，代理人損失函數中的標籤轉換機率是在外部計算出來之後再與模型的輸出進行計算，並不是在模型中學習出來的。這些方法雖然可以處理更加複雜的噪音資料，但是這些方法是將標籤轉換機率放到網路中讓網路變成特定的架構，這樣的話並不能將這些方法套用至其他不同的網路架構上。

因為噪音資料會對模型的分類結果造成巨大的影響，所以有些學者想要在訓練過程中去進行樣本選擇，希望能挑選到乾淨資料，並讓模型只用乾淨資料去進行訓練[14][15][23][37]。這類方法在模型訓練的過程中，藉由減少噪音資料的數量來減少噪音資料的影響，其優點是可以應用在不同的資料集，也不會有太高的計算成本。一般進行挑選會挑選損失較小的資料將其視為乾淨資料，有些學者發現只挑選損失較小的資料可以很好的區別乾淨資料與噪音資料，並稱這種現象為 memorization effect [2]

Decoupling [23]是同時訓練兩個網路，它們的挑選方法是兩個網路如果對某筆資料預測的結果不同就留下這筆資料，若是相同則不留下這筆資料，這裡並沒有用到 memorization effect 的概念，只使用他們自己的挑選規則，但是這樣做並不能保證能很好的挑選出噪音資料，圖 2.4 為架構示意圖。

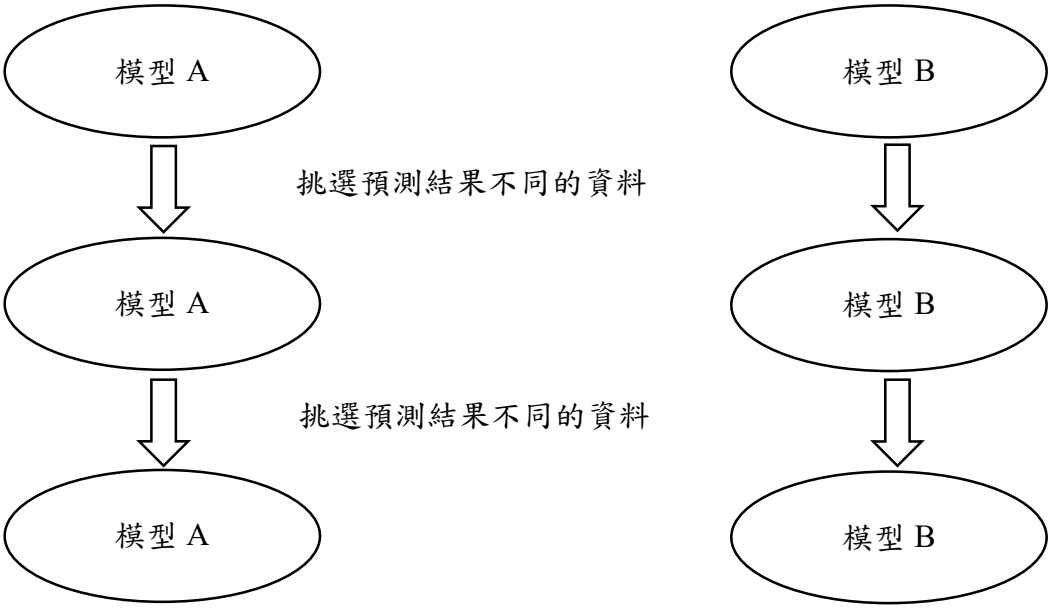


圖 2.4: Decoupling 示意圖

MentorNet [15]可以視為有兩個網路的網路架構，兩個網路分別為老師網路 (teacher net) 與學生網路 (student net)，老師網路是一個經過預訓練的網路。MentorNet 的目的是希望藉由老師網路來挑選資料給學生網路進行訓練，訓練完的學生網路才是最後要使用的模型，這裡也是挑選損失較小的資料。但這裡有一個缺點是老師網路訓練時要使用乾淨的資料集進行訓練，否則就需要有一套預先定義的標準來訓練老師網路。

Co-teaching [14]和 Co-teaching+ [37]這兩種方法都是同時訓練兩個網路，圖 2.5 為 Co-teaching 方法的示意圖，在每一個 Epoch 開始時兩個網路都使用一樣的訓練資料 D 進行訓練。這個方法也有使用 memorization effect 的概念，所以在訓練之前會分別使用模型 A 與模型 B 計算出損失，接著模型 A 根據損失大小只留下損失較小的資料挑選出 DA，同理模型 B 挑選出 DB，最後模型 A 會使用模型 B 挑選出的 DB 進行訓練，模型 B 會使用模型 A 挑選出的 DA 進行訓練，到此 Epcoh 結束。

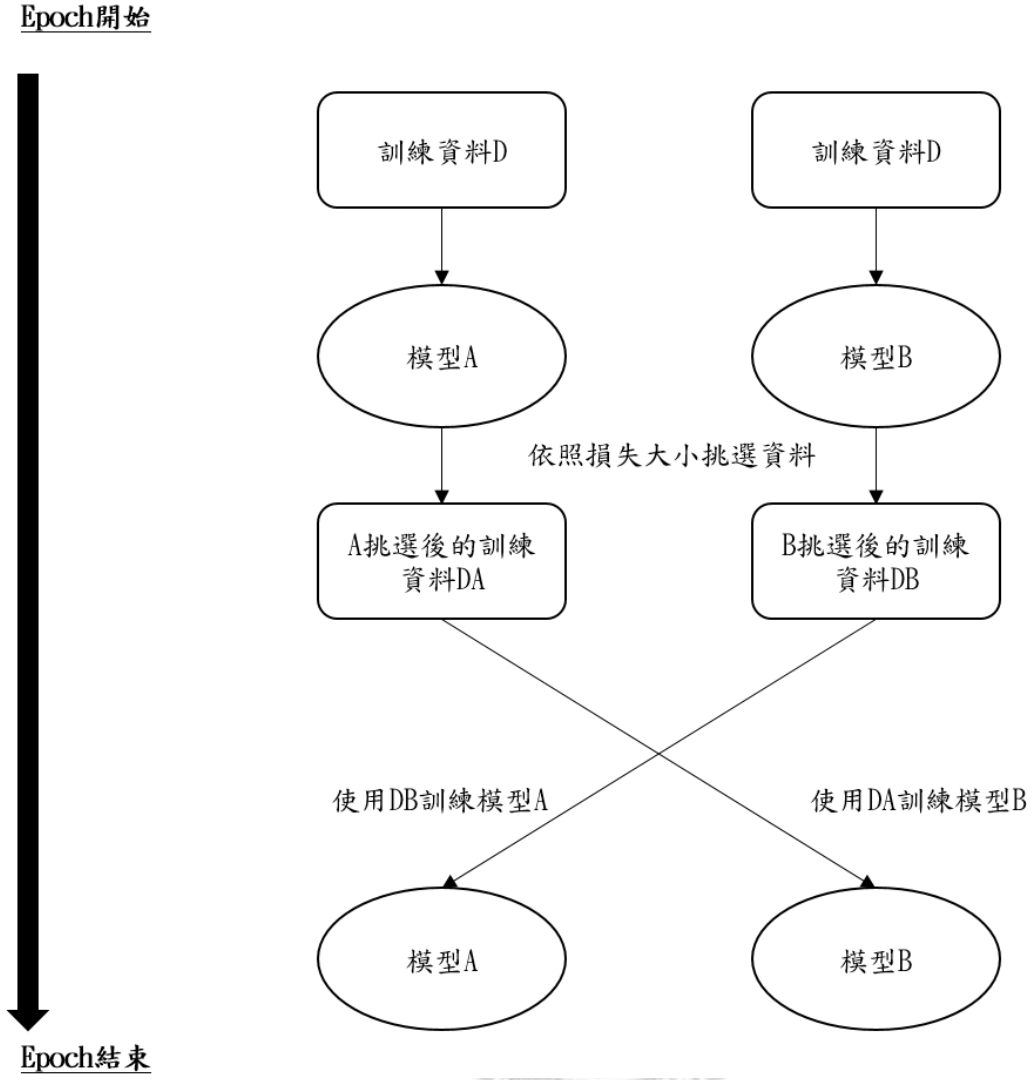


圖 2.5: Co-teaching 方法示意圖

Co-teaching 的核心概念是根據[2]提出的 memorization effect 的想法來挑選損失比較小的資料。Co-teaching 的挑選資料與資料前處理的挑選資料差異是資料前處理在挑選之後就將不需要的資料完全移除，在整個訓練中那些資料都不會進入訓練，但是 Co-teaching 是在每一個 Epoch 開始時才對資料進行挑選，每一次的挑選都是使用全部資料來進行挑選，這樣的好處是模型在前幾個 epoch 挑選能力較弱時有可能會誤判乾淨資料，導致乾淨資料被移除，但因為每次都是使用全部資料進行挑選，所以到了後面的 epoch 挑選能力已經提升時，就會減少誤判的機率了。Co-teaching 在挑選時會隨著 epoch 的增加挑選越來越少的資料，他們希望模型在開始時能使用較多的資料先建立一個簡單的挑選標準，但隨著 epoch 數的增加模型的挑選能力提升之後為了避免過多的噪音資料會影響模型的挑選能力，後期

會留下較少的資料來確保留下的資料為乾淨資料的可能性較高。

Co-teaching 是使用兩個網路互相幫對方挑選資料，這樣子挑選的優點是只用一個網路可能會有盲點，盲點的意思代表網路對某些資料的判斷就是錯的，從一開始到最後都是錯的，像是有些乾淨資料一直被誤判且損失被計算成很大，這樣的話這些乾淨資料會一直被去除不會被挑選到，這樣子就會造成挑選能力的下降。如果使用兩個網路的話就有可能可以避免盲點的問題，例如有兩位程度一樣的學生 A 和 B 在讀書，A 讀出來的心得和學習到的內容會和 B 不完全相同，兩人一定有些地方有不同的看法，A 可能有些地方不懂，但剛好 B 那些地方懂，這樣兩人一起學習就可以把能互補的地方補起來。使用兩個網路的話因為是使用對方幫忙挑出來的資料，這樣就可以避免單一網路學習可能會出現的盲點。

使用兩個網路進行長時間的訓練的話兩個網路可能都會收斂到太過相似，這樣的話就會失去使用兩個網路互相幫對方挑選的意義。為了避免這個問題 Co-teaching+ 被提出來，Co-teaching+ 和 Co-teaching 的差異只差在挑選資料的部分。圖 2.6 為 Co-teaching+的方法示意圖，Co-teaching 中模型 A 挑出的 DA 會直接給模型 B 訓練，模型 B 也會挑出 DB 給模型 A 訓練，但在 Co-teaching+中模型 A 挑出來的 DA 會對模型 B 挑出來的 DB 進行比較，DA 會去除 DB 同時也選到的那些資料，模型 B 挑出來的 DB 也會對 DA 進行比較，DB 去除 DA 同時也有選到的那些資料，因為持續挑選不同的資料讓另一個網路進行訓練，這樣的步驟可以確保兩個網路經過長時間的訓練不會收斂到太過於相似。

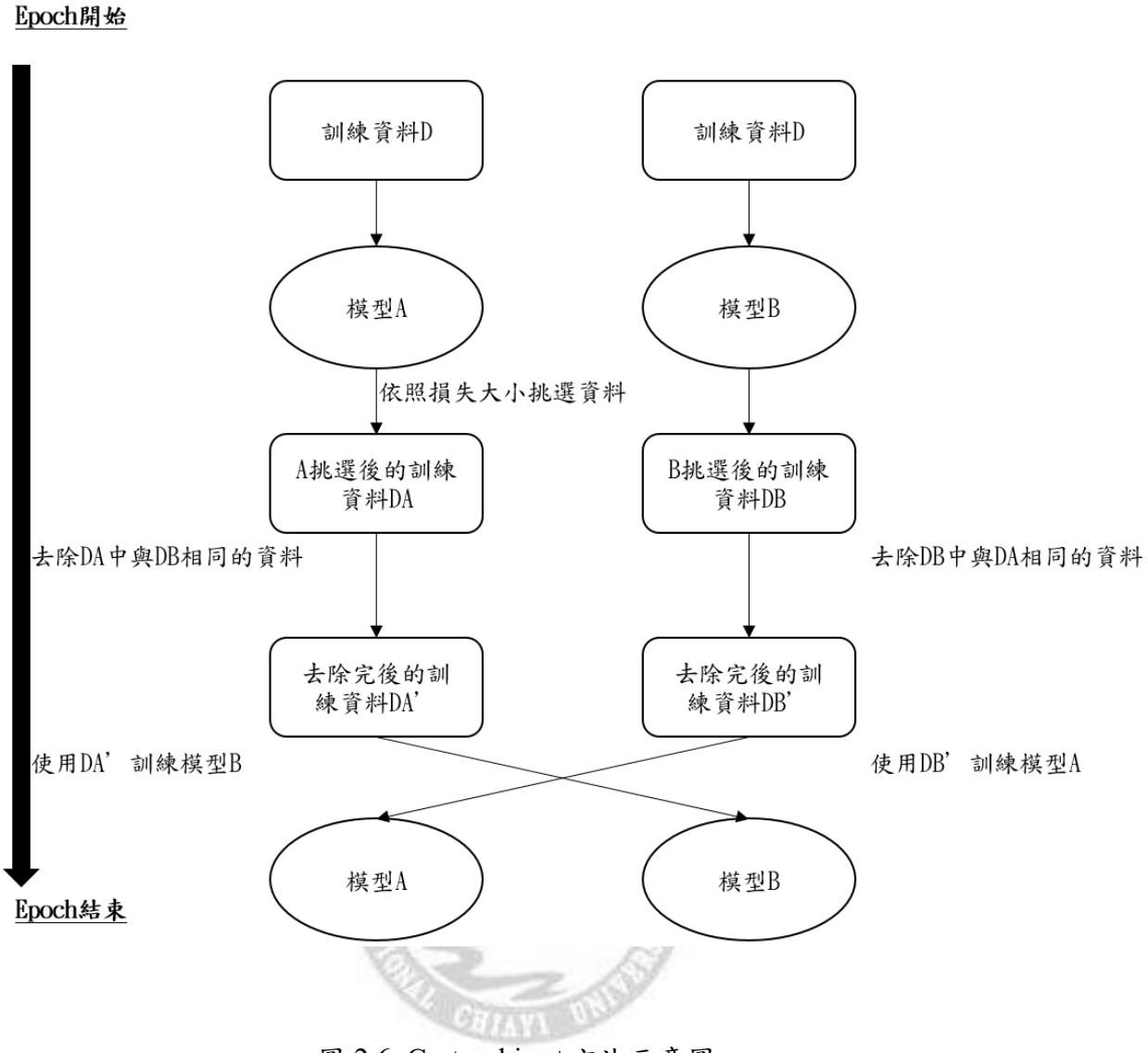


圖 2.6: Co-teaching+方法示意圖

表 2.1: 我們方法與其他兩種方法挑選差異

方法	挑選差異
樣本選擇	每個 epoch 開始時去除資料
資料前處理	訓練開始前就去除資料
我們的方法	每個階段結束時去除資料

表 2.1 為我們的方法與另外兩種方法的挑選差異，樣本選擇是在每一個 epoch 開始時才去除資料，資料前處理是在訓練開始前就去除資料，我們的方法則是在每一個階段結束時才使用兩次挑選來去除資料。

近年來有些學者改使用半監督式學習 (Semi-supervised learning)來降低標注的成本，這方法適用於乾淨與噪音混和的資料集，因為人工標注是非常耗成本的，所以就把資料集中可能為噪音的資料視為沒有標籤的資料，剩下的資料將視為乾淨資料，這樣就可以把問題定義為半監督式學習的問題[9] [30][35]。

標籤聚合 (label aggregation) [35]先使用一個小型乾淨的資料集來訓練出多個網路來當作標注者，這些標注者會對混合噪音的資料集中的資料進行標籤的預測，並將標注者們預測出來的標籤做一個標籤的聚合，聚合的方法可為集成 (ensemble)，接著將這些聚合後的標籤當成新的資料標籤。

Two-stage framework [9]分成兩個階段，第一個階段會使用有噪音的資料集訓練一個模型，並使用此模型對每一個訓練資料給予一個預測機率，並會訂出一個門檻，當預測機率高於此門檻時則重新定義為有標籤的資料，小於此門檻時也重新定義為沒有標籤的資料。第二階段時則使用這些被重新定義為有標籤與重新定義為沒標籤的資料混和然後訓練。

SELF [30]使用了自主集成 (self-ensemble)的想法，他們主要的想法也是要去挑選出乾淨的資料，並將被移除的噪音資料視為沒有標籤的資料。自主集成是一種時間軸上的集成方法，這個方法會將模型當前 epoch 之前每一個 epoch 的預測標籤記錄起來進行集成。模型在每一個 epoch 結束時會對所有資料進行標籤預測並記錄，接著對標籤進行集成，只要集成出來的標籤與資料標籤不同就將其視為噪音資料，相同視為乾淨資料，在下一個 epoch 訓練時就使用這些挑選完的乾淨資料進行訓練，因為挑選資料每一次都是使用所有資料進行挑選，當前 epoch 未被選中的資料在之後的 epoch 也是有可能會被選上。在半監督式學習時挑選出來的噪音資料可以視為沒有標籤的資料，接著使用非監督式學習的損失計算方式對這些資料計算損失就可利用這些噪音資料。

表 2.2 為相關研究的差異比較，我們比較的方向有三個，分別是有無使用外部資料或額外的乾淨資料;是否適用於大量噪音的情形以及訓練時資料總數是否更改。資料前處理、

代理人損失函數與半監督式學習有些方法有使用外部資料，類神經網路偵測噪音資料與集成學習都有使用外部資料，修改模型結構與我們的方法都沒有使用外部資料。在是否適用於大量的噪音情形下類神經網路偵測噪音資料、樣本選擇、半監督式學習與我們的方法都是適用於大量的噪音。在訓練時資料總數方面，資料前處理與類神經網路偵測噪音資料的總數都會明顯減少，我們的方法會隨著階段數的進行逐漸減少，其他五種方法都是不變的。

表 2.2: 相關研究差異比較

方法	是否需要外部資料或額外的乾淨資料集	是否適用於有大量噪音資料的訓練資料	進行訓練時的訓練資料總數
資料前處理	有些有	否	減少
類神經網路偵測噪音資料	是	是	減少
集成學習	是	否	不變
代理人損失函數	有些有	否	不變
修改模型結構	否	否	不變
樣本選擇	否	是	不變
半監督式學習	有些有	是	不變
我們的方法	否	是	逐漸減少

第三章 研究方法

本章一開始將說明我們方法的概念，接著說明方法的流程。

3.1. 方法概念

在分類問題中噪音資料對分類的準確率影響非常大，會讓準確率下降很多，解決這類問題的方法有先對訓練資料做前處理，或是增強模型對噪音資料的對抗能力。本論文使用的是強健訓練中樣本選擇的方法，我們會將實驗過程分多個階段進行，每一個階段會對資料進行兩次挑選以組成下一階段的訓練資料，第一次的挑選是希望能將噪音資料從訓練資料中移除，第二次的挑選則是希望加回一些被移除的資料來增加訓練資料的多樣性。我們的方法想要藉由兩次的挑選來讓模型更加的強健。

3.2. 方法流程

這裡先說明第一次挑選的方法，假設我們有一個 80%預測正確率的模型 M。這裡的正確率是指針對乾淨資料的預測正確比率，也就是如果有 100 筆具有正確標籤的乾淨資料，模型 M 的預測結果與正確標籤一致的有 80 筆，預測的結果與正確標籤不一致的有 20 筆。因為噪音資料的錯誤標籤與資料的真實標籤相反，且模型 M 又有 80%的預測正確率，所以用模型 M 預測 100 筆的噪音資料，預測結果與錯誤標籤一致的有 20 筆，代表模型 M 正確分類了這 20 筆資料，而預測結果與錯誤標籤不一致的有 80 筆，代表模型 M 錯誤分類了這 80 筆資料。我們可以從表 3.1 的例子中更了解我們的方法，假設有 10 筆乾淨的二分類資料要分成正負評兩類，模型也對這 10 筆資料進行預測，標籤 1 為正評，0 為負評，如表格中左半邊所示，我們可以看到模型分類結果有 8 筆標籤一致，2 筆標籤不一致。接著如果

將這 10 筆資料標籤全部變成噪音的話，如表中右半邊所示，分類結果會變成 8 筆標籤不一致，2 筆標籤一致，從這裡可以看出來若資料全部都是噪音資料的話分類結果會與全部都是乾淨資料的結果完全相反，所以模型 M 在乾淨資料中的 80%預測正確率到了噪音資料就會變成 20%的預測正確率，我們將這想法延伸至較大的資料集進行更進一步的說明。

表 3.1: 假設 10 筆二分類資料全為乾淨或噪音的分類結果

正確類別	標籤 (乾淨)	預測標籤	是否一致		正確類別	標籤 (噪音)	預測標籤	是否一致
1	1	1	是		1	0	1	否
1	1	1	是		1	0	1	否
0	0	0	是		0	1	0	否
0	0	0	是		0	1	0	否
1	1	1	是		1	0	1	否
0	0	0	是		0	1	0	否
1	1	1	是		1	0	1	否
0	0	0	是		0	1	0	否
1	1	0	否		1	0	0	是
0	0	1	否		0	1	1	是

我們使用表 3.2 來說明剛剛方法的延伸。假設有 23000 筆要分成正負評兩類的資料，其中包含 9200 筆噪音資料，噪音率為 9200 / 23000 = 40.0%，且從表 3.2 可以看出，所有預測標籤與資料標籤一致的資料有 11040 + 1840 = 12880 筆，其中噪音率為 1840 / 12880 = 14.3%，而預測標籤與資料標籤不一致的資料有 1960 + 7360 = 9320 筆，其中噪音率為 7360

/ 9320 = 79.0%。由以上範例可以看出，用有較高預測正確率的模型挑選預測標籤與資料標籤一致的資料，可以大幅降低資料集的噪音率。

表 3.2: 有 40%噪音的 23000 筆二分類資料經過 80%預測正確率模型 M 分類

	乾淨的資料 (13800 筆)	噪音的資料 (9200 筆)
預測標籤一致的比例： 不一致的比例	80%：20%	20%：80%
預測標籤一致的數量： 不一致的數量	11040：1960	1840：7360

接著我們將剛剛表 3.1 的 10 筆資料改成多類別的情況如表 3.3 所示。假如今天有 10 筆要分成三類的資料集，表格左半邊一樣全部都是乾淨資料，右半邊全部都是噪音資料，多類別與二類別的差異就是乾淨資料變成噪音資料時的標籤會有多種可能，這裡因為是三類所以標籤就有兩種可能，這裡我們假設有一個模型 N 對乾淨資料有 80%預測正確率，因為這裡是多類別的情況，所以到了全部都是噪音資料時的預測正確率就不會是剛好 20%，我們可以觀察右半邊最後兩筆資料，因為多分類時更動標籤不像二分類時是直接更改成另一類，而是有可能變成其他的，所以最後兩筆資料就會有兩種可能，剛好改成對的或剛好改成錯的，我們只考慮兩個剛好都改成與預測標籤一致的情況與兩個剛好都改成與預測標籤不一致的情況的話在噪音資料中的預測正確率就會分別是 20%或 0%，其他情況都會介於 0%~20%之中。

表 3.3: 假設 10 筆多分類資料全為乾淨或噪音的分類結果

正確類別	標籤(乾淨)	預測標籤	是否一致		正確類別	標籤(噪音)	預測標籤	是否一致
1	1	1	是		1	0 or 2	1	否
1	1	1	是		1	0 or 2	1	否
0	0	0	是		0	1 or 2	0	否
2	2	2	是		2	0 or 1	2	否
1	1	1	是		1	0 or 2	1	否
2	2	2	是		2	0 or 1	2	否
1	1	1	是		1	0 or 2	1	否
0	0	0	是		0	1 or 2	0	否
1	1	2	否		1	0 or 2	2	否或是
0	0	1	否		0	1 or 2	1	否或是

我們使用 23000 筆有 40%噪音比例的多分類資料來驗證我們的想法，我們將表 3.3 的結論延伸至表 3.4 來觀察多分類時挑選標籤一致的資料是否也可以有效地降低噪音率，在乾淨資料的情況下會和二分類時一樣，有 11040 筆標籤一致的資料與 1960 筆標籤不一致的資料，但在噪音資料時我們比較兩種情況，剛好一致與剛好不一致，第一種是有 1840 筆標籤一致的資料與 7360 標籤不一致的資料，這時我們將所有標籤一致的資料相加 11040+1840 = 12880，其中噪音率為 1840/12880 = 14.3%，不一致的資料相加 1960+7360 =

9320，其中噪音率為 7360/9320＝79.0%。如果是第二種時會有 0 筆標籤一致的資料與 9200 筆標籤不一致的資料，這時所有標籤一致的資料相加 11040+0 = 11040，其中噪音率為 0/11040＝0%，不一致的資料相加 1960+9200＝11160，其中噪音率為 9200/11160＝82.4%。我們可以發現就算是多類別的情況時兩種極端情況我們留下與預測標籤一致的資料也可以有效地降低噪音比例。

表 3.4: 有 40%噪音的 23000 筆多分類資料經過 80%預測正確率模型 N 分類

	乾淨的資料 (13800 筆)	噪音的資料 (9200 筆)
預測標籤一致的比例： 不一致的比例	80%：20%	20%：80% (一致) 0% : 100% (不一致)
預測標籤一致的數量： 不一致的數量	11040：1960	1840：7360 (一致) 0 : 9200 (不一致)

在第二次的挑選時我們會從分類錯誤 (預測標籤與資料標籤不一致) 的資料中挑選部分的資料加入至新的訓練資料。這裡我們先討論二分類的資料集，因為二分類與多分類的模型輸出不一樣，所以我們需要分開討論。二分類的模型對每筆資料的分類結果會是一個 [0,1] 之間的分數，0.5 為判斷的標準，小於 0.5 為負評、大於等於 0.5 為正評。模型分類的分數越靠近 0 或 1 表示模型對很確定這筆資料的分類結果，也是較容易分類的資料，若分類分數離 0 或 1 越遠則表示模型不太確定這筆資料的分類結果，我們在實驗的過程中發現標籤不一致的資料中乾淨資料的分類分數大部分會分布在 0.5 附近，我們推測可能是因為較容易分類的乾淨資料已經都在標籤一致的資料中，在標籤不一致中的乾淨資料可能是較不容易分類的資料，所以模型對這些資料的類別並不是很確定，所以就誤判導致標籤不一致，但是因為不確定這些資料的類別，所以預測分數就不會離判斷標準很遠。為了增加訓練資料的數量及多樣性，且同時要避免加入太多的噪音資料，我們挑選分類分數在 0.5 附近的標籤不一致資料，加入至新的訓練資料。我們一開始挑選的範圍比較寬，隨著階段的

進行會逐漸縮小範圍，這個步驟可以讓模型在開始的階段學習較多的資料，到了後面的階段會逐漸專注在乾淨的資料上。圖 3.1 是我們方法的演算法步驟，而圖 3.2 為演算法的流程圖。

表 3.5 為二分類第二次挑選的挑選範例說明，這裡假設有四筆不一致資料，兩筆噪音兩筆乾淨。乾淨資料會像我們上一段所說明的情況一樣分部在 0.5 的附近，所以我們想要將這些資料取回來。

表 3.5: 二分類二次挑選範例說明

資料	正確類別	資料標籤	預測結果	預測標籤
A (乾淨)	0	0	0.59	1
B (噪音)	1	0	0.91	1
C (噪音)	0	1	0.13	0
D (乾淨)	1	1	0.44	0

迴圈開始 k = 1,2,…,9,10, k 為階段數

1. 使用訓練資料D_k訓練出模型M_k
2. 使用模型M_k分類訓練資料D_k並產生分類標籤一致資料D_c與分類不一致資料D_w
3. 將D_c加入至下一個階段的訓練資料D_{k+1}並以逐步減少方式從D_w挑選判斷邊界附近的資料加入至D_{k+1}

迴圈結束

圖 3.1: 二分類演算法步驟

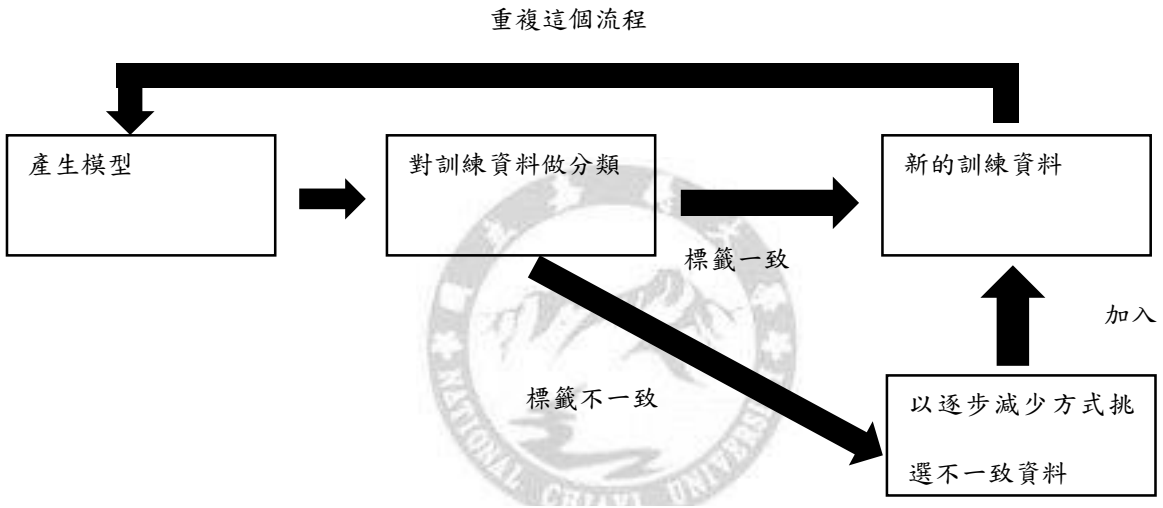


圖 3.2: 二分類演算法流程圖

多分類的模型輸出會由多個 0~1 之間的數值組成，且所有數值相加總合為 1，其中數值最大的那個位置即為預測出來的標籤。例如今天有一筆從四個類別的資料集中取出的資料，假設這筆資料的資料標籤是 2，代表這筆資料是屬於第二類的資料，假設模型對這筆資料的分類結果為 (0.01, 0.97, 0.01,0.01)，我們會對所有數值找出最大的那個數值代表預測出來的標籤，所以這筆資料的標籤為 2，表示被模型分到第二類。這樣和二分類的輸出並不一樣，因此多分類問題並不能直接使用二分類問題的挑選方法。這裡我們挑選的想法和二分類相似，我們希望挑選不一致資料中只是因為比較難被分類而被模型分類錯誤導致標籤不一致的這些資料，多分類的分類結果中最大值如果越大表示模型越相信這筆資料是屬於該類別，越小表示模型較不確定這筆資料是否屬於該類別，這裡分類結果就類似於機率，

最大值越大屬於該類別的機率越大，反之機率越小。有些乾淨資料比較難被分類而被分類錯誤導致標籤不一致的資料與其他標籤不一致資料中的噪音資料有一些不同，大部分標籤不一致的噪音資料的分類數值的最大值都很大，這表示這些資料對模型來說是較容易分類的，但因為他們是噪音資料，所以變成了標籤不一致，這些資料會讓模型表現的更差，所以我們捨棄這些標籤不一致且最大值都很大的資料改留下最大值比較小的那些資料，但因為只挑選最大值較小的資料也是有可能挑選到噪音資料，所以這裡我們還加上了一個規則，最大值減去資料標籤對應的數值要小於某一個門檻值，這樣我們可以確保模型對這筆資料的預測數值不會太低，但只是最大值並不是在標籤數值而已，我們希望隨著模型分類能力的提升能將這些資料分類正確讓預測標籤與資料標籤一致。接著我們觀察多分類問題中標籤不一致資料的分類結果，如上述所說其中多數標籤不一致資料中乾淨資料的分類結果的最大值不會很大，但是會與資料標籤不同。例如有一筆預測標籤與資料標籤不一致的乾淨資料的資料標籤是第一類，預測標籤是第二類，這樣的話第二類標籤的數值就會是最大值，模型對這筆資料的預測就會像是 (0.14, 0.63, 0.03, 0.20)。但這些資料可能是乾淨資料，我們希望挑選這些資料並加入至訓練資料集。我們會記錄數值中最大的數值與數值中資料標籤對應的數值，並將最大值減去標籤對應到的數值。我們會訂定一個門檻值，當這最大值與標籤對應值的差小於等於門檻值時即為要挑選的資料。用 (0.14, 0.63, 0.03, 0.20)來看就是 0.63-0.14 = 0.49，若是門檻值為 0.4，那這筆資料就會被挑選，若門檻值為 0.5，這筆資料就不會被挑選。圖 3.3 為多分類演算法的步驟，圖 3.4 為多分類演算法的流程圖。

表 3.6 為多分類的二次挑選範例說明，這裡我們假設有四筆分四類的不一致資料，兩筆乾淨，兩筆噪音。乾淨資料會像二分類一樣，因為模型比較不確定這些資料屬於哪一類，所以最大值會相對噪音資料比較小一點，我們可以看到每筆資料中預測最大值是粗體的數字，資料標籤對應的數值是有底線的數值，從資料 A 來看就是 0.34-0.32=0.02，資料 B 是 0.95-0.02=0.93，資料 C 是 0.74-0.11=0.63，資料 D 是 0.56-0.20=0.36。

表 3.6: 多分類二次挑選範例說明

資料	正確類別	資料標籤	預測結果	預測標籤	最大值減去資料標籤對應的數值
A (乾淨)	1	1	(<u>0.32</u>, 0.11, 0.23, **0.34**)	4	0.34-0.32= 0.02
B (噪音)	1	2	(**0.95**, <u>0.02</u>, 0.01, 0.02)	1	0.95-0.02= 0.93
C (噪音)	2	4	(0.13, **0.74**, 0.02, <u>0.11</u>)	2	0.74-0.11= 0.63
D (乾淨)	2	2	(0.05, <u>0.20</u>, 0.19, **0.56**)	4	0.56-0.20= 0.36

迴圈開始 k = 1,2,…,9,10, k 為階段數

1. 使用訓練資料D_k訓練出模型M_k
2. 使用模型M_k分類訓練資料D_k並產生分類標籤一致資料D_c與分類標籤不一致資料D_w
3. 將D_c加入至下一個階段的訓練資料D_{k+1}並從D_w挑選最大值與標籤對應值的差小於等於門檻值的資料放入D_{k+1}中

迴圈結束

圖 3.3: 多分類演算法步驟

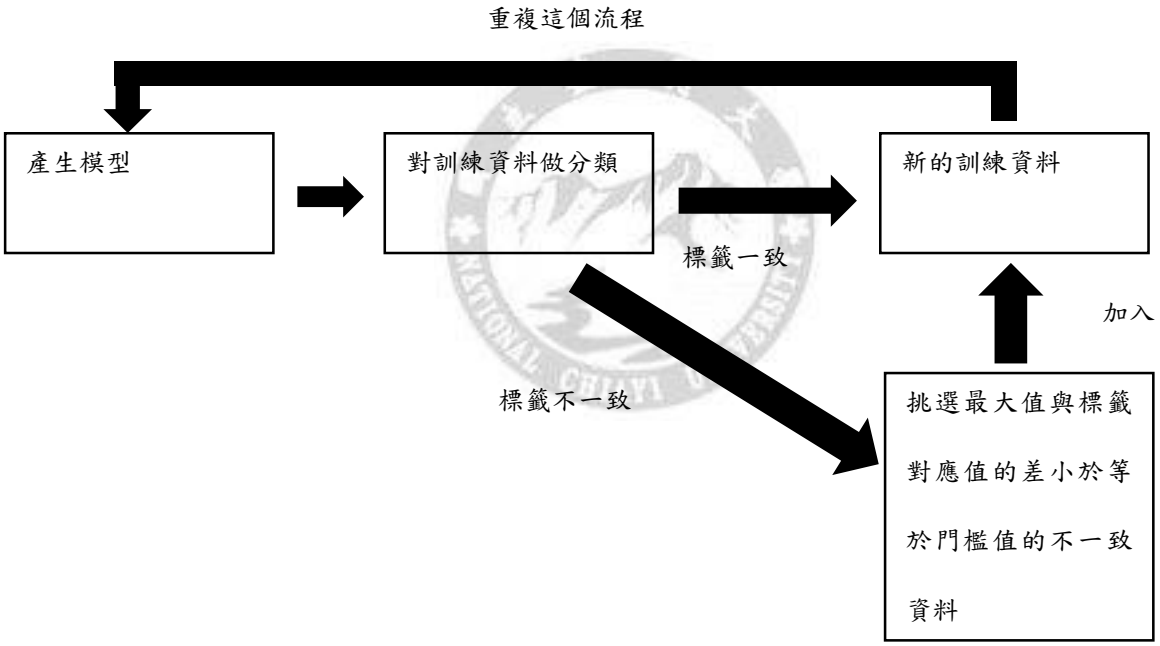


圖 3.4: 多分類演算法流程圖

第四章 實驗

我們會在這章介紹使用到的資料集、評量公式、實驗環境、實驗設計、實驗結果與實驗討論。

4.1. 資料集

我們實驗使用了四個資料集，分別為 IMDb [22]、Elec [19]、AG's News [39]與 Sogou [33]。所有資料集的資訊與細節如表 4.1 所示，IMDb 與 Elec 為二分類的資料集，AG's News 與 Sogou 為多分類的資料集。IMDb 資料集為 Maas 等人從網路電影資料庫 (IMDb Internet Movie Database) 之中隨機挑選的評論，其中每部電影不超過 30 則評論。IMDb 的評分方式為 10 星制，這裡 7 星以上的評論定義為正評，低於 4 星的評論定義為負評，5~6 星的中立評論不納入考慮。挑選出的評論有 50000 筆，其中有 25000 筆訓練資料與 25000 筆測試資料，正評與負評的數量各占一半，訓練資料有 12500 筆正評、12500 筆負評，測試資料也是 12500 筆正評、12500 筆負評，最長的評論有 2122 個字，最短的評論有 6 個字，平均的評論長度為 230.95 個字，標準差為 170.88。Elec 資料集是從 Amazon 電子產品的評論中取出來的，其訓練資料與測試資料的數量是模仿 IMDb 的結構去生成出來的，評分方式為 5 星制，這裡 4~5 星為正評，1~2 星為負評，3 星不考慮。挑選出來的評論有 50000 筆，其中也是 25000 筆訓練資料與 25000 筆測試資料，正負評數量也是各占一半，訓練資料有 12500 筆正評、12500 筆負評，測試資料也是 12500 筆正評、12500 筆負評，最長的評論有 5308 個字，最短為 1 個字，平均長度為 109.13 個字，標準差為 120.01。AG's News 資料集是從網路上超過 2000 個來源取出來的新聞，目的是要將這些新聞分成 4 類，挑選出來的新聞有 127600 筆，其中 120000 筆為訓練資料，4 個類別各有 30000 筆，7600 筆為測試資

料，也是 4 個類別各有 1900 筆，最長的新聞有 173 個字，最短為 1 個字，平均長度為 30.92 個字，標準差為 10.64。Sogou 是結合了 SogouCA [33]和 SogouCS [33]這兩個資料集，目的是要將資料依照 5 個不同領域分成 5 類，這個資料集比較特別的是這些資料本來都是中文的，但他們透過漢語拼音轉換工具 pypinyin 和 jieba 斷詞系統將其轉換成英文字母與數字，所以我們人類是沒辦法直接看得懂資料內容的，但是因為有進行斷詞所以還是可以用於訓練模型。資料集有 510000 筆資料，其中訓練資料有 450000 筆，5 個類別各有 90000 筆資料，測試資料有 60000 筆，也是 5 個類別各有 12000 筆，最長字數有 45740 個字，最短為 0 個字，因為是經過轉換的，所以最長字數才會這麼長，0 個字則是那個資料只有符號所以經過轉換就不會有字。

表 4.2 為四個訓練集中訓練資料、驗證資料與測試資料的數量，IMDb 我們取 2000 筆驗證資料，正負評各 1000 筆，Elec 也是取 2000 筆，正負評也各 1000 筆，AG's News 每個類別取 3000 筆總共取 12000 筆，Sogou 每個類別取 9000 筆總共取 45000 筆。

表 4.1: 使用的資料集資訊

資料集	IMDb	Elec	AG's News	Sogou
訓練資料	25,000	25,000	120,000	450,000
測試資料	25,000	25,000	7,600	60,000
類別	2	2	4	5
最短/長字數	6/2122	1/5308	1/173	0/45740
平均長度 (字)	230.95	109.13	30.92	533.06
標準差	170.88	120.01	10.64	639.33

表 4.2: 資料集中訓練、驗證與測試資料數量

資料集	IMDb	Elec	AG's News	Sogou
訓練資料	23,000	23,000	108,000	405,000
驗證資料	2,000	2,000	12,000	45,000
測試資料	25,000	25,000	7,600	60,000

4.2. 實驗環境

OS : Ubuntu 18.04.3LTS

CPU : i9-9820x

RAM : 128G

GPU : nvidia GTX 2080ti 11g

我們方法執行的環境的 OS 是在 Ubuntu 18.04.3LTS 上面執行的，硬體方面使用的 CPU 是 i9-9820x，RAM 有 128G，GPU 是 nvidia GTX 2080ti 11g。Tensorflow 版本是 2.20，Keras 也是 2.20，Python 是 3.8.5。

4.3. 實驗設計

本論文想了解從一開始的每一個階段結束時都進行兩次的挑選，每一個階段都經過 40 個 epochs 的訓練，到最後一個階段結束後是否能訓練出更不受噪音資料影響的模型。我們的實驗都會取部分的訓練資料當成驗證資料，驗證資料用於決定是否提早停止，讓每一個階段訓練不需要都經過 40 個 epochs，可以節省一些時間也可以避免模型過擬合 (overfitting)。

實驗會比較 Co-teaching、傳統類神經網路、二次挑選與以及比較只使用一次挑選的差異，總共四種方法，我們會在 0%、10%、20%、30%、40%這幾種噪音率情況下每一個方法都進行三次取平均。

傳統類神經網路指的是一個四層的淺層模型，圖 4.1 為其二分類時的參數數量，圖 4.4 為架構圖。第一層為輸入密集層，第二層也是一個密集層，第三層為 Dropout，最後一層為 Sigmoid 輸出。Sigmoid 是將輸入該層的資料轉換至[0,1]之間的數值。圖 4.2 與圖 4.3 分別為 AG's News 與 Sogou 中傳統類神經網路參數數量，圖 4.5 與圖 4.6 為架構圖，圖中的問號指的是輸入資料的 batch size。多分類與二分類的差別只在最後一層更改為 Softmax。多分類兩個資料集的差別只在最後一層 AG's News 為因為要分 4 類所以為 4，Sogou 要分 5 類所以為 5。

我們的方法因為是著重在資料的挑選，所以我們使用的是最簡單的類神經網路，圖 4.7 為二分類資料集 IMDb 與 Elec 中的網路參數，圖 4.8 與圖 4.9 分別為 AG's News 與 Sogou 中的網路參數。圖 4.10 為 IMDb 與 Elec 中的網路架構，圖 4.11 與圖 4.12 分別是 AG's News 與 Sogou 的網路架構圖，圖中的問號指的是輸入資料的 batch size。

Co-teaching 的方法有兩個網路，兩個網路的架構與參數數量是一樣的，但兩個網路的初始參數值是隨機設定的，圖 4.13 為 Co-teaching 在 IMDb 與 Elec 中的網路參數數量，圖 4.14 是 Co-teaching 在 AG's News 中的網路參數數量，圖 4.15 是 Co-teaching 在 Sogou 中的網路參數數量，Co-teaching 的方法因為兩個網路長的是一樣的所以參數數量是一樣，但是參數值是不同的。圖 4.16 是 Co-teaching 在 IMDb 與 Elec 中的網路架構圖，圖 4.17 是 Co-teaching 在 AG's News 中的網路架構圖，圖 4.18 是 Co-teaching 在 Sogou 中的網路架構圖，圖中的問號指的是輸入資料的 batch size。這裡只顯示網路的架構，並沒有顯示交換訓練資料的步驟，交換訓練資料的步驟在圖 2.5 有詳細的說明。

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 16)                160016
_________________________________________________________________
dense_1 (Dense)              (None, 16)                272
_________________________________________________________________
dropout (Dropout)            (None, 16)                0
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 17
=================================================================
Total params: 160,305
Trainable params: 160,305
Non-trainable params: 0
_________________________________________________________________
```

圖 4.1: 傳統類神經網路參數數量

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_3 (Dense)              (None, 16)                160016
_________________________________________________________________
dense_4 (Dense)              (None, 16)                272
_________________________________________________________________
dropout_1 (Dropout)          (None, 16)                0
_________________________________________________________________
dense_5 (Dense)              (None, 4)                 68
=================================================================
Total params: 160,356
Trainable params: 160,356
Non-trainable params: 0
_________________________________________________________________
```

圖 4.2: AG's News 中傳統類神經網路參數數量

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_6 (Dense)              (None, 16)                160016
_________________________________________________________________
dense_7 (Dense)              (None, 16)                272
_________________________________________________________________
dropout_2 (Dropout)          (None, 16)                0
_________________________________________________________________
dense_8 (Dense)              (None, 5)                 85
=================================================================
Total params: 160,373
Trainable params: 160,373
Non-trainable params: 0
_________________________________________________________________
```

圖 4.3: Sogou 中傳統類神經網路參數數量

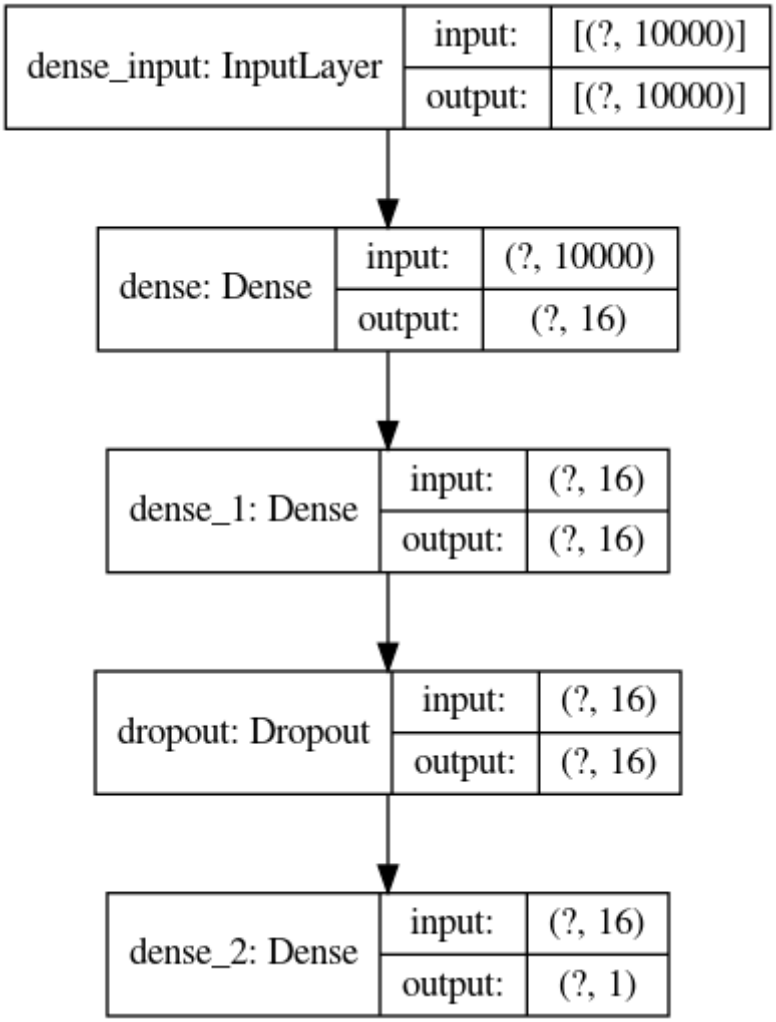


圖 4.4: 傳統類神經網路在 IMDb 與 Elec 架構圖

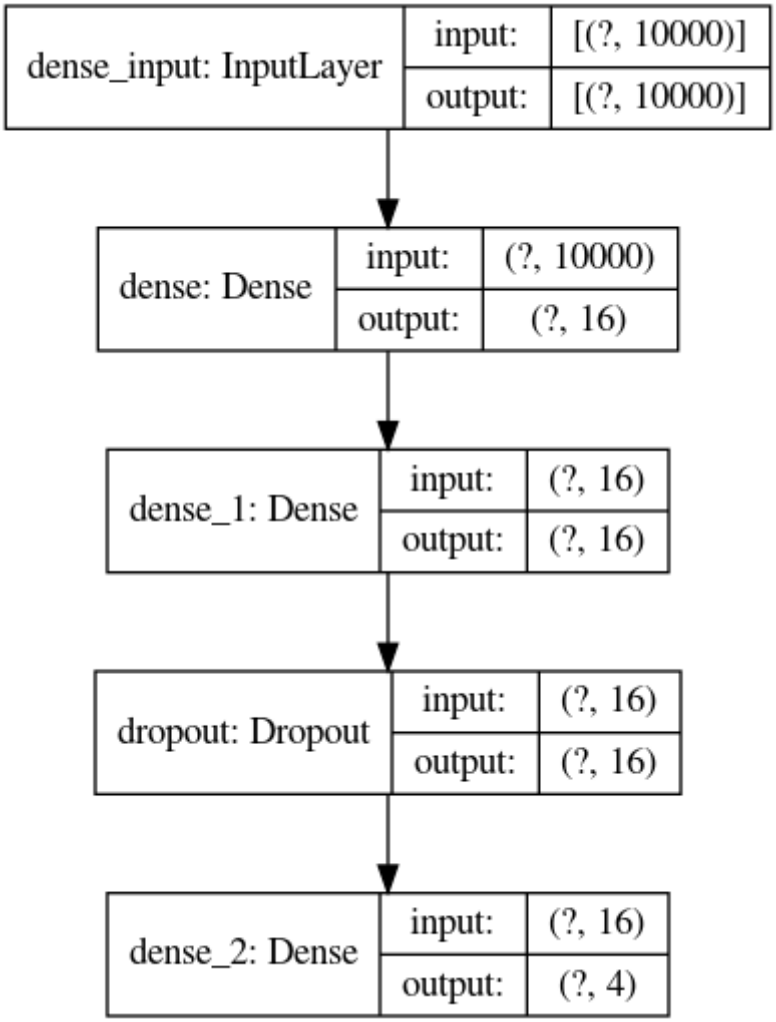


圖 4.5: 傳統類神經網路在 AG's News 架構圖

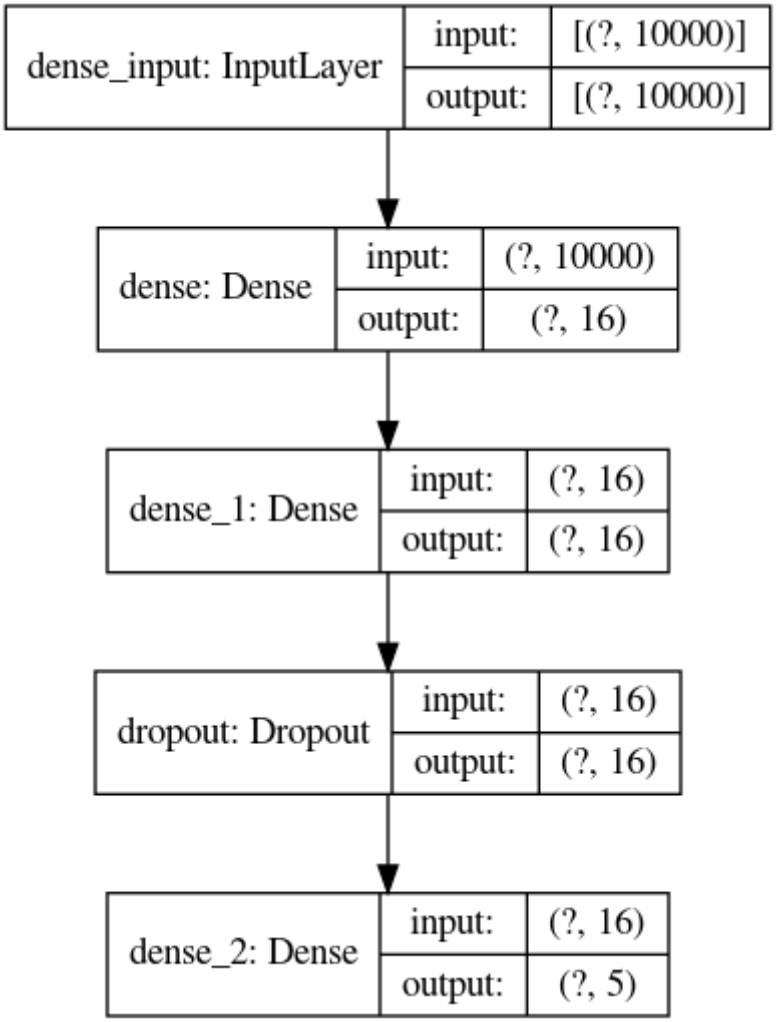


圖 4.6: 傳統類神經網路在 Sogou 架構圖

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_15 (Dense)             (None, 16)                160016
_________________________________________________________________
dense_16 (Dense)             (None, 16)                272
_________________________________________________________________
dropout_5 (Dropout)          (None, 16)                0
_________________________________________________________________
dense_17 (Dense)             (None, 1)                 17
=================================================================
Total params: 160,305
Trainable params: 160,305
Non-trainable params: 0
_________________________________________________________________
```

圖 4.7: IMDb 與 Elec 中我們方法的網路參數數量

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_12 (Dense)             (None, 16)                160016
_________________________________________________________________
dense_13 (Dense)             (None, 16)                272
_________________________________________________________________
dropout_4 (Dropout)          (None, 16)                0
_________________________________________________________________
dense_14 (Dense)             (None, 4)                 68
=================================================================
Total params: 160,356
Trainable params: 160,356
Non-trainable params: 0
_________________________________________________________________
```

圖 4.8: AG's News 中我們方法的網路參數數量

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_9 (Dense)              (None, 16)                160016
_________________________________________________________________
dense_10 (Dense)             (None, 16)                272
_________________________________________________________________
dropout_3 (Dropout)          (None, 16)                0
_________________________________________________________________
dense_11 (Dense)             (None, 5)                 85
=================================================================
Total params: 160,373
Trainable params: 160,373
Non-trainable params: 0
_________________________________________________________________
```

圖 4.9: Sogou 中我們方法的網路參數數量

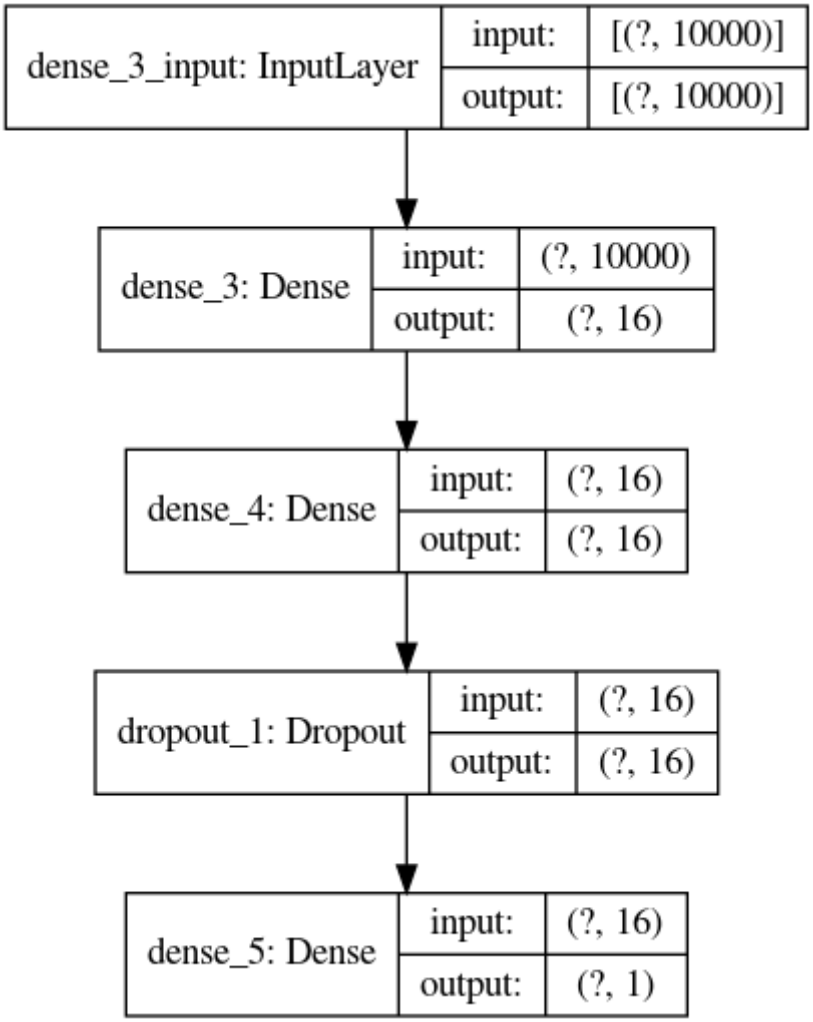


圖 4.10: 我們方法使用的網路在 IMDb 與 Elec 架構圖

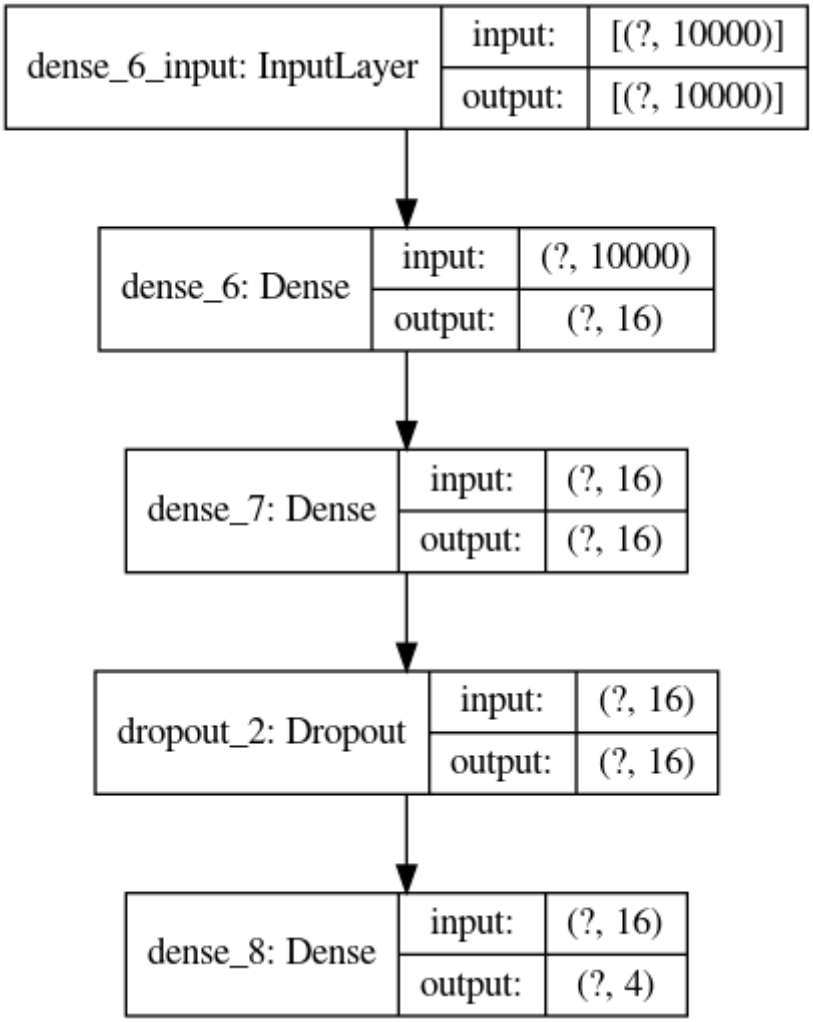


圖 4.11: 我們方法使用的網路在 AG's News 架構圖

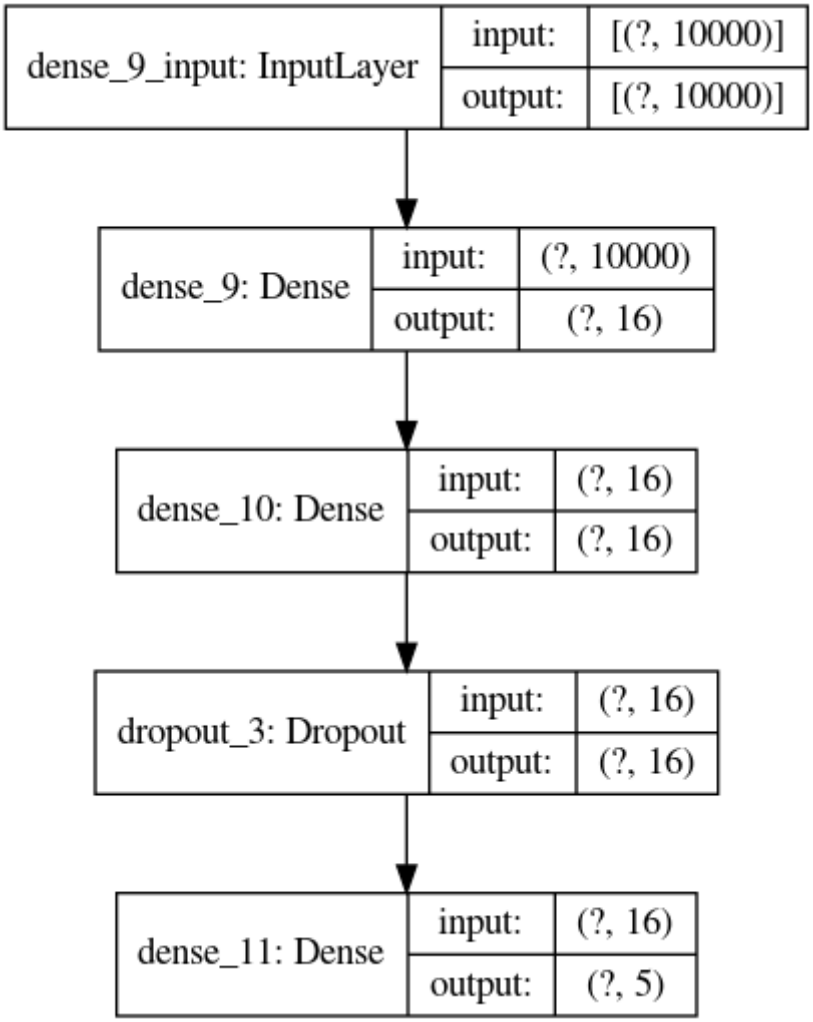


圖 4.12: 我們方法使用的網路在 Sogou 架構圖

Layer (type)	Output Shape	Param #
dense_30 (Dense)	(None, 16)	160016
dense_31 (Dense)	(None, 16)	272
dropout_10 (Dropout)	(None, 16)	0
dense_32 (Dense)	(None, 1)	17

Total params: 160,305
Trainable params: 160,305
Non-trainable params: 0

Layer (type)	Output Shape	Param #
dense_30 (Dense)	(None, 16)	160016
dense_31 (Dense)	(None, 16)	272
dropout_10 (Dropout)	(None, 16)	0
dense_32 (Dense)	(None, 1)	17

Total params: 160,305
Trainable params: 160,305
Non-trainable params: 0

圖 4.13: Co-teaching 在 IMDb 與 Elec 中網路參數數量

Layer (type)	Output Shape	Param #
dense_27 (Dense)	(None, 16)	160016
dense_28 (Dense)	(None, 16)	272
dropout_9 (Dropout)	(None, 16)	0
dense_29 (Dense)	(None, 4)	68

Total params: 160,356
Trainable params: 160,356
Non-trainable params: 0

Layer (type)	Output Shape	Param #
dense_27 (Dense)	(None, 16)	160016
dense_28 (Dense)	(None, 16)	272
dropout_9 (Dropout)	(None, 16)	0
dense_29 (Dense)	(None, 4)	68

Total params: 160,356
Trainable params: 160,356
Non-trainable params: 0

圖 4.14: Co-teaching 在 AG's News 中網路參數數量

Layer (type)	Output Shape	Param #
dense_24 (Dense)	(None, 16)	160016
dense_25 (Dense)	(None, 16)	272
dropout_8 (Dropout)	(None, 16)	0
dense_26 (Dense)	(None, 5)	85

Total params: 160,373
Trainable params: 160,373
Non-trainable params: 0

Layer (type)	Output Shape	Param #
dense_24 (Dense)	(None, 16)	160016
dense_25 (Dense)	(None, 16)	272
dropout_8 (Dropout)	(None, 16)	0
dense_26 (Dense)	(None, 5)	85

Total params: 160,373
Trainable params: 160,373
Non-trainable params: 0

圖 4.15: Co-teaching 在 Sogou 網路參數數量

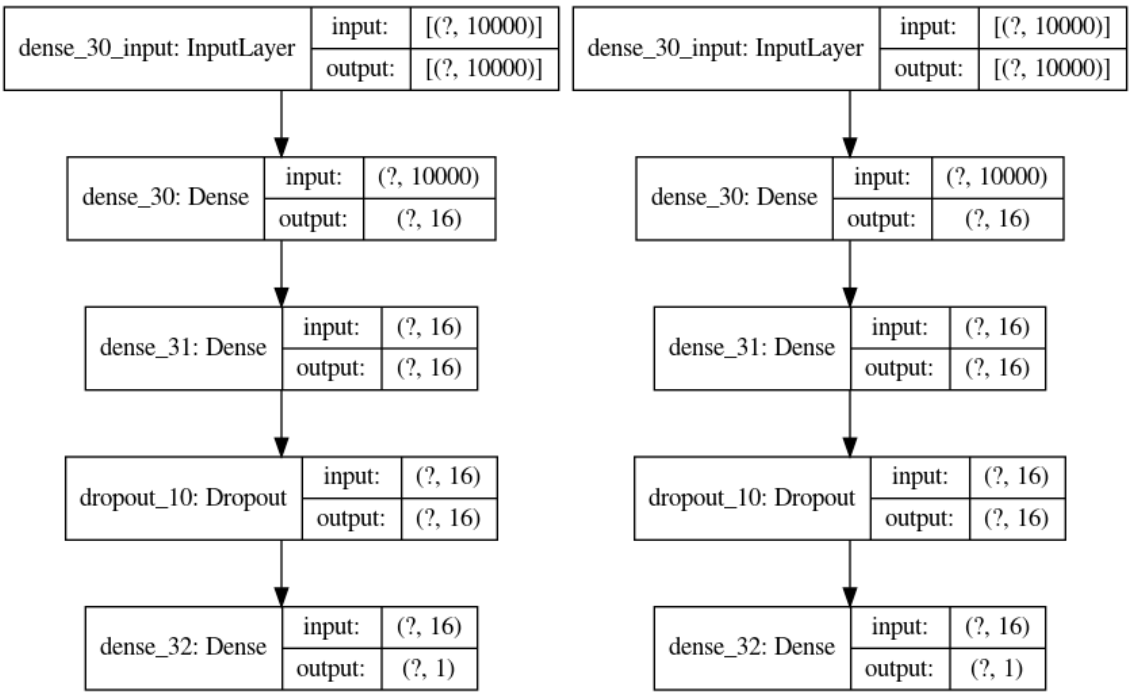


圖 4.16: Co-teaching 在 IMDb 與 Elec 網路架構圖

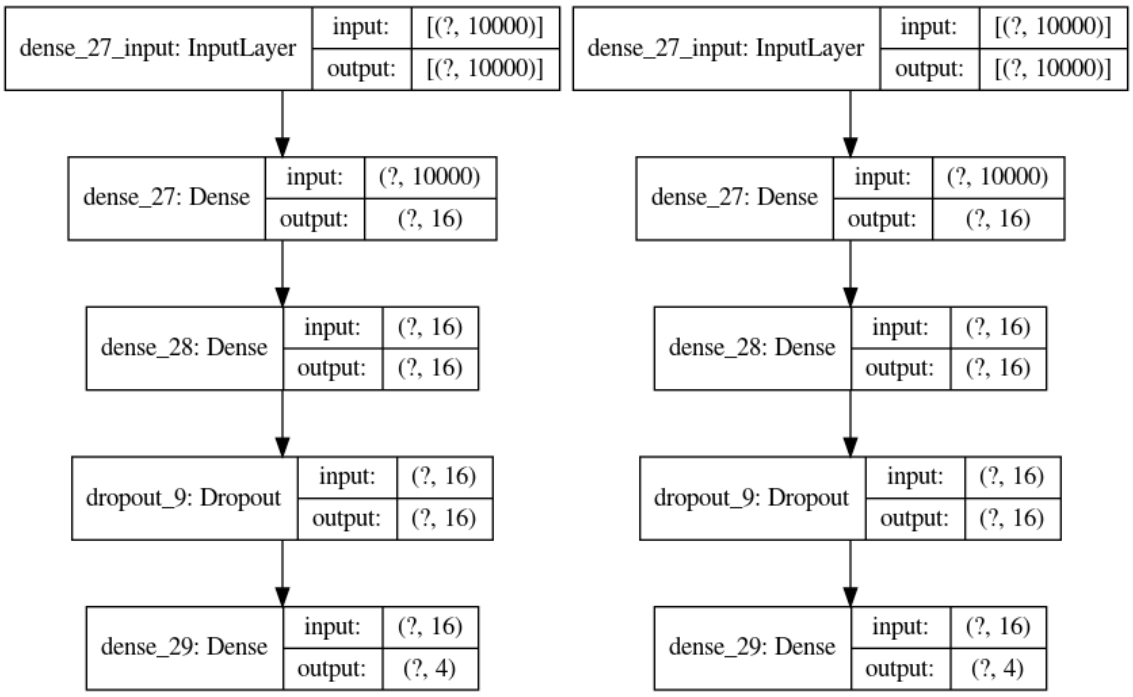


圖 4.17: Co-teaching 在 AG's News 網路架構圖

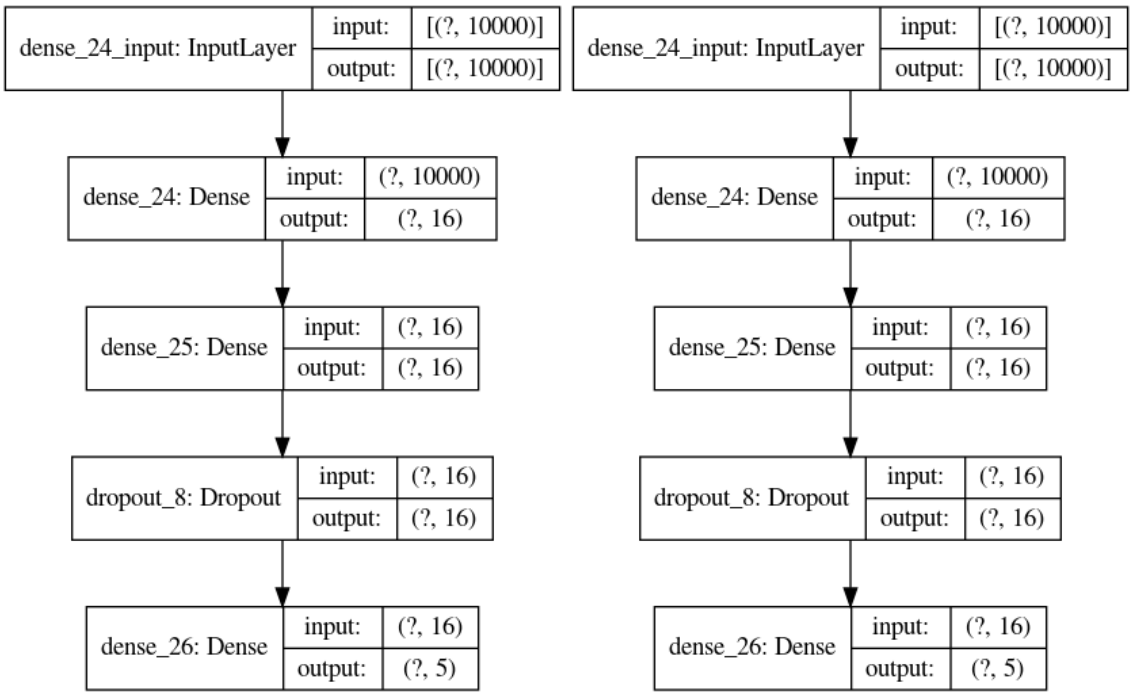


圖 4.18: Co-teaching 在 Sogou 網路架構圖

4.3.1. 評量公式

在測試階段我們需要一個標準來評估我們的模型好壞，模型會將每筆測試資料給一個標籤，這個標籤只要與資料本身的標籤相同就是分類正確，反之就是錯誤，統計完之後我們使用測試資料準確率圖 4.19 來評估。

$$準確率\ (\text{Accuracy}) = \frac{分類正確的資料數量}{全部的資料數量}$$

圖 4.19: 準確率公式

4.3.2. 噪音加入方法

在二分類的資料集加入噪音的方法較為容易，我們會根據噪音率去選擇資料，並將選擇到的資料標籤直接更改成另外一類，舉例來說: 現在有一個要分成 (0, 1) 兩類的乾淨資料集，假設噪音率為 40%，我們就會取其中 40%的資料更改其標籤，若取得資料的標籤為 0 就會更改成 1，1 就會更改成 0。

在多分類的資料集加入噪音的方法為了確保每類的噪音資料數量都是公平的被加入，

我們使用對稱式翻轉 (Symmetry Flipping) [31]，對稱式翻轉的意思為我們根據噪音率翻轉某一類的資料，被選出來要被翻轉的資料會平均的翻轉成其他類別的標籤。我們使用表 4.3 來說明這個方法，表 4.3 中我們假設有一個五類的乾淨資料集，噪音率為 40%，我們每類都會留下 60%的資料，剩下 40%的資料標籤會被更改成另外四類的標籤。從第一列來看類別一的資料我們留下了 60%，剩下 40%的標籤我們平均改成類別二、三、四、五讓每一個類別都有 10%的資料，第二列到第五列都是做一樣的事情。這樣子可以確保每一個類別被更改標籤的資料數量是一致的，標籤更改完之後每一個類別的資料數量也不會被更改。

表 4.3: 五類別資料集在 40%噪音率標籤更改情形

	類別一	類別二	類別三	類別四	類別五
類別一	60%	10%	10%	10%	10%
類別二	10%	60%	10%	10%	10%
類別三	10%	10%	60%	10%	10%
類別四	10%	10%	10%	60%	10%
類別五	10%	10%	10%	10%	60%

4.3.3. 訓練

在二分類訓練的兩次挑選過程中，每一個階段的第一次挑選都是留下訓練資料中預測標籤與資料標籤一致的資料，接著我們會對預測標籤與資料標籤不一致的資料進行第二次的挑選。第二次的挑選如同我們在方法流程中說明的一樣，因為我們要挑選判斷邊界 0.5 附近的資料，一開始會將分類分數在 [0.4,0.6] 之間的資料加入至新的訓練資料中，我們假設一個變動值為 0.01，接著每進行一個階段會將挑選區間最大值減少 0.01、最小值增加 0.01，

直到區間變成 [0.49,0.51] 為止，總共變動 9 次，區間為 [0.49,0.51] 的那一個階段即為最後一個階段，總共會進行 10 個階段。

在多分類訓練的兩次挑選過程中，每一個階段的第一次挑選也是留下訓練資料中預測標籤與資料標籤一致的資料，接著也會對預測標籤與資料標籤不一致的資料進行第二次的挑選。第二次的挑選我們也和方法流程中說明的因為多分類的結果與二分類不同，我們會去計算最大的數值減去資料標籤對應的數值之後的差要小於等於門檻值來判斷是否被挑選，所以需要訂定一個門檻值，我們在兩個多分類的資料集中從經過測試之後將門檻值在 AG's News 資料集設定為 0.5，Sogou 資料集設定為 0.6，最大的數值與標籤的數值差小於門檻值我們才會將此資料加入至新的訓練資料，多分類也是進行 10 個階段，但門檻值為固定的不會更動。

表 4.4: 我們方法與另外兩種方法的表示法與網路

方法	表示法	網路
傳統類神經網路	One-hot	四層類神經網路
Co-teaching	One-hot	四層類神經網路
我們的方法	One-hot	四層類神經網路

本論文的方法與 Co-teaching 與傳統類神經網路都是使用 One-hot [24]表示法，如表 4.4 所示，我們方法與傳統類神經網路與 Co-teaching 的模型使用的是四層傳統類神經網路。

我們實驗中優化器使用的是 RMSprop，學習率設定為 0.0001，激活函數在網路前幾層都是 Leaky Relu，最後一層會根據二分類或多分類有所不同，二分類為 Sigmoid，多分類為 Softmax。

Co-teaching 的優化器也是 RMSprop，學習率設定為 0.0001，激活函數在前幾層都是

Leaky Relu，最後一層會根據二分類或多分類有所不同，二分類為 Sigmoid，多分類為 Softmax，Co-teaching 還有多了一個遺忘率，這個遺忘率表示每一個 epoch 要遺忘的資料數量，遺忘率是會更改的，原論文中遺忘率的最大值設為和噪音率一樣，例: 噪音率為 0.4 則遺忘率最大值設為 0.4。在強健訓練中有提到 Co-teaching 到後期會留下較少的資料，換言之是遺忘較多的資料，遺忘率在前面的 epoch 會設定較小，直到最後一個 epoch 的遺忘率為噪音率，每一個 epoch 遺忘率會增加 噪音率/epoch 數量 的值，直到最後一個 epoch 遺忘率達到最大值為止。

傳統類神經網路的優化器使用的是 RMSprop，學習率設定為 0.0001，激活函數在網路前幾層都是 Leaky Relu，最後一層也會根據二分類或多分類有所不同，二分類為 Sigmoid，多分類為 Softmax。

資料長度的部份三種方法我們都使用一樣的資料長度，但在四個不同的資料集會有不同的資料長度，在 IMDb 中資料長度為 1000 字，Elec 中資料長度為 1000 字，AG's News 中資料長度為 500 字，Sogou 中資料長度為 2000 字。

Co-teaching 因為其論文只在影像的資料集進行訓練，這裡我們將其程式自行撰寫套用至文字的資料集中，根據相關研究中 Co-teaching 的說明我們設計兩個傳統類神經網路當成網路 A 與網路 B，我們也根據原論文的設定將遺忘率最大值設定為噪音率，接著在每一個 epoch 開始時將資料分別交給網路 A 與網路 B，網路 B 會對 A 的訓練資料進行預測並計算損失，再根據損失排序，排序完畢之後依照當前 epoch 的遺忘率來遺棄資料，遺棄完畢後剩下的資料才是網路 A 當前 epoch 的訓練資料，反之網路 A 也會挑選 B 的訓練資料，一樣會先進行預測並計算損失接著排序，然後根據遺忘率遺忘資料，挑選完之後網路 B 使用 A 挑選的資料作為當前 epoch 的訓練資料。

4.3.4. 測試

我們的目的是在有噪音的環境中訓練出一個不受噪音影響的模型，所以我們只會在訓練時加入噪音，在測試時全部都為乾淨資料。我們藉由計算在乾淨資料中的準確率來評估模型的好壞。

4.4. 實驗結果

圖 4.20、圖 4.21、圖 4.22、圖 4.23 這四張圖分別為 IMDb、Elec、AG's News、Sogou 在四種不同噪音率的實驗結果。圖中有只做一次挑選和兩次挑選與另外兩種方法的比較，我們的方法是做兩次挑選的方法，先挑選一致的資料，再從不一致中挑選資料。我們會比較只挑選一致資料的結果，即為一次挑選的實驗結果，藉此來比較第一次與第二次的挑選分別改進了多少。X 軸為噪音率，Y 軸為測試準確率。圖中的四條線代表四種方法，圖中的叉為只用一次挑選，三角形為兩次挑選，正方形為傳統類神經網路，圓形為 Co-teaching。

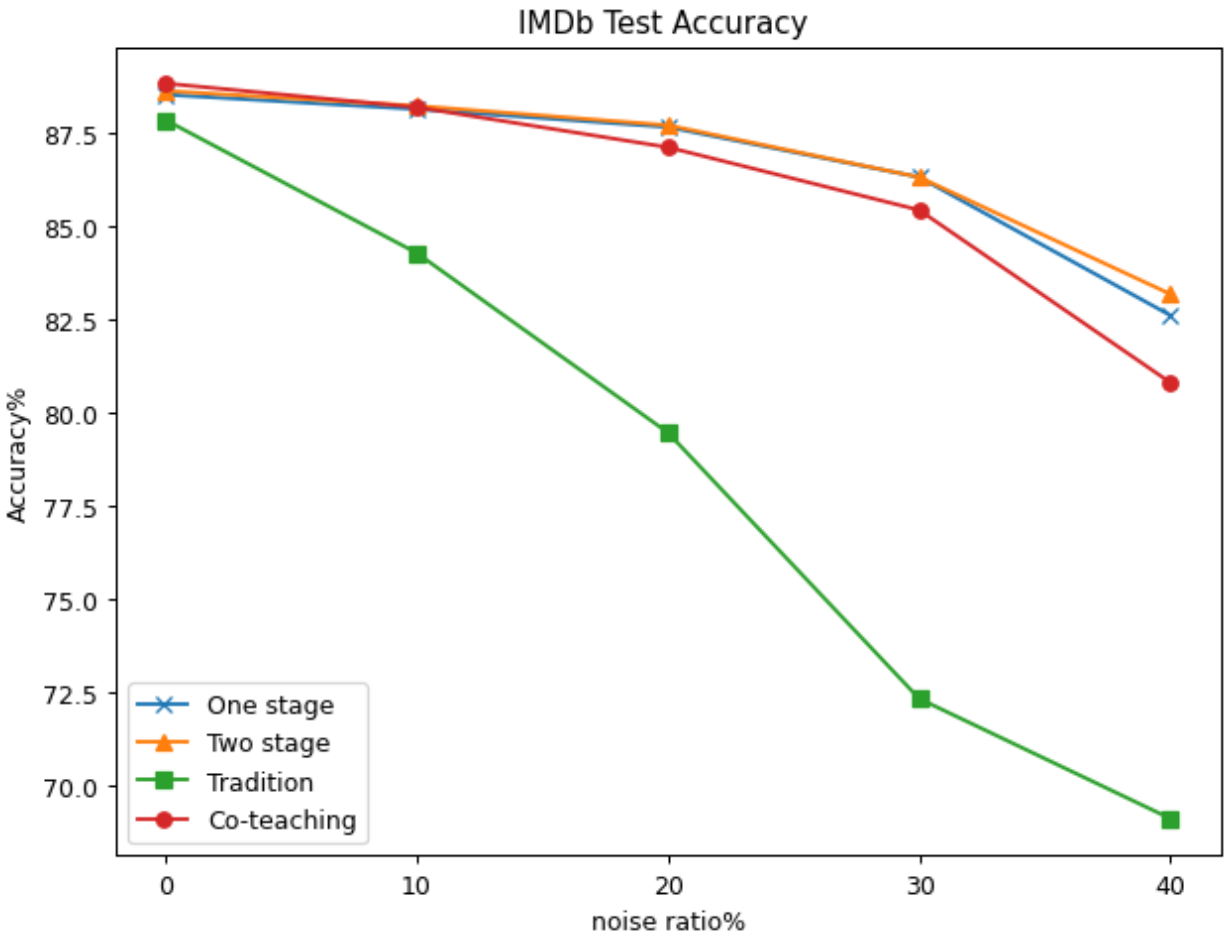


圖 4.20： IMDb 在不同噪音率下測試資料準確率的折線圖

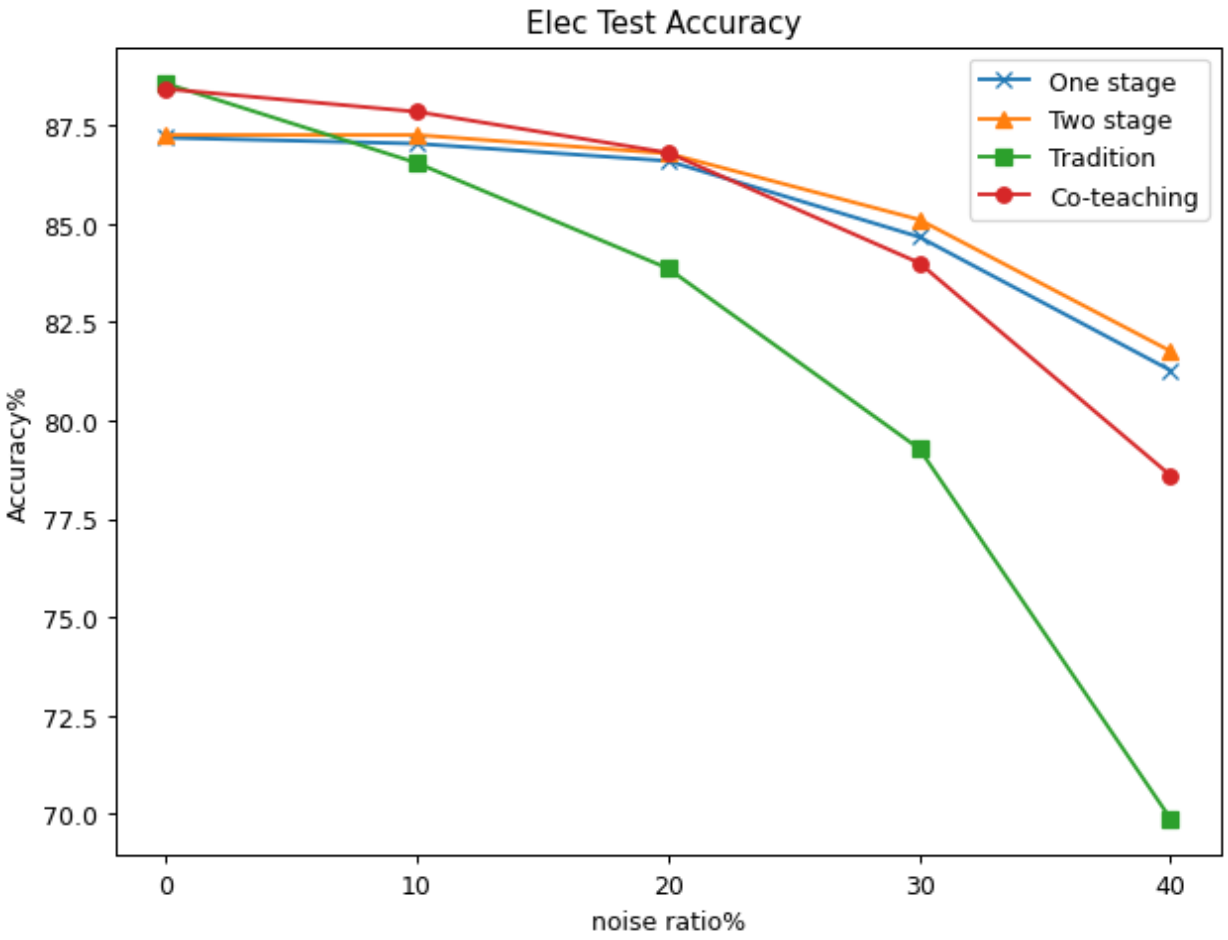


圖 4.21: Elec 在不同噪音率下測試資料準確率的折線圖

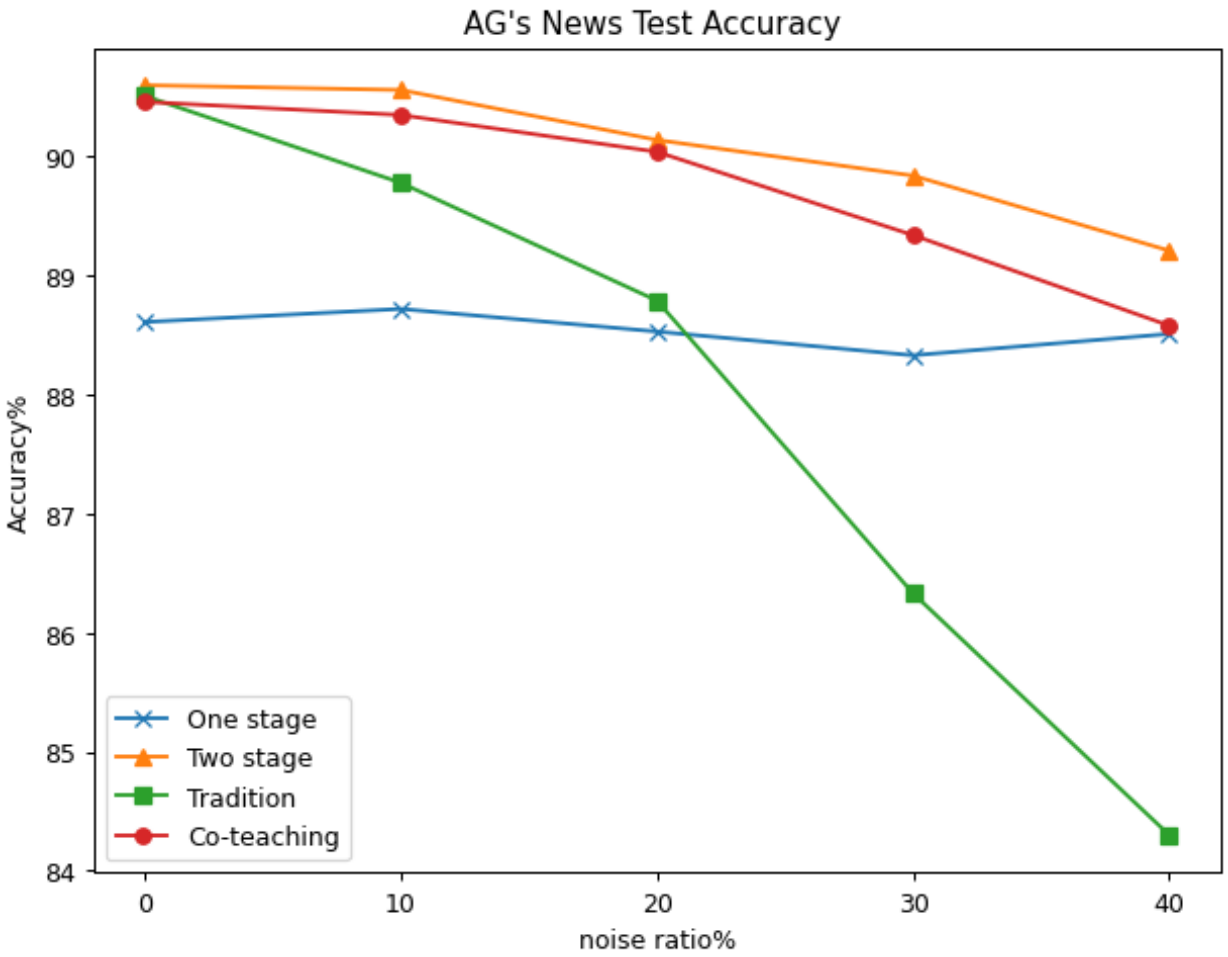


圖 4.22: AG’s News 在不同噪音率下測試資料準確率的折線圖

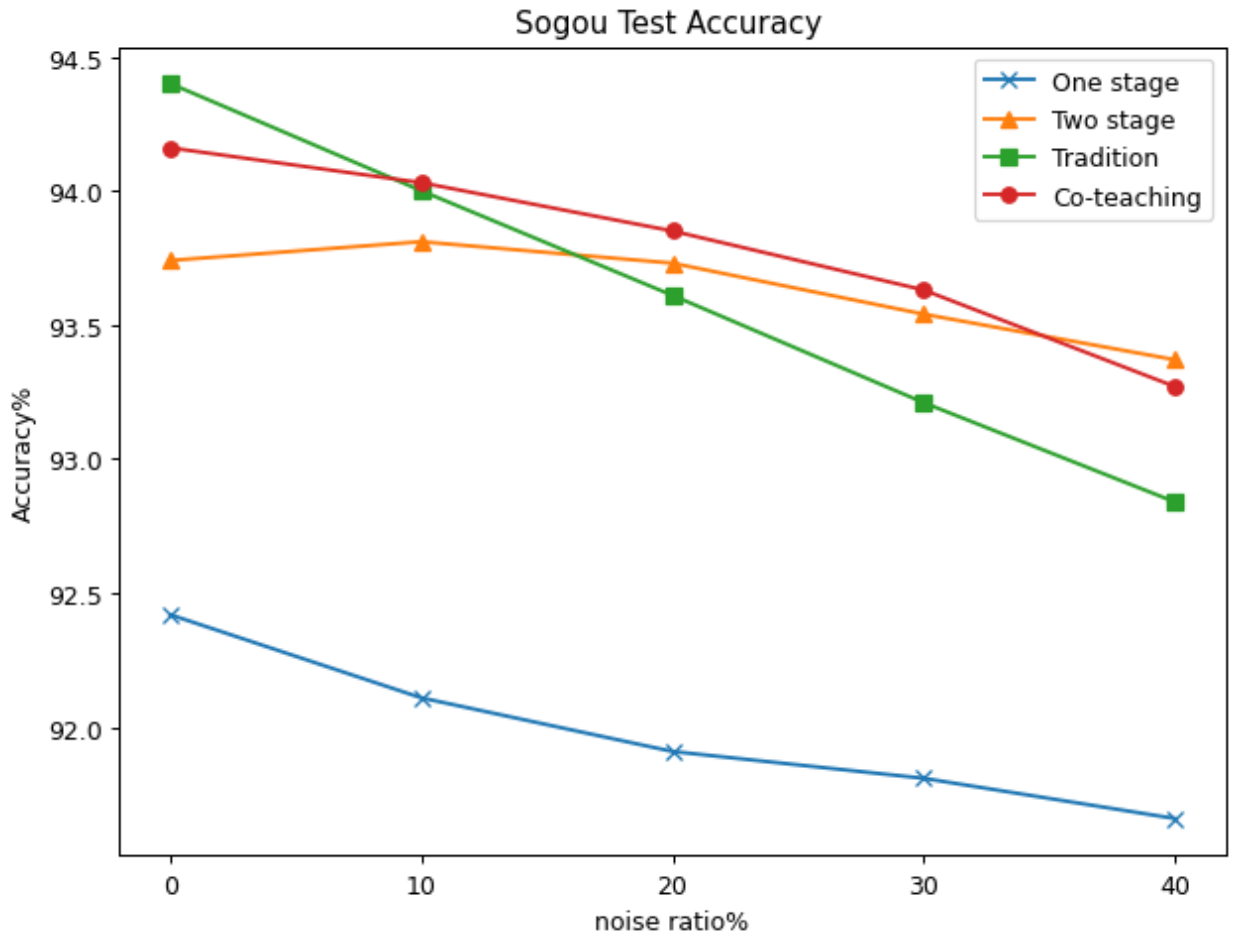


圖 4.23: Sogou 在不同噪音率下測試資料準確率的折線圖

表 4.5、表 4.6、表 4.7、表 4.8 分別為 IMDb、Elec、AG's News、Sogou 四個資料集在四種不同的噪音率中的測試準確率表格。粗體字為該噪音率中表現最好的結果。

表 4.5: IMDb 在不同噪音資料下測試資料準確率

噪音率 / 方法	0%	10%	20%	30%	40%
一次挑選	88.50%	88.11%	87.63%	86.29%	82.59%
用兩次挑選	88.60%	**88.20%**	**87.68%**	**86.29%**	**83.16%**
傳統類神經網路	87.82%	84.26%	79.45%	72.33%	69.12%
Co-teaching	**88.80%**	88.16%	87.09%	85.41%	80.79%

表 4.6: Elec 在不同噪音資料下測試資料準確率

噪音率 / 方法	0%	10%	20%	30%	40%
一次挑選	87.18%	87.03%	86.59%	84.66%	81.27%
用兩次挑選	87.25%	87.25%	86.78%	**85.10%**	**81.76%**
傳統類神經網路	88.56%	86.54%	83.86%	79.28%	69.88%
Co-teaching	**88.42%**	**87.84%**	**86.80%**	84.00%	78.60%

表 4.7: AG's News 在不同噪音資料下測試資料準確率

噪音率 / 方法	0%	10%	20%	30%	40%
一次挑選	88.61%	88.72%	88.53%	88.33%	88.51%
用兩次挑選	**90.60% (0.9 三次)**	**90.56% (0.8 三次)**	**90.14% (0.7 五次)**	**89.84% (0.6 五次)**	**89.21% (0.5 十次)**
傳統類神經網路	90.51%	89.78%	88.79%	86.33%	84.30%
Co-teaching	90.46%	90.35%	90.04%	89.34%	88.58%

表 4.8: Sogou 在不同噪音資料下測試資料準確率

噪音率 / 方法	0%	10%	20%	30%	40%
一次挑選	92.42%	92.11%	91.91%	91.81%	91.66%
用兩次挑選	93.74% (0.9 三次)	93.81% (0.9 三次)	93.73% (0.8 五次)	93.54% (0.7 八次)	**93.37% (0.6 十次)**
傳統類神經網路	**94.40%**	94.00%	93.61%	93.21%	92.84%
Co-teaching	94.16%	**94.03%**	**93.85%**	**93.63%**	93.27%

我們先觀察二分類的資料集，在表 4.5 中的數據可以發現在 IMDb 的資料集中 0%的情況大家表現得都一樣好，10%噪音率開始到 40%噪音率都是我們兩次挑選的方法表現得比較好，隨著噪音率的增加我們兩次挑選的準確率下降越少，在 40%噪音率時兩次挑選的最明顯。

在表 4.6 中 Elec 資料集 0%表現最好的方法是 Co-teaching，我們的方法反而表現的比傳統類神經網路差，這是因為我們的方法是將不一致資料移除，但在 0%的時候並沒有噪音資料，其實應該保留所有資料進行訓練，因為我們的方法移除不一致的資料中都是乾淨的資料，這樣會導致準確率下降。在 10%和 20%時雖然 Co-teaching 還是最好的，但我們的方法只差了一點，到了 30%和 40%的噪音率我們的方法就是最好的了。

我們接著觀察多分類的資料集，表 4.7 為 AG's News 資料集的詳細結果，我們的方法在 0%~40%都是最好的，表格中每一個準確率下括號中表示的是門檻值與執行的階段數，例如在 0%噪音率 (0.9 三次) 表示的是門檻值為 0.9，階段數是 3。表 4.8 為 Sogou 資料集的詳細結果，0%表現最好的一樣是傳統類神經網路，10%到 30%都是 Co-teaching 表現較好，我們的方法到了 40%就是表現最好的了，表格中每一個準確率下括號中表示的也是門檻值與執行的階段數，例如在 0%噪音率 (0.9 三次) 表示的是門檻值為 0.9。

4.5. 實驗討論

我們的方法在所有資料集中 40%噪音率中都是表現的最好的，在 IMDb 資料集中我們的方法在有噪音的情況下都是表現得最好，表現第二好的都是 Co-teaching。但在 Elec 資料集中我們的方法只有 30%和 40%噪音率表現的最好，0%~20%表現最好的是 Co-teaching，我們的方法在 10%~20%噪音率表現的第二好。在 AG's News 資料集中我們的方法在 0%~40%噪音率表現的都是最好，在 Sogou 資料集中我們的方法只在 40%噪音率表現的最好，在 10%~30%是 Co-teaching 表現的最好，0%時傳統類神經網路表現的最好，關於這個

現象我們推測應該是多分類的這個資料集的訓練資料數量和其他相比多了許多，訓練類神經網路本來就是越多的資料能訓練出越好的網路，在低噪音率的情況乾淨的訓練資料又很多時，去進行我們方法中的資料挑選可能就不是很好的方法，但是 Co-teaching 也是樣本選擇的方法，它們卻可以在 10%~30%表現得很好，這裡我們需要說明我們與 Co-teaching 之間的差異。

Co-teaching 與我們的方法之間的不同是在挑選資料的方式，Co-teaching 是在每一個 epoch 開始訓練之前對全部訓練資料進行損失排序並保留損失較小的資料，我們的方法則是整個訓練結束時對全部的訓練資料進行一致與不一致的挑選。Co-teaching 每一個 epoch 都會對資料進行挑選，但我們只有在每一個階段結束時對資料進行挑選，所以我們的總資料數量會隨著挑選階段的進行數量逐漸越少，但 Co-teaching 的總訓練資料數量不會減少，只會在 epoch 開始時減少數量，但到了新的 epoch 又是全部的訓練資料開始挑選。在挑選的規則上我們與 Co-teaching 也不同，Co-teaching 是照著損失來挑選，我們是一致與不一致。

因此，我們的方法可能就不適用在資料數量多且表現得不錯的資料集。從實驗結果來看，Sogou 資料集 10%噪音率的狀況，傳統類神經網路就可以達到 94.00%的準確率，若我們進行一致與不一致的挑選，因為噪音率不高訓練資料又多，所以不一致資料中乾淨資料的比例可能就會比較高，就算我們有從不一致資料進行挑選並加入至新的訓練資料，也是會捨棄掉部分的乾淨資料，這樣子多做幾個階段就會導致乾淨資料的數量越來越少，模型的準確率就下降了，這種現象可能就是我們方法在資料數量較多的資料集 0%~30 噪音率的情況下沒辦法表現的更好的原因。Co-teaching 的方法因為在每一個 epoch 都是使用全部的資料進行挑選，所以他們若是在這個 epoch 捨棄掉乾淨資料，在後面的 epoch 也有機會將其保留，並且他們使用兩個網路互相挑選，因此更可以降低誤判的可能性。所以他們在 10%~30%噪音率才可以表現得那麼好。

在 40%的噪音率時因為噪音資料的數量比起 10%多了許多，此時我們的方法就可以更有效地將噪音資料移除，Co-teaching 因為每一個 epoch 都是使用全部資料進行挑選，此時噪音資料的數量太多讓他們也沒辦法很好的將噪音資料移除，導致他們的表現就變差了。

第五章 結論與展望

早期噪音資料的處理方法大部分都是針對資料進行前處理之後再將處理後的資料當作訓練資料，但由於無法確切得知資料中噪音資料的比例，只能透過假設或機率分布來設定噪音資料。強健訓練中都是針對訓練過程去修改或調整，所以並不需要知道噪音資料的比例。我們的方法是強健訓練中挑選資料的方法，我們不需要藉由外部資料的幫忙，只需要從訓練資料中重複地進行兩次挑選的過程就可以得到比之前的方法更好的模型。我們的貢獻是提出了一種方法藉由重複地進行兩次的挑選就可以有效地在有噪音的資料集中訓練出比之前方法更好的模型。

在重複進行兩次挑選時我們與之前樣本選擇的差異是從不同的角度挑選資料，在之前的樣本選擇方法大多使用損失來挑選資料，我們改成從一致與不一致來挑選。我們使用重複兩次挑選有兩個優點，第一個是我們挑選了一致的資料，這樣可以去除噪音資料，可以有效地降低噪音比例，第二個是為了保持資料的多樣性和數量，希望能從不一致的資料中挑選乾淨資料來增加資料的多樣性與數量。

我們的方法可以應用在二分類與多分類的資料集中並得到比之前的方法更好的表現。在二分類的 IMDb 資料集中，10%~40%噪音率情況下我們都是表現最好的，在 40%噪音率還可以達到 83.16%的準確率，比表現第二好的 Co-teaching 還要高了 3%，在二分類的 Elec 資料集中，30%~40%噪音率我們的方法也都表現的最好，在 40%噪音率下我們準確率有 81.76%，比第二好的 Co-teaching 也高了 3%。在多分類的 AG's News 資料集中，我們的方法在 0%~40%噪音率都是表現最好的，在 40%我們的準確率有 89.21%。在多分類的 Sogou 資料集中，在 10%~30%噪音率是 Co-teaching 表現較好，但在 40%噪音率的我們準確率有 93.37%。我們發現在資料數量較多的資料集並且表現已經不錯的情況下我們方法的表現不如預期。我們在實驗討論中也有探討其可能的原因，我們的方法也還有參數可以進行調整，

像是執行的階段數與多類別時的門檻值。我們希望未來能透過調整參數、修改挑選方法或規則來改善這個情況，讓我們的方法能在這種情況下也能表現得更好，以改善含有噪音資料的文章分類結果。

我們的方法只針對在資料挑選的部份進行調整，並沒有針對模型架構上進行修改，我們希望強調我們方法並不是靠著更好的模型達到改善，所以我們使用基礎的模型。我們在二分類與多分類分別提出各自專屬的挑選規則，我們希望能將這些挑選規則應用至其他的模型上，或是加入更多的挑選規則，像是加入以往樣本選擇使用的損失大小。我們希望能在其他的模型像是 BERT、XLNET 等等使用外部資料建立的模型套用我們的方法，若是這些模型使用我們的挑選方法也能達到改善就可以進一步推廣我們的方法到更多不同的模型上面。加入更多的挑選規則或許可以在更高的噪音率中保留更多的乾淨資料，或是在較低的噪音率中也可以保留更多的乾淨資料，藉此讓我們的方法可以在更多不同的噪音率下達到改善。

參考文獻

[1] E. Arazo, D. Ortego, P. Albert, N. E. O'Connor, and K. McGuinness, "Unsupervised label noise modeling and loss correction," in Proc. ICML, 2019.

[2] D. Arpit, S. Jastrzebski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio et al., "A closer look at memorization in deep networks," in Proc. ICML, 2017, pp. 233–242.

[3] B. Biggio, B. Nelson, and P. Laskov, "Support vector machines under adversarial label noise," in Proc. ACML, pp. 97-112, 2011.

[4] L. Breiman, "Arcing Classifier (with Discussion and a Rejoinder by the Author)," The Annals of Statistics, vol. 26, no. 3, pp. 801-849, 1998.

[5] L. Breiman, "Bagging Predictors," Machine Learning, vol. 24, no. 2, pp. 123-140, August, 1996.

[6] L. Breiman, "Stacked Regressions," Machine Learning, vol. 24, no. 2, pp. 49-64, July, 1996.

[7] H.-S. Chang, E. Learned-Miller, and A. McCallum, "Active Bias: Training more accurate neural networks by emphasizing high variance samples," in Proc. NeurIPS, 2017, pp. 1002–1012.

[8] J. Chen, S. Sathe, C. Aggarwal, and D. Turaga, "Outlier Detection with Autoencoder Ensembles," in Proc. SIAM International Conference on Data Mining, pp. 90-98, 2017.

[9] Y. Ding, L. Wang, D. Fan, and B. Gong, "A semi-supervised two-stage approach to learning from noisy labels," in Proc. WACV, 2018, pp. 1215–1224.

[10] A. Ganapathiraju and J. Picone, "Support vector machines for automatic data cleanup," in Proc. ICSLP, 2000.

[11] A. Ghosh, H. Kumar, and P. Sastry, "Robust loss functions under label noise for deep neural networks," in Proc. AAAI, 2017.

[12] J. Goldberger and E. Ben-Reuven, "Training deep neural-networks using a noise adaptation layer," in Proc. ICLR, 2017.

[13] B. Han, J. Yao, G. Niu, M. Zhou, I. Tsang, Y. Zhang, and M. Sugiyama, "Masking: A new perspective of noisy supervision," in Proc. NeurIPS, 2018, pp. 5836–5846.

[14] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. Tsang, and M. Sugiyama, “Co-teaching: Robust training of deep neural networks with extremely noisy labels,” in Proc. NeurIPS, pp. 8527-8537, 2018.

[15] L. Jiang, Z. Zhou, T. Leung, L.-J. Li, and L. Fei-Fei, “MentorNet: Learning data-driven curriculum for very deep neural networks on corrupted labels,” in Proc. ICML, 2018.

[16] I. Jindal, M. Nokleby, and X. Chen, “Learning deep networks from noisy labels with dropout regularization,” in Proc. ICDM, 2016, pp. 967–972.

[17] I. Jindal, D. Pressel, B. Lester, and M. Nokleby, “An Effective Label Noise Model for DNN text Classification,” in Proc. NAACL, 2019.

[18] A. J. Bekker and J. Goldberger, “Training deep neural-networks based on unreliable labels,” in Proc. ICASSP, 2016, pp. 2682–2686.

[19] R. Johnson, T. Zhang, “Effective Use of Word Order for Text Categorization with Convolutional Neural Networks,” in Proc. The 2015 Annual Conference of the North American Chapter of the ACL, pp. 103-112, 2015.

[20] D. Kingma and M. Welling, “Auto-Encoding Variational Bayes,” in Proc. ICLR, 2014.

[21] Q. Le and T. Mikolov, "Distributed Representation of Sentences and Documents," in Proc. International Conference on Machine Learning, vol. 32, pp. 1188-1196, 2014.

[22] A.L. Maas, R.E. Daly, P.T. Pham, D. Huang, A.Y. Ng, and C. Potts, "Learning Word Vectors for Sentiment Anaysis," in Proc. ACL, vol. 1, pp. 142-150, 2011.

[23] E. Malach and S. Shalev-Shwartz, “Decoupling” when to update” from” how to update”,” in Proc. NeurIPS, 2017, pp. 960–970.

[24] C.D. Manning, P. Raghavan, and H. Schütze, Introduction to information retrieval, Cambridge University press, 2008.

[25] N. Manwani and P. Sastry, “Noise tolerance under risk minimization,” IEEE Transactions on Cybernetics, vol. 43, no. 3, pp. 1146-1151, 2013.

[26] V. Mnih and G. E. Hinton, “Learning to label aerial images from noisy data,” in Proc. ICML, pp. 567-574, 2012.

[27] G. Patrini, A. Rozza, A. Krishna Menon, R. Nock, and L. Qu, “Making deep neural networks robust to label noise: A loss correction approach,” in Proc. CVPR, 2017, pp. 1944–1952.

[28] J. Pennington, R. Socher, and C.D. Manning, "GloVe: Global Vectors for Word Representation," in Proc. Empirical Methods in Natural Language Processing (EMNLP), pp. 1532-1543, 2014.

[29] S. Reed, H. Lee, D. Anguelov, C. Szegedy, D. Erhan, and A. Rabinovich, “Training deep neural networks on noisy labels with bootstrapping,” in Proc. ICLR, 2015.

[30] D. T. Nguyen, C. K. Mummadi, T. P. N. Ngo, T. H. P. Nguyen, L. Beggel, and T. Brox, “Self: Learning to filter noisy labels with self-ensembling,” in Proc. ICLR, 2020.

[31] B. Van Rooyen, A, Menon, and B. Williamson, “Learning with symmetric label noise: The importance of being unhinged,” in Proc. NIPS, 2015.

[32] R. Wang, T. Liu, and D. Tao, “Multiclass learning with partially corrupted labels,” IEEE Transactions on Neural Networks and Learning Systems, vol. 29, no. 6, pp. 2568–2580, 2017.

[33] C. Wang, M. Zhang, S. Ma and L. Ru, “Automatic online news issue construction in web environment,” in Proc. World Wide Web, pp. 457–466, 2008.

[34] T. Xiao, T. Xia, Y. Yang, C. Huang, and X. Wang, “Learning from massive noisy labeled data for image classification,” in Proc. CVPR, pp. 2691-2699, 2015.

[35] Y. Yan, Z. Xu, I. W. Tsang, G. Long, and Y. Yang, “Robust semisupervised learning through label aggregation,” in Proc. AAAI, 2016.

[36] J. Yao, J. Wang, I. W. Tsang, Y. Zhang, J. Sun, C. Zhang, and R. Zhang, ”Deep learning from noisy image labels with quality embedding,” IEEE Transactions on Image Processing, vol. 28, no. 4, pp. 1909-1922, 2018.

[37] X. Yu, B. Han, J. Yao, G. Niu, I. W. Tsang, and M. Sugiyama, “How does disagreement help generalization against label corruption?” in Proc. ICML, 2019.

[38] Z. Zhang and M. Sabuncu, “Generalized cross entropy loss for training deep neural networks with noisy labels,” in Proc. NeurIPS, pp. 8778-8788, 2018.

[39] X. Zhang, J. Zhao and Y. LeCun, “Character-level convolutional networks for text classification”, in Proc. NeurIPS, pp. 649–657, 2015.