國立臺灣大學電機資訊學院電信工程學研究所

碩士論文

Graduate Institute of Communication Engineering

College of Electrical Enginnering and Computer Science

National Taiwan University

Master Thesis

以深層與卷積類神經網路建構聲學模型之大字彙連續語音辨識

Deep and Convolutional Neural Networks for Acoutic Modeling in Large Vocabulary Continuous Speech Recognition

周伯威

Po-Wei Chou

指導教授：李琳山 教授

Advisor: Lin-Shan Lee, Ph.D.

中華民國一百零四年二月

Feb, 2015

國立臺灣大學碩士學位論文
口試委員會審定書

以深層與卷積類神經網路建構聲學模型之大字彙連續語音辨識

Deep and Convolutional Neural Networks for Acoutic Modeling in Large Vocabulary Continuous Speech Recognition

本論文係周伯威君 (R01942135) 在國立臺灣大學電信工程學研究所完成之碩士學位論文，於民國 104 年 2 月 2 日 承下列考試委員審查通過及口試及格，特此證明

口試委員：

(簽名)

(指導教授)

系主任、所長 (簽名)

摘要

在語音辨識中，以深層類神經網路 (deep neural network, DNN) 取代傳統的高斯混合模型 (Gaussian mixture model, GMM) 來建構聲學模型 (acoustic model, AM) 的作法，因其優異的表現已逐漸成為主流。在本論文中，我們以深層類神經網路及卷積類神經網路 (convolutional neural network, CNN) 來產生隱藏式馬可夫模型 (hidden Markov model, HMM) 所需的狀態 (state) 機率，發展出大字彙連續語音辨識 (large-vocabulary continuous speech recognition, LVCSR) 中的聲學模型，在英文的評效語料 (benchmark corpus) 上進行了一系列的實驗。實驗結果顯示不論是深層類神經網路還是卷積類神經網路，其辨識準確率均能大幅地超越傳統基於高斯混合模型的作法，而其中又以深層類神經網路的表現最為出色。

由於不同語者的語音永遠是不一樣的，本文也探討了如何在深層類神經網路的聲學模型架構上，執行語者調適 (speaker adaptation) 以解決受測目標語者 (target speaker) 的語音與訓練語料 (training corpus) 之間不匹配 (mismatch) 的問題。透過對特徵空間上鑑別式線性迴歸 (feature-space discriminative linear regression, fDLR) 的改進，我們提出了一套將隱藏式馬可夫模型的狀態分群 (state-clustered) 的作法，更精細地考慮隱藏式馬可夫模型中各狀態不同的聲學結構，分群進行調適，並透過兩階段的方式進行辨識，提升目標語者的辨識準確度。在一系列的以 Facebook 個人動態 (status) 錄製而成的中英雙語 (bilingual) 語料的實驗中，可以發現不論是少量或是大量的調適語料，運用此方法建立的個人化 (personalized) 聲學模型皆能有相當良好的表現。

此外，我們也實作了一套透過圖形處理器 (graphics processing unit, GPU) 加速的深層類神經網路函式庫。文中除了介紹基本的使用說明以外，也詳細地記載了該程式的軟體架構與設計原理，並探討了圖形處理器上幾個重要的實作細節。

Contents

圖目錄

表目錄

第一章　導論

1.1　研究背景與動機

近年來由於網路產業的蓬勃發展，使得網路上可供瀏覽的多媒體內容如線上課程、影片等大幅地增加；而軟硬體科技的突飛猛進，也使得更快更小的智慧型手機與穿戴型裝置，能夠迅速地普及於人們的日常生活之中，並取代傳統的個人電腦，成為人們透過網路存取這些多媒體內容不可或缺的管道之一。此外，隨著科技不斷地進步，漸臻成熟的語音技術已經達到了一般大眾願意接受並使用的程度，許多基於多媒體內容的應用及行動裝置上的語音服務也因此孕育而生。不論是 Apple 前幾年所提出的個人語音助理 Siri ，讓使用者能夠透過自然的對話方式，搜尋網路上的各種資訊、查詢天氣或是設定鬧鈴及個人行程等，還是 Google 的語音地圖檢索及 Google Now 問答系統 (question answering system) 、 Facebook 的語音輔助輸入通訊軟體、以及線上影片的自動語音轉寫字幕等等，都大量地使用到語音與語言處理的各項技術。

今日所提到的自動語音辨識 (automatic speech recognition, ASR) 大多是指大字彙連續語音辨識 (large-vocabulary continuous speech recognition, LVCSR) ，也就是本論文的主題。由於許多語音技術與相關的應用，都圍繞著大字彙語音辨識，以其為基礎做進一步的延伸，也因此，大字彙連續語音辨識可以說是扮演了一個相當重要的角色。這些相關應用包括了語音輸入 (speech input) 、自動語音轉寫 (automatic transcription) 、語音文件摘要 (spoken document summarization) 、數位語音內容檢索 (spoken content retrieval) 、口述語彙偵測 (spoken term detection) 、問答系統 (question answering system) 、電腦輔助語言學習 (computer-assisted language learning) 等等。因此，如何有效地提高大字彙連續語音辨識系統的準確率、改善

系統的效能，便是當前最熱門的主題之一。

早期的自動語音辨識系統多半是透過隱藏式馬可夫模型 (hidden Markov model, HMM) [3] [4] [5] [6] 以及高斯混合模型 (Gaussian mixture model, GMM) [7] 為聲學訊號建模而成；雖然早在九零年代時，就有學者成功地利用簡單的類神經網路，搭配隱藏式馬可夫模型建立初步的聲學模型 [8] [9] 。然而受限於早年的時空背景，軟體、硬體、及訓練語料等各項資源的不足，科學家遲遲無法在大量的資料上，發展出效能足以大幅超越高斯混合模型的類神經網路聲學模型。

近年來，由於機器學習 (machine learning) 演算法 [10] 的大幅進展及電腦硬體科技如分散式運算 (distributed computing) 與圖形處理器 (graphics processing unit, GPU) [11] 的普及，使得更深更寬的深層類神經網路 (deep neural networks, DNN)，得以透過更有效率的方式進行訓練。有相當多的研究與實驗顯示，以深層類神經網路建立聲學模型的大字彙連續語音辨識系統，在一系列語料庫上都能有大幅超越傳統高斯混合模型的表現 [12] [13] [14] [15] 。 此外，由於卷積類神經網路 (convolutional neural network, CNN) [2] 在影像辨識上優異的表現 [16] [17] [18] ，也開始有人將其用在自動語音辨識上，並取得了相當不錯的成績 [19] 。因此，類神經網路聲學模型漸漸地受到重視且成為主流，而如何在這上面進行語者調適 (speaker adaptation) ，解決受測目標語者 (target speaker) 與訓練語料 (training corpus) 之間不匹配 (mismatch) 的問題，是其中一個相當值得研究的議題。

本論文便以此為出發點，詳細地探討深層類神經網路與卷積類神經網路是如何取代傳統的高斯混合模型，在大字彙連續語音辨識系統中建立聲學模型，提升系統的辨識率。同時，我們也實作了一整套的深層學習函式庫，並以此作為訓練深層類神經網路與卷積類神經網路的工具，進行了一系列的實驗。 最後，我們也根據特徵空間上鑑別式線性迴歸 (feature-space discriminative linear regression,

fDLR) [20] ，提出了一套在深層類神經網路上基於狀態分群的語者調適演算法，改善目標語者的辨識率。

1.2 研究方向與貢獻

本論文的研究方向與貢獻，包含了以下幾點：

- 基於類神經網路的聲學模型 (Acoustic Modeling Based on Neural Networks)

  從前後文相關深層類神經網路隱藏式馬可夫模型 (context-dependent deep neural network hidden Markov model, CD-DNN-HMM) 出發，以實作的角度，仔細地探討在大字彙連續語音辨識中，深層類神經網路與卷積類神經網路是如何透過反向傳播演算法 [21] 進行訓練，並透過一系列的實驗、分析與討論，比較了各種類神經網路架構的優劣及其對系統效能的影響。

- 深層類神經網路語者調適 (Speaker Adaptation on Deep Neural Networks)

  本論文提出了一個改進傳統的特徵空間上鑑別式線性迴歸的語者調適作法。我們透過 K 平均分群演算法 (k-means clustering) ，把那些在特徵空間上較為相近的隱藏式馬可夫模型狀態分成一群，並以鑑別式訓練 (discriminative training) 的方式將原本相近的狀態拉開，達到提升目標語者辨識率的作用。

- 深層學習函式庫與工具 (Deep Learning Library and Toolkit)

  本論文也詳細地記載了研究過程中所開發的深層學習函式庫的軟體架構、記憶體使用與佈局 (memory usage and layout) 、類神經網路訓練過程中各步驟的實作方式、還有在圖形處理器上進行平行運算時會遇到的各種問題、以及相對應的解決方案和優化方式等。

1.3 章節安排

本論文之章節安排如下：

- 第二章：介紹本論文相關背景知識。
- 第三章：介紹如何以深層類神經網路改善聲學模型。
- 第四章：介紹如何以卷積類神經網路改善聲學模型。
- 第五章：介紹如何在基於深層類神經網路的聲學模型上進行語者調適。
- 第六章：介紹深層類神經網路函式庫與實作。
- 第七章：本論文之結論與未來研究方向。

第二章　背景知識

本章將回顧大字彙連續語音辨識中最基本的聲學模型，包括了如何透過隱藏式馬可夫模型模擬不同語音特徵狀態之間的轉移機率與如何利用高斯混合模型描述聲學特徵在向量空間上的分佈。 除此之外，我們也會介紹最基本的類神經網路模型及其運作原理。

2.1　自動語音辨識

圖 2.1 是大字彙連續語音辨識系統之流程圖。輸入的聲音訊號經過一連串的前端訊號處理 (front-end signal processing) 後，便會通過圖中紅色區塊的解碼器 (decoder) ，利用預先建立好的聲學模型 (acoustic model) 、語言模型 (language model) 、辭典 (lexicon) 以及搜尋演算法進行解碼，以取得最佳詞序列 (optimal word sequence) w^* 作為最終的辨識結果。

$$w^* = \underset{w}{\text{argmax}}\, p(w|\mathbf{x}) = \underset{w}{\text{argmax}}\, \frac{p(\mathbf{x}|w)p(w)}{p(\mathbf{x})}, \tag{2.1}$$

其中 $\mathbf{x}$ 是經過前端訊號處理後的聲學特徵， $p(w)$ 是語言模型機率、 $p(\mathbf{x}|w)$ 是聲學模型機率。以下將會分別就聲學模型、語言模型以及辭典依序簡單地介紹。

2.1.1　聲學模型

音位 (phoneme) 是一個語言中能夠區別意義的最小單位，而音素 (phone) 則是人們於說話時音位的發音呈現方式。由於說話時相同的音位可以有千變萬化的發音呈現方式，因此在語音辨識系統中，如何透過模型描述這些千變萬化的發音，便是

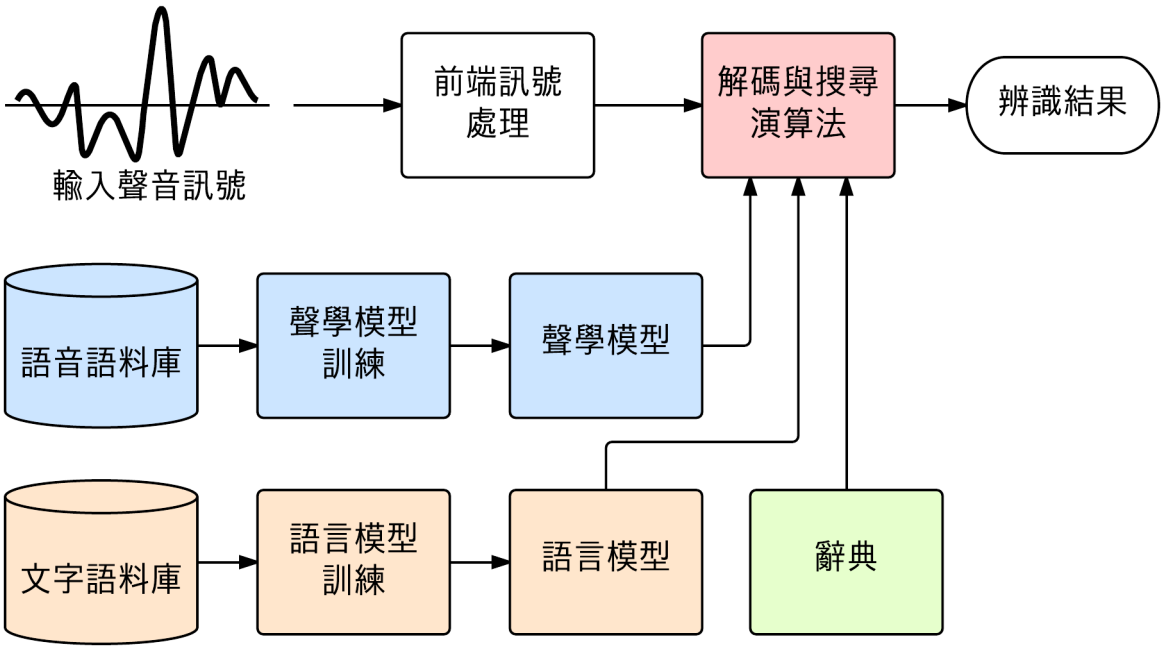


圖 2.1: 大字彙連續語音辨識系統之流程圖。圖中藍色的部份屬於聲學模型，橘色的部份屬於語言模型。

一個相當重要的問題，而常見的作法是透過隱藏式馬可夫模型及高斯混合模型來描述。

隱藏式馬可夫模型

在隱藏式馬可夫模型中，除了有一組用來描述狀態與狀態間相互轉移的轉移機率 (transition probability) ，每一個狀態中也會用一組輸出機率 (output probabilities) 用來描述可能的輸出結果。在語音辨識中，一個隱藏式馬可夫模型會有若干個不等的狀態數，其中包含了至少兩個特殊的狀態：起始狀態 (initial state) 和中止狀態 (exit state) ，如圖 2.2 所示。 一般來說，隱藏式馬可夫模型中的一個狀態會對應到聲音訊號中的一個音框 (frame) 。 因此，一組觀察到的聲學特徵向量序列對一個

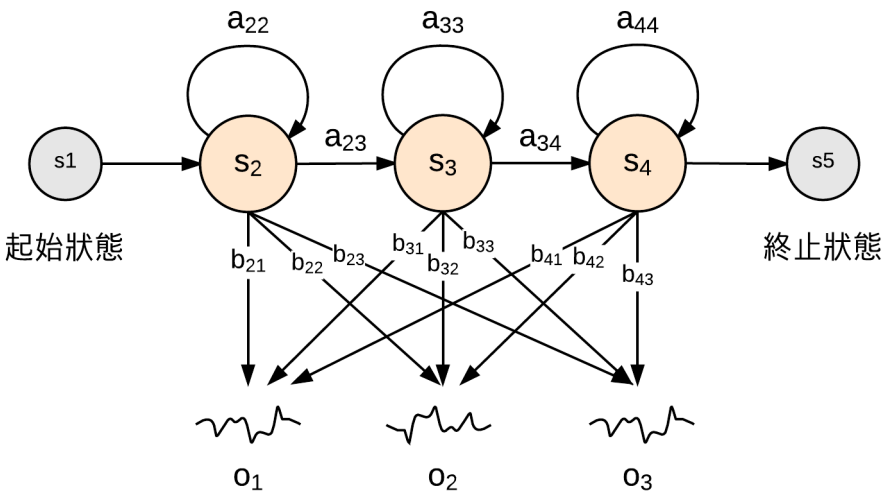


圖 2.2: 隱藏式馬可夫模型示意圖。

隱藏式馬可夫模型 $\boldsymbol{\lambda}$ 的相似度 (likelihood) 可以表示成：

$$P(\mathbf{x}|\boldsymbol{\lambda}) = \sum_{\mathbf{q}} \pi_{q_1} p(\mathbf{x}_1|q_1) \prod_{t=2}^{T} a_{q_{t-1},q_t} p(\mathbf{x}_t|q_t) \tag{2.2}$$

其中 t 代表時間， $\mathbf{x}$ 是觀察到的聲學特徵向量序列，由 $\mathbf{x}_1$ 到 $\mathbf{x}_T$ 共 T 個音框的特徵向量所組成。$\mathbf{q}$ 是一個可能的狀態序列， q_t 是該狀態序列在時間 t 時的狀態索引值 (state index) ， π_{q_1} 是起始狀態為 q_1 的機率，a_{q_{t-1},q_t} 是從狀態 q_{t-1} 到狀態 q_t 的轉移機率，而 $P(\mathbf{x}_t|q_t)$ 則是給定一個狀態 q_t ，觀測到特徵向量 $\mathbf{x}_t$ 的機率，或稱作特徵向量對於狀態的相似度，一般是透過高斯混合模型來描述。

高斯混合模型

給定一個隱藏式馬可夫模型中的狀態，假設觀察到的聲音訊號在特徵空間上的機率分佈，可以由 K 個 N 維空間中的高斯分布 (Gaussian distribution) 所組成的高斯混合模型 (Gaussian mixture model, GMM) 來描述，則特徵向量對於每一個狀態的

相似度可以表示為：

$$P(\mathbf{x}_t|q) = \sum_{k=1}^{K} \frac{w_k}{(2\pi)^{N/2}|\mathbf{\Sigma}_k|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}_t-\boldsymbol{\mu}_k)^T\mathbf{\Sigma}_k^{-1}(\mathbf{x}_t-\boldsymbol{\mu}_k)} \quad (2.3)$$

其中 w_k 、 $\boldsymbol{\mu}_k$ 、 $\mathbf{\Sigma}_k$ 分別是第 k 個高斯混合 (Gaussian mixture) 的比重 (weighting) 、平均向量 (mean vector) 及共變異矩陣 (covariance matrix) 。 常見的作法是透過「期望值最大化演算法」 (Expectation-Maximization algorithm, EM algorithm) [22] 及迭代 (iterative) 的方式調整高斯混合模型的隱藏式馬可夫模型中的參數，最大化語料與隱藏式馬可夫模型的相似度，以訓練出我們想要的聲學模型。

由於一個音素的發音，往往會受到前後音素的影響，現行的語音辨識系統多半已不再使用傳統的單音素 (monophone) 聲學模型，改採用前後文相關音素 (context-depedent phone) 聲學模型，譬如說三連音 (triphone) 。 以中文為例，常見的音素有 35 個，例如： CH_b （ㄅ）、 CH_p （ㄆ）、 CH_m （ㄇ）等等。 在考慮前後文音素的情況下，音素集合 (phone set) 的大小將會大幅擴展，理論上此集合內可能存在的三連音數量，可達原本單音素數量的三次方之多。然而考慮到有些三連音組合根本不存在或是無法發音，或是有些因為訓練語料太少訓練不起來，實際的三連音數量大概在數千左右。 雖然考慮前後文音素的影響，可以更多更細微的變化，但由於傳統上訓練資料的不足，當我們從單音素的音素集合，拓展到前後文相關音素的音素集合時，龐大的分類目標將會導致訓練資料被過度地分散，模型中的參數也會更加難以估計。 因此，常見的做法是透過決策樹 (decision tree) [23] 及狀態分享 (tied-state) [24] ，將語音特性相似的三連音以共享參數的方式合併。

2.1.2 辭典

辭典 (lexicon) 或發音辭典是由語言學家所共同制定的、用以描述一個語言中各字詞該如何發音的一套規則，例如教育部重編辭典修訂本 [25]。發音辭典中的每一條規則都是一個字或詞的發音方式，以音素序列 (phone sequence) 的方式呈現。而對於那些沒有出現在辭典內的辭典外字彙 (out-of-vocabulary, OOV) ，我們也可以透過發音辭典內的規則，以及對訓練文字語料 (text corpus) 的分析，額外地為這些辭典外字彙新增發音規則。在語音辨識的過程中，這些規範了語言中音素是否能夠前後相接的規則，便會被用來建立辭典網路 (lexicon net) ，作為語音辨識中的解碼演算法的搜尋依據。

2.1.3 語言模型

語言模型 (language model, LM) 是用來描述一個語言中各字詞出現的可能性的機率模型。給定一個詞序列 $w = (w_t)_{t=1}^{T}$ ，語言模型的目標便是計算出該詞序列出現的機率 $p(w)$

$$p(w) = p(w_1, w_2, \cdots w_n) \tag{2.4a}$$

$$= \prod_{i=1}^{n} p(w_i | w_1, w_2, \cdots w_{i-1}) \tag{2.4b}$$

其中 $p(w_i|w_1, w_2, \cdots w_{i-1})$ 可以透過統計的方式，從訓練文字語料 (text corpora) 中估計得出。然而上式這樣的作法，會使得我們無法從有限的訓練文字語料中精確地估計出所有可能的機率 $p(w_i|w_1, w_2, \cdots w_{i-1})$ ，因此常見的作法是使用 N 連

(N-gram) 語言模型。假設一個字或詞出現的機率只受到前 N 個字或詞的影響

$$p(w_i|w_1, w_2, \cdots w_{i-1}) = p(w_i|w_{i-N}, w_{i-N+1}, \cdots w_{i-1}), \tag{2.5}$$

如此一來便可以將式 2.4b 中語言模型對於詞序列的機率估計 $p(w)$ 簡化成以下的形式，進而大幅減少我們需要估計的參數

$$p(w) = p(w_1)p(w_2|w_1)\cdots \prod_{i=N+1}^{n} p(w_i|w_{i-1}, w_{i-2}, ...w_{i-N}). \tag{2.6}$$

除了 N 連語言模型以外，另一種常見的作法則是透過遞迴式類神經網路 (recurrent neural network, RNN) 來描述。 [26] [27]

2.2 類神經網路

類神經網路 (Neural Networks, NN) 是一種模仿生物神經網路的結構與功能的數學模型。在本節中，我們將會詳細介紹類神經網路的數學模型及其運作原理。

2.2.1 順向傳遞式類神經網路

順向傳遞式類神經網路 (Feedforward Neural Network) 是眾多類神經網路 (Artificial Neural Network, ANN) 中最簡單的一種，如圖 2.3 所示。這種類神經網路也因其層狀的架構常被稱為多層感知器 (Multilayer Perceptron, MLP) ，其中的每一層都是由相當多的感知器 (perceptron) 所組成。 一般來說，我們用「深度」 (depth) 代表多層感知器的層數，用「寬度」 (width) 代表每一層中感知器的數量。一個多層感知器中，至少包含了一個輸入層 (input layer) 、一個輸出層 (output layer) 、以及一至

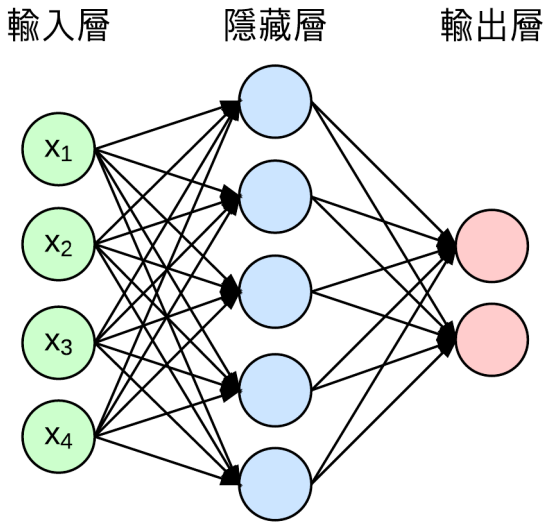


圖 2.3: 類神經網路示意圖

多層的隱藏層 (hidden layer) 。

感知器是二元分類器 (binary classifier) 中的一種（見圖 2.4 ），它是由相當多的神經元 (neuron) 所組成，我們可以將感知器輸入與輸出的關係表示如下

$$y_j = \phi\left(\sum_{i=1}^{M} a_{ji}x_i + b_j\right), \quad j = 1, 2, ...N \tag{2.7}$$

其中 N 代表感知器的總數， M 代表輸入特徵向量 $\mathbf{x}$ 的維度， y_j 代表第 j 個感知器的輸出， a_{ji} 與 b_j 分別是該感知器對應到 x_i 的加權係數 (weighting) 以及偏移量 (bias) ， ϕ 則是活化函數 (activiation function) ，例如邏輯函數 (logistic function) 或雙曲正切函數 (hyperbolic tangent) 等（詳見附錄三） 。

我可以透過仿射轉換 (affine transformation) 的方式，用權重矩陣 $\mathbf{A} = \{a_{ji}\}$ 及偏移向量 $\mathbf{b} = [b_1, b_2, ..., b_N]^T$ 來描述多層感知器中層與層之間的變換關係

$$\mathbf{y} = \phi\left(\mathbf{A}\mathbf{x} + \mathbf{b}\right) \tag{2.8}$$

一般來說，我們會為 M 維的輸入特徵向量新增一個維度，並設 $x_{M+1} = 1$ ，

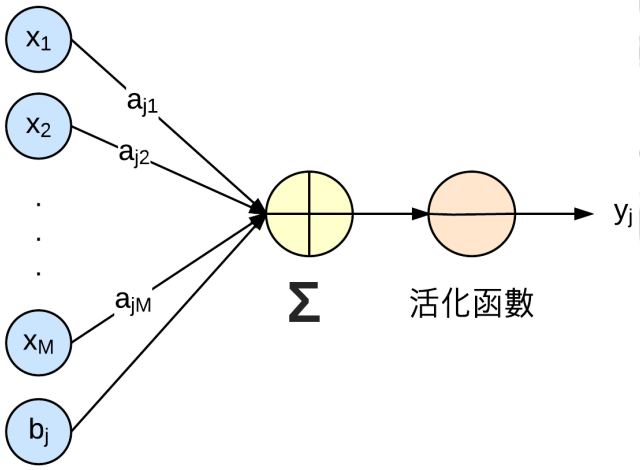


圖 2.4: 感知器示意圖

藉此將偏移量 b_j 視為第 j 個感知器中對應到 x_{M+1} 的加權係數，並利用增廣矩陣 (augmented matrix) 的概念，把偏移量 $\mathbf{b}$ 放進矩陣乘法運算中，將式2.7 重寫成

$$\begin{bmatrix} \mathbf{y} \\ 1 \end{bmatrix} = \phi\left(\begin{bmatrix} \mathbf{A} & \mathbf{b} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix}\right) \tag{2.9a}$$

$$= \phi\left(\mathbf{W}\begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix}\right), \tag{2.9b}$$

2.2.2 訓練類神經網路

假設一個多層感知器的輸出層寬度為 S ，並以向量 $\mathbf{o}$ 及 $\mathbf{t}$ 分別表示「模型的預測結果」與「我們期望模型預測出的結果」，則平方差 (squared error) 損失函數 (loss function) 定義為

$$E = \frac{1}{2}\sum_{i=1}^{S}(o_i - t_i)^2 \tag{2.10a}$$

$$= \frac{1}{2}(\mathbf{o} - \mathbf{t})^T(\mathbf{o} - \mathbf{t}) \tag{2.10b}$$

有了損失函數後，我們便可以透過反向傳播演算法 [21] ，求得損失函數對類神經網路模型中各項參數的偏微分，並透過梯度下降法 (gradient descent) 更新模型內的參數，以降低損失函數的值，如下所示

$$\boldsymbol{\theta}(i+1) = \boldsymbol{\theta}(i) + \Delta\boldsymbol{\theta}(i) \tag{2.11a}$$

$$\Delta\boldsymbol{\theta}(i) = -\epsilon \cdot \left.\frac{\partial E}{\partial \boldsymbol{\theta}}\right|_{\boldsymbol{\theta}=\boldsymbol{\theta}(i)} \tag{2.11b}$$

$$E(\boldsymbol{\theta}(i+1)) \leq E(\boldsymbol{\theta}(i)). \tag{2.11c}$$

其中 $\boldsymbol{\theta}$ 是類神經網路模型中所有可訓練參數的集合，包括了各層的權重矩陣 $\mathbf{A}$ 及偏移向量 $\mathbf{b}$ 等； $\boldsymbol{\theta}(i)$ 則是經過第 i 次迭代後的結果； ϵ 稱為學習率 (learning rate)，其值大約在 0.001 到 0.1 不等 [28] 。 一般來說，我們會在滿足式 2.11c 的情況下，傾向選擇一個較大的值，以加速類神經網路訓練的過程。 然而，要找出一個最適的 (optimal) 學習率，使得類神經網路的訓練可以在最短的時間內完成，並非一件容易的事。 太小學習率會導致損失函數下降緩慢；太大的學習率則會展現出如圖 2.5a 中之字型 (zigzag) 般的更新路徑，過與不及都會導致類神經網路的訓練時間增加。 因此，常見的作法是將式2.11b 中的 $\left.\frac{\partial E}{\partial \boldsymbol{\theta}}\right|_{\boldsymbol{\theta}=\boldsymbol{\theta}(i)}$ 視為物體速度，並引入慣量 (momentum) [21] 的概念，讓每次更新的方向 $\Delta\boldsymbol{\theta}(i)$ 都是前次更新方向 $\Delta\boldsymbol{\theta}(i-1)$ 和 $\left.\frac{\partial E}{\partial \boldsymbol{\theta}}\right|_{\boldsymbol{\theta}=\boldsymbol{\theta}(i)}$ 的線性組合，以避免在相鄰幾次的迭代中 $\dfrac{\partial E}{\partial \boldsymbol{\theta}}$ 差異過大。引入慣量後的梯度下降法可以寫成

$$\Delta\boldsymbol{\theta}(i) = \alpha \cdot \Delta\boldsymbol{\theta}(i-1) - \epsilon \left.\frac{\partial E}{\partial \boldsymbol{\theta}}\right|_{\boldsymbol{\theta}=\boldsymbol{\theta}(i)} \tag{2.12}$$

其中 α 稱作慣量係數，通常在 0.5 至 0.9 之間。

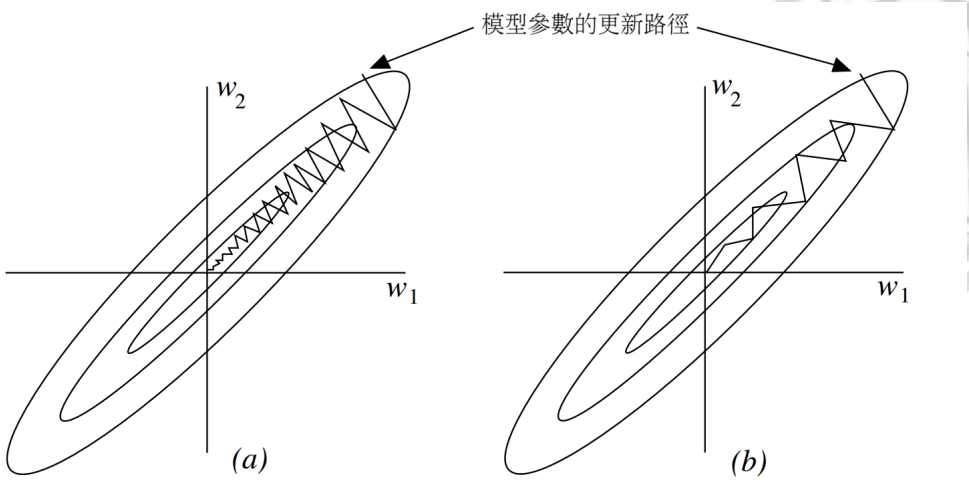


圖 2.5: 反向傳播演算法分別在 (a) 沒有慣量 (b) 有慣量 時的模型參數更新路徑示意圖。 [1]

2.3 類神經網路之正規化

正因為類神經網路中有非常多的可訓練參數，其高複雜度的模型常被用在諸多困難的工作上。 然而，這麼多的可訓練參數常常會導致模型出現過度貼合 (over-fitting) 的現象，使得該模型在新的測試資料中表現不盡理想。 因此，本節將探討如何透過類神經網路的正規化，來避免此一現象的發生。

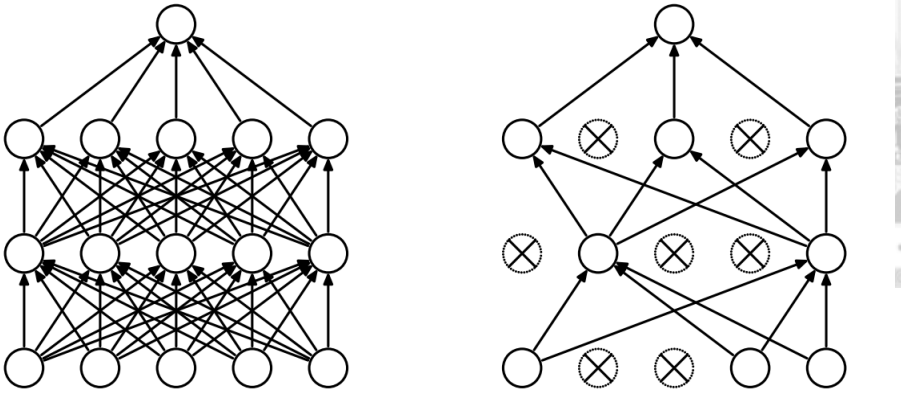
2.3.1 一次與二次正規化

一次與二次正規化 (L1 and L2 regularization) 是在原本的損失函數中， 加了一項 $\|\boldsymbol{\theta}\|_p$ 以作為新的衡量標準，如下所示：

$$E(\boldsymbol{\theta}) \leftarrow E(\boldsymbol{\theta}) + \lambda \|\boldsymbol{\theta}\|_p^p \tag{2.13a}$$

$$\|\boldsymbol{\theta}\|_p = \left(\sum_{\theta_i \in \boldsymbol{\theta}} |\theta_i|^p \right)^{\frac{1}{p}}, \quad p = 1, 2 \tag{2.13b}$$

其中 $\|\boldsymbol{\theta}\|_p$ 稱為 L^p 範數 (L^p norm) ， λ 則是一個介於 0 到 1 之間的係數。



(a) 標準的類神經網路模型　　(b) 使用了丟棄法的類神經網路模型

圖 2.6: 丟棄法正規化。左圖是標準的類神經網路模型，共有兩層隱藏層。右圖則是使用了丟棄法的類神經網路模型，以 $\times$ 代表暫時被丟棄的神經元。

當 $p = 1$ 時，此種作法稱為一次正規化。它會使得模型參數 $\boldsymbol{\theta}$ 中傾向出現較多的 0 ，有助於減少模型參數的數量並減緩過度貼合現象的發生，故常稱為稀疏解 (sparse solution) [29] [30] 。

當 $p = 2$ 時，此種作法稱為二次正規化。對於模型 $\boldsymbol{\theta}$ 中絕對值較大的參數，它傾向於給予較多的懲罰，避免其中的 θ_i 跑到太大或太小的值，有助於減少模型的複雜度 (model complexity) 以避免過度貼合現象的發生。由於類神經網路中，$\boldsymbol{\theta}$ 多半是層與層之間的權重矩陣 $\mathbf{W}$ ，故二次正規化也常被稱為權重衰減 (weight decay) 。

2.3.2 丟棄法

另一種常見的作法是丟棄法 (dropout) [31] [32] 。 在類神經網路的訓練過程中，每一次的順向傳遞都會隨機地關閉一部分的感知器，將其輸出值設為 0 ，並在進行反向傳播演算法時，將錯誤訊號沿著相反的方向，順著當時未被關閉的感知器，往回傳播以更新模型參數。這樣的作法等效上會使得這些未被關閉的感知器，在每一次的迭代過程中，隨機地組出一個寬度較窄且與前次完全不同的類神經網路

架構（如圖 2.6 所示）。 假設一個類神經網路中共有 K 個感知器，由於每一個感知器都有一定的機率會被關閉，因此所有可能出現的類神經網路架構便會高達 2^K 種之多。 當類神經網路透過丟棄法完成訓練，面對新的測試資料進行預測時， 所有感知器便會同時打開，此舉相當於同時使用了非常多的類神經網路模型進行預測， 而這些模型平均 (model averaging) 起來的結果，便能夠有效地降低過度貼合現象的發生 [33] [34] 。

2.4 本章總結

本章介紹了自動語音辨識的背景，包含了基礎的隱藏式馬可夫模型、高斯混合模型、及語音辨識的基本概念。並介紹了基本的類神經網路模型，以及如何透過正規化來避免過度貼合的現象發生。

第 三 章　深層類神經網路聲學模型

本章將敘述如何在大字彙連續語音辨識中，利用深層類神經網路取代傳統的高斯混合模型作為聲學模型，並透過一系列的實驗，探討不同的聲學特徵對辨識效能的影響。

3.1 以深層類神經網路取代高斯混合模型作為聲學模型

一般來說，深層類神經網路 (deep neural network, DNN) 是指具有一層以上隱藏層的多層感知器。近年來，由於深層類神經網路在諸多領域上的優異表現，以其作為聲學模型的研究也越來越多。其中最主流的作法是在不改動以隱藏式馬可夫模型為架構的情況下，透過貝式定理 (Bayes's theorem) ，將高斯混合模型取代成深層類神經網路，作為隱藏式馬可夫模型中用來描述可能的輸出結果之機率分佈。

在大字彙語音辨識尋找最佳的詞序列 w^* 的解碼過程中 (見式 2.1 及 2.2) ，聲學模型機率 $p(\mathbf{x}|w)$ 可以透過貝式定理，將本來透過高斯混合模型來描述的特徵向量對於狀態的相似度 $p(\mathbf{x}_t|q_t)$ 轉換成給定特徵向量的情況下，狀態的事後機率分佈 (state posterior probability) $p(q_t|\mathbf{x}_t)$ ，如下所示

$$p(\mathbf{x}_t|q_t) = \frac{p(q_t|\mathbf{x}_t)p(\mathbf{x}_t)}{p(q_t)}, \tag{3.1}$$

其中 $p(q_t)$ 是狀態的事前機率 (prior probability) ，可以透過計算各個狀態在訓練資料中出現的次數得出 [9] ，而 $p(q_t|\mathbf{x}_t)$ 則可以透過深層類神經網路優異的多元分類 (multiclass classification) 能力估計得出。

研究顯示 [14] ，這樣的作法在前後文相關音素隱藏式馬可夫模型 (CD-HMM) 上的表現，遠遠超過其在單音素上的表現。 因此，一般的作法是先訓練一個基於高斯混合模型的前後文相關音素隱藏式馬可夫模型 (CD-GMM-HMM) 做為基準 (baseline) ，並透過强制對齊 (force alignment) 的方式，將訓練語料中已經標記好的音素序列 (phone sequence) 轉成經過狀態共享的三連音狀態序列 (state sequence) $(q_t)_{t=1}^{T}$ ，以此作為訓練深層類神經網路所需要的目標標記 (target label) 。

建構深層類神經網路聲學模型時，一般會選用處理多元分類問題時最常見的平滑最大值 (softmax) 作為輸出層的活化函數

$$y_i = \frac{e^{x_i}}{\sum_{i=1}^{M} e^{x_i}}, \tag{3.2}$$

以其輸出值 y_i 作為事後機率分佈 $p(q_t|\mathbf{x}_t)$ ，並使用衡量兩個機率分佈之間的差異的交叉熵 (cross entropy) 作為損失函數

$$CE = -\sum_{i=1}^{M} d_i(t) \cdot \ln(p(q_i|\mathbf{x}_t)), \tag{3.3}$$

其中 $d_i(t)$ 是透過前述狀態序列 $(q_t)_{t=1}^{T}$ 產生的目標機率分佈 (target probability distribution) ，只有在被標記的狀態上才有值為 1 ，其餘狀態皆為 0

$$d_i(t) = \begin{cases} 1, & \text{if } i = q_t \\ 0, & \text{otherwise} \end{cases} \tag{3.4}$$

搭配平滑最大化函數的交叉熵損失函數相較於使用邏輯函數的最小平方差損失函數，有以下幾個優點及原因。首先，平滑最大值活化函數不但可以保證輸

出結果 y_i 為一個機率分佈，它本身也可以視為邏輯迴歸 (logistic regression) 從二元結果 (outcome) 推廣到多元結果的一般化形式。其次，由於輸出結果與目標都是機率分佈，以交叉熵的形式衡量兩者之間的差異相較於使用最小平方差的作法，明顯較為合理且具有意義。此外，由於目標機率分佈只有在被標記的狀態上才有非零值 1 ，因此使用交叉熵當作損失函數的作法便相當於最大相似度估計 (maximum likelihood estimation, MLE) 。最後，這種作法也可以避免深層神經網路中最後一層隱藏層與輸出層之間的權重矩陣，在透過反向傳播演算法進行更新時，因為邏輯函數的偏微分值在零點兩側過小，導致學習速率緩慢的狀況的發生。

有了損失函數後，便可以透過反向傳播演算法，對深層類神經網路進行訓練。 有時候也會利用訓練完的深層類神經網路，重新對音素序列進行强制對齊，以求得更好的狀態序列作為訓練目標，進行第二輪的深層類神經網路訓練。上述這一整套的作法，通常被稱作前後文相關深層類神經網路隱藏式馬可夫模型 (CD-DNN-HMM) 。

傳統上我們為了簡化「期望值最大化演算法」中的運算，會假設輸入特徵向量 $\mathbf{x}_t$ 中的各個維度均互相獨立 (independent) ，以便將高斯混合模型中的共變異矩陣簡化成對角線 (diagonal) 矩陣。然而深層類神經網路卻不受此一限制，這使得我們得以選用各式各樣的特徵向量當作深層類神經網路的輸入，例如串接 (concatenate) 前後各 L 個與 R 個音框的梅爾倒頻譜係數 (mel-frequency cepstral coefficient, MFCC) ，或是串接其他語音特徵如音高 (pitch) 或是身份向量 (i-vector) [35] 的輸入特徵向量等。

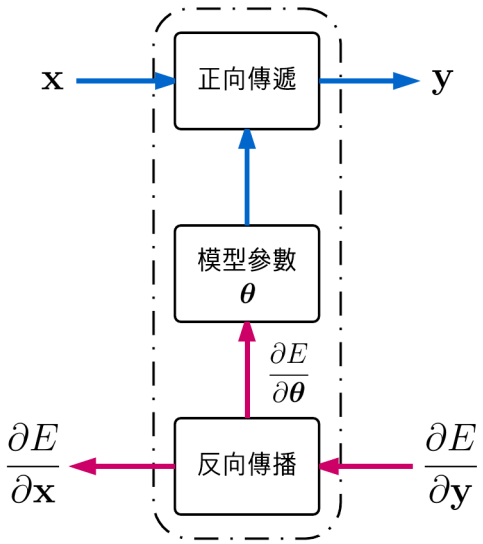


圖 3.1: 反向傳播演算法示意圖。虛線框是一個抽象的介面，框內代表一個特徵轉換。以箭頭的方向代表系統的輸入與輸出，藍色的箭頭是順向傳遞的路徑，紅色的箭頭則是反向傳播的路徑。

3.1.1 反向傳播演算法

在訓練深層類神經網路時，由於層數比起傳統的多層感知器來得要多，因此我們需要一套有系統的方式逐步求出 $\frac{\partial E}{\partial \boldsymbol{\theta}}$ 。

首先，我們定義了一個抽象的介面，如圖 3.1 所示。 虛線框內代表一個將特徵向量 $\mathbf{x} \in \mathbb{R}^M$ 轉換成 $\mathbf{y} \in \mathbb{R}^N$ 的特徵轉換 $f(\boldsymbol{\theta})\colon \mathbb{R}^M \to \mathbb{R}^N$ ，其中 $\boldsymbol{\theta}$ 是特徵轉換內的模型參數。 這樣的一個特徵轉換可以是前述 2.2.1 小節中的仿射轉換，也可以是活化函數如邏輯函數或平滑最大值等。

接著引進兩個很重要的中間變數：前級錯誤訊號 $\frac{\partial E}{\partial \mathbf{x}}$ 與後級錯誤訊號 $\frac{\partial E}{\partial \mathbf{y}}$ 。可以從圖 3.1 中清楚得看到，通過虛線框所定義的介面的箭頭中， $\mathbf{x}$ 及 $\frac{\partial E}{\partial \mathbf{y}}$ 是唯二的輸入。 換言之，我們可以將 $\frac{\partial E}{\partial \mathbf{x}}$ 及 $\frac{\partial E}{\partial \boldsymbol{\theta}}$ 寫成完全由 $\boldsymbol{\theta}$ 、 $\mathbf{x}$ 、及 $\frac{\partial E}{\partial \mathbf{y}}$ 所決定的函數，並輕易地透過連鎖律求得（見附錄一）。

有了上述的概念後，我們便可以將看似複雜的深層類神經網路，想像成一連串特徵轉換函數所構成的複合函數 (function composition) ，並透過以下的步驟，

進行反向傳播演算法：

1. 根據損失函數 $E(\mathbf{y})$ 計算出 $\dfrac{\partial E}{\partial \mathbf{y}}$，當作最後一層特徵轉換的後級錯誤訊號。
2. 根據 $\mathbf{x}$ 、 $\boldsymbol{\theta}(i)$ 、及 $\left.\dfrac{\partial E}{\partial \mathbf{y}}\right|_{\boldsymbol{\theta}=\boldsymbol{\theta}(i)}$，計算出 $\left.\dfrac{\partial E}{\partial \boldsymbol{\theta}}\right|_{\boldsymbol{\theta}=\boldsymbol{\theta}(i)}$，並透過式 2.11b 將模型參數更新成 $\boldsymbol{\theta}(i+1)$ 。
3. 根據 $\mathbf{x}$ 、 $\boldsymbol{\theta}(i)$ 、及 $\left.\dfrac{\partial E}{\partial \mathbf{y}}\right|_{\boldsymbol{\theta}=\boldsymbol{\theta}(i)}$，計算出前級錯誤訊號 $\left.\dfrac{\partial E}{\partial \mathbf{x}}\right|_{\boldsymbol{\theta}=\boldsymbol{\theta}(i)}$ 並往前傳遞，當作前一層特徵轉換的後級錯誤訊號。
4. 重複步驟 2. 及步驟 3. ，直到所有模型參數都被更新為止。

這樣的作法有幾個優點，其一是我們可以在程式的實作中，定義一個如圖 3.1 中虛線框所示的介面類別 (interface class) ，並規範一系列的方法 (method) ，以遵守物件導向程式設計 (object-oriented programming, OOP) 的設計守則 [36] ；此外，不論哪種的類神經網路架構，均可以透過這樣的方式執行反向傳播演算法，更新模型內的參數。例如遞迴式類神經網路 [37] 及長短期記憶神經網路 (long short-term memory, LSTM) [38] 等。 以平滑最大值作為活化函數、交叉熵作為損失函數的深層類神經網路聲學模型，透過反向傳播法的完整訓練流程如圖 3.2 所示。

3.2 實驗與分析

本節將敘述本論文中的實驗環境，包括實驗語料、訓練及辨識工具、前端處理及各項參數設定等，並就實驗的結果進行討論與分析。

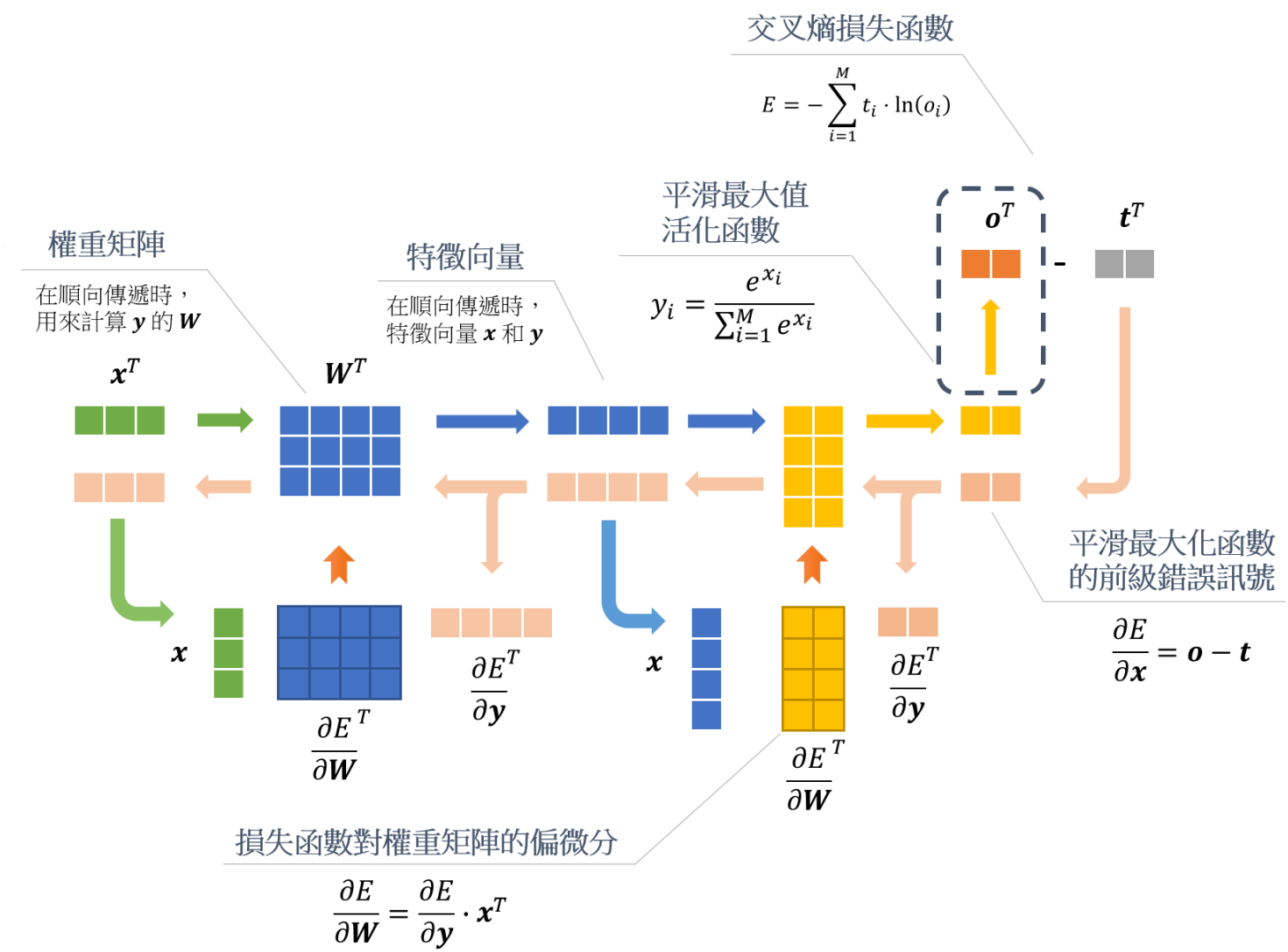


圖 3.2: 使用反向傳播演算法訓練深層類神經網路之示意圖。 圖中的矩陣與向量是由很多小方塊所組成，每個小方塊代表一個值。 深藍色的虛線框代表平滑最大值活化函數。 橘紅色和灰色方塊分別代表模型預測的結果與我們期望模型輸出的結果，以轉置向量 $\mathbf{o}^T$ 、 $\mathbf{t}^T$ 表示。 粉紅色的箭頭代表反向傳播演算法的傳播路徑。 橘紅色的箭頭代表模型參數 $\mathbf{W}$ 是透過式 2.11b 所描述的方式（梯度下降法）進行更新。

	總句數（句）	總時間長度（分鐘）
訓練語料	3696	189
調適語料	400	34
測試語料	192	9

表 3.1: 語料時間長度與句數

3.2.1 實驗設定

實驗語料

本實驗中，我們選用 TIMIT 作為語音辨識的語料。TIMIT 是一套由德州儀器 (Texas Instruments, TI) 及美國麻省理工學院 (Massachusetts Institute of Technology, MIT) 共同錄製的全英文語料。錄音語者有男女生共 630 位，其中包含了來自美國不同地區的方言（見附錄表 7.1 ）。 在 TIMIT 官方的文件中，也詳細地規定了一份的核心測試語料 (core test set) 供大家參考與比較，而訓練語料 (training corpus) 及調適語料 (development corpus) 的部份則如表 3.1 所示。

訓練與辨識工具

本實驗中，我們整合並使用了 2 套訓練及辨識工具，分別是美國約翰·霍普金斯大學 (Johns Hopkins University, US) 的 KALDI、以及我們自己開發的一套深層卷積類神經網路工具 **libdnn** 。前者負責了前端聲學特徵訊號處理、資料貯存、訓練基於高斯混合模型的前後文相關音素隱藏式馬可夫模型、以及基於加權有限狀態轉換器的解碼器；後者則負責了深層類神經網路的訓練與預測。其餘的程式均係由台大語音實驗室所提供。

前端處理

本論文中所使用的聲學特徵有兩種，一種是梅爾倒頻譜係數，另一種則是梅爾濾

波器組 (mel filter banks) ，記作 fbank 。 梅爾倒頻譜係數是以標準的作法取得：每平移 10 微秒即以 25 微秒的漢名窗 (Hamming window) 取值作為一個音框，通過 18 維的梅爾濾波器組，再透過離散餘弦轉換 (discrete cosine transform) 降至 13 維，最後再以串接的方式，搭配上對時間的一階導數及二階導數所組成的 39 維向量 [39]； 而梅爾濾波器組則是使用與上述相同的漢名窗，通過 23 維的梅爾濾波器組，配上對時間的一階導數及二階導數所形成的 69 維向量。

基準實驗與聲學模型設定

本實驗中的基準實驗是以基於高斯混合模型的前後文相關音素隱藏式馬可夫模型作為架構，其中隱藏式馬可夫模型共包含了 2425 個前後文相關狀態 (context-dependent state) ，高斯混合模型中的高斯機率分佈之總數量則是從一開始的 2425 個，經過 36 次的最大相似度的重新估計，漸漸增加至 12155 個。 我們選用了兩個深度為 4 層、寬度為 2048 、分別使用串接前後各 4 個音框、橫跨 9 個音框的梅爾濾波器組與梅爾倒頻譜係數的深層類神經網路作為聲學模型。

在訓練深層類神經網路的過程中，我們使用梯度坡降法的變形 — 批次隨機坡降法 (mini-batch stochastic gradient descent) — 作為更新模型參數的方法，並以 256 筆資料作為一個批次 (mini-batch) 進行迭代。

本實驗中，我們選用了 0.008 作為初始的學習率，往後每看過一次完整的訓練資料，就會根據該模型在調適語料上的表現決定下一次的學習率。一般是以狀態正確率或是音框正確率 (frame accuracy) 作為初步衡量模型效能的標準。若在調適語料上的音框正確率相較於前一次的進步量小於 0.5% ，便會在往後每看完一次完整的訓練資料時將學習率減半，直到調適語料上的音框正確率進步量小於 0.1% 或是達到最大訓練次數上限 35 為止。

聲學模型架構	音素錯誤率 (%)
高斯混合模型基準實驗 (MFCC)	28.29
深層類神經網路 (fbank)	23.41
深層類神經網路 (MFCC)	22.74

表 3.2: 在 TIMIT 核心測試語料上的辨識結果。以音素錯誤率作為衡量標準。

丟棄率 (%)	深層類神經網路 (fbank)	深層類神經網路 (MFCC)
0%	23.41	22.74
10%	22.80	22.69
20%	22.91	22.66
30%	**22.73**	**22.63**
40%	23.23	23.52
50%	23.77	23.99

表 3.3: 使用丟棄法正規化進行深層類神經網路訓練，在 TIMIT 核心測試語料上的辨識結果。以音素錯誤率作為衡量標準。

3.2.2 實驗結果與分析

表 3.2 是各深層類神經網路聲學模型在 TIMIT 核心測試語料上的實驗結果，以音素錯誤率 (phone error rate, PER) 作為衡量辨識率的標準。

在初步的實驗中，我們發現以深層類神經網路作為聲學模型的作法，其辨識效能很明顯地優於傳統基於高斯混合模型的作法，音素錯誤率下降了 20% 之多 (見表 3.2)。此外，兩種不同的聲學特徵都有相當不錯的表現，唯以梅爾濾波器組作為輸入特徵向量的深層類神經網路，其效能比起使用傳統作為聲學特徵向量的梅爾倒頻譜係數顯得略為遜色。

我們也針對丟棄法正規化，做了一系列的實驗。 在表 3.3 中，不論是以梅爾濾波器組還是梅爾倒頻譜係數作為輸入的深層類神經網路，在丟棄率小於 30% 的情況下，大致上都能有不錯的進步。而當丟棄率進一步提高到 40% 至 50% 時，兩者的效能便開始出現退步的情形，其中尤以使用梅爾倒頻譜係數的深層類神經網路最為嚴重。 由於越寬的深層類神經網路擁有較高的模型複雜度 (model complexity) ，反之則越低，因此我們推測是在深層類神經網路每一層不夠寬的情

況下，若是大幅地提高丟棄率，則會對深層類神經網路造成破壞性的影響。以我們實驗中選用寬度為 2048 的深層類神經網路架構為例，當丟棄率提高到 50% 時便出現效能上的退步，所以我們推論在 TIMIT 中，像 1024 這樣的寬度並不足以模擬前後文相關音素隱藏式馬可夫模型中，各狀態與輸入聲學特徵間高度複雜的關係。因此，在使用丟棄法正規化時，必須慎選丟棄率以避免過度丟棄造成以上現象的發生。

3.3 本章總結

本章介紹了如何使用深層類神經網路取代傳統的高斯混合模型作為聲學模型，並透過一系列在 TIMIT 上的實驗，發現使用梅爾濾波器組與梅爾倒頻譜係數作為輸入的深層類神經網路，其辨識效能明顯優於傳統上基於高斯混合模型的聲學模型，進而驗證了此作法的可行性。

第 四 章　卷積類神經網路聲學模型

4.1　卷積類神經網路

卷積類神經網路 (convolutional neural network) 是影像辨識 (image recognition) 中常用的一種類神經網路架構，最早被用來處理手寫識別 (handwriting recognition, HWR) 及光學字元識別 (optical character recognition, OCR) 等相關問題 [40] [2] 。近年來，隨著深層學習的興起，也有越來越多的研究團隊，將語音辨識中常用的聲學特徵訊號，如梅爾濾波器組或是頻譜圖 (spectrogram) 當作一張張的影像，利用卷積類神經網路替代深層類神經網路作為聲學模型 [19] [41] 。

4.1.1　簡介

卷積類神經網路顧名思義就是一種使用卷積 (convolution) 運算的類神經網路架構。 有別於傳統的卷積運算，在卷積類神經網路中，我們事先並不知道卷積運算中的脈衝響應 (impulse response) 長什麼樣子，而是透過大量的資料學出這些脈衝響應。 在卷積類神經網路中，除了會使用到傳統多層感知器中的仿射轉換及活化函數以外，還另外定義了兩種新的特徵轉換：卷積層 (convolutional layer) 及減縮取樣層 (subsampling layer) ，我們將在下面的章節中一一介紹。

4.1.2　卷積層 (Convolutional Layer)

在卷積類神經網路中，通常使用特徵圖 (feature map) 一詞來代表一張二維的影像。 如圖 4.1 所示， $\mathbf{X}$ 是一張高為 H_x 、寬為 W_x 的輸入特徵圖， $\mathbf{K}$ 是一個高 H_k 、寬 W_k 的卷積核心 (convolutional kernel) 。輸入特徵圖 $\mathbf{X}$ 通過卷積核心 $\mathbf{K}$

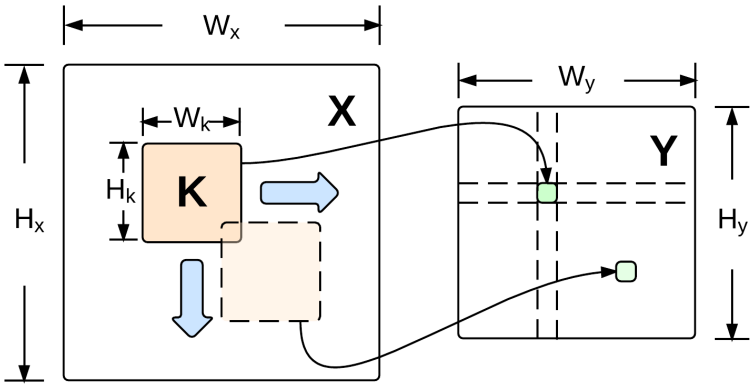


圖 4.1: 卷積類神經網路中，卷積運算的示意圖。

後，所得到的輸出特徵圖 $\mathbf{Y}$ ，可以表示成

$$\mathbf{Y} = \mathbf{X} * \mathbf{K} \tag{4.1}$$

其中 $*$ 是在離散 (discrete) 域上的二維卷積運算，可以定義成

$$\mathbf{Y}[a,b] \stackrel{\text{def}}{=} \sum_{m=0}^{H_k-1} \sum_{n=0}^{W_k-1} \mathbf{X}[a+m, b+n] \cdot \mathbf{K}[H_k-1-m, W_k-1-n] \tag{4.2}$$

其中 $0 \le a \le H_y - 1$ 且 $0 \le b \le W_y - 1$ ， H_y 和 W_y 分別是輸出特徵圖 $\mathbf{Y}$ 的高和寬。 由於卷積核心在 $\mathbf{X}$ 上滑動進行卷積運算時，其覆蓋範圍不得超出 $\mathbf{X}$ 所定義的 $H_x \times W_x$ 範圍之外，因此透過卷積運算得到的輸出特徵圖 $\mathbf{Y}$ ，其大小 $H_y \times W_y$ 會略小於輸入特徵圖的 $H_x \times W_x$ 。從圖 4.1 上，我們可以看出 $\mathbf{X}$ 、 $\mathbf{K}$ 、 $\mathbf{Y}$ 三者之間的大小關係為

$$H_y = H_x - H_k + 1 \tag{4.3a}$$

$$W_y = W_x - W_k + 1 \tag{4.3b}$$

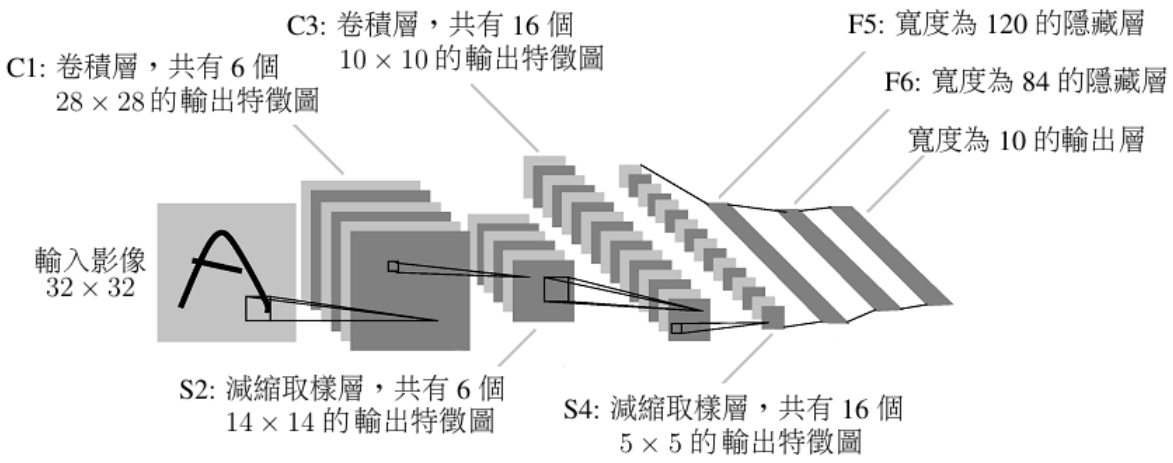


圖 4.2: 卷積類神經網路的示意圖 [2] 。

在卷積類神經網路中，一個卷積層可以同時有多個輸入及輸出，如圖 4.2 所示。 假設一個卷積層可以同時接收 M 個不同的輸入特徵圖，對於這 M 個不同的輸入特徵圖，我們使用 M 個不同的卷積核心分別進行卷積運算取得 M 個不同的輸出，並將這 M 個不同的輸出與相對應的偏移量疊加起來，作為一個輸出特徵圖。一個卷積層若是定義了 N 個不同的輸出特徵圖，我們便重複上述的過程 N 次，使用 $M \times N$ 個不同的卷積核心進行運算。如下所示

$$\mathbf{Y}_j = b_j + \sum_{i=1}^{M} \mathbf{X}_i * \mathbf{K}_{ij}, \quad j = 1, 2 \ldots N \tag{4.4}$$

其中 b_j 是第 j 個輸出特徵圖的偏移量，它與特徵圖之間的加法是透過逐元素 (element-wise) 的運算來完成； $\mathbf{X}_i$ 是第 i 個輸入特徵圖， $\mathbf{Y}_j$ 是第 j 個輸出特徵圖， $\mathbf{K}_{ij}$ 則是相對應的卷積運算核心，是整個卷積類神經網路中最核心也最重要的部份。 一般來說，通過卷積層的輸出特徵圖 $\mathbf{Y}_j$ ，都會在進入下一層當做輸入前通過一個活化函數，例如邏輯函數等。 在一般影像相關的問題中，我們會透過各式各樣的濾波器 (filter) 進行影像的處理，例如使用高斯濾波器達到高斯模糊 (Gaussian Blur) 的效果，或是用加伯濾波器 (Gabor filter) 進行邊緣檢測 (edge

detection) 等，並以此作為特徵做進一步的處理。 這些濾波器通常是經過一系列的數學推導與實驗證明才得出的結果，然而在卷積類神經網路中，這些作為濾波器的卷積核心 $\mathbf{K}_{ij}$ ，完全是以隨機的方式進行初始化，並透過與深層類神經網路相同的反向傳播演算法，一步步地調整而學出來的。

4.1.3 減縮取樣層 (Subsampling Layer)

一個取樣比率 (sampling rate) 為 r 的減縮取樣層 (subsampling layer) 中，可以透過與 $r \times r$ 單位矩陣 $\mathbf{I}_r$ 的卷積運算，對輸入的特徵圖進行平均取樣 (average sampling)，將其高寬分別縮小 r 倍，如下所示

$$\mathbf{Y}_i[a,b] = a_i(\mathbf{X}_i * \mathbf{I}_r)[ra, rb] + b_i, \quad i = 1, 2 \dots M \tag{4.5}$$

其中 a_i 和 b_i 則分別是第 i 個減縮取樣層的縮放係數及偏移量，均為模型中可訓練的參數。

圖 4.2 是一個標準的卷積類神經網路架構圖，共有 2 個卷積層、 2 個減縮取樣層、以及一個深度為 3 的深層類神經網路。在圖中，最左方的圖片是訓練資料中的一張張影像，也是第一層卷積取樣層唯一的輸入特徵圖，會由左至右地通過一連串的卷積運算與減縮取樣而逐漸縮小。通常，我們會在這些輸出特徵圖小到不再適合進行卷積運算的時候，將所有的輸出特徵圖取出，將其中的值拉直排列成一個向量，當作深層類神經網路的輸入特徵向量，繼續往前傳遞。

4.1.4 反向傳播演算法

根據前述 3.1.1 節的作法， 損失函數 E 對卷積核心 $\mathbf{K}_{ij}$ 的偏微分，可以透過連鎖

律對 $\mathbf{Y}_j$ 展開（參考附錄二式 B.11 ），寫成 $\mathbf{X}_i$ 與 $\dfrac{\partial E}{\partial \mathbf{Y}_j}$ 的函數，如下所示

$$\frac{\partial E}{\partial \mathbf{K}_{ij}} = \mathbf{X}_i^{180^\circ} * \frac{\partial E}{\partial \mathbf{Y}_j} \tag{4.6}$$

其中 $\mathbf{X}_i^{180^\circ}$ 是旋轉 180 度後的輸入特徵圖，可以寫成

$$\mathbf{X}_i^{180^\circ}[a,b] = \mathbf{X}_i[H_x - 1 - a, W_x - 1 - b] \tag{4.7}$$

透過相同的作法，可以將損失函數 E 對輸入特徵圖 $\mathbf{X}_i$ 的偏微分，透過連鎖律對 $\mathbf{Y}_j$ 展開（參考附錄二式 B.8c ），寫成 $\dfrac{\partial E}{\partial \mathbf{Y}_j}$ 與 $\mathbf{K}_{ij}$ 的函數

$$\frac{\partial E}{\partial \mathbf{X}_i} = \sum_{j=1}^{N} \left(\frac{\partial E}{\partial \mathbf{Y}_j} \overset{full}{*} \mathbf{K}_{ij}^{180^\circ} \right) \tag{4.8}$$

其中預算子 $\overset{full}{*}$ 是指以 $\dfrac{\partial E}{\partial \mathbf{Y}_j}$ 為中心，將周圍以 0 填滿 (padding) ，並輸出與 $\mathbf{X}$ 相同大小結果的卷積運算，如圖 4.3 所示；而 $\mathbf{K}_{ij}^{180^\circ}$ 是旋轉 180 度後的卷積核心，與式 4.7 有相同的定義方式。 損失函數 E 對偏移量 b_j 的偏微分則為

$$\frac{\partial E}{\partial b_j} = \sum_{a=0}^{H_y-1} \sum_{b=0}^{W_y-1} \frac{\partial E}{\partial \mathbf{Y}_j[a,b]} \cdot \frac{\partial \mathbf{Y}_j[a,b]}{\partial b_j} \tag{4.9a}$$

$$= \sum_{a=0}^{H_y-1} \sum_{b=0}^{W_y-1} \frac{\partial E}{\partial \mathbf{Y}_j[a,b]} \tag{4.9b}$$

減縮取樣層也可以利用同樣的方式進行推導，我們便不再贅述。有了這些前後級錯誤訊號間的關係，以及損失函數對各模型參數的偏微分值，我們便可以透過梯度下降法訓練卷積類神經網路。

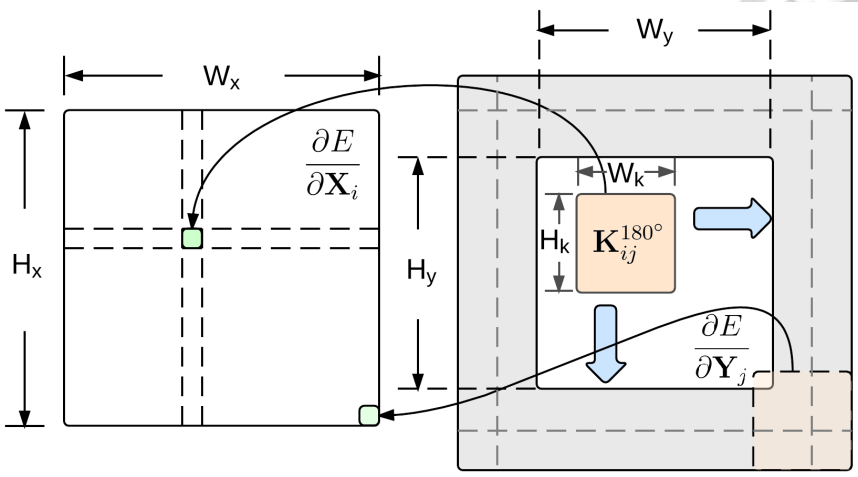


圖 4.3: 在反向傳播演算法中，以式 4.8 對後級錯誤訊號 $\frac{\partial E}{\partial \mathbf{Y}_j}$ 進行反向卷積運算，求得前級的錯誤訊號 $\frac{\partial E}{\partial \mathbf{X}_i}$ 的示意圖。圖中 $\frac{\partial E}{\partial \mathbf{Y}_j}$ 周圍的灰色區塊代表補 0 的部份。

4.2 實驗與分析

4.2.1 實驗設定

基準實驗

在本實驗中，我們選用與第三章相同的前後文相關音素隱藏式馬可夫模型作為基礎的架構，並使用 TIMIT 作為評效語料進行一系列的實驗。

聲學模型設定

本論文中，我們實驗了 4 種不同的卷積類神經網路作為聲學模型。 在系統一中，我們使用了 64 維的梅爾濾波器組，並串接前 39 後 24 共 64 個音框，形成 64×64 的時頻圖作為輸入特徵圖。 第一層為卷積層，其中每個卷積核心的大小為 7×7 ，並輸出成 20 個 58×58 的特徵圖。 第二層為 $r = 2$ 的減縮取樣層，將大小為 58×58 的輸入特徵圖縮小至 29×29 。 第三層為卷積層，其中每個

卷積核心的大小為 6×6 ，並輸出成 20 個 24×24 的特徵圖。 第四層為 $r = 2$ 的減縮取樣層，將大小為 24×24 的輸入特徵圖縮小至 12×12 。 接著我們將 20 個 12×12 的輸出映像拉直成一個 2880 的特徵向量，當作寬度為 2048 、深度為 2 、輸出層寬度為 2425 的深層類神經網路的輸入，並將上述的架構記成 `1x64x64-20x7x7-2s-20x6x6-2s-2048-2048-2425` ，如圖 4.4 a 所示。

系統二是將 64 維梅爾濾波器組，切成高頻 (33-64) 與低頻 (1-32) 兩個頻帶，並搭配上其對時間的一階及二階導數、分別串接前 21 後 10 共 32 個音框，形成 6 張 32×32 的時頻圖（如圖 4.4 b 所示）。整個系統的架構可以用前述的方法，表示為 `6x32x32-32x5x5-2s-48x5x5-64x3x3-2s-4095-4095-2425` 。

系統三是系統二的變形。除了上述的高頻與低頻以外，我們還新增了一個橫跨高低頻 (17-48) 的 32 維聲學特徵作為輸入，形成 9 張 32×32 的時頻圖（如圖 4.4 c 所示），其餘設定都與系統二相同。

系統四是將 64 維梅爾濾波器組，切成四個不重疊的頻帶，每個頻帶各 16 維，並搭配上其對時間的一階及二階導數、分別串接前 10 後 5 共 16 個音框，形成 12 張 16×16 的時頻圖（如圖 4.4 d 所示）。整個系統的架構可以表示為 `12x16x16-30x3x3-30x3x3-2s-2048-2048-2425` 。

在上述的四種系統中，我們一律使用邏輯函數當作深層類神經網路與卷積層後方的活化函數，並在深層類神經網路中使用 10% 的丟棄法正規化。透過反向傳播演算法對卷積類神經網路進行訓練時，我們以 256 筆訓練資料作為一個批次，並選用 0.5 作為動量、 0.01 作為初始學習率，並使用與第三章相同的策略，根據系統在調適語料上的辨識率動態地調整學習率。

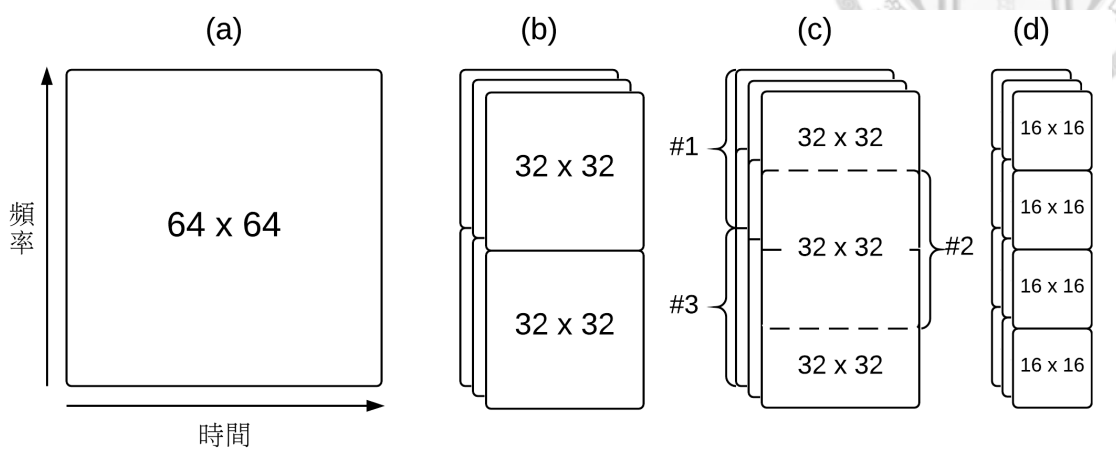


圖 4.4: 各種不同的卷積類神經網路架構圖。橫軸代表時間，綜軸代表頻率（梅爾濾波器組）。

聲學模型	丟棄率 (%)	音素錯誤率 (%)
高斯混合模型基準實驗 (MFCC)	-	28.29
(A) 深層類神經網路 (fbank)	30%	22.73
(B) 深層類神經網路 (MFCC)	30%	22.63
(a) 卷積類神經網路系統一 (fbank, $1 \times 64 \times 64$)	10%	24.30
(b) 卷積類神經網路系統二 (fbank, $6 \times 32 \times 32$)	10%	24.68
(c) 卷積類神經網路系統三 (fbank, $9 \times 32 \times 32$)	10%	24.13
(d) 卷積類神經網路系統四 (fbank, $12 \times 16 \times 16$)	10%	27.18

表 4.1: 四種卷積類神經網路系統在 TIMIT 核心測試語料上的辨識結果，以音素錯誤率作為衡量標準。此外，我們也放入了第三章中使用丟棄法的深層類神經網路的辨識結果，一併進行比較。

4.2.2 實驗結果與分析

上述四種系統在 TIMIT 的核心測試語料上的實驗結果如表 4.1 所示。 從實驗的結果我們可以看出，使用卷積類神經網路的辨識效能明顯優於傳統基於高斯混合模型的聲學模型，其中又以系統三的表現最為出色，音素錯誤率下降了 14.7% 之多；然而跟同樣是類神經網路的作法相比，如表 4.1 中的 (A) 及 (B) ，則顯得略有不足，音素錯誤率大約差了 1.40% 。 我們推論是卷積網路中，各卷積層中使用的卷積核心數量不足所導致。 由於卷積類神經網路在訓練的過程中，需要進行相當多次的卷積運算，其時間複雜度遠遠超過深層類神經網路中的矩陣向量運算。訓練一個如表 4.1 中 (a) – (d) 這樣的卷積類神經網路，所需的時間大約是深層類神經網路 (A) 與 (B) 的 10 倍之多。這使得我們難以在合理的時間範圍內，透過大量增加卷積核心的數量作法提高辨識率。

在一系列的卷積類神經網路實驗中，也可以發現系統四的辨識錯誤率明顯高於其他三者。我們推論是系統四將梅爾濾波器組切成過多的小頻帶，導致每個輸入特徵圖太小，使得能選擇的卷積核心大小也連帶受到限制。而太小的卷積核心，如 3×3 或 2×2 ，其所能描述的輸入輸出間關係也就相當有限。

此外，在進行系統三的實驗時，我們是直接從系統二已經訓練好的模型中，加上新的卷積核心並用隨機亂數為其初始化。這樣的作法可以讓系統三中的各項參數，以較好的初始值作為出發點，大幅地減少訓練的時間。從表 4.1 中也可以明顯的看到，系統三新增一段中間頻帶作為輸入特徵圖的作法，確實能夠減少系統的音素錯誤率。 最後，我們也從 (a) - (c) 這三個系統中發現，加上對時間的一階導數及二階導數作為輸入的作法，似乎對語音辨識系統的效能沒有太大的幫助，但仍需要進一步更完整的實驗才能證明。

4.3 本章總結

本章介紹了最基本的卷積類神經網路及其運作原理，並利用卷積類神經網路作為語音辨識系統的聲學模型，在 TIMIT 上進行了一系列的實驗。實驗結果顯示，雖然卷積類神經網路的表現比傳統高斯模型的作法還要好，但是與相同的運算資源下的深層類神經網路作法相比，仍略顯不足。

第 五 章　深層類神經網路之語者調適

本章將探討如何在深層類神經網路聲學模型上進行語者調適，以解決受測目標語者與訓練語料之間不匹配的問題。我們針對現有的語者調適演算法，提出了一個適當的改進方案，以達到更好的辨識效果。

5.1 簡介

語者調適 (speaker adaptation) 是語音辨識中一門探討如何解決聲學模型在面對新的受測語者時，因為受測語者與訓練語料之間的不匹配 (mismatch) ，而造成系統辨識效能下降的研究。 一般來說，在進行語者調適時，我們會有一個已經事先訓練好的聲學模型，稱作語者不特定模型 (speaker-independent model, SI model) 。接著會利用目標語者的少量資料，重新對這個模型進行訓練、調整其中的參數，使得該模型在面對受測目標語者時，能夠展現出較好的辨識率表現。而這個經過重新訓練的模型，便稱作語者調適模型 (speaker-adaptive model, SA model) 。

傳統上以高斯混合模型作為聲學模型的語音辨識中，有相當多的語者調適演算法，可以對抗此一問題，常見的包括了「最大相似度線性迴歸」 (maximum likelihood linear regression, MLLR) [42] [43] [44] [45] 、「最大事後機率調適法」 (maximum a posteior probability, MAP) [46] [47] [48] 、「聲道長度正規化」 (Vocal Tract Length Normalization, VTLN) [49] 及「特徵聲音」 (eigenvoice) [50] 等。

近年來，由於以深層類神經網路作為聲學模型的作法已逐漸成為主流，也因此有越來越多的研究開始談討如何有效地在深層類神經網路上進行語者調適。其中包括了在輸入層、輸出層、及隱藏層中加入線性轉換 (linear transformation) 的線性輸入網路 (linear input network, LIN) [51] 、線性輸出網路

(linear output network) [52] 及線性隱藏網路 (linear hidden network) [53] 、特徵空間上鑑別式線性迴歸 (feature-space discriminative linear regression, fDLR) [20] 、透過矩陣分解 (matrix factorization) 或是奇異值分解 (singular value decomposition, SVD) 的作法 [54] [55] [56] 、以及串接了身份向量 (identity vector, i-vector) [57] 的語者調適演算法等。

因此，本論文以此為出發點，介紹幾種語者調適的演算法，並提出利用狀態分群的方式，改進現有的「特徵空間上的鑑別式線性回歸」，增進目標語者的辨識準確率。

5.2 基於奇異值分解的語者調適

在語音辨識中，一般來說會選擇使用很深很寬的深層類神經網路架構作為聲學模型，來描述輸入的聲學特徵向量與成千上萬個隱藏式馬可夫模型狀態之間，高度複雜的非線性關係。而面對這樣的一個聲學模型，如何有效地透過少量的目標語者語料，調整其中高達百萬甚至上億的參數來提昇辨識的準確率，便是一個相當困難的問題。

基於奇異值分解的語者調適演算法，是一個透過奇異值分解，剖析深層類神經網路中最靠近輸出層的仿射轉換矩陣，去除其中不重要的資訊並降低維度，使其參數減少以利於參數調整的一套作法。

考慮一個已經事先訓練好的語者不特定深層類神經網路聲學模型，假設最後一層隱藏層的寬度為 H 、輸出層寬度為 S ，並用 $\mathbf{W}_{S\times H}$ 代表最後一層隱藏層中，用來描述輸出與輸入之間的關係的仿射轉換矩陣。透過奇異值分解，可以將 $\mathbf{W}_{S\times H}$ 寫成

$$\mathbf{W}_{S\times H} = \mathbf{U}_{S\times S} \cdot \mathbf{\Sigma}_{S\times H} \cdot \mathbf{V}^{T}_{H\times H} \tag{5.1}$$

其中 $\mathbf{\Sigma}_{S\times H}$ 是一個在對角線 (diagonal) 上才有值的矩陣。這些值稱為奇異值 (singular values) ，會在對角線上根據大小依序由左上至右下排列。接著便把對角線上較小的奇異值丟掉，只保留其中前 k 大的部份

$$\mathbf{W}_{S\times H} \approx \mathbf{U}_{S\times k} \cdot \mathbf{\Sigma}_{k\times k} \cdot \mathbf{V}^{T}_{H\times k} \tag{5.2}$$

透過奇異值分解降維後的 $\mathbf{W}_{S\times H}$ ，由於多多少少損失了一些資訊，因此會再透過 1 至 2 次反向傳播演算法與梯度下降法，微調整個深層類神經網路中各參數的值，進行修復以彌補這些損失。 接著，將式 5.2 後兩項 $\mathbf{\Sigma}_{k\times k} \cdot \mathbf{V}^{T}_{H\times k}$ 合併成另一個矩陣 $\mathbf{N}_{k\times H}$ ，並在中間差入一個單位矩陣 (identity matrix) $\mathbf{I}_{k\times k}$ ，寫成以下形式

$$\mathbf{W}_{S\times H} \approx \mathbf{U}_{S\times k} \cdot \mathbf{N}_{k\times H} \tag{5.3a}$$

$$= \mathbf{U}_{S\times k} \cdot \mathbf{I}_{k\times k} \cdot \mathbf{N}_{k\times H} \tag{5.3b}$$

在固定除了 $\mathbf{I}_{k\times k}$ 以外的模型參數的情況下，便可以透過反向傳播演算法及梯度下降法，利用目標語者的訓練資料，對原本的語者不特定模型進行調適。

5.3 特徵空間上鑑別式線性迴歸

另一種運行在深層類神經網路上的語者調適演算法為「特徵空間上鑑別式線性迴歸」，其概念源自於傳統高斯混合模型的「特徵空間上的最大相似度線性迴歸演算法」 (feature-space maximum likelihood linear regression, fMLLR) 。 由於每個人的唇、齒、喉構造均不相同，發音的習慣也不同，故發出的聲音自然不盡相同，這也使得諸如頻域 (frequency domain) 上的共振峰頻率 (formant frequency) ，會以

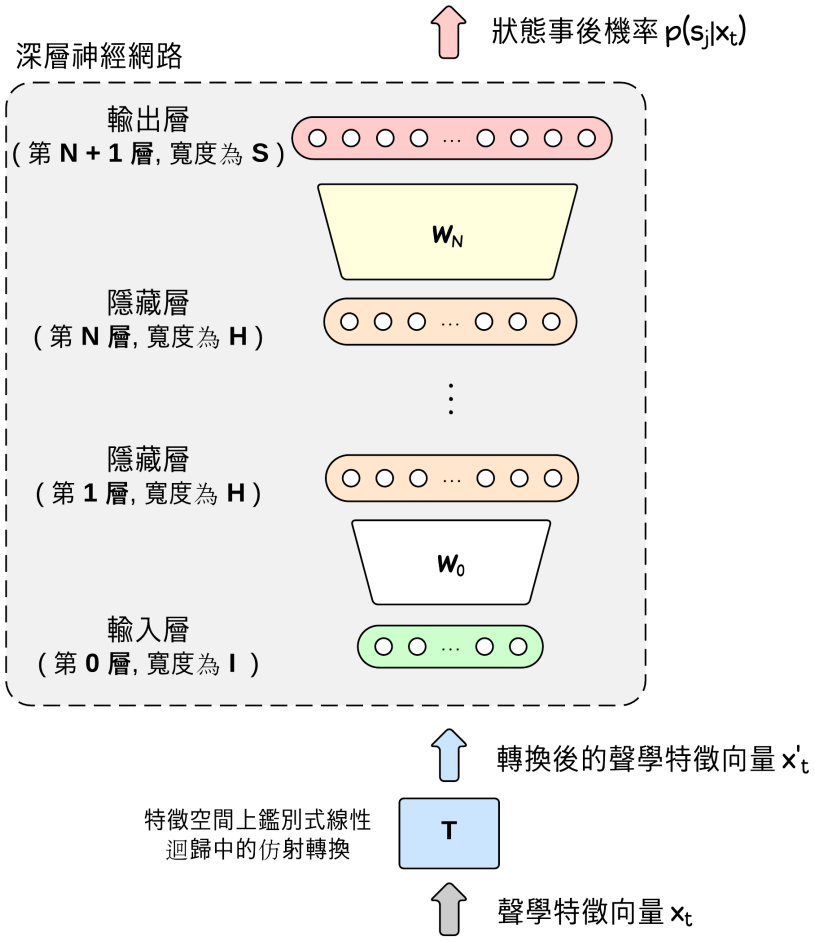


圖 5.1: 特徵空間上鑑別式線性迴歸之示意圖

平移、縮放、甚至是其他更複雜的變化，表現在串接了前後音框的梅爾倒頻譜係數聲學特徵向量上。因此，如果聲學特徵向量在進入語者不特定深層神經網路之前，能先通過一個在特徵空間上的轉換，便有機會補償受測語者與訓練語料之間的不匹配的問題，進而提昇受測語者的辨識準確率。

首先，如圖 5.1 所示，在深層類神經網路的最前端額外地加上一個可訓練的特徵轉換矩陣 $\mathbf{T}$

$$\mathbf{x}_t^{'} = \mathbf{T} \cdot \mathbf{x}_t = \mathbf{M} \cdot \mathbf{x}_t + \mathbf{c}. \tag{5.4}$$

其中的線性轉換矩陣 $\mathbf{M}$ 及偏移量 $\mathbf{c}$ 分別以單位矩陣 $\mathbf{I}$ 及零向量 (zero vector) 作為初始值。透過反向傳播演算法，便得以調整特徵轉換矩陣裡的值，以補償訓練語料與目標語者間不匹配的問題而達到語者調適的效果。

5.4 基於狀態分群的特徵空間上鑑別式線性迴歸

本論文中，我們提出了「基於狀態分群的特徵空間上鑑別式線性迴歸」(state-clustered fDLR)，期望能進一步提升語者調適的效果，達到更佳的辨識準確率。此方法跟傳統方法不一樣的地方在於，對每位語者來說，除了原本就有的 $\mathbf{T}$ 以外，我們還額外訓練了 K 個仿射轉換 $\mathbf{T}_k$。

首先，我們將隱藏式馬可夫模型中聲學特性相似的三連音狀態，透過 K 平均分群 (k-means clustering) 演算法的方式，分成 K 個互相不交集的集合 (disjoint set)，例如不送氣爆破音 /ㄅ, ㄉ, ㄎ/ 的狀態可能會分在一起、送氣爆破音 /ㄆ, ㄊ, ㄎ/ 的狀態可能會分在一起、鼻音 /ㄇ, ㄋ/ 的狀態可能會分在一起等。根據群聚演算法的結果以及語者調適語料中每筆訓練資料所對應到的狀態標記 (state label)，便可以將語者調適語料分成相對應的 K 群，並為每一群訓練一個專屬於該群的仿射轉換 $\mathbf{T}_k$，最後再透過兩階段的方式進行解碼辨識。 在第一階段中，經過 $\mathbf{T}$ 轉換的聲學特徵向量會被輸入至語者不特定深層類神經網路聲學模型，並根據其預測結果，估計出該筆資料屬於上述 K 個狀態群的機率為何；在第二階段中，相同的一筆資料會被重新輸入至 K 個不同的仿射轉換 $\mathbf{T}_k$，取得 K 個不同的狀態事後機率 (state posterior probabilities)，並用第一階段的機率作為權重，為這些不同的狀態事後機率分佈進行加總。

5.4.1 隱藏式馬可夫模型之狀態分群

我們將前述的 $\mathbf{W}_{S\times H}$ 轉置，並表示成 S 個 H 維行向量 (column vector) 的組合，如下所示

$$\mathbf{W}_{S\times H}^{T} = [\mathbf{w}_1, ..., \mathbf{w}_j, ..., \mathbf{w}_S], \tag{5.5}$$

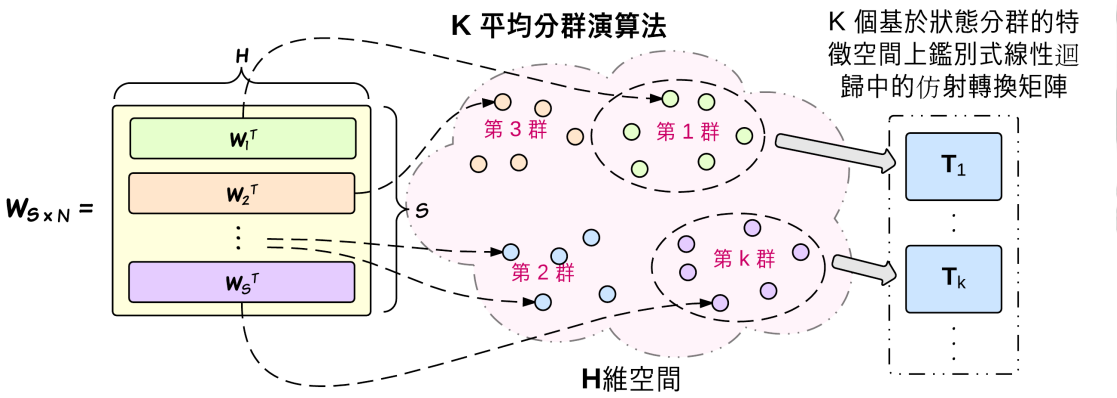


圖 5.2: 狀態分群之示意圖

其中 $\mathbf{w}_j$ 代表 $\mathbf{W}_{S\times H}^T$ 中第 j 個 H 維的行向量。 我們認為這個仿射矩陣 $\mathbf{W}_{S\times H}$ 帶有相當重要的資訊，且這 S 個向量應該非常適合用來代表隱藏式馬可夫模型中的那 S 個狀態。 因此，透過 K 平均演算法 (k-means clustering) 將這 S 個 H 維的向量分成 K 群：

$$C_k = \{s_j | \mathbf{w}_j \in \text{cluster } k\}, \quad k = 1, 2, \ldots K \tag{5.6}$$

其中 s_j 代表隱藏式馬可夫模型中的第 j 個狀態。 接著，對於每一位目標語者，我們根據前述的 C_k 將其調適用的訓練語料 (training corpus for speaker adaptation) D 分成 K 個不互相交集的子集合 $\{D_1, D_2, ..., D_K\}$

$$D = \{(\mathbf{x}_t, y_t)\} \tag{5.7a}$$

$$D_k = \{(\mathbf{x}_t, y_t) \,|\, y_t \in C_k\}, \tag{5.7b}$$

其中 y_t 是聲學特徵向量 $\mathbf{x}_t$ 的狀態標記 (state label) 。換句話說，如果聲學特徵向量 $\mathbf{x}_t$ 的狀態標記 y_t 屬於第 k 群，我們便將該特徵向量納入資料群 D_k 之中。最後，除了使用標準的特徵空間上鑑別式線性迴歸演算法，在整份的語者調適用的

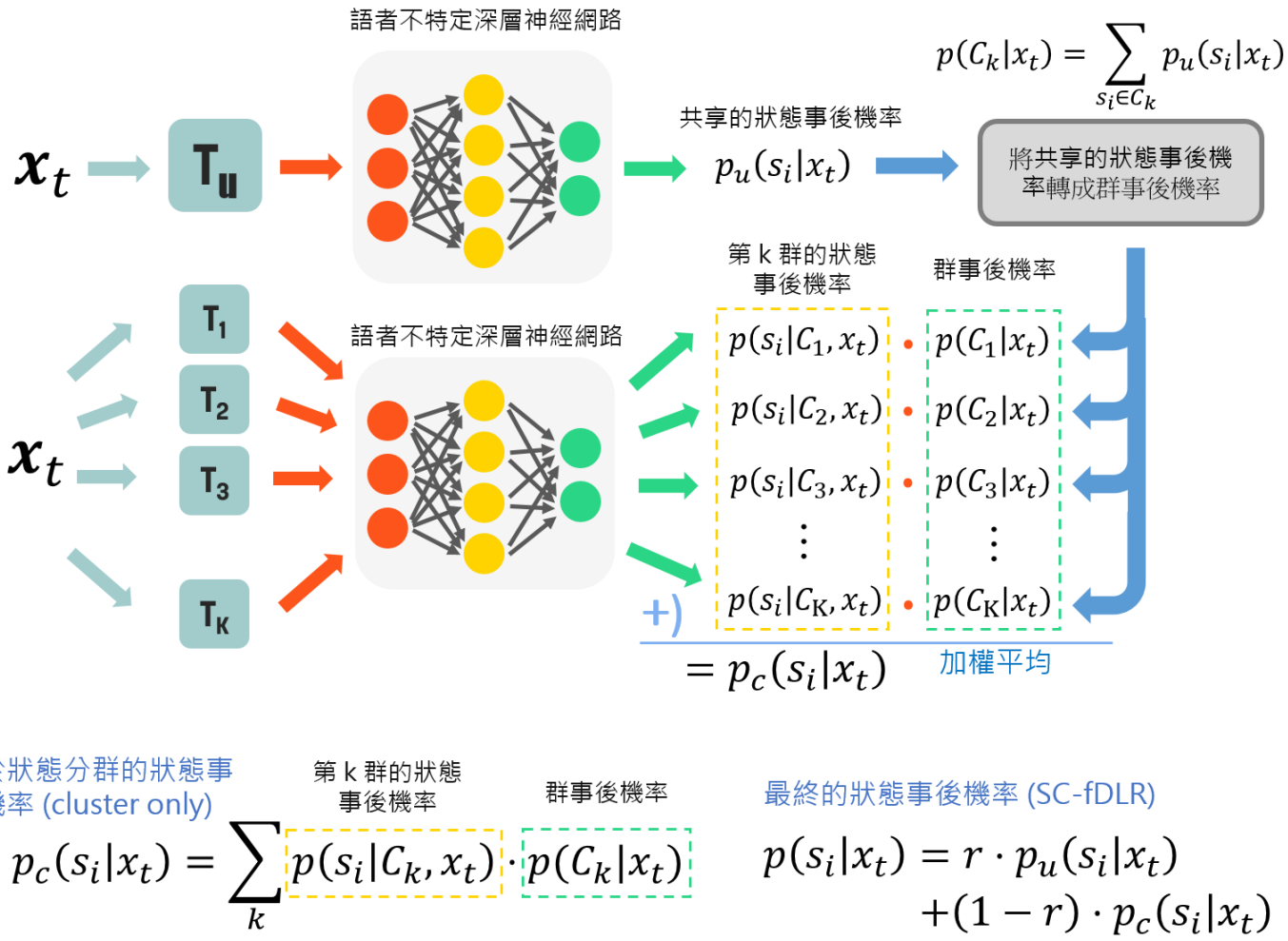


圖 5.3: 兩階段式解碼

訓練語料 D 上訓練一個共享的 (universal) 特徵轉換 $\mathbf{T}_u$ 以外，我們也另外為這 K 群語者調適用的訓練語料 D_k ，分別訓練出專屬於該群的 $\mathbf{T}_k$ ，一共 K 個不同的特徵轉換矩陣 $\{\mathbf{T}_1, \mathbf{T}_2, ..., \mathbf{T}_K\}$ 。

5.4.2 兩階段式解碼

由於我們事先並不知道每一筆輸入的聲學特徵向量 $\mathbf{x}_t$ 屬於隱藏式馬可夫模型中的哪個狀態，也就不知道該使用哪個相對應的仿射轉換 $\mathbf{T}_k$ 為聲學特徵向量進行轉換，因此我們得透過兩階段式的解碼 (two-pass decoding) 來完成語音辨識。

在第一階段中（見圖 5.3 的左半部份 ），我們先依照傳統特徵空間上鑑別式

線性迴歸的作法，將聲學特徵向量輸入至共享的 $\mathbf{T}_u$ 進行特徵轉換，再使其進入到語者不特定的深層類神經網路聲學模型中，以取得隱藏式馬可夫模型中各狀態的事後機率，此稱為「共享的狀態事後機率」，記作 $p_u(s_j|\mathbf{x}_t)$ 。 接著便可以透過狀態與各群的對照表，將同一群中各狀態的事後機率加總起來，取得該群的群事後機率 (cluster posterior probability) ，如下所示：

$$p(C_k|\mathbf{x}_t) = \sum_{s_j \in C_k} p_u(s_j|\mathbf{x}_t). \tag{5.8}$$

在第二階段中，我們將同一份的輸入聲學特徵向量 $\mathbf{x}_t$ ，以平行地方式分別輸入至 K 個經由狀態分群所訓練出來的仿射轉換 $\mathbf{T}_k$ 及語者不特定深層類神經網路聲學模型，取得 K 份不同的事後機率 $p(s_j|C_k, \mathbf{x}_t)$ ，如圖 5.3 右半部所示。接著，再利用前述的群事後機率，對這 K 份不同的事後機率透過進行權重加總 (weighted sum) ，以取得「基於狀態分群的狀態事後機率」 $p_c(s_j|\mathbf{x}_t)$ 。

$$p_c(s_j|\mathbf{x}_t) = \sum_{k=1}^{K} p(s_j|C_k, \mathbf{x}_t) \cdot p(C_k|\mathbf{x}_t). \tag{5.9}$$

最後，為了避免過多可訓練參數導致過度貼合現象的發生，我們便透過線性內插 (linear interpolation) 的方式，以係數 r 結合共享事後機率與基於狀態分群的事後機率，作為最終的狀態事後機率 $p(s_j|\mathbf{x}_t)$ 。

$$p(s_j|\mathbf{x}_t) = r \cdot p_u(s_j|\mathbf{x}_t) + (1 - r) \cdot p_c(s_j|\mathbf{x}_t). \tag{5.10}$$

5.5 實驗與分析

5.5.1 實驗設定

本實驗中，受測語者有 5 男 5 女共 10 位。 我們從各受測語者的 Facebook 狀態中隨機挑選出 1000 句，當作該語者的文字語料。 在這些文字語料中，只有約 4.1% 的字或片語屬於英文，其餘均屬中文。 我們請各語者分別以朗讀及自發性的方式，錄製了朗讀式語音 (read speech) 與自發性語音 (spontaneous speech) 兩個版本，作為本次語者調適實驗所需的語音語料，總長度為 20,000 句、共 15.6 小時。對於每個語者，不論是哪種說話方式，我們都從 1000 句中選出 500 句作為語者調適用的訓練語料、 250 句作為實驗中調整參數用的調適語料，並以剩下的 250 句進行測試。

我們選用了與前述第三章中相同的 39 維梅爾倒頻譜係數，並以串接了前 4 後 4 共 9 個音框的 351 維聲學特徵向量，作為深層類神經網路的輸入。 實驗中所需的語者不特定之共享三連音隱藏式馬可夫模型，是透過標準的最大相似度估計，在中英混合語料上訓練而成，其中共有 3445 狀態及 17269 個高斯分佈。 語料中屬於中文的部份係由聲碩麥克風 (AST Microphone) 錄製而成，共 31.8 小時的朗讀式語料； 英文的部份則是長度為 29.7 小時、由台灣語者所錄製的台灣口音英語 (English Across Taiwan Microphone, EATMIC) 朗讀式語料。 描述狀態相似度的前後文相關深層類神經網路中，包含了 4 層寬度為 2048 的隱藏層及一層寬度為 3445 的輸出層。

本實驗中，我們選用來自 PTT 的文字語料產生辭典及訓練詞三連 (trigram) 語言模型。 PTT 是台灣一個非常熱門的電子佈告欄系統，擁有超過 150,000 位同時在線的使用者、及超過 20,000 個佈告欄。由於 PTT 是時下年輕人聚集並討論事務

的地方，故相當適合作為辭典及語言模型。 計算辨識錯誤率時，中文的部份以中文字 (character) 為單位，而語料中較少的英文部份則用英文單字 (word) 為單位。以下所有的實驗都是以前述 10 位語者的平均辨識錯誤率作為衡量的標準。

5.5.2 基準實驗

在第一個初步的實驗中，我們比較了當群聚數量 K 分別為 4, 8, 16, 32 時，式 5.9 中未經線性內插的特徵空間上的鑑別式線性回歸（記作僅分群），與其他三組基準實驗的辨識準確率，包含了「語者不特定模型」 (記做為 SI) 、「傳統的特徵空間上的鑑別式線性回歸」 (記作 fDLR) 及保留了 40% 總奇異值的「基於奇異值分解的語者調適」 (記作 SVD) 。 我們可以清楚的看到狀態分群能有效地改善語者調適的效果，但辨識率也隨著群聚數量 K 的變大而略有降低，很明顯是因為缺乏足以估計 眾多參數的資料而導致了過度貼合。

在第二個初步的實驗中，我們考慮了式 5.10 中，不同線性內插係數 r 與辨識準確率的關係。 如圖 5.5 所示，以 8 群 (K = 8) 及 500 句的朗讀式語音作為例子，可以發現只靠基於狀態分群的狀態事後機率 $p_c(s_j|\mathbf{x}_t)$ (當 r = 0 時），其錯誤率大約比傳統的特徵空間上的鑑別式線性回歸 (當 r = 1 時) 多了約 1.3 個百分比，且兩者互補後，最低的辨識錯誤率大約發生在 r 為 0.4 左右。雖然我們可以在獨立出來的 250 句調適語料上進行實驗，選出較好的參數 r ，但幾次的實驗結果都一致指出 r 在 0.5 附近就能有不錯的表現，故以下的實驗均已 r 為 0.5 作為式 5.10 中的線性內插係數。

5.5.3 實驗結果與分析

本論文所提出的「基於狀態分群的特徵空間上的鑑別式線性回歸」經過式 5.10 的

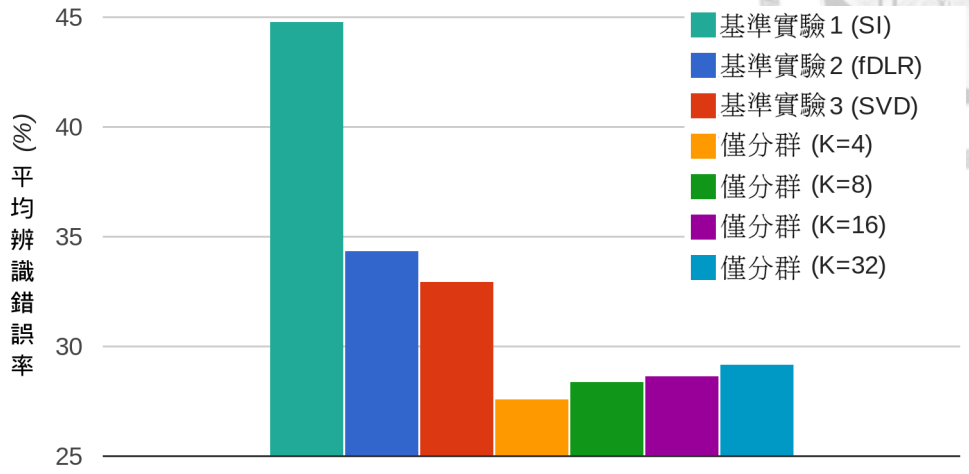


圖 5.4: 做在 10 句朗讀式語音的基礎實驗。當群聚數量 K 分別為 4, 8, 16, 32 時，未經線性內插（見式 5.9 ）的特徵空間上的鑑別式線性回歸與「語者不特定模型」、「傳統的特徵空間上的鑑別式線性回歸」及「基於奇異值分解的語者調適」的辨識準確率比較圖。

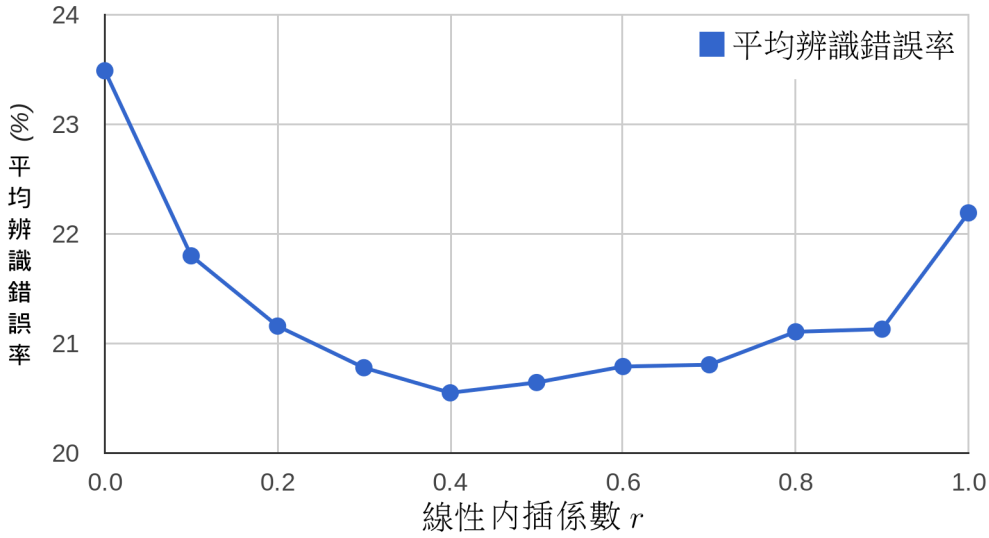


圖 5.5: 在式 5.10 中，不同線性內插係數 r 對辨識錯誤率的影響。

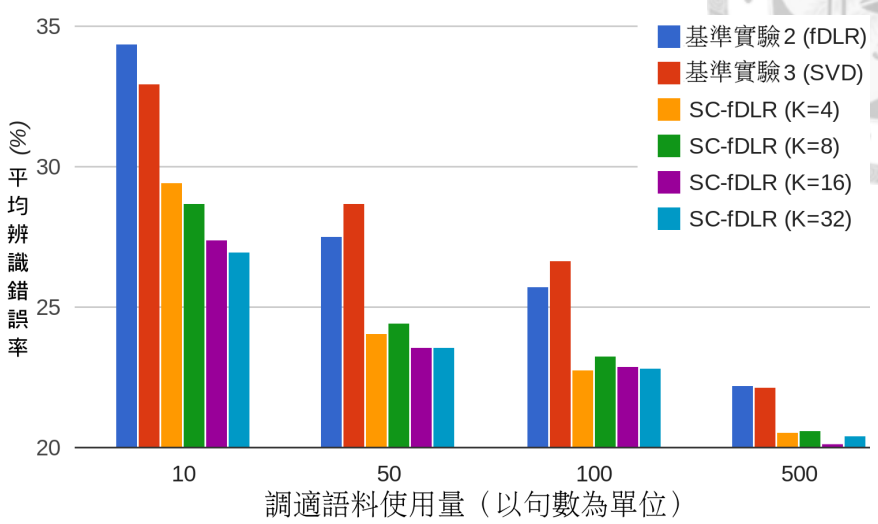


圖 5.6: 本論文所提出的「基於狀態分群的特徵空間上的鑑別式線性回歸」、「共享的特徵空間上的鑑別式線性回歸」和「基於奇異值分解語者調適」在朗讀式語音上的平均錯誤率之比較。

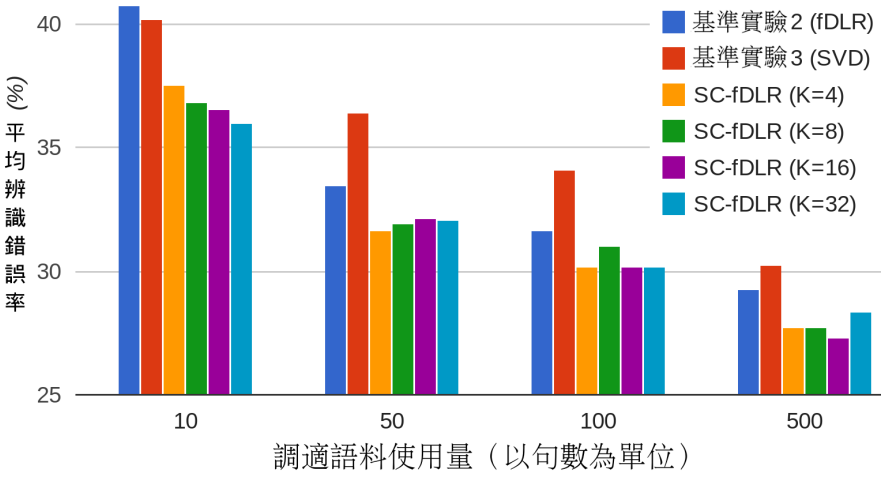


圖 5.7: 本論文所提出的「基於狀態分群的特徵空間上的鑑別式線性回歸」、「共享的特徵空間上的鑑別式線性回歸」和「基於奇異值分解語者調適」在自發性語音上的平均錯誤率之比較。

線性內插後，記作 SC-fDLR ，其在 10 、 50 、 100 、 500 句朗讀式和自發性語音上、群聚數量 K 為 4 、 8 、 16 、 32 時進行的完整實驗結果分別如圖 5.6 及圖 5.7 所示。 實驗結果顯示，在大多數的情況下，將狀態分成越多群，越能更細緻地描述隱藏式馬可夫模型中各狀態的聲學架構，但也越容易受到資料匱乏和過度貼合的影響。 雖然這兩個因素會相互競爭，但我們可以透過式 5.10 中線性內插的方式，合併兩種不同的模型的優點以降低影響。（譬如圖 5.4 中 10 句朗讀式語料上的辨識效能會隨著 K 的變多而變差，但圖 5.6 中的效能卻能隨 K 的變多而持續進步。） 雖然圖5.6 及圖 5.7 中的辨識錯誤率有高有低、在不同參數下表現略有不同，但大體來說，本論文所提出的 SC-fDLR 在各情況下均能有優越的表現。 以 $K = 32$ 為例，在 10 、 50 、 100 、 500 句的朗讀式語音上，其辨識效能的絕對進步量 (absolute improvement) 比傳統的特徵空間上的鑑別式線性回歸分別高出 7.42 、3.91 、 2.89 、 1.75 個百分點，也比基於奇異值分解的語者調適分別高出了 5.99 、 5.10 、 3.79 、 1.72 的百分點。除了朗讀式語音外，在可以在自發性語音的實驗上觀察到相同的趨勢，唯進步量略低一點，推測是自發性語音與訓練自朗讀式語音的隱藏式馬可夫模型不匹配的結果。

5.6 本章總結

本論文提出了利用狀態分群的概念改進傳統特徵空間上的鑑別式線性回歸的作法，並用以處理基於前後文相關深層類神經網路的個人化辨識及語者調適問題。不論是朗讀式或自發性語音，實驗顯示在不同群聚數量及不同長度的語者調適用訓練語料下，其辨識效能均能大幅超越傳統的作法與基於奇異值分解語者調適。

第六章　深層類神經網路函式庫與工具

本章將敘述 **libdnn** 這套深層學習函式庫的基礎用法及程式架構，並說明如何運用此工具訓練基於深層類神經網路及卷積類神經網路的聲學模型。

6.1 簡介

libdnn 是一個輕量的、由 C++ 和統一計算架構程式語言撰寫而成的深層學習函式庫。除了支援 LibSVM [58] 的稀疏 (sparse) 資料格式，也支援緊密排列 (dense) 及 KALDI 資料庫 (archive) 的資料格式。

6.2 基礎用法

在這套工具中，我們提供了三個程式： `nn-init` 、 `nn-train` 、及 `nn-predict` ，分別用來「初始化類神經網路模型」、「利用資料訓練類神經網路模型」、「透過訓練後的類神經網路對資料進行預測」。 茲分別就各程式的使用方式進行說明。

6.2.1 初始化類神經網路模型

像深層學習這樣的一個非凸最佳化 (non-convex optimization) 問題，要在其中找到最佳解，已被證明是 NP 困難的問題 [59] 。 一般來說，深層類神經網路或是深層卷積類神經網路中的參數，少則數百萬多則上億，就算退而求其次，要在這數千萬維的參數空間中，找到一個次佳 (sub-optimal) 的參數組合絕非一件容易的事。一個好的初始值，除了能取得叫好的次佳解以外，也能避免因為活化函數過度飽

和，使得梯度過小而在參數空間中無止盡地遊走的狀況。因此，在常見的隨機坡降法中，如何避免上述情況發生，便是一門重要的課題。 在 `nn-init` 中，使用者可以透過參數 `--type` ，選擇使用以下三種方式進行初始化。

隨機初始化

使用連續型均勻分布 (uniform distributed) 或是常態分佈 (normal distributed) 的虛擬亂數 (pseudorandom numbers) ，為卷積核心或是仿射矩陣內的值進行初始化。 以仿射轉換矩陣為例，如果扇入 (fan-in) 及扇出 (fan-out) 分別為 m 及 n （也就是輸入向量和輸出向量的維度）， 根據 [60] ，我們選用

$$W \sim U\big[-\frac{\sqrt{6}}{\sqrt{m+n}}, \frac{\sqrt{6}}{\sqrt{m+n}}\big] \tag{6.1}$$

作為分布進行初始化，以避免活化函數過度飽和而難以訓練的狀況。

限制性波茲曼機（白努利 — 白努利）

將輸入特徵向量的每一維視作非 0 即 1 的白努利分佈 (Bernoulli distribution) ，透過高斯 — 白努利限制性波茲曼機初始化深層類神經網路。

限制性波茲曼機（高斯 — 白努利）

將輸入特徵向量的每一維視作平均為 0 ，變異數為 1 的高斯分佈 (Gaussian distribution) 或常態分佈，透過高斯 — 白努利限制性波茲曼機初始化深層類神經網路。若輸入特徵向量是高斯分佈，唯平均值非 0 或變異數非 1 ，則可以使用程式提供的 `--normalize 2` 進行特徵縮放 (feature scaling) ，將每一維標準化 (standardize) 成標準分數 (standard score) 。

字串	說明
$20 \times 5 \times 5$	20 個卷積核心。第一個 5 是高度，第二個 5 則是寬度（以像素為單位）
2s	減縮取樣的比例是 2
$10 \times 3 \times 3$	20×10 個卷積核心。
2s	減縮取樣的比例是 2
512-512	兩層寬度均為 512 的隱藏層。

表 6.1: 使用 `nn-init` 初始化類神經網路模型架構時，參數 `--struct` 的使用方式與說明。

使用範例

給定一組存放至 `train.dat` ，特徵向量維度為 351 的訓練資料，可以透過以下指令，為其初始化一個深 4 層、各層寬 2048 、輸出目標共 2425 個類別的深層類神經網路，並將完成初始化完成的類神經網路模型儲存至 `init.xml` 。

```
bin/nn-init train.dat -o init.xml --input-dim 351 --output-dim
   2425 --struct 2048-2048-2048-2048
```

也可以初始化一個深層卷積類神經網路，如下所示：

```
nn-init --input-dim 32x24 --struct 20x5x5-2s-10x3x3-2s-512-512
   --output-dim 10
```

這邊的 32×24 代表這是一張高 32 像素，寬 24 像素的影像，而後面的 20x5x5-2s-10x3x3-2s-512-512 則代表：

在這套工具中，類神經網路模型會以可延伸標記式語言 (eXtensible Markup Language, XML) 的格式儲存，如下所示：

```
<transform type="Affine" input-dim="351" output-dim="2048">
 <weight> ... </weight>
 <bias> ... </bias>
</transform>
<transform type="Tanh" input-dim="2048" output-dim="2048" />
```

```
<transform type="Affine" input-dim="2048" output-dim="2048">
 <weight> ... </weight>
 <bias> ... </bias>
</transform>
<transform type="Tanh" input-dim="2048" output-dim="2048" />
...
<transform type="Affine" input-dim="2048" output-dim="2425">
 <weight> ... </weight>
 <bias> ... </bias>
</transform>
<transform type="Softmax" input-dim="2425" output-dim="2425" />
```

其中 `<weight>` 、 `</weight>` 和 `<bias>` 、 `</bias>` 便是儲存仿射轉換的權重矩陣與偏移向量。

6.2.2 利用資料訓練類神經網路模型

對類神經網路模型進行訓練時，除了第二章中提到的各種正規化方法外，一般會根據訓練集 (training set) 及驗證集 (validation set) 上的錯誤率，選擇停止的時機，以避免過度貼合的狀況發生，此稱為提早結束 (early stopping) 。 在 `nn-train` 中，使用者可以透過 `--max-epoch` 決定最多訓練多少個紀元就停止，或是透過 `--min-acc` 選擇至少在驗證集上達到多少準確率才停止訓練。 此外，拿到一組新資料，進行首次的類神經網路訓練時，可以透過參數 `--save-model-per-epoch` 儲存每個紀元所訓練出的模型，以供後續實驗參考。

使用範例

以前述的 `init.xml` 作為起始點開始訓練，選定學習率為 0.01 ，最多訓練 20 個紀元，並將訓練完的模型儲存至 `mature.xml` 。

```
bin/nn-train train.dat init.xml mature.xml --input-dim 351 --
   learning-rate 0.01 --max-epoch 20
```

6.2.3 透過訓練後的類神經網路對資料進行預測

使用 `nn-predict` 對資料進行預測時，可以透過參數 `--output` 選擇輸出每一筆資料 (**x**) 的 (1) 預測結果、 (2) 事後機率分佈 $p(c_i|\mathbf{x})$ ，其中 c_i 是第 i 個類別、(3) 事後機率分佈的對數值 $\log p(c_i|\mathbf{x})$ 。 除此之外，也可以透過參數 `--prior` ，提供一個存放著目標類別事前機率 $p(c_i)$ 的檔案，讓程式自動透過式 3.1 計算相似度 $p(\mathbf{x}|c_i)$ 與相似度的對數值 $\log p(\mathbf{x}|c_i)$ 。

使用範例

用訓練完成的類神經網路模型 `mature.xml` ，預測一組新的測試資料 `test.dat` 。

```
bin/nn-predict test.dat mature.xml --input-dim 351
```

6.3 程式碼架構

實作上，不論是仿射轉換 `Affine` 、活化函數如雙曲正切函數 `Tanh` 、還是丟棄法正規化 `Dropout` 等等，都被視為是一種特徵轉換，並繼承了

`FeatureTransform` 這個類別。 透過陣列的方式，將這些仿射轉換和活化函數串起來，便能構成一個深層類神經網路。

6.3.1 記憶體佈局

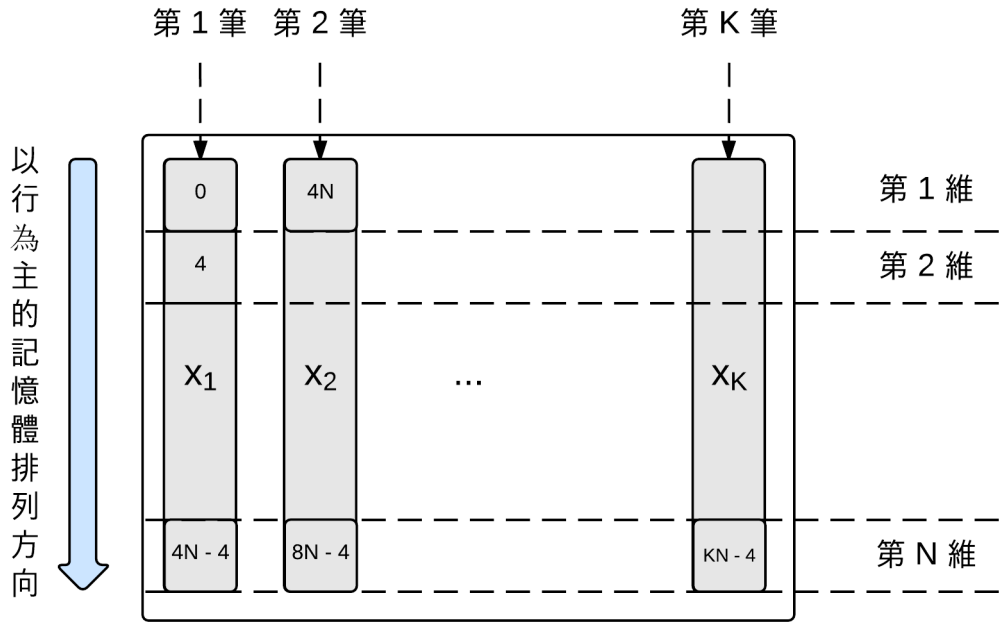


圖 6.1: 以行為主的記憶體佈局。橫向第一列代表第一維，第二列代表第二維，以此類推。縱向的一行代表一筆筆的資料。

由於在基礎線性代數程式集 (Basic Linear Algebra Subprograms, BLAS) 與基於統一計算架構整合技術的基礎線性代數程式集 (CUDA Basic Linear Algebra Subroutine, cuBLAS) 中，記憶體都是「以行為主」 (column-major) 的方式排列，因此在這套程式中，我們也選擇使用「以行為主」的方式排列，如圖 6.1 所示。在卷積類神經網路中，記憶體也以前述的方式進行排序，唯圖中的每個直行代表該筆資料所對應的 M 個特徵圖，分別拉直排列而成的結果。 (見圖 6.2) 。

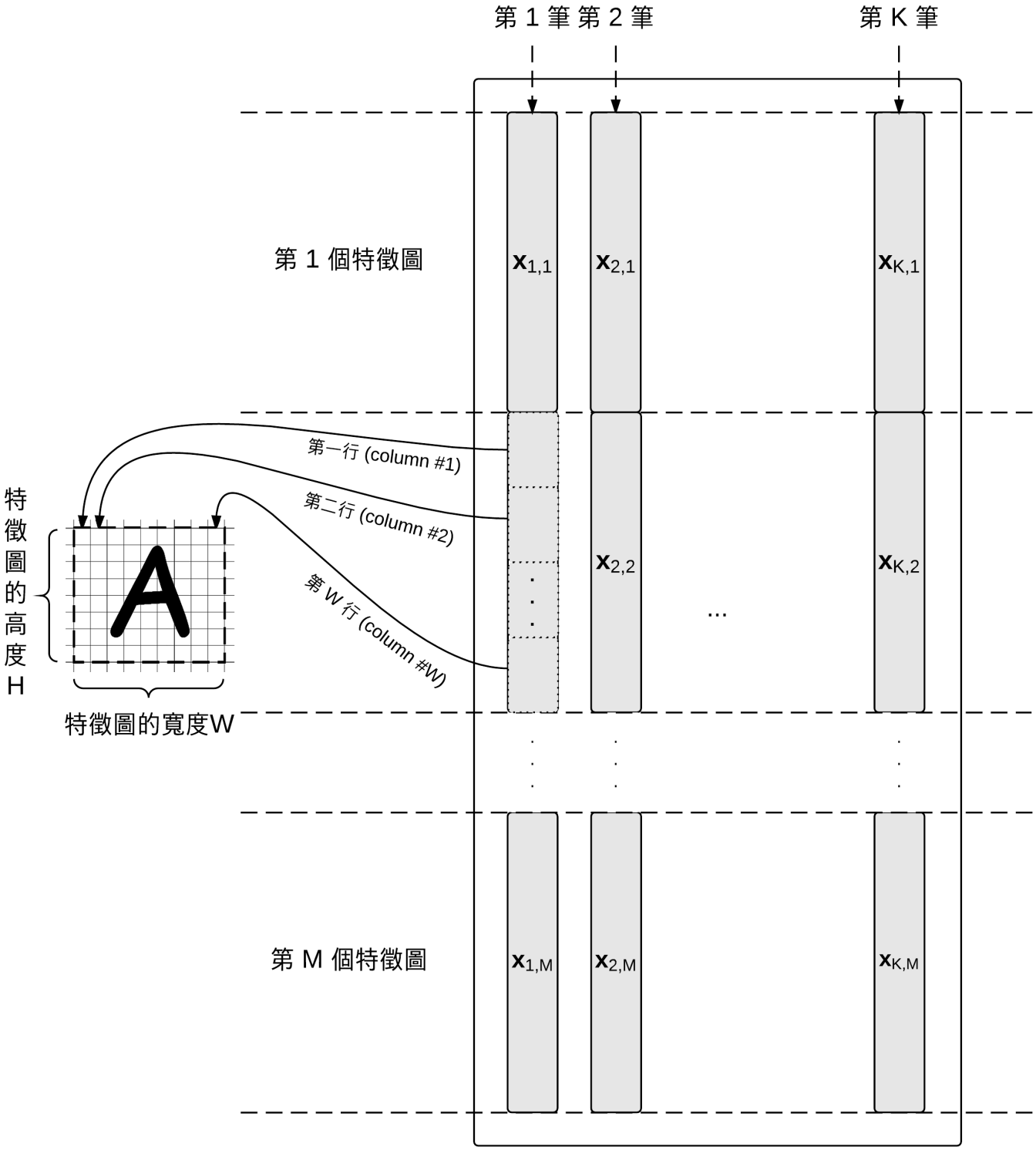


圖 6.2: 深層卷積類神經網路中的記憶體佈局。

6.3.2 性能調校與優化

一般來說，圖形處理器中數個至十數個的串流多處理器 (streaming multiprocessor, SM) ，負責管理與指揮數百至數千個的計算核心 (CUDA cores) 。 以 NVIDIA GeForce GTX 980 為例，GTX 980 有 16 個串流多處理器，共 2048 個計算核心，其運算巔峰效能 (peak performance) 可達每秒 4,616 個拾億次 (Giga) 的浮點運算 (Floating-point operations per second, FLOPS) ，約為一般個人電腦中，相同價位的中央處理器的 50 至 100 倍。 為了汲取圖形處理器的最大效能，首要目標便是想盡辦法避免這些計算核心處於閒置 (idle) 的狀態。 以類神經網路中的運算作為例子 (見圖 6.1) ，若是以一筆資料作為一批，每經過一次仿射轉換，需計算一次向量矩陣乘法，但光這樣並不足以塞滿圖形處理器中數以千計的計算核心。而若是以 128 筆或 256 筆資料作為一批，則經過一次仿射轉換時，所需的矩陣矩陣乘法，相當於 128 個同時運行向量矩陣乘法，以避免閒置的狀況發生。

暫存

在統一計算架構中，程式每次透過 `cudaMalloc` 跟圖形處理器取得記憶體時，或是透過 `cudaFree` 釋放記憶體時，系統便會自動執行一次 `cudaDeviceSynchronize` 。 執行此操作時，會阻擋 (block) 任何尚未執行的指令進到圖形處理器中進行計算，直到所有正在執行的運算都完成並且成功取得或釋放記憶體為止，因此大幅地降低程式的執行效能。 透過暫存 (cache) 的作法，將每次釋放的記憶體暫存起來，略過本來要執行的 `cudaFree` ，便可省下一次 `cudaDeviceSynchronize` ，而下次需要記憶體時，便將前次暫存起來的記憶體回傳給該程式，略過本來要執行的 `cudaMalloc` ，再省下一次 `cudaDeviceSynchronize` ，如此便可以大幅地增加效能。

6.4 本章總結

本章介紹了如何使用 **libdnn** 這套工具訓練深層類神經網路及卷積類神經網路，同時探討了程式架構與記憶體佈局，並提出了利用暫存和批次處理的方式，解決圖形處理器運算中可能遇到的效能瓶頸。

第 七 章　結論與展望

7.1　總結

深層類神經網路由於其傑出的效能，近年來已廣泛地被應用在大字彙連續語音辨識及其他語音處理相關的研究中。本論文使用深層類神經網路與卷積類神經網路取代傳統的高斯混合模型，作為語音辨識中的聲學模型，並探討了如何在基於深層類神經網路的聲學模型上進行語者調適。

在大字彙連續語音辨識中，以深層類神經網路或是深層卷積類神經網路作為聲學模型的作法，相較於傳統的高斯混合模型的基礎實驗，分別少了 19.6% 與 14.7% 的音素錯誤率。此外，藉由丟棄法正規化，可以避免類神經網路模型在訓練語料上過度貼合的狀況發生，取得更進一步的提升，

在深層類神經網路上進行語者調適的情境中，我們利用分群的概念，將隱藏式馬可夫模型中相似的狀態分成一群，並分別利用特徵空間上的鑑別式線性迴歸，將這些原本相似的狀態分得更開。透過兩階段式的解碼以及線性內插的方式，我們混合了傳統特徵空間上的鑑別式線性迴歸中單一仿射轉換的作法，有效地改善了過度貼合的問題，更進一步地提升語者調適模型的效能表現。實驗結果顯示，相較於原本的特徵空間上的鑑別式線性迴歸，及另一套基於奇異值分解的語者調適演算法，不論是在朗讀式或是自發性的語音、還是在數量不同的語者調適語料上，本論文提出的方法都能有最佳且一致性的表現。

最後，我們也介紹了 **libdnn** 這套使用圖形處理器加速的深層類神經網路函數庫，並詳細地記載基礎的使用方法、程式碼架構、以及實作細節等，以供後人參考之用，並期待這套工具能夠在眾人的參與及開發之下更臻完善。

7.2 未來展望

近年來有越來越多的研究團隊，試圖運用遞迴式神經網路、長短期記憶及其他非順向傳遞式的類神經網路架構，作為大字彙連續語音辨識中的聲學模型，或是對付自然語言處理 (natural language processing, NLP) 及數位語音處理中各式各樣的問題。此外，也有相當多正規化的研究，希望受過訓練的類神經網路能夠更加一般化 (generalization) ，在測試資料上有更高的準確率。

我們在此提出以下幾點作為未來的改進方向，為通篇做結。

1. 從數學及理論的角度出發，詳細地分析與探討類神經網路中各種活化函數及正規化的優點與缺點，使得類神經網路的訓練不再只是透過經驗法則，而是有一套標準化的訓練流程。並找出一套自動選擇卷積核心的大小及輸出特徵圖數量的演算法，改善現有的卷積類神經網路訓練流程，以及運用類似自動編碼機 (autoencoder) [61] 的概念，學習出更有效的編碼方式，為深層卷積類神經網路中的卷積核心找到更好的參數起始點，以減少深層卷積網路的訓練時間與失敗的次數。

2. 在英文連續大字彙語料例如 Aurora 4 上進行語者調適實驗，以更進一步的驗證本論文中提出的基於狀態分群的特徵空間上鑑別式線性迴歸。

3. 持續地擴充並更新 **libdnn** 這套工具。首先，將原本深層類神經網路或卷積類神經網路中用來儲存特徵轉換的容器 (container) ，從原本 C++ 標準模板庫 (standard template library, STL) 中的 `std::vector` ，改成以有向圖 (directed graph) 的結構儲存，讓這套工具能夠支援各式各樣的類神經網路架構，如遞迴式類神經網路及長短期記憶等，並利用 HTML5 及 JavaScript 等技術，製作出跨平台的圖形使用者界面 (graphical user interface, GUI) ，提供

使用者更簡單的方式繪製出各式各樣的類神經網路架構。此外，也新增對多圖形處理器的支援，以加速類神經網路的訓練，使其可以更快速地佈署在 Amazon 或 Google 所提供的雲端計算平台上。

參 考 文 獻

[1] Raúl Rojas, *Neural Networks: A Systematic Introduction*, Springer-Verlag New York, Inc., New York, NY, USA, 1996.

[2] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[3] James Baker, "The dragon system–an overview," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 23, no. 1, pp. 24–29, 1975.

[4] Lawrence Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[5] Sadaoki Furui, *Digital speech processing: synthesis, and recognition*, CRC Press, 2000.

[6] Janet Baker, Li Deng, James Glass, Sanjeev Khudanpur, Chin-Hui Lee, Nelson Morgan, and Douglas O'Shaughnessy, "Developments and directions in speech recognition and understanding, part 1 [dsp education]," *Signal Processing Magazine, IEEE*, vol. 26, no. 3, pp. 75–80, 2009.

[7] B-H Juang, "Maximum-likelihood estimation for mixture multivariate stochastic observations of markov chains," *AT&T technical journal*, vol. 64, no. 6, pp. 1235–1249, 1985.

[8] Richard P Lippmann, "An introduction to computing with neural nets," *ASSP Magazine, IEEE*, vol. 4, no. 2, pp. 4–22, 1987.

[9] Herve A Bourlard and Nelson Morgan, *Connectionist speech recognition: a hybrid approach*, vol. 247, Springer, 1994.

[10] Geoffrey Hinton, Simon Osindero, and Yee-Whye Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[11] John Nickolls, Ian Buck, Michael Garland, and Kevin Skadron, "Scalable parallel programming with cuda," *Queue*, vol. 6, no. 2, pp. 40–53, 2008.

[12] Abdel-rahman Mohamed, Tara N Sainath, George Dahl, Bhuvana Ramabhadran, Geoffrey E Hinton, and Michael A Picheny, "Deep belief networks using discriminative features for phone recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*. IEEE, 2011, pp. 5060–5063.

[13] Abdel-rahman Mohamed, George E Dahl, and Geoffrey Hinton, "Acoustic modeling using deep belief networks," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 20, no. 1, pp. 14–22, 2012.

[14] George E Dahl, Dong Yu, Li Deng, and Alex Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 20, no. 1, pp. 30–42, 2012.

[15] George Dahl, Abdel-rahman Mohamed, Geoffrey E Hinton, et al., "Phone recognition with the mean-covariance restricted boltzmann machine," in *Advances in neural information processing systems*, 2010, pp. 469–477.

[16] Yann LeCun, Fu Jie Huang, and Leon Bottou, "Learning methods for generic object recognition with invariance to pose and lighting," in *Computer Vision and Pattern*

Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on. IEEE, 2004, vol. 2, pp. II–97.

[17] Yann LeCun and Yoshua Bengio, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, 1995.

[18] Honglak Lee, Peter Pham, Yan Largman, and Andrew Y Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks," in *Advances in neural information processing systems*, 2009, pp. 1096–1104.

[19] Ossama Abdel-Hamid, Abdel-rahman Mohamed, Hui Jiang, and Gerald Penn, "Applying convolutional neural networks concepts to hybrid nn-hmm model for speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, 2012, pp. 4277–4280.

[20] Frank Seide, Gang Li, Xie Chen, and Dong Yu, "Feature engineering in context-dependent deep neural networks for conversational speech transcription," in *Automatic Speech Recognition and Understanding (ASRU), 2011 IEEE Workshop on*. IEEE, 2011, pp. 24–29.

[21] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams, "Learning representations by back-propagating errors," *Cognitive modeling*, 1988.

[22] Jeff A Bilmes et al., "A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models," *International Computer Science Institute*, vol. 4, no. 510, pp. 126, 1998.

[23] Lalit R Bahl, PV deSouza, PS Gopalakrishnan, D Nahamoo, and MA Picheny, "Decision trees for phonological rules in continuous speech," in *Acoustics, Speech, and Signal Processing, 1991. ICASSP-91., 1991 International Conference on*. IEEE, 1991, pp. 185–188.

[24] Steve J Young, JJ Odell, and Philip C Woodland, "Tree-based state tying for high accuracy acoustic modelling," in *Proceedings of the workshop on Human Language Technology*. Association for Computational Linguistics, 1994, pp. 307–312.

[25] "教育部重編國語辭典修訂本," `http://dict.revised.moe.edu.tw/`, Accessed: 2014-02-09.

[26] Tomas Mikolov, Stefan Kombrink, Anoop Deoras, Lukar Burget, and Jan Cernocky, "Rnnlm-recurrent neural network language modeling toolkit," .

[27] Tomas Mikolov, Anoop Deoras, Stefan Kombrink, Lukas Burget, and Jan Cernockỳ, "Empirical evaluation and combination of advanced language modeling techniques.," in *INTERSPEECH*, 2011, number s 1, pp. 605–608.

[28] John Duchi, Elad Hazan, and Yoram Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *The Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011.

[29] David L Donoho, "For most large underdetermined systems of linear equations the minimal l1-norm solution is also the sparsest solution," *Communications on pure and applied mathematics*, vol. 59, no. 6, pp. 797–829, 2006.

[30] Trevor Hastie, Robert Tibshirani, Jerome Friedman, T Hastie, J Friedman, and R Tibshirani, *The elements of statistical learning*, vol. 2, Springer, 2009.

[31] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[32] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[33] Ludmila I Kuncheva and Christopher J Whitaker, "Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy," *Machine learning*, vol. 51, no. 2, pp. 181–207, 2003.

[34] Peter Sollich Anders Krogh, "Learning with ensembles: How over-fitting can be useful," in *Proceedings of the 1995 Conference*, 1996, vol. 8, p. 190.

[35] Najim Dehak, Patrick Kenny, Réda Dehak, Pierre Dumouchel, and Pierre Ouellet, "Front-end factor analysis for speaker verification," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 19, no. 4, pp. 788–798, 2011.

[36] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides, *Design patterns: elements of reusable object-oriented software*, Pearson Education, 1994.

[37] Anthony J Robinson, "An application of recurrent nets to phone probability estimation," *Neural Networks, IEEE Transactions on*, vol. 5, no. 2, pp. 298–305, 1994.

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

[39] Sadaoki Furui, "Cepstral analysis technique for automatic speaker verification," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 29, no. 2, pp. 254–272, 1981.

[40] Kunihiko Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological cybernetics*, vol. 36, no. 4, pp. 193–202, 1980.

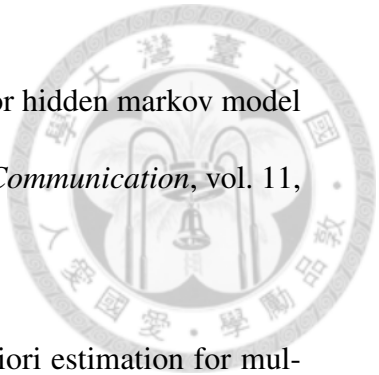
[41] Ossama Abdel-Hamid, Li Deng, and Dong Yu, "Exploring convolutional neural network structures and optimization techniques for speech recognition.," in *INTERSPEECH*, 2013, pp. 3366–3370.

[42] CJ Leggetter and Philip C Woodland, "Speaker adaptation of continuous density hmms using multivariate linear regression.," in *ICSLP*, 1994, vol. 94, pp. 451–454.

[43] Christopher J Leggetter and Philip C Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models," *Computer Speech & Language*, vol. 9, no. 2, pp. 171–185, 1995.

[44] Mark JF Gales and Philip C Woodland, "Mean and variance adaptation within the mllr framework," *Computer Speech & Language*, vol. 10, no. 4, pp. 249–264, 1996.

[45] Mark JF Gales, "Maximum likelihood linear transformations for hmm-based speech recognition," *Computer speech & language*, vol. 12, no. 2, pp. 75–98, 1998.



[46] Jean-Luc Gauvain and Chin-Hui Lee, "Bayesian learning for hidden markov model with gaussian mixture state observation densities," *Speech Communication*, vol. 11, no. 2, pp. 205–213, 1992.

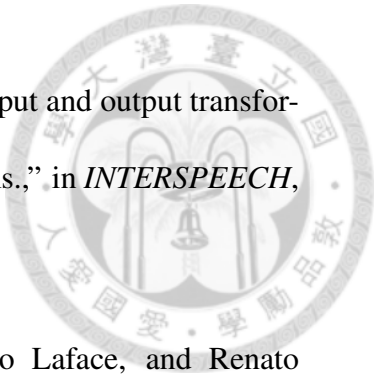
[47] Jean-Luc Gauvain and Chin-Hui Lee, "Maximum a posteriori estimation for multivariate gaussian mixture observations of markov chains," *Speech and audio processing, ieee transactions on*, vol. 2, no. 2, pp. 291–298, 1994.

[48] G Zavaliagkos, R Schwartz, and John McDonough, "Maximum a posteriori adaptation for large scale hmm recognizers," in *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on*. IEEE, 1996, vol. 2, pp. 725–728.

[49] Michael Finke, Petra Geutner, Hermann Hild, Thomas Kemp, Klaus Ries, and Martin Westphal, "The karlsruhe-verbmobil speech recognition engine," in *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on*. IEEE, 1997, vol. 1, pp. 83–86.

[50] Roland Kuhn, Jean-Claude Junqua, Patrick Nguyen, and Nancy Niedzielski, "Rapid speaker adaptation in eigenvoice space," *Speech and Audio Processing, IEEE Transactions on*, vol. 8, no. 6, pp. 695–707, 2000.

[51] Joao Neto, Luís Almeida, Mike Hochberg, Ciro Martins, Luís Nunes, Steve Renals, and Tony Robinson, "Speaker-adaptation for hybrid hmm-ann continuous speech recognition system," 1995.



[52] Bo Li and Khe Chai Sim, "Comparison of discriminative input and output transformations for speaker adaptation in the hybrid nn/hmm systems.," in *INTERSPEECH*, 2010, pp. 526–529.

[53] Roberto Gemello, Franco Mana, Stefano Scanzio, Pietro Laface, and Renato De Mori, "Linear hidden transformations for adaptation of hybrid ann/hmm models," *Speech Communication*, vol. 49, no. 10, pp. 827–835, 2007.

[54] Jian Xue, Jinyu Li, Dong Yu, Mike Seltzer, and Yifan Gong, "Singular value decomposition based low-footprint speaker adaptation and personalization for deep neural network," .

[55] Tara N Sainath, Brian Kingsbury, Vikas Sindhwani, Ebru Arisoy, and Bhuvana Ramabhadran, "Low-rank matrix factorization for deep neural network training with high-dimensional output targets," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 6655–6659.

[56] Jian Xue, Jinyu Li, and Yifan Gong, "Restructuring of deep neural network acoustic models with singular value decomposition.," in *INTERSPEECH*, 2013, pp. 2365–2369.

[57] George Saon, Hagen Soltau, David Nahamoo, and Michael Picheny, "Speaker adaptation of neural network acoustic models using i-vectors," in *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*. IEEE, 2013, pp. 55–59.

[58] Chih-Chung Chang and Chih-Jen Lin, "Libsvm: a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, pp. 27, 2011.

[59] Katta G Murty and Santosh N Kabadi, "Some np-complete problems in quadratic and nonlinear programming," *Mathematical programming*, vol. 39, no. 2, pp. 117–129, 1987.

[60] Xavier Glorot and Yoshua Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *International Conference on Artificial Intelligence and Statistics*, 2010, pp. 249–256.

[61] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 1096–1103.

附　　錄

附錄一

首先，假設兩個向量變數 $\mathbf{y}$ 及 $\mathbf{x}$ 之間的關係可以用一個矩陣 $\mathbf{W}$ 來表達

$$\mathbf{y} = \mathbf{W} \cdot \mathbf{x} \tag{A.1}$$

給定一個純量 E ，並假設 E 是 $\mathbf{y}$ 的函數，寫成 $E = E(\mathbf{y})$ 。根據連鎖律，函數 E 對於 x_i 的偏微分可以寫成

$$\frac{\partial E}{\partial x_i} = \sum_{j=1}^{m} \frac{\partial y_j}{\partial x_i} \cdot \frac{\partial E}{\partial y_j} \tag{A.2a}$$

$$= \sum_{j=1}^{m} w_{ji} \cdot \frac{\partial E}{\partial y_j} \qquad (\because y_j = \sum_{i=1}^{m} w_{ji} x_i) \tag{A.2b}$$

將上式重新寫成向量矩陣形式便可得到

$$\frac{\partial E}{\partial \mathbf{x}} = \left(\frac{\partial \mathbf{y}}{\partial \mathbf{x}} \right)^T \cdot \frac{\partial E}{\partial \mathbf{y}} = \mathbf{W}^T \cdot \frac{\partial E}{\partial \mathbf{y}}. \tag{A.3}$$

純量函數 E 當然也可以視為 $\mathbf{W}$ 的函數，亦即 $E = E(\mathbf{W})$ 。根據連鎖律，函數 E 對 w_{ji} 的偏微分可以寫成

$$\frac{\partial E}{\partial w_{ji}} = \sum_{l=1}^{m} \frac{\partial E}{\partial y_l} \cdot \frac{\partial y_l}{\partial w_{ji}} \tag{A.4a}$$

$$= \sum_{l=1}^{m} \frac{\partial E}{\partial y_l} \cdot \frac{\sum_{k=1}^{n} w_{lk} x_k}{\partial w_{ji}} \tag{A.4b}$$

其中右手邊的 $\frac{\sum_{k=1}^{n} w_{lk}x_k}{\partial w_{ji}}$ 只有在 $i=k, j=l$ 時才有值 x_k ，否則為 0 。 因此我們可以將 $i=k, j=l$ 帶入上式，把 $\sum_{l}^{m}$ 去掉得到

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial y_j} \cdot x_i \tag{A.5}$$

有了 $\frac{\partial E}{\partial \mathbf{W}}$ 中的每個元素 $\frac{\partial E}{\partial w_{ji}}$ ，我們便可以將 $m \times n$ 的矩陣 $\frac{\partial E}{\partial \mathbf{W}}$ 寫成一個由 $m \times 1$ 和 $1 \times n$ 向量相乘所展開的乘積，如下所示：

$$\frac{\partial E}{\partial \mathbf{W}} = \begin{bmatrix} \frac{\partial E}{\partial y_1} \\ \frac{\partial E}{\partial y_2} \\ \vdots \\ \frac{\partial E}{\partial y_m} \end{bmatrix} \cdot \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix} = \frac{\partial E}{\partial \mathbf{y}} \cdot \mathbf{x}^T \tag{A.6}$$

附錄二

給定一個純量 E ，並假設 E 是一個二維離散訊號 $\mathbf{Y}$ 的函數，寫成 $E = E(\mathbf{Y})$ ，且 $\mathbf{Y}$ 、 $\mathbf{X}$ 、 $\mathbf{K}$ 這三個二維的離散訊號滿足式 4.2 所描述的卷積關係。根據連鎖律，函數 E 對於 $\mathbf{X}[a,b]$ 的偏微分可以寫成

$$\frac{\partial E}{\partial \mathbf{X}[a,b]} \tag{B.7a}$$

$$= \sum_{\alpha=0}^{H_y-1} \sum_{\beta=0}^{W_y-1} \frac{\partial E}{\partial \mathbf{Y}[\alpha,\beta]} \frac{\partial \mathbf{Y}[\alpha,\beta]}{\partial \mathbf{X}[a,b]} \tag{B.7b}$$

$$= \sum_{\alpha=0}^{H_y-1} \sum_{\beta=0}^{W_y-1} \frac{\partial E}{\partial \mathbf{Y}[\alpha,\beta]} \frac{\partial \left(\sum_{\gamma=0}^{H_k-1} \sum_{\delta=0}^{W_k-1} \mathbf{K}[H_k-1-\gamma, W_k-1-\delta] \mathbf{X}[\alpha+\gamma, \beta+\delta] + b_j \right)}{\partial \mathbf{X}[a,b]} \tag{B.7c}$$

透過簡單的觀察即可發現，上式右手邊第二項只有在 $\alpha+\gamma=a$ 及 $\beta+\delta=b$ 時，才會有非零的微分值 $\mathbf{K}_{ij}[H_k-1-\gamma, W_k-1-\delta]$。因此，將 γ 及 δ 帶回上式可得

$$\frac{\partial E}{\partial \mathbf{X}[a,b]} = \sum_{\alpha=0}^{H_y-1}\sum_{\beta=0}^{W_y-1} \frac{\partial E}{\partial \mathbf{Y}[\alpha,\beta]} \mathbf{K}[H_k-1-(a-\alpha), W_k-1-(b-\beta)] \tag{B.8a}$$

$$= \sum_{\alpha=0}^{H_y-1}\sum_{\beta=0}^{W_y-1} \frac{\partial E}{\partial \mathbf{Y}[\alpha,\beta]} \mathbf{K}^{180^\circ}[a-\alpha, b-\beta] \tag{B.8b}$$

$$= \frac{\partial E}{\partial \mathbf{Y}} \overset{full}{*} \mathbf{K}^{180^\circ} \tag{B.8c}$$

其中預算子 $\overset{full}{*}$ 是指以 $\frac{\partial E}{\partial \mathbf{Y}}$ 為中心，將周圍以 0 填滿 (padding) ，並輸出與 $\mathbf{X}$ 相同大小的結果的卷積運算。

根據連鎖律，函數 E 對 $\partial \mathbf{K}[a,b]$ 的偏微分可以寫成

$$\frac{\partial E}{\partial \mathbf{K}[a,b]} \tag{B.9a}$$

$$= \sum_{\alpha=0}^{H_y-1}\sum_{\beta=0}^{W_y-1} \frac{\partial E}{\partial \mathbf{Y}[\alpha,\beta]} \frac{\partial \mathbf{Y}[\alpha,\beta]}{\partial \mathbf{K}[a,b]} \tag{B.9b}$$

$$= \sum_{\alpha=0}^{H_y-1}\sum_{\beta=0}^{W_y-1} \frac{\partial E}{\partial \mathbf{Y}[\alpha,\beta]} \frac{\partial \left(\sum_{\gamma=0}^{H_k-1}\sum_{\delta=0}^{W_k-1} \mathbf{K}[H_k-1-\gamma, W_k-1-\delta]\mathbf{X}[\alpha+\gamma,\beta+\delta] + b_j \right)}{\partial \mathbf{K}[a,b]} \tag{B.9c}$$

同樣地，上式右手邊第二項只有在 $H_k-1-\gamma=a$ 且 $W_k-1-\delta=b$ 時，才有非零的微分值 $\mathbf{X}_i[\alpha+\gamma,\beta+\delta]$ 。因此，將 γ 及 δ 帶回上式即可得到：

$$\frac{\partial E}{\partial \mathbf{K}[a,b]} = \sum_{\alpha=0}^{H_y-1}\sum_{\beta=0}^{W_y-1} \frac{\partial E}{\partial \mathbf{Y}[\alpha,\beta]} \cdot \mathbf{X}[\alpha+H_k-1-a, \beta+W_k-1-b] \tag{B.10}$$

接著將 $\mathbf{Y}, \mathbf{X}, \mathbf{K}$ 三者大小的關係式 4.3b 帶入上式，即可簡化成

$$\frac{\partial E}{\partial \mathbf{K}} = \mathbf{X}^{180^\circ} * \frac{\partial E}{\partial \mathbf{Y}} \tag{B.11}$$

附錄三　活化函數

常見的活化函數包括了：

1. S型函數 (sigmoid)
2. 雙曲正切函數 (hyperbolic tangent)
3. 整流線性單元 (rectified linear unit)
4. 平滑加法 (softplus)
5. 平滑最大值活化函數 (softmax)

上述活化函數，除了平滑最大值活化函數以外，其餘都是逐元素 (element-wise) 的運算。

S型函數

S 型函數是最常見的活化函數（見圖 7.1)，可以表示如下：

$$y = \frac{1}{1 + e^{-x}} \tag{C.12a}$$

$$\frac{\partial y}{\partial x} = \frac{e^{-x}}{(e^{-x} + 1)^2} = y \cdot (1 - y) \tag{C.12b}$$

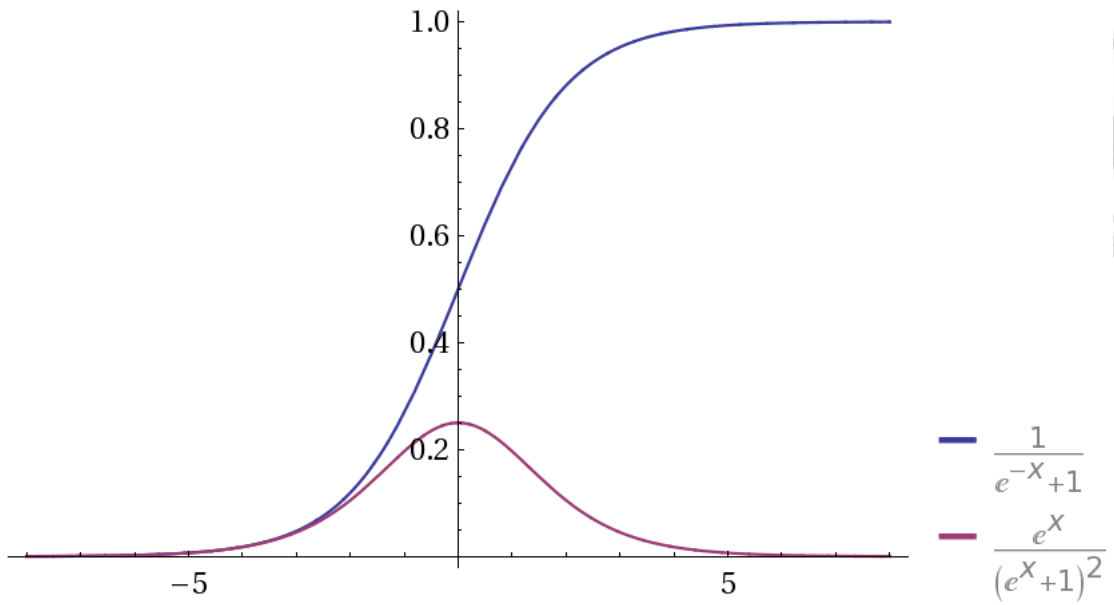


圖 7.1: S 型函數（藍色線）及其微分函數（紅色線）

雙曲正切函數

另一個常見的活化函數則為雙曲正切函數 (hyperbolic tangent) ，其函數圖形與前述的 S 型函數非常相似（如圖 7.2 所示），其式子如下：

$$y = tanh(x) = \frac{e^{2x} - 1}{e^{2x} + 1} \tag{C.13a}$$

$$\frac{\partial y}{\partial x} = 1 - tanh^2(x) = 1 - y^2 \tag{C.13b}$$

整流線性單元

整流線性單元 (rectified linear units) 是為了要改進在 S 型函數及雙曲正切函數中，在離零點較遠的兩側微分值過小，而造成活化函數的值一直處在 0 或是一直處在

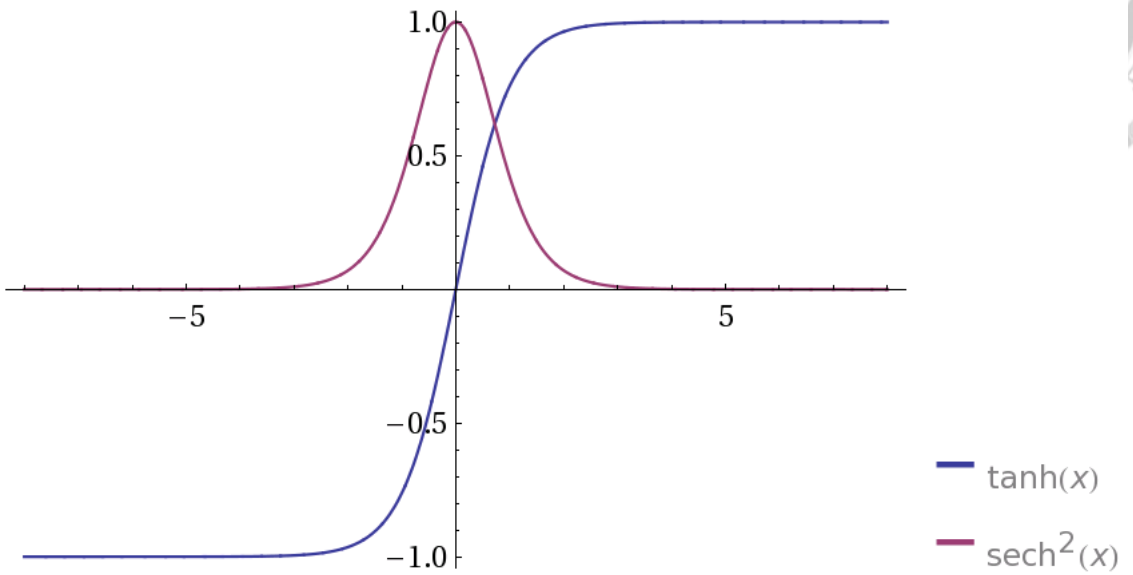


圖 7.2: 雙曲正切函數（藍色線）及其微分函數（紅色線）

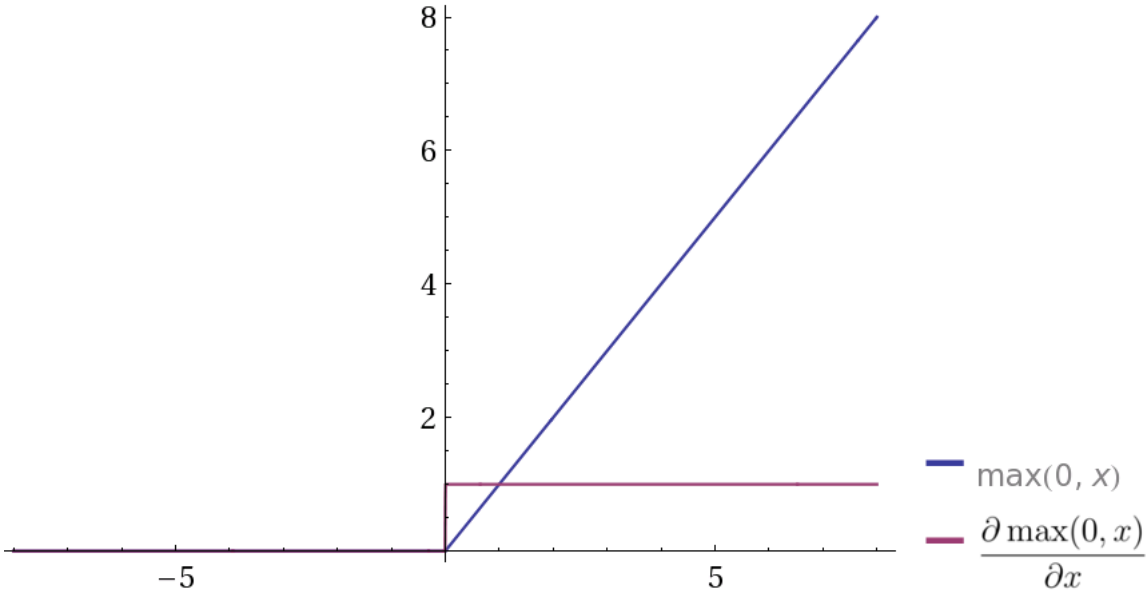


圖 7.3: 整流線性單元（藍色線）及其微分函數（紅色線）

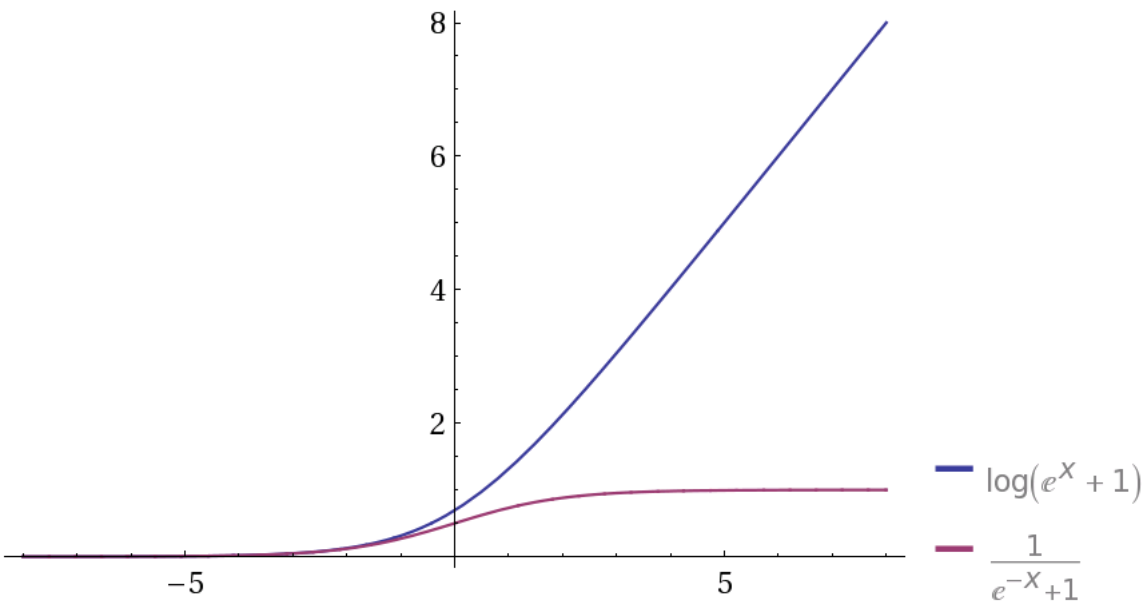


圖 7.4: 平滑加法（藍色線）及其微分函數（紅色線）

1 的問題的一種活化函數 (見圖7.3) ，其式子如下所示：

$$y = max(0, x) \tag{C.14a}$$

$$\frac{\partial y}{\partial x} = \begin{cases} 1, & \text{if } x > 0 \\ 0, & \text{otherwise} \end{cases} \tag{C.14b}$$

平滑加法

平滑加法是對整流線性單元的改進（見圖 7.4 ）。 由於整流線性單元其微分值在 $x < 0$ 時為 0 ，常常會導致後級錯誤訊號沒辦法被傳遞至前端，而導致輸入訊號 x 被困在零點左側難以跨越的狀況。透過這項改進，平滑加法能在零點有較滑順的

微分值，並同時保留整流線性單元的優點，如下所示：

$$y = \log(1 + \exp^{x}) \tag{C.15a}$$

$$\frac{\partial y}{\partial x} = sigmoid(x) \tag{C.15b}$$

平滑最大值活化函數

在神經網路中，如果預測的目標函數 **t** 是一個機率分佈，通常會選用交叉熵當作損失函數，並搭配平滑最大值活化函數放在深層類神經網路的最後一層一併使用。其式如下：

$$y_j = \frac{e^{x_j}}{\sum_{i=1}^{m} e^{x_i}} \tag{C.16a}$$

$$\frac{\partial y_j}{\partial x_i} = y_j(\delta_{ij} - y_i) \tag{C.16b}$$

其中 δ_{ij} 是克羅內克函數 (Kronecker delta function) ，只有當 $i = j$ 時才有值 1 ，而當 $i \neq j$ 時其值為 0 。 此時交叉熵損失函數對模型預測結果 o_j （即前級輸出 y_j ）的偏微分為：

$$\frac{\partial}{\partial o_j} E = \frac{\partial \left(-\sum_{i=1}^{m} t_i \cdot \ln(o_i) \right)}{\partial o_j} \tag{C.17a}$$

$$= -\frac{t_i}{o_i} \tag{C.17b}$$

根據上式，我們可以將反向傳播演算法中，平滑最大值活化函數的前級錯誤訊號 $\frac{\partial E}{\partial \mathbf{x}}$ 寫成：

$$\frac{\partial}{\partial x_i} E = \sum_{j=1}^{m} \frac{\partial E}{\partial y_j} \cdot \frac{\partial y_j}{\partial x_i} \tag{C.18a}$$

$$= \sum_{j=1}^{m} \frac{-t_j}{y_j} \cdot y_j(\delta_{ij} - y_i) \tag{C.18b}$$

$$= -\sum_{j=1}^{m} t_j(\delta_{ij} - y_i) \tag{C.18c}$$

$$= -\sum_{j=1}^{m} t_j \delta_{ij} + y_i \sum_{j=1}^{m} t_j \tag{C.18d}$$

其中第一項只有在 $i = j$ 時才有值，第二項則因為我們期望的目標 $\mathbf{t}$ 是一個機率分佈，其總和 $\sum_{j=1}^{m} t_j$ 必為 1 ，帶入上式可得：

$$\frac{\partial}{\partial x_i} E = y_i - t_i \tag{C.19}$$

附錄四

地區	男性語者（人）	女性語者（人）	總語者數（人）	各區比例 (%)
New England	31	18	49	8
Northern	71	31	102	16
North Midland	79	23	102	16
South Midland	69	31	100	16
Southern	62	36	98	16
New York City	30	16	46	7
Western	74	26	100	16
Army Brat	22	11	33	5

表 7.1: TIMIT 語料中各方言之男女語者人數及比例分佈