國立陽明交通大學

工業工程與管理學系

碩士論文

Department of Industrial Engineering and Management

National Yang Ming Chiao Tung University

Master Thesis

模糊倒傳遞類神經網路於晶圓廠之晶圓批生產週期時間預測

A fuzzy back-propagation neural network for lot cycle time prediction in a wafer fab

研究生： 許庭銓 (Hsu, Ting Chuan)

指導教授： 陳亭志 (Chen, Tin Chih)

中華民國一一零年六月

June 2021

模糊倒傳遞類神經網路於晶圓廠之晶圓批生產週期時間預測

A fuzzy back-propagation neural network for lot cycle time prediction in a wafer fab

研究生：許庭銓　　Student：Ting Chuan Hsu
指導教授：陳亭志 博士　　Advisor：Dr. Tin Chih Chen

國立陽明交通大學

工業工程與管理學系

碩士論文

A Thesis
Submitted to Department of Industrial Engineering and Management
College of Management
National Yang Ming Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Master of Science
in
Industrial Engineering and Management

June 2021
Taiwan, Republic of China

中華民國一一零年六月

模糊倒傳遞類神經網路於晶圓廠之晶圓批生產週期時間預測

學生：許庭銓　　指導教授：陳亭志

國立陽明交通大學工業工程與管理學系(研究所)碩士班

摘　　要

晶圓批之生產週期時間指該晶圓批從投入到產出所歷經的時間。晶圓批週期時間的預測對於晶圓廠而言是至關重要的任務，然而現有的預測方法並不能達到足夠高的準確率。因此，若能夠預測生產週期時間的範圍，將會很有參考價值。為此目的，本研究結合模糊理論與倒傳遞類神經網路。首先，透過輸入與一晶圓批相關的變數進入一類神經網路模型中，以得到生產週期時間之預測值。接著，藉由模糊化類神經網路中的參數以產生生產週期時間之區間。此方法與傳統統計或最佳化方法相比，節省了許多計算的時間且不須許多假設。在保持高準確率的同時，生產週期時間之區間寬度也不會過度增加。實驗結果顯示，在準確率相近的條件下，相較於統計方法，本研究所提出方法大約提升了百分之八十的預測精確度。

關鍵字：類神經網路、倒傳遞類神經網路、模糊理論、模糊類神經網路、生產週期時間預測、模糊區間

A fuzzy back-propagation neural network for lot cycle time prediction in a wafer fab

Student：Ting-Chuan Hsu　　Advisor：Dr. Tin-Chih Chen

Department of Industrial Engineering and Management
National Yang Ming Chiao Tung University

Abstract

The cycle time of a wafer lot refers to the time that the wafer lot goes through the factory. Predicting the cycle time of a wafer lot is a critical task of a wafer fab. However, existing prediction methods may not achieve sufficient accuracy. Therefore, if the range of cycle time can be estimated, it will provide valuable information. To this end, this study combines fuzzy theory and back-propagation neural network, to generate fuzzy cycle time forecast by fuzzifying parameters in a back-propagation neural network. Variables related to wafer lot production are inputted into this fuzzy neural network model to estimate the value and range of cycle time. Compared with statistical or optimization methods, the proposed method does not require a lot of assumptions and saves computing time. While maintaining high accuracy, the cycle time interval will not be expanded excessively. Experimental results showed that the proposed methodology narrowed the average range of cycle times by 80% than existing methods.

Keywords: neural network, back-propagation neural network, fuzzy theory, fuzzy neural network, prediction of cycle time, fuzzy interval

目錄

圖目錄

表目錄

第一章 緒論

1.1 研究背景

半導體產業可以說是 21 世紀最重要的產業之一，在日常生活中處處可見內含半導體的產品，舉凡像是現今人手一台的手機、電腦，甚至在交通工具方面，高鐵、飛機的控制裝置等都是半導體的應用。在半導體產業方面，台灣半導體產值在 2019 年已達 2.7 兆台幣，位居全球第二名，並佔台灣該年度 GDP 的約 15%，可見半導體產業是台灣不可或缺的經濟成長關鍵(經濟日報, 2020)。新興產業的發展也為半導體產業帶來更多的需求。高通資深副總裁陳若文表示未來 5G 之發展不僅是手機的應用，也將攜及到智慧製造及自駕車的應用，這些應用都會增加市場對半導體晶片的需求(財經新報, 2020)。台灣 2020 年 1 至 9 月積體電路之產值為 1 兆 2,755 億元，創下歷年同期新高，年增率 23.2%；半導體封裝測試產值達 3,626 億元，亦為歷年同期新高，年增率 4.5%(經濟日報, 2020)。另外，世界半導體貿易組織(World Semiconductor Trade Statistics, WSTS)估計，2021 年全球半導體之市場規模將同比增長 8.4%，達到 4694 億美元，創歷史新高。可看出半導體不僅在台灣，甚至在全世界都有著十分蓬勃的發展(新浪新聞, 2020)。

一晶圓批之產出時間為該晶圓批從投入到產出所歷經的時間，晶圓批產出時間的預測對於晶圓廠而言是至關重要的任務，預測晶圓廠中每個批次的產出時間不僅對晶圓廠本身、對客戶而言都是十分重要的議題(Chen, 2006)。若能準確預測晶圓廠中每批次的產出時間，將可以讓企業遂行幾個管理目標，其中包括內部預期產出日期之決定、產出預測、協助訂單決策、增強客戶關係以及指導後續作業等(Chen, 2006)。估計晶圓批的產出時間等同於估計晶圓批的生產週期時間(cycle time)，因為前者可透過將後者加上投產時間(常數)來求得。晶圓製造廠是一個極為複雜的生產系統，有許多因素會影響最終的生產週期時間，例如需求的變化、不同的產品組合、生產優先順序、設備機台的可靠度、產能不均等。一晶圓批的生產週期時間往往長達數個月、標準差亦高達數百小時。因此許多研究指出要精準地預測在此類大型系統中的生產週期時間是非常困難的(Chen,

2013)。

類神經網路已被廣泛運用至各種領域。例如，Wilson (1994)利用類神經網路來預測公司是否會破產，與傳統判別分析(discriminant analysis)方法做比較後，發現類神經網路的表現明顯優於判別分析。Li (2010)提出一倒傳遞類神經網路模型用於預測校園內建築物之維護成本，由於維護管理方面成本往往很高，若沒有完善的維護管理機制，容易造成維護預算分配不均且浪費。研究最後透過模型的結果提供出幾項對於維護管理的建議，也顯示出類神經網路模型在預測成本準確率上面優於多元迴歸模型。Gill (2010)結合基因演算法(genetic algorithm, GA)與類神經網路建構模型以預測天氣狀況，選擇的輸入變數有空氣溫度、風速、降雨量、日照小時等。結果顯示。透過基因演算法得出最佳的類神經參數組合之模型表現優於傳統的類神經網路模型。Moghaddam (2016)以類神經網路預測那斯達克綜合指數的每日變化。其研究使用連續 99 天的股票價格作為輸入資料，其中前 70 天作為訓練數據、後 29 天作為測試數據，最終建構出兩種類型的那斯達克指數預測類神經網路。Alimissis (2018)結合類神經網路和多元線性迴歸以建構依空氣汙染估計模型，用來估計不受監視的位置中的汙染物濃度。其並比較了五種受管制的空氣汙染物(二氧化氮、一氧化氮、臭氧、一氧化碳與二氧化硫)的預測結果，發現在大多數情況下類神經網路的模型準確度優於多元線性迴歸。Salah (2018)將年齡、年級、性別、社會地位、固定醫療費用等變數作為類神經網路的輸入，並以 5574 個案例數據來訓練類神經網路模型，藉此預測一個地區中個人或家庭的醫療支出率。Shenfield (2018)利用一些良性的網路流量數據(圖像、音檔、文字檔)以及從資料庫中獲取的惡性流量數據建構一類神經網路模型，用以預測惡意網路之流量。模型預測的準確度平均可達到 98%，可看出此類神經模型對於惡意網路流量的檢測十分有效。Hu (2019)建構一倒傳遞類神經網路模型與一高斯過程回歸(gaussian process regression, GPR)模型以預測船舶燃料消耗，傳統預測船舶油耗方法主要有三種：實驗模擬、統計分析與機器學習，前兩者分別有研究成本過高、假設性強的問題，因此透過機器學習方法能獲得更完善的答案。實驗結果表明，類神經網路模型雖然在預測準確度上略低於高斯模型，但運行速度卻大幅領先，因

此類神經網路模型更適合應用於船舶油耗預測。

1.2 研究動機

雖然已有許多預測晶圓批生產週期時間之方法。然而這些方法仍有著相當程度的預測誤差。因此，若能估計生產週期時間的範圍，將很有參考價值。文獻中對此議題的探討較少(Chen and Wu, 2015)。

考量到生產週期時間本身為一個時間序列問題且本身充滿許多不確定性，每日甚至每分每秒的生產狀況都不盡相同(Chen, 2013)。因此若想透過傳統統計方法，例如點估計、區間估計等方法進行生產週期時間的預測，除了會提高方法實行過程中的複雜度之外，預測結果的準確率可能也不盡理想。因此透過模糊理論結合類神經網路的方式，便可以在考慮到不確定性的同時也保持良好的準確率，本篇研究所提出方法比起上述統計方法也相對容易執行。Chang (2008)提出一模糊類神經網路用於半導體廠中的流動時間預測，其研究中使用隸屬函數的概念將輸入值進行模糊化，在計算的過程略為複雜；而本篇研究使用三角模糊數對類神經網路參數進行模糊化，希望以更容易、快速的方式進行求解。

預測晶圓廠中每個批次的產出時間不僅對晶圓廠本身、對客戶而言都是十分重要的議題(Chen, 2006)。若預測時高估生產週期時間，會造成晶圓廠本身產能閒置、庫存成本增加、排擠到後續訂單等問題；而低估生產週期時間則會造成與客戶之間協商的交期不準確、面臨延遲賠償、給客戶留下不良印象等問題。在過去，晶圓廠通常會藉由過去經驗增加交期時間提供給客戶以確保能準時交貨；而在遇到客戶有急單需求詢問最短交貨時間時，也是參考過去經驗提供給客戶一個粗估的時間，這種預估的方式通常都是較沒有根據且不可靠的，因此本篇研究想藉由模糊類神經網路所產出的結果，提供給生產者與客戶一個精準且可靠的生產週期時間上下限做為參考。

1.3 研究目的

本研究的研究目的為：

(1) 以類神經網路預測一晶圓批之生產週期時間。

(2) 模糊化此類神經網路之參數，以產生生產週期時間的上下限。

(3) 藉由模擬找出模糊參數的最佳設定。

1.4 研究方法

本研究首先回顧相關文獻，包括(1)與模糊類神經網路應用之相關文獻，以及(2)與生產週期時間預測相關之文獻。而後，本研究建構一類神經網路以預測一晶圓批之生產週期時間。傳統的統計方法建構預測值之信賴空間，來估計生產週期時間之範圍。這樣做並不能保證實際值一定落在此範圍內，因此，本研究將類神經網路的參數模糊化，以產出一模糊生產週期時間預測值，其包含實際值。接下來，為了最佳化預測之精確度，本研究提出一非線性規劃模式。然而，此非線性規劃模式不容易求解。為此，本研究以數值模擬分析協助非線性規劃問題之求解。

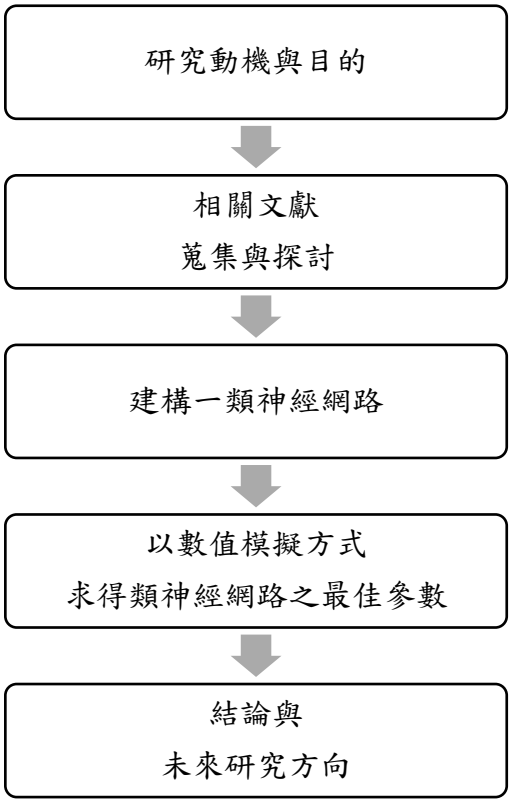

圖 1.1 研究流程

第二章 文獻回顧

2.1 晶圓製造

半導體(semiconductor)或積體電路(Integrated circuit, IC)是在幾乎所有類型的電子設備中都能發現的一種複雜裝置，半導體製造的過程既困難又耗時。半導體製造過程中最重要的步驟就是晶圓製造(wafer fabrication)。所謂晶圓製造是指在晶圓上形成積體電路的步驟。晶圓製造過程在無塵室內進行，該過程可能需要 10 至 30 天才能完成。以最新的 5nm 生產技術，一個半導體元件約包含 100 個光罩層，每層約需耗時 0.8 至 1.5 天。晶圓製造包含下列步驟，流程如圖 2.1 所示：

1. 熱氧化或沉積 (Thermal Oxidation or Deposition)

熱氧化是在晶片表面形成氧化物薄膜的過程，該過程通常在 900℃至 1100℃之間的溫度進行。由於二氧化矽(SiO_2)具有各種優異性質，如高電阻率、蝕刻抵抗性、不易造成環境汙染等(John X.J., 2019)，被廣泛運用在此道製程中。二氧化矽薄膜有乾氧化(dry oxidation)與濕氧化(wet oxidation)兩種形成方式，分別使用氧氣與水蒸氣做為熱氧化使用的氣體，不同的形成方式也會造成薄膜厚度的差異。化學氣相沉積(chemical vapor deposition, CVD)則是另一種產生氧化物薄膜的方式，透過將基底(substrate)暴露在欲產生薄膜的前驅物蒸氣中，當前驅物蒸氣碰觸到基底便會發生不同的化學反應，沉積也會是其中一種反應。半導體產業即是以晶圓當作基底暴露在不同蒸氣下來得到想要的薄膜沉積，例如二氧化矽、氮化矽(Si_3N_4)等。

2. 微影 (Photolithography)

微影通常被視為晶圓製造中的瓶頸製程(Ghasemi et al., 2020)，是將光罩(mask)上的圖形狀圖案轉換於覆蓋在晶圓上之光阻的一道製程。微影製程主要包括三個步驟，塗抹、曝光、顯影，首先在晶片上塗抹一層感光聚合物薄膜，經過烘烤過後進行曝光，使用紫外線(UV)曝光使光罩上的圖案顯影至晶片上，最後再使用化學溶劑洗去不需要的部分，也就是接下來的蝕刻製程。

3. 蝕刻 (Etching)

蝕刻製程主要目的是晶片的電路成型，在晶圓製造中，微影與蝕刻製程通常會重複進行許多次。蝕刻有分為濕蝕刻(Wet etching)與乾蝕刻(Dry etching)，濕蝕刻是利用化學藥劑將不需要的部分洗去，優點為速度快、成本較低；缺點則是需要大量用水且汙染高，而乾蝕刻主要透過電漿與離子的反應來去除不需要的部分，優點為可以精準控制消除範圍、不易造成汙染；缺點則是速度慢且成本昂貴。二氧化矽薄膜主要透過氫氟酸(HF)來去除，若人體不慎接觸會將造成嚴重後果。

4. 摻雜 (Doping)

所謂摻雜是將指定的摻質(dopant)引入半導體內，主要目的為改變半導體的電性，形成晶片上的P、N接面，臨場摻雜(in situ doping)、擴散(diffusion)、離子佈植(ion implantation)為摻雜的主要三種方式。臨場摻雜是指在形成薄膜的過程中就將摻植加入薄膜內。擴散是將晶片放進預熱的烤箱中，惰性氣體隨著管子夾帶摻質進入烤箱流動後使晶片表面吸附摻質，適合用於大面積的晶片摻雜。離子佈植則是直接透過含有摻質離子的離子束植入晶片中，跟擴散相比此種方法具有更高的精準度。

5. 介電沉積與金屬披覆 (Dielectric Deposition and Metallization)

將經過上述製程後的半成品晶片進行導電材料的沉積，提供晶片上單獨區塊之間的關鍵互連路徑。與微影、蝕刻製程相同，摻雜與金屬披覆也能重複多次，在晶片上建構多層結構。

6. 鈍化 (Passivation)

為避免上段製程的金屬披覆層氧化，會再進行最後的鈍化製程，同時也避免外界因素傷害晶片。

7. 電氣測試 (Electrical Test)

測試製作出來的晶片功能是否完善。

8. 組裝 (Assembly)

將晶片密封在防靜電塑料袋中，等待後續儲藏或配送。

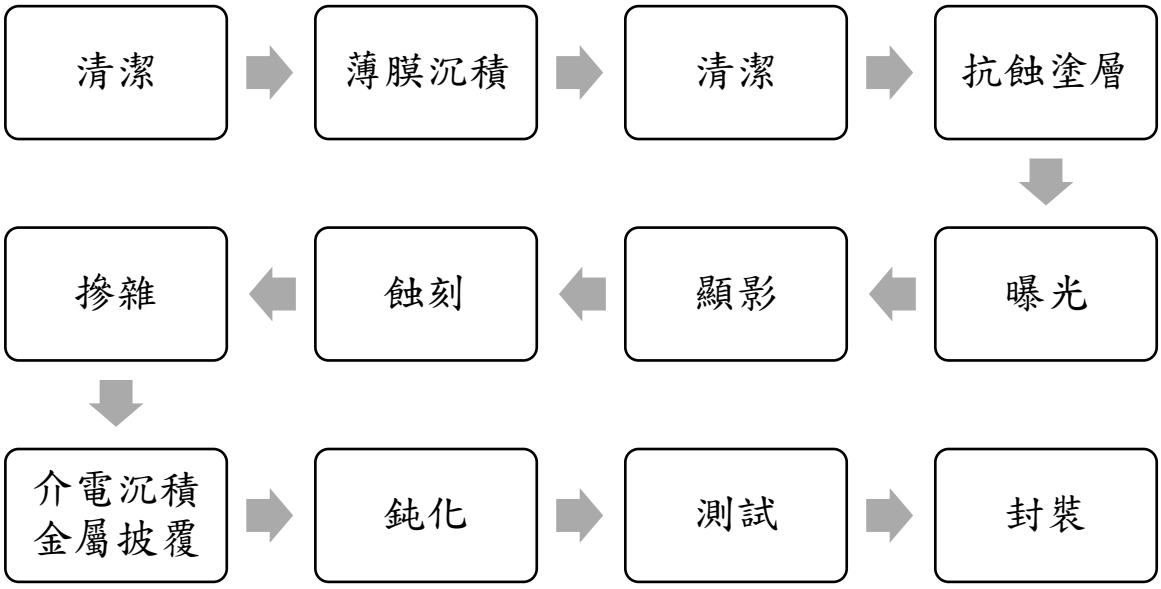

圖 2.1 晶圓製造流程圖

2.2 模糊理論之應用

模糊理論是由 Zadeh (1965)所提出，適合解決具有不確定性的問題。模糊推論系統(fuzzy inference system, FIS)、模糊類神經網路(fuzzy neural network, FNN)、模糊迴歸(fuzzy regression, FR)等是較常見的模糊理論之應用。

Chen (2008)結合模糊 C 平均(fuzzy c-means, FCM)與倒傳遞神經網路用於預測半導體廠內的作業完成時間。先使用模糊 C 平均對作業進行分類，再對這些不同的群體建構相對應的倒傳遞神經網路模型。研究最後透過模擬方式表明該方法的預測精確度明顯優於現有方法。

Khashei (2008)提出了一混合方法用於預測金融市場，進而改進決策與投資的表現。其提出的方法結合了類神經網路與模糊迴歸，結合兩者的優點後使該方法可以在不完整的數據下產生更準確的結果。最後透過實際的預測結果表明該模型可以有效提高預測準確性。

Chang (2008)提出一模糊類神經網路用於半導體廠中的流動時間預測，該模型使用幾個影響流動時間的關鍵因素來建構模型，即訂單處理時間、總在製品數量、瓶頸機台

的工作量等。模擬實際工廠內的數據評估結果表明，模糊類神經網路的表現優於案例式管理與多層類神經網路。

Wu (2010)為了提高晶片的良率預測準確度，提出一基於模糊類神經網路的良率預測模型，該模型同時考慮了影響晶片成品良率的關鍵因素與關鍵的幾個電氣測試參數。最後透過上海半導體生產線的歷史數據進行比較，相較於三種傳統的方法，模糊類神經網路的模型準確性具有相當的改進。

Mirahadi and Zayed (2016)利用一模糊推論系統預測一營建事業的產能，其中隸屬函數(membership function)的參數是透過基因演算法來最佳化。

Baumers (2016)提出一模糊產能預測方法來評估兩種 3D 列印生產系統的產能是否足以負擔大量生產的需求。

為了提高模糊產能預測區間包含真實值的可能性，Chen and Wang (2016)提出一模糊協同預測(fuzzy collaborative forecasting, FCF)方法，其中以一模糊學習過程(fuzzy learning process)來預測未來產能。在模糊協同系統中，一些具有不同背景的專家、顧問或系統試圖實現一個共同的目標。由於他們擁有不同的知識與觀點，因此他們可能會使用各種方式來建模、識別或控制共同的目標。此系統的關鍵為這些擁有不同知識背景的專家、系統們彼此交換意見、經驗與知識，即為模糊協同系統的特點，且不同於多個模糊系統的集合。

Akano and Asaolu (2017)建構一自適應類神經模糊推論系統(adaptive neuro-fuzzy inference system, ANFIS)去預測生產系統的產能。Jan (1993)提出的自適應神經模糊推論系統是一種將模糊邏輯與神經網路結合的新型模糊推論系統結構，採用倒傳遞傳播法和最小平方法的混合算法調整參數，並能自動產生 If-Then 規則。基於自適應神經網路的模糊推論系統將神經網路與模糊推論系統結合起來，既發揮了兩者的優點，也彌補了各自的不足(百度百科)。

Chen (2017)以模糊學習過程來描述產能的提高，其中參數的決定使用一類神經網路(artificial neural network, ANN)來決定。

傳統的庫存模型通常可以滿足已知的需求和充足的庫存量，但在許多行業中往往不具備這些條件，因此 Aengchuan (2018)分別以類神經網路、自適應類神經模糊推論系統與模糊推論系統結合，建構出應用於未知需求與供給量的庫存模型。經比較過後，與傳統經濟訂單量(economic order quantity, EOQ)建構出的模型相比，結合自適應類神經模糊推論系統與模糊推論系統的模型庫存成本節省約 75%以上。

Alyousufi (2020)使用模糊時間序列馬可夫鏈(fuzzy time series Markov chain, FTSMC)來預測每日空氣汙染指數。該模型使用 RMSE、MAPE 等指標進行績效驗證，最後結果顯示，該研究所提出方法之結果優於其他統計模型。

Hao (2020)結合模糊理論與支撐向量機(support vector machine, SVM)以預測股市價格走勢。提取社交網路上大量用戶的情感數據做為指標，並數入至模糊支撐向量機中訓練，藉此預測股票價格的趨勢。結果表明，此篇研究所提出之方法明顯優於傳統支撐向量機模型。

Yang (2021)結合模糊理論與類神經網路應用於外貿出口預測，透過對台灣國內外貿出口預測研究成果的分析，訂定出外貿出口預測之指標，並結合各種因素的影響，將模糊類神經網路應用於外貿出口預測，得出預測績效後並與迴歸分析、支撐向量機(support vector machine, SVM)等方法進行比對。結果顯示，模糊類神經網路的預測準確度較高於其他方法。

2.3 精確度與準確度

本研究建構之模糊前傳遞類神經網路(fuzzified feedforward neural network, FFNN)以一前傳遞神經網路作為核心，而後將參數模糊化，再來使用非線性規劃最佳化每個模糊參數的上下限。目的是為了同時提升預測晶圓批生產週期時間之精確度與準確度，過去之方法多以提升準確度為目標，見下表。

表 2.1 既有方法多以測量準確度為目標

方法	類型	最佳化準確度或精度	解決極端案例的能力
Mirahadi & Zayed (2016)	FIS	準確度	無
Akano & Asaolu (2017)	ANFIS	準確度	無
Chen & Wang	FCF	先精確度後準確度	差
Chen (2019)	ANN	準確度	差
Repina et al (2019)	Statistical analysis	準確度	無
The proposed method	FBPN	先準確度後精確度	強

在精確度方面既有的方法多半估計生產週期時間之信賴區間：

1. 週期預測方法未達到百分之百的準確率(Tirkel, 2011)。
2. 週期時間的上限代表作業進度可能延遲的程度，必須加快這項工作。因此週期時間的上限必須越低越好。
3. 週期時間的下限代表能立即實施從作業中獲得利潤。因此任何高估都會造成預算的浪費。

建構週期區間最常見的方法為訂定其範圍(Pearn et al., 2007; Tai et al., 2012)。為了訂定範圍，需要週期時間的標準差(通常以均方誤 RMSE 來估計)，然而：

1. 理論上信賴區間並非每次都會包含實際值。
2. 人們更在乎週期時間的上限，但大多數信賴區間都是對稱的，並不滿足實際需求(Janssens et al., 2009; Bernardi & Campos, 2013)。
3. 大多數信賴區間都有嚴格的假設(Sivakumar & Chong, 2001)，例如殘差符合常態。

但可對用於預測週期時間的參數進行模糊處理，進而得到一個模糊預測結果是被期待包含實際值的，稱之為包含區間。與信賴區間的差別在於：

1. 信賴區間通常是對稱的(Pearn et al., 2007; Tai et al., 2012)；包含區間則不對稱的(Chen, 2011)。
2. 對於訓練數據來說，信賴區間包含實際值的機率為 100*(1-α)%；在沒有異常值且無

對上下限寬度嚴格限制時，包含區間包含實際值的機率為 100%。

3. 建立包含區間不需要嚴格的假設。

Chen & Lin (2011)建構了一個倒傳遞網路(back propagation network, BPN)根據作業的屬性及作業開始時工廠的狀況來預測晶圓廠中每個作業的週期時間，然後將 BPN 的參數模糊化以得到模糊時間預測結果。為了決定模糊參數的值並最大化預測精確度，建構兩個非線性規劃問題(NLP)來求解。但解決 NLP 問題並不容易且得到的解通常是局部最佳解，為了解決這個問題 Chen (2013a)僅模糊單個參數(輸出節點上的閥值)，使用兩個簡單的方程式便可以解決。雖然使用此方法的預測精確度令人滿意，但對於下限的控制並不好。

為了更加精確估算週期時間的範圍，對 Chen 的方法進行了修改並提出。在提出的方法中將 BPN 的輸出替換為兩個線性函數。理論上來說這樣的方式使包含區間的上下限收縮，進而提高預測精確度，但必須找到兩個函數的可行參數區域。此方法找出的包含區間是不對稱的，符合現實中所需。

2.4 預測方法

在過去通常有六種主要方法可用於預測晶圓批的產出/生產週期時間：多因子線性組合(multiple-factor linear combination, MFLC)、生產模擬(production simulation, PS)、倒傳遞神經網路(back propagation neural network, BPN)、案例式推理(case-based reasoning, CBR)、模糊模型以及混合方法。在這六種方法中 MFLC 在實際應用上是最簡單、快速以及普遍的，但它最大的缺點是缺乏預測準確性。而 PS 雖然對於產出時間擁有最準確的預測能力，但並須建立在不斷更新數據庫才得以保持足夠的有效性，因此需要大量的數據以及漫長的模擬時間為最大的缺點。考慮到有效性與效率，Chang 等人以及 Chang 與 Hsieh 都使用了單層倒傳遞神經網路(single layer back propagation neural network)來預測晶圓批的產出/生產週期時間，與 MFLC 相比，使用 BPN 大幅降低了預測結果的均方

根誤差(root mean squared error, RMSE)，在 Chang 等人的論文中 BPN 的方法相較於 MFLC 約可提高 40%的準確率。此外，與 PS 相比使用 BPN 所需的數據以及資料也減少許多。Chang 等提出了一種基於 K 個近鄰演算法(k nearest neighbor, KNN)的 CBR 方法，此方法在預測準確性方面也優於 BPN。Chen 也建構了一個模糊的倒傳遞神經(fuzzy back propagation neural network, FBPN)模型，它在處理 FBPN 的輸入時結合了專家的意見，此方法的效率優於一般的 BPN 模型。

(1) MFLC 透過下列參數來估算晶圓批的生產週期時間：

(a)作業屬性

(b)生產週期時間與等待時間

(c)工作量資訊

優點為應用簡單、快速、普遍；缺點為缺乏預測準確性。

(2) PS：需持續更新資料庫以維持足夠的有效性，需要模擬多次以充份考慮所有不確定或隨機的事件。優點為預測十分準確、允許進行計算實驗與後續分析且無區耗費任何實際執行成本；缺點為需維護大量數據、模擬時間長。

(3) BPN：在許多研究中表示，在時間序列預測中人工神經網路的表現優於傳統方法。優點為預測速度快、適合模擬複雜系統。在 Chang 等人的論文中 BPN 的方法相較於 MFLC 都大大提高預測的精度。

(4) CBR：Chang 等提出了一種基於 K 個近鄰演算法(k nearest neighbor, KNN)的 CBR 方法，該方法具有動態因子權重和非線性相似度函數，此方法在預測準確性方面優於傳統 BPN 方法。

(5) 模糊模型：Chang 等用一種簡單的基因演算法修改了 WM 方法的第一步(將每個輸入的變數轉化成模糊數)，並提出 EFR 方法來預測晶圓批的生產週期時間。此方法在預測準確性上優於 CBR 與 BPN，基因演算法以證明能夠對參數進行全面性的最佳化。

(6) 混合方法：Chen 建構了一個 FBPN，在輸入參數時結合了專家意見。在效率方面優

於傳統 BPN 方法，且在預測準確性也略優於傳統 BPN 方法。

表 2.2 預測方法比較

方法	資料需求	處理時間	準確性	方便使用度	普及度
PS	大	長	非常高	容易	不普遍
MFLC	小	非常短	低	非常容易	普遍
BPN/FPBN	小	短	高	難	不普遍
CBR	小	短	高	難	不普遍
EFR	小	短	高	難	不普遍

2.5 倒傳遞類神經網路之應用

倒傳遞類神經網路是一種具有監督式學習功能的前饋神經網路，所謂的前饋是指訊號從輸入經由單一或多層隱藏層後再輸出，而監督式學習是指在已知正確答案的情況之下，比較每次權重調整後的類神經網路的輸出值，最後得出一個預測誤差最小的最佳網路。透過倒傳遞學習方法的類神經網路由於易於訓練，因此在使用上非常普遍。據估計，正在開發的所有類神經網路項目中，約有 80%以上使用倒傳遞神經網路(Abhishek, 2012)。

Kim (2002)建構一倒傳遞類神經網路模型用於預測微處理器(central processing unit, CPU)之製造良率，使用 40 種在製程中電氣測試(electrical test, ET)階段之數據，建構出的模型預測結果之均方根誤差(root mean square error, RMSE)僅有 5.4%，與傳統多元回歸(multiple regression, MR)模型相比改進了約 41.09%。為了降低不必要的封裝成本，也使用相似的方法建構一用於預測微處理器運行速度之倒傳遞類神經網路模型，實驗結果表明預測結果與實際微處理器運行速度差異僅有 1.7%，足以有效降低製造成本與提升產品品質。

Chang (2006)結合模糊邏輯與類神經網路，建構一模糊倒傳遞類神經網路(fuzzy back-propagation network, FBPN)模型用於預測印刷電路板產業的銷售情況，可靠的銷售預測可以有效幫助業務策略的訂定。建構出的模型透過印刷電路板公司提供的真實數據印證，實驗結果表明該模糊倒傳遞神經網路模型在平均絕對百分誤差(mean absolute

percentage error, MAPE)中的表現優於其他三種模型。

Kadir (2014)建構一倒傳遞類神經網路模型用於預測小麥產量，小麥產量被視為全球糧食供應的良好指標，因此若能準確預測其產量將可能對糧食短缺問題提出貢獻。此模型所使用的輸入變數為天氣數據，包括陽光、雨水、溫度等因素，以 2772 筆數據進行訓練後模型的預測結果可達到 98%左右的準確率。

Joshi (2016)使用倒傳遞類神經網路為架構設計出一檢測糖尿病之工具，該神經網路的輸入層為八個參數、使用具十個神經元的單一隱藏層，並建構出一圖形使用者介面(graphical user interface, GUI)方便醫護人員操作，希望透過此工具減少對病患進行指微血管血糖測驗(finger stick, FS)的次數。實驗指出該工具預測準確率為 81%，優於過去所使用的方法。

Rhee (2016)建構一倒傳遞類神經網路模型用於預測電影是否能夠獲利，與其他現有的方法不同，他蒐集大量社群網站上對電影的評價作為參考，一共挑選出 14 個他認為具有影響獲利的關鍵因素做為模型的輸入變數。以 375 筆資料進行實驗後，最終結果表明該模型預測的準確率為 91%，與支撐向量機(support vector machine, SVM)模型比較過後，倒傳遞類神經網路模型具有更高的準確率。

Concha (2019)為了減少建築物拆除成本的估算時間與人力成本，建構一倒傳遞類神經網路模型來解決此問題。該模型使用菲律賓奎松(Quezon)市的 90 筆拆除項目之詳細資料來進行訓練與驗證，將建築材料、分類、樓層數、位置、拆遷方法等九個因素設為模型的輸入變數，經過訓練與學習後，該模型的預測平均準確率達 90.21%，結果表明此倒傳遞類神經模型能可靠的預測奎松市的建築拆除成本。

Idris (2020)為了使汽車製造商能夠更合理的訂定具最新規格汽車之價格，建構一倒傳遞類神經網路模型來預測汽車之價格。該模型透過輸入 14 個與汽車相關的規格以得到預測價格，使用 264 筆 BMW 的資料訓練模型。實驗結果顯示模型的預測價格與實際價格相近，隨機抽查 15 筆結果之平均誤差為 11.46%，能有效提供車商訂定價格時的參考依據。

在整理完現有文獻後，可看出模糊理論與倒傳遞類神經網路分別在各個領域中均有廣泛的應用，但對於生產週期區間預測方面並沒有相關應用，且大多數的預測方法皆著重於準確度方面，並未對精確度進行最佳化。為此，本研究欲結合模糊理論與倒傳遞類神經網路，透過模糊化類神經網路中的參數，得到一模糊生產週期區間，再透過調整模糊化參數的數量與組合，縮減生產週期區間的寬度，以提升精確度。

第三章 研究方法

3.1 預測晶圓批生產週期時間之類神經網路

本節首先定義倒傳遞網路之參數以及其運算方式。

(1) 輸入參數：

許多之前的研究收集一晶圓批投入晶圓廠時之生產狀況並依這些生產狀況來預測此晶圓批之生產週期時間(Chen, 2014; Chen, 2015)。文獻中曾用到生產狀況整理如表 3-1。

表 3.1 用以預測晶圓批之生產狀況

q_n	批量
U_n	廠內前一天的平均設備利用率
Q_n	該批晶圓生產線上的等候批數
BQ_n	瓶頸站前的等候批數
WIP_n	廠內的在製品批數
D_n	最近幾批之平均延遲時間
FQ_n	所有生產線上的等候批數

(2) 輸出值(a)：

該晶圓批之生產週期時間預測值。

(3) 轉換函數：

輸入層到隱藏層的轉換函數為線性函數：

$$f(x) = x \tag{1}$$

其他地方的轉換函數為 logistic 函數：

$$f(x) = \frac{1}{1+e^{-x}} \tag{2}$$

(4) 學習速率(η)：0.01-1.0

(5) 批次學習：

訓練階段可以分為兩個步驟。首先是向前傳遞，輸入值乘以連結權重後被傳送至第一層隱藏層，經過整合、轉換之後輸出為：

$$h_{jl} = \frac{1}{1+e^{-n_{jl}^{h}}}, l = 1\sim L, \tag{3}$$

其中，

$$n_{jl}^{h} = I_{jl}^{h} - \theta_{l}^{h}, \tag{4}$$

$$I_{jl}^{h} = \sum_{i=1}^{I} w_{il}^{h} x_{ji}, l = 1\sim \mathrm{L}, \tag{5}$$

再透過相同方式傳遞至第二層隱藏層，經過整合、轉換之後輸出為：

$$h_{jk} = \frac{1}{1+e^{-n_{jk}^{h}}}, k = 1\sim K, \tag{6}$$

其中，

$$n_{jk}^{h} = I_{jk}^{h} - \theta_{k}^{h}, \tag{7}$$

$$I_{jk}^{h} = \sum_{l=1}^{L} w_{lk}^{h} h_{jl}, k = 1\sim K, \tag{8}$$

接著，h_k 透過同樣的方式傳遞至輸出層。最終，可以得到此類神經網路的輸出結果為：

$$o_j = \frac{1}{1+e^{-n_j^o}} , \tag{9}$$

其中，

$$n_j^o = I_j^o - \theta^o , \tag{10}$$

$$I_j^o = \sum_{k=1}^{K} w_k^o h_{jk} , \tag{11}$$

接著，將類神經網路的輸出 o_j 與實際值 a_j 比較，並以均方根誤差(root mean square error, RMSE)作為判斷預測結果是否準確的評估準則。均方根誤差的計算公式如下：

$$\text{RMSE} = \sqrt{\frac{\sum_{j=1}^{n}(o_j - a_j)^2}{n}} , \tag{12}$$

接著是修正網路參數以減少誤差。在陡降坡度(gradient descend, GD)法中，預測誤差以倒傳遞的方式影響修正值。首先，根據輸出值 o_j 與實際值 a_j 之間的差異，可以將隱藏層與輸出層中神經元的誤差項計算出來，其公式如下：

$$\delta_j^o = o_j(1-o_j)(a_j-o_j) , \tag{13}$$

$$\delta_{jk}^h = h_{jk}(1-h_{jk})\, w_k^o \delta_j^o , \tag{14}$$

$$\delta_{jl}^h = h_{jl}(1-h_{jl})\, w_{lk}^h \delta_{jk}^h , \tag{15}$$

根據上述公式求得的誤差項，可以將連權重與閾值所需的調整值計算出來：

$$\Delta w_{jl}^o = \eta \delta_j^o h_{jk} , \tag{16}$$

$$\Delta w_{jlk}^h = \eta \delta_{jk}^h h_{jl} , \tag{17}$$

$$\Delta w_{jil}^h = \eta \delta_{jl}^h x_{ji} , \tag{18}$$

$$\Delta\theta_j^h = -\eta\delta_j^o \ , \tag{19}$$

$$\Delta\theta_{jk}^h = -\eta\delta_{jk}^h \ , \tag{20}$$

$$\Delta\theta_{jl}^h = -\eta\delta_{jl}^h \ , \tag{21}$$

為了加快過程，可以將一動量加至修正公式中，例如：

$$\Delta w_{jl}^o = \eta\delta_j^o h_{jl} + \alpha(w_{jl}^o(\mathrm{t}) - w_{jl}^o(\mathrm{t}\text{-}1)) \ , \tag{22}$$

理論上來講，當 RMSE 降低至給定的水準後，或者隨著更多的學習，RMSE 的改善幅度極小時，學習過程就會停止。最後，此類神經網路可用於預測新的晶圓批的生產週期時間。新晶圓批的相關參數被記錄下來，輸入至此類神經網路中即可預測其生產週期時間。除了陡降坡度法之外，還有其他更具效率之演算法，例如 LM (Levenberg-Marquardt) 演算法。

3.2 預測晶圓批生產週期時間之模糊類神經網路

本研究所提出的模糊參數皆使用三角模糊數(triangular fuzzy numbers, TFNs)表示。首先定義三角模糊數與其運算。

一個三角模糊數$\tilde{A}=(A_1, A_2, A_3)$具有以下隸屬函數：

$$\mu_A(x) = \begin{cases} \frac{x-A_1}{A_2-A_1} & if\ A_1 \le \mathrm{x} < A_2 \\ \frac{A_3-x}{A_3-A_2} & if A_2 \le \mathrm{x} < A_3 \\ 0 & otherwise \end{cases} \tag{23}$$

三角模糊數的運算方式如下：

(1) 模糊加法：

$$\tilde{A}(+)\tilde{\mathrm{B}} = (A_1+B_1,\ A_2+B_2,\ A_3+B_3) \tag{24}$$

(2) 模糊減法：

$$\tilde{A}(-)\tilde{\mathrm{B}} = (A_1\text{-}B_1,\ A_2\text{-}B_2,\ A_3\text{-}B_3) \tag{25}$$

(3) 模糊乘法：

$$\tilde{A}(\times)\tilde{\mathrm{B}} \cong (A_1B_1,\ A_2B_2,\ A_3B_3) \text{ if } A_1, B_1 \geq 0 \tag{26}$$

(4) 模糊除法：

$$\tilde{A}(/)\tilde{\mathrm{B}} \cong (A_1/B_3,\ A_2/B_2,\ A_3B_1) \text{ if } A_1 \geq 0, B_1 > 0 \tag{27}$$

(5) 指數運算：

$$e^{\tilde{A}} \cong (e^{A_1}, e^{A_2}, e^{A_3}) \text{ if } A_1 \geq 0 \tag{28}$$

(6) 對數運算：

$$\ln\tilde{A} \cong (\ln A_1, \ln A_2, \ln A_3) \text{ if } A_1 > 0 \tag{29}$$

接著我們透過將網路參數系數模糊化，以產生模糊生產週期時間用以估計生產週期時間之範圍。首先是向前傳遞，輸入值乘以連結權重後被傳送送至隱藏層，經過整合、轉換之後輸出為：

$$\widetilde{h_{Jl}} = \frac{1}{1(+)e^{-\widetilde{n_{Jl}^h}}},\ l = 1\sim L, \tag{30}$$

其中，

$$\widetilde{n_{Jl}^h} = \widetilde{I_{Jl}^h}\ (\text{-})\ \widetilde{\theta_l^h}\ , \tag{31}$$

$$\widetilde{I_{Jl}^h} = \sum_{i=1}^{I} \widetilde{w_{il}^h} x_i\ ,\ l = 1\sim L \tag{32}$$

再透過相同方式傳遞至第二層隱藏層，經過整合、轉換之後輸出為：

$$\widetilde{h_{Jk}} = \frac{1}{1(+)e^{-\widetilde{n_{Jk}^h}}},\ k = 1\sim K, \tag{33}$$

其中，

$$\widetilde{n_{Jk}^h} = \widetilde{I_{Jk}^h}\ (\text{-})\ \widetilde{\theta_k^h}\ , \tag{34}$$

$$\widetilde{I_{Jk}^h} = \sum_{l=1}^{L} \widetilde{w_{lk}^h}\ \widetilde{h_{Jl}}\ ,\ k = 1\sim K, \tag{35}$$

接著，$\widetilde{h_k}$透過同樣的方式被傳遞至輸出層。最終，我們可以得到此模糊類神經網路的輸出結果：

$$\widetilde{o_j} = \frac{1}{1+e^{-\widetilde{n_j^o}}} , \tag{36}$$

其中，

$$\widetilde{n_j^o} = \widetilde{I_j^o} \text{ (-) } \widetilde{\theta^o} , \tag{37}$$

$$\widetilde{I_j^o} = \sum_{k=1}^{K} \widetilde{w_k^o}(\times)\widetilde{h_{jk}} , \tag{38}$$

接著是修正網路參數以減少誤差。為此目的，陡降坡度(gradient descend)法可以被模糊化。首先，根據輸出值$\widetilde{o_j}$與實際值$\widetilde{a_j}$之間的差異，可以將隱藏層與輸出層中神經元的誤差項計算出來，其公式如下：

$$\widetilde{\delta_j^o} = \widetilde{o_j}(\times)(1(-)\widetilde{o_j})(\times)(\widetilde{a_j}(-)\widetilde{o_j}) , \tag{39}$$

$$\widetilde{\delta_{jk}^h} = \widetilde{h_{jk}}(\times)(1(-)\widetilde{h_{jk}})(\times)\widetilde{w_k^o}(\times)\widetilde{\delta_j^o}, \tag{40}$$

$$\widetilde{\delta_{jl}^h} = \widetilde{h_{jl}}(\times)(1(-)\widetilde{h_{jl}})(\times)\widetilde{w_{lk}^h}(\times)\widetilde{\delta_{jk}^h}, \tag{41}$$

透過上述公式求得的誤差項，可以將權重與閾值所需的調整值計算出來：

$$\Delta\widetilde{w_{jl}^o} = \eta\widetilde{\delta_j^o}(\times)\widetilde{h_{jk}} , \tag{42}$$

$$\Delta\widetilde{w_{jlk}^h} = \eta\widetilde{\delta_{jk}^h}(\times)\widetilde{h_{jl}} , \tag{43}$$

$$\Delta\widetilde{w_{jil}^h} = \eta\widetilde{\delta_{jl}^h}x_{ji}, \tag{44}$$

$$\Delta\widetilde{\theta_j^h} = -\eta\widetilde{\delta_j^o} , \tag{45}$$

$$\Delta\widetilde{\theta_{jk}^h} = -\eta\widetilde{\delta_{jk}^h} , \tag{46}$$

$$\Delta\widetilde{\theta_{jl}^{h}} = -\eta\widetilde{\delta_{jl}^{h}} \ , \tag{47}$$

然而，根據 Chen (2003)指出，模糊陡降坡度法會導致模糊性(不確定性)的累積，如此一來，所估計的生產週期時間之範圍會變得非常寬。為了解決此問題，可以求解模糊網路參數以最小化模糊生產週期時間預測值的範圍之和：

$$\text{Min } \sum_{j=1}^{n}(o_{j3} - o_{j1}) \tag{48}$$

s.t.

$$o_{j3} \geq aj \geq o_{j1} \tag{49}$$

$$o_{j3} = \frac{1}{1+e^{-n_{j3}^{o}}} \tag{50}$$

$$o_{j1} = \frac{1}{1+e^{-n_{j1}^{o}}} \tag{51}$$

$$n_{j3}^{o} = I_{j3}^{o} - \theta_{1}^{o} \tag{52}$$

$$n_{j1}^{o} = I_{j1}^{o} - \theta_{3}^{o} \tag{53}$$

$$I_{j3}^{o} = \max(\sum_{k=1}^{K} w_{k1}^{o} h_{jk3} , \sum_{k=1}^{K} w_{k3}^{o} h_{jk3}) \tag{54}$$

$$I_{j1}^{o} = \min(\sum_{k=1}^{K} w_{k1}^{o} h_{jk1} , \sum_{k=1}^{K} w_{k3}^{o} h_{jk1}) \tag{55}$$

$$h_{jk3} = \frac{1}{1+e^{-n_{jk3}^{h}}} \tag{56}$$

$$h_{jk1} = \frac{1}{1+e^{-n_{jk1}^{h}}} \tag{57}$$

$$n_{jk3}^{h} = I_{jk3}^{h} - \theta_{k1}^{h} \tag{58}$$

$$n_{jk1}^{h} = I_{jk1}^{h} - \theta_{k3}^{h} \tag{59}$$

$$I_{jk3}^{h} = \max(\sum_{l=1}^{L} w_{lk1}^{h} h_{jl3}, \sum_{l=1}^{L} w_{lk3}^{h} h_{jl3}) \quad (60)$$

$$I_{jk1}^{h} = \min(\sum_{l=1}^{L} w_{lk1}^{h} h_{jl1}, \sum_{l=1}^{L} w_{lk3}^{h} h_{jl1}) \quad (61)$$

$$h_{jl3} = \frac{1}{1+e^{-n_{jl3}^{h}}} \quad (62)$$

$$h_{jl1} = \frac{1}{1+e^{-n_{jl1}^{h}}} \quad (63)$$

$$n_{jl3}^{h} = I_{jl3}^{h} - \theta_{l1}^{h} \quad (64)$$

$$n_{jl1}^{h} = I_{jl1}^{h} - \theta_{l3}^{h} \quad (65)$$

$$I_{jl3}^{h} = \max(\sum_{i=1}^{I} w_{il1}^{h} x_{ji}, \sum_{i=1}^{6} w_{il3}^{h} x_{ji}) \quad (66)$$

$$I_{jl1}^{h} = \min(\sum_{i=1}^{I} w_{il1}^{h} x_{ji}, \sum_{i=1}^{6} w_{il3}^{h} x_{ji}) \quad (67)$$

$$o_{j3} \geq o_{j1} \quad (68)$$

$$n_{j3}^{o} \geq n_{j1}^{o} \quad (69)$$

$$I_{j3}^{o} \geq I_{j1}^{o} \quad (70)$$

$$\theta_{3}^{o} \geq \theta_{1}^{o} \quad (71)$$

$$w_{l3}^{o} \geq w_{l1}^{o} \quad (72)$$

$$h_{jl3} \geq h_{jl1} \quad (73)$$

$$w_{k3}^{o} \geq w_{k1}^{o} \quad (74)$$

$$h_{jk3} \geq h_{jk1} \quad (75)$$

$$n_{jk3}^{h} \geq n_{jk1}^{h} \quad (76)$$

$$I_{jk3}^{h} \geq I_{jk1}^{h} \quad (77)$$

$$\theta_{k3}^{o} \geq \theta_{k1}^{o} \quad (78)$$

$$n_{jl3}^{h} \geq n_{jl1}^{h} \tag{79}$$

$$I_{jl3}^{h} \geq I_{jl1}^{h} \tag{80}$$

$$\theta_{l3}^{o} \geq \theta_{l1}^{o} \tag{81}$$

$$w_{il3}^{o} \geq w_{il1}^{o} \tag{82}$$

此為一非線性規劃模式，常見的非線性規劃求解方式有：牛頓法、梯度法、拉格朗日乘子法等，這些方法通常需要許多計算且得到之結果也未必為最佳。因此，本研究擬採取數值模擬之方式來協助求解模糊網路參數之值。

第四章 實例分析

本章以一實例驗證所提出之方法之有效性。本篇研究使用 R 語言建構一倒傳遞類神經網路，再對網路中的參數進行模糊化，並以模擬的方式找出最佳的模糊化參數。最後再評估、比較其之預測績效。實例分析的進行過程如圖 4.1 所示。

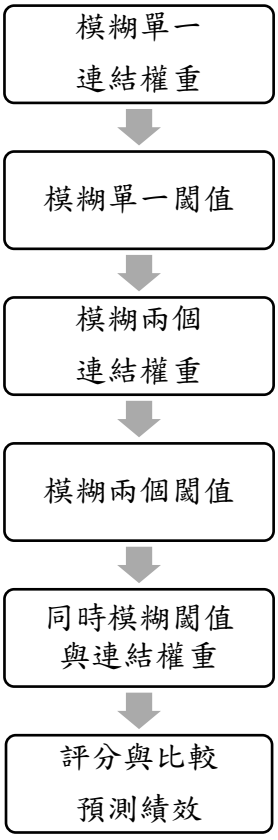

圖 4.1 實例分析之進行流程

4.1 輸入數據

許多之前的研究收集一晶圓批投入晶圓廠時之生產狀況，並依這些生產狀況來預測此晶圓批之生產週期時間(Chen, 2014; Chen, 2015)。於本實驗中，六個與第 j 批次有關的參數在標準化後輸入至類神經網路中，其中包括：批量(q_n)、廠內前一天的整體設備平均利用率(U_n)、該晶圓批生產線上的等候批數(Q_n)、瓶頸站前的等候批數(BQ_n)、所有生產線上的等候批數(FQ_n)、廠內的在製品批數(WIP_n)、最近幾批的平均延遲時間(D_n)。由於類神經網路模型預測的結果落在 0 與 1 的機率非常小，因此為了提升預測績效，本研究使標準化後的數值介於 0.1 至 0.9 之間，標準化之公式如下：

$$f(x) = \left(\frac{x - x_{min}}{x_{max} - x_{min}}\right) * 0.8 + 0.1 \tag{83}$$

透過圖 4.2 之箱型圖(boxplot)可以發現在批量(q_n)、最近幾批的平均延遲時間(D_n)以

及真實生產週期時間(a_n)有離群值(outlier)的出現。此外，可以發現 q_n 的資料點若將離群值去除後大部分都集中在數值 1 的地方，對預測模型的幫助並不大。因此，在後續建構類神經網路模型時便不使用該參數。而在 a_n 的部分，由於將原本的離群值去除後會再出現出新的離群值，因此就不對此參數做移除的動作。圖 4.2 為處理過後的資料及箱型圖。

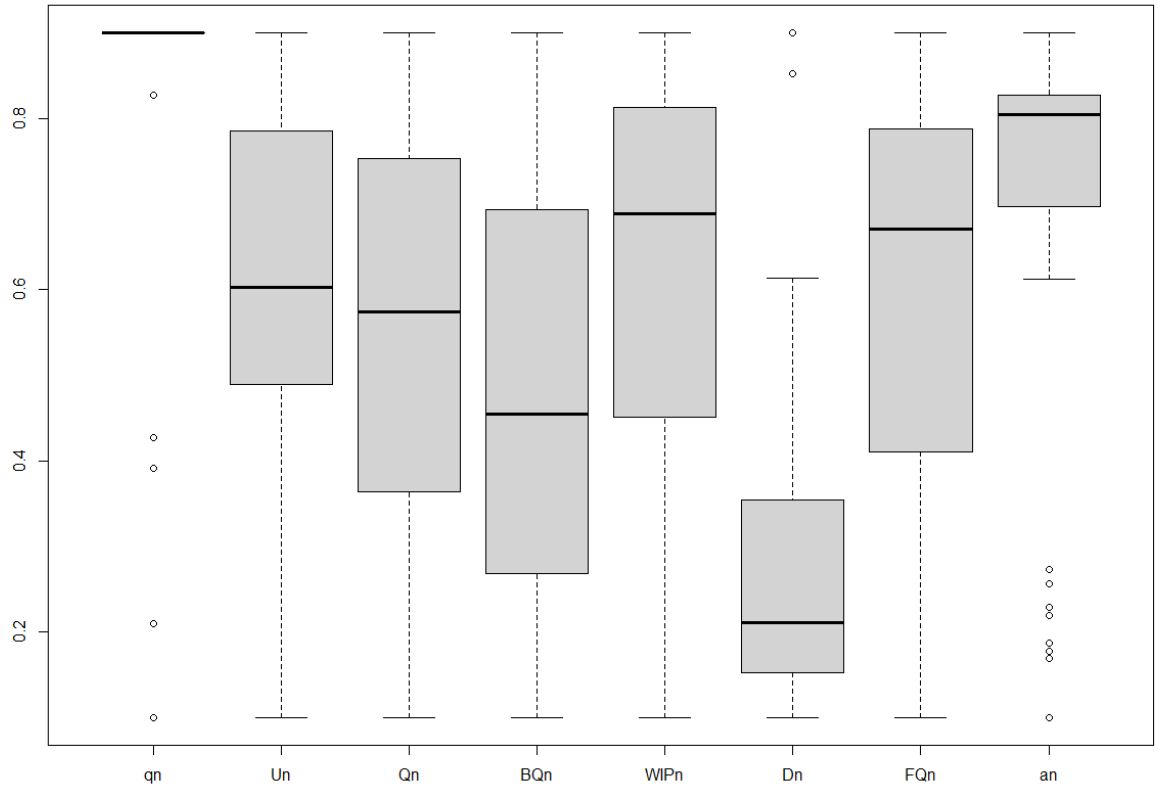

圖 4.2 資料集之箱型圖

如下圖 4.3 所示，將資料刪減過後由原本 56 筆資料減少至 47 筆，輸入類神經網路模型之參數由 7 個減少為 6 個。

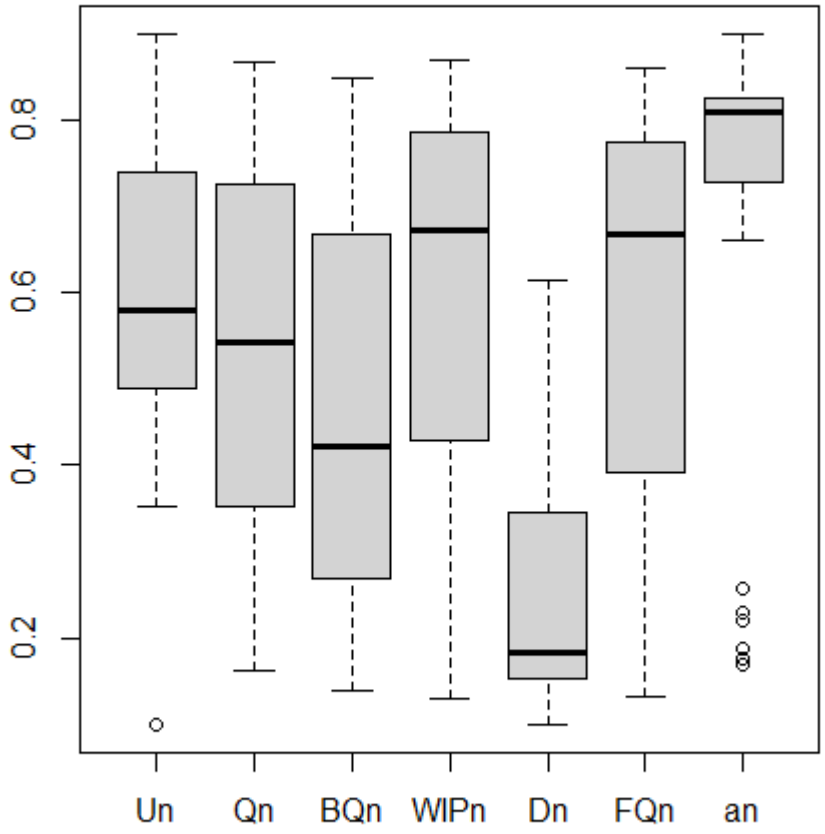

圖 4.3 處理後資料集之箱型圖

4.2 倒傳遞網路架構

本研究所使用之倒傳遞類神經網路其架構之決定是根據模擬實驗後所得出的結果：事先設定好類神經網路中的架構參數，如隱藏層數量、節點數量等，再透過程式模擬比較各種參數之組合的模型預測準確度。最後選出最佳的類神經網路架構參數以提供後續步驟使用，如圖 4.4 所示。透過比較均方根誤差(root mean square error, RMSE)與平均絕對誤差(mean square error, MAE)兩者的表現，可以發現在隱藏層為 2 層、第一層隱藏層之節點個數為 1、第二層隱藏層之節個點數為 3 時，能夠得到 RMSE 最低、MAE 次低

的結果。最後，最佳化之類神經網路架構如圖 4.5 所示。

```
No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 37, 37, 37, 37, 37, 37, ..
Resampling results across tuning parameters:

  layer1  layer2  RMSE       Rsquared   MAE
  1       0       0.2450997  0.2357067  0.1561532
  1       1       0.2456428  0.1987496  0.1582040
  1       2       0.2439283  0.1995888  0.1604090
  1       3       0.2327545  0.1914998  0.1478951
  1       4       0.2492079  0.1886644  0.1594845
  1       5       0.2504939  0.1811616  0.1591906
  2       0       0.2586639  0.1961673  0.1607691
  2       1       0.2368233  0.2367646  0.1474785
  2       2       0.2652554  0.1572947  0.1686841
  2       3       0.2671939  0.2139037  0.1691058
  2       4       0.2597038  0.2030391  0.1601231
  2       5       0.2622941  0.2366114  0.1639367
  3       0       0.2535356  0.2206833  0.1658332
  3       1       0.2664947  0.1832958  0.1710416
  3       2       0.2969847  0.1992360  0.1718524
  3       3       0.2676958  0.1861946  0.1648828
  3       4       0.2649551  0.1687991  0.1681840
  3       5       0.2939490  0.2010916  0.1791732
  4       0       0.3029650  0.1712587  0.1824600
  4       1       0.2751245  0.1699775  0.1719074
  4       2       0.2883490  0.1960376  0.1755037
  4       3       0.2672642  0.1865021  0.1704929
  4       4       0.2665390  0.2139817  0.1688024
  4       5       0.2683487  0.2083187  0.1688270
  5       0       0.3302595  0.1931277  0.2025899
  5       1       0.2575546  0.2099517  0.1588411
  5       2       0.2543471  0.2024792  0.1634873
  5       3       0.3383672  0.2075511  0.1892779
  5       4       0.2633548  0.2194984  0.1661199
  5       5       0.2777251  0.1527275  0.1762639
```

圖 4.4 類神經網路架構不同參數組合之預測表現(訓練演算法為試誤法)

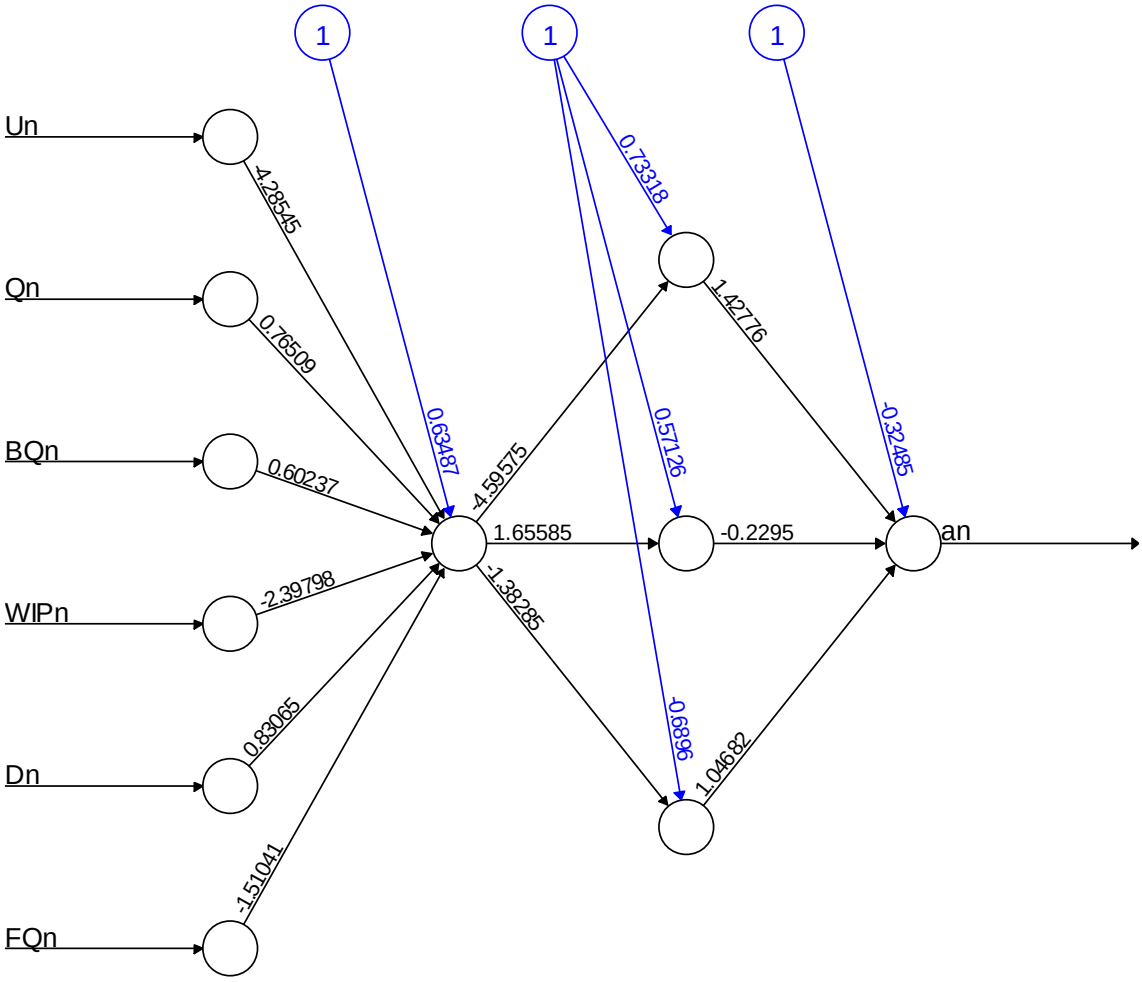

圖 4.5 類神經網路之架構

4.3 參數之模糊化

接下來對類神經網路之參數進行模糊化之處理，採模擬(試誤法)之方式。首先，挑選單一權重或閾值增加或減少其值百分之十，之後推導出模糊輸出值，並紀錄結果(見圖4.6)。接著再對兩個權重或閾值進行模糊化，得出最後之模糊輸出值後記錄。最後模擬同時對權重及閾值模糊化。另外，雖然本研究模糊化參數的幅度是對稱的(上下各百分之十)，但其模型產出的模糊區間結果為非對稱。本研究由模糊化單一權重開始，隨機挑選單一權重增加/減少其值百分之十，觀察並紀錄結果。為方便觀察並紀錄，將各參數編號如圖 4.7 所示。

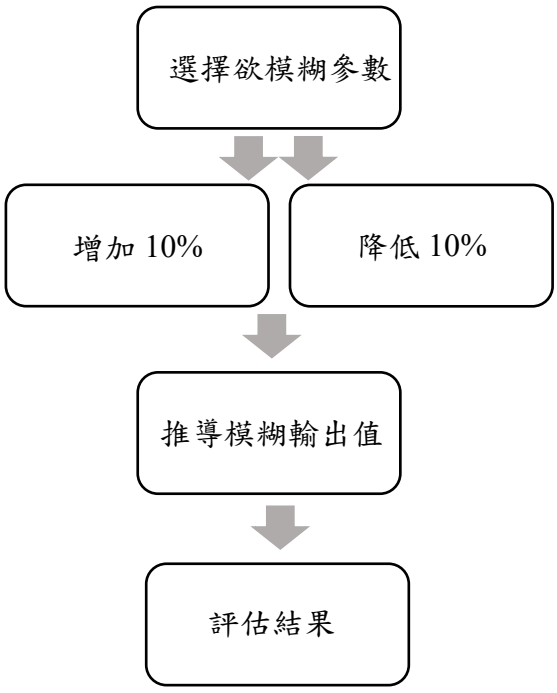

圖 4.6 模糊化參數流程

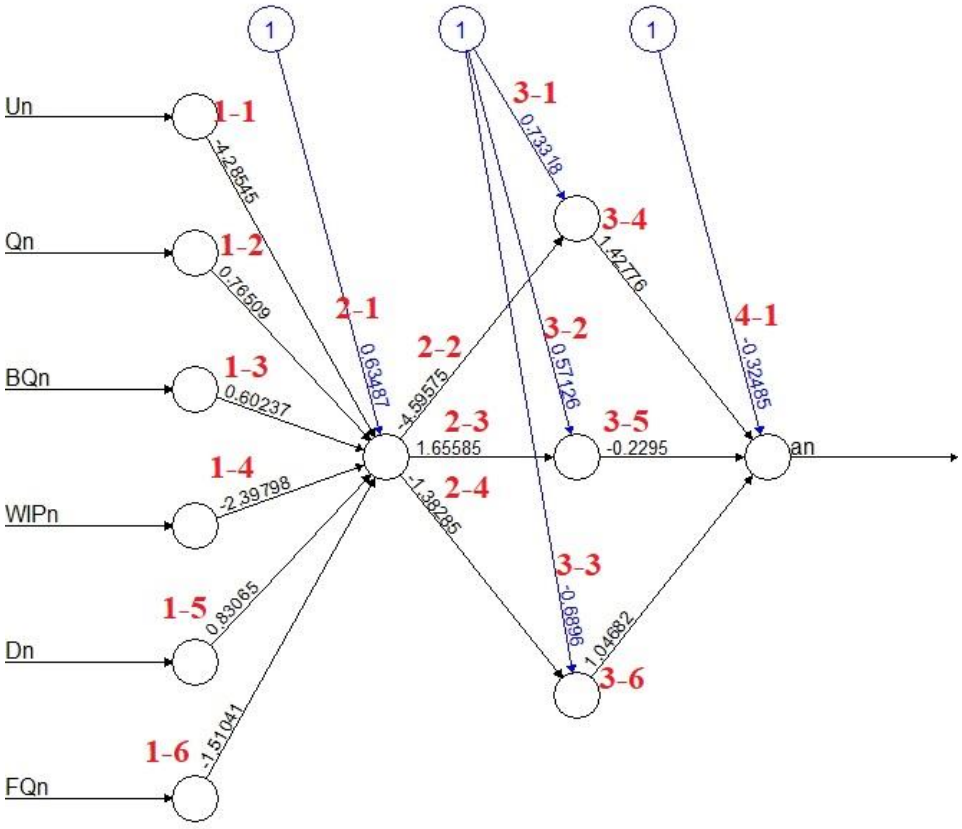

圖 4.7 類神經網路各參數之編號

模糊化後之類神經網路之預測績效評估指標為模糊預測值之平均區間大小(average range, AR)、區間包含真實值之機率(Hit Rate)。目標為平均區間越窄越好、且包含真實值之機率越高越佳。

4.3.1 模糊化單一權重

本研究所使用的模糊化方法是增加或減少該參數值的 10%。首先，選擇單一權重進行模糊化。可看到前兩次選擇之權重 1-1 與 2-2 模糊化的效果(圖 4.8 上半部)並沒有太好。雖然平均區間窄但包含真實值的機率也低。模糊化權重 3-4 後的命中率非常高(圖 4.8 左下)，且僅有一個樣本落在區間外。模糊化權重 3-6 後的命中率雖然沒有很高(圖 4.8 右下)，但可看出若將區間寬度進行微調，便可能包含更多真實值，因此在進行後續之兩權重模糊化時可優先考慮該權重。藉由這些實驗結果可以看出模糊化越靠近輸入層的權

重對最後的輸出值之影響越小，所得到的平均區間也越窄；越靠近輸出層則反之。

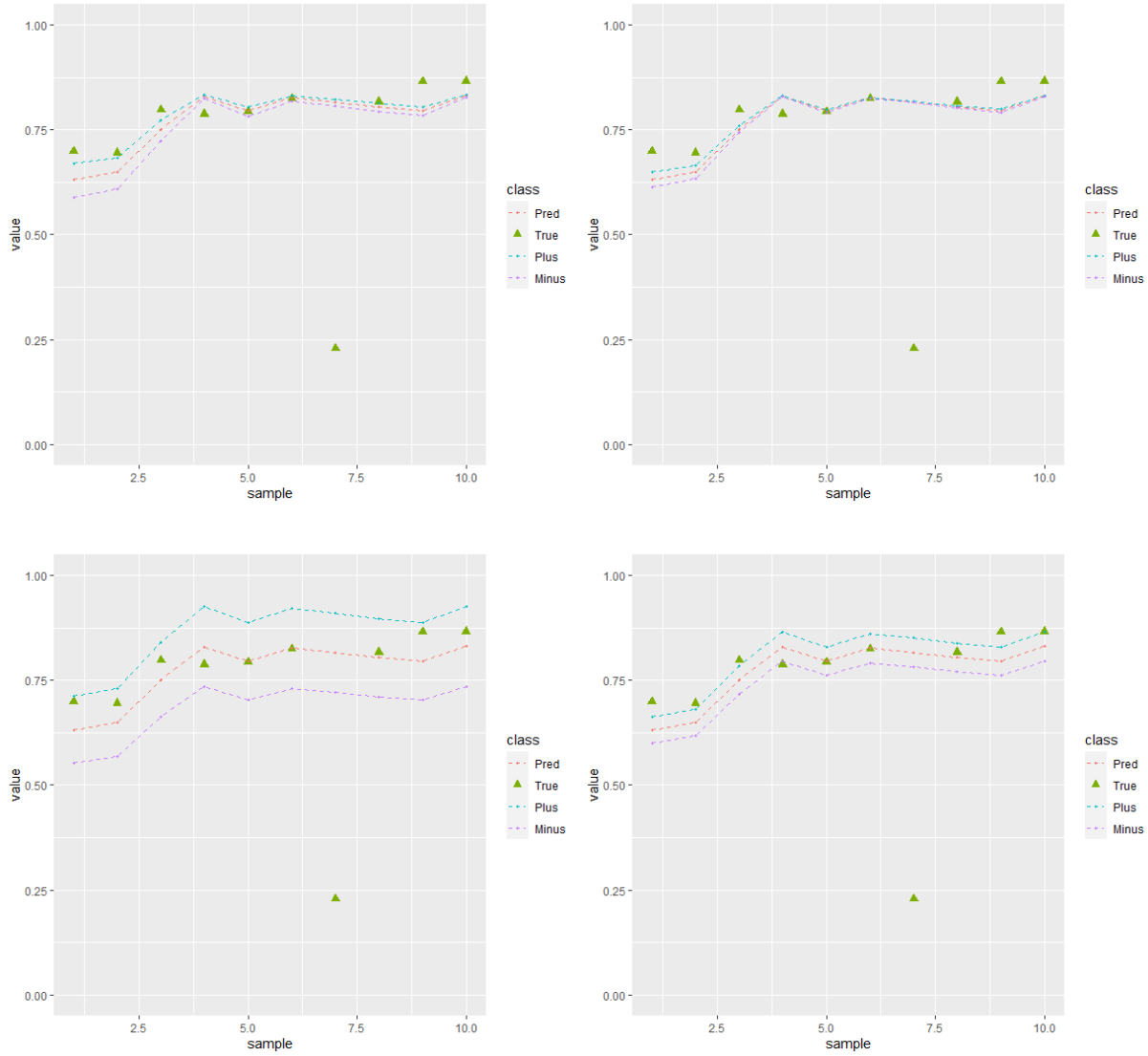

圖 4.8 模糊化單一權重之結果

表 4.1 模糊化單一權重之績效

模糊化之參數	Average Range	Raw Range	Hit Rate
1-1	0.03163538	29.1441	2/10
2-2	0.01139914	10.50146	2/10
3-4	0.1818987	167.5742	9/10
3-6	0.06764678	62.3196	4/10

4.3.2 模糊化單一閾值

從模擬結果(圖 4.9)可以看出，比起模糊化權重，模糊化閾值對於區間寬度之影響較小、所得到的區間寬度也較窄，因此包含真實值的命中率也較低。換言之，在進行後續多參數模糊化模擬實驗時可透過模糊化閾值對其結果進行微調。

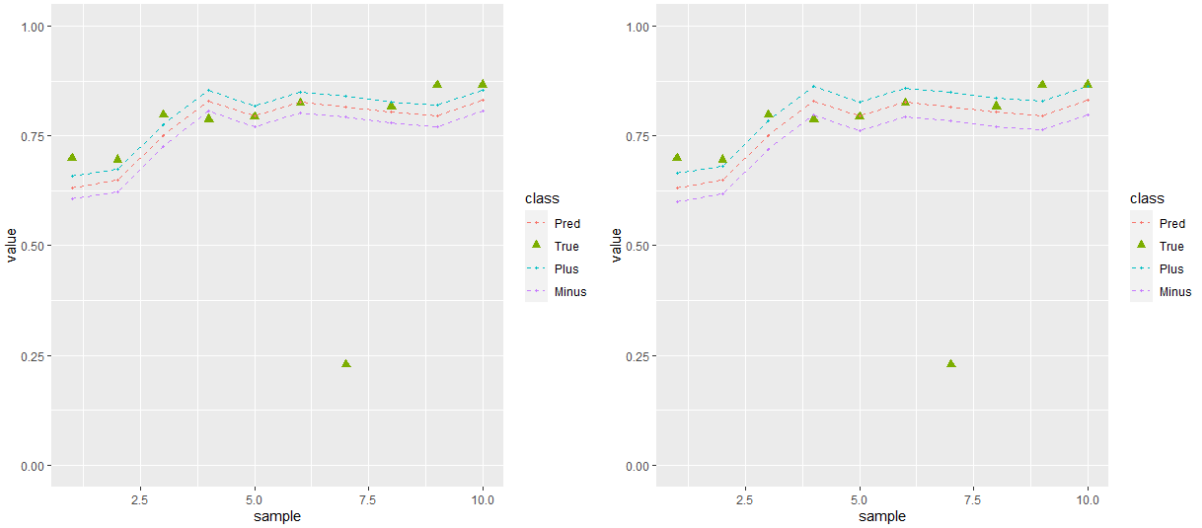

圖 4.9 模糊化單一閾值之結果

表 4.2 模糊化單一閾值之績效

模糊化之參數	Average Range	Raw Range	Hit Rate
3-1	0.04807596	44.28998	3/10
4-1	0.06496915	59.85283	3/10

4.3.3 模糊化兩個權重

接續 4.3.1 的實驗結果，在模糊化兩個權重時候挑選另一個權重的模糊化以改善權重 3-6 模糊化的效果。因此我們分別選擇權重 1-1、2-2、3-5 與權重 3-6 同時模糊化，結果可以看出 1-1/3-6 與 3-5/3-6 在各方面的表現皆非常相近(圖 4.10 左半部)且優於 2-2/3-6 的表現(圖 4.10 右上)。因此在後續多參數模糊化時可優先考慮這些組合以進行調整。

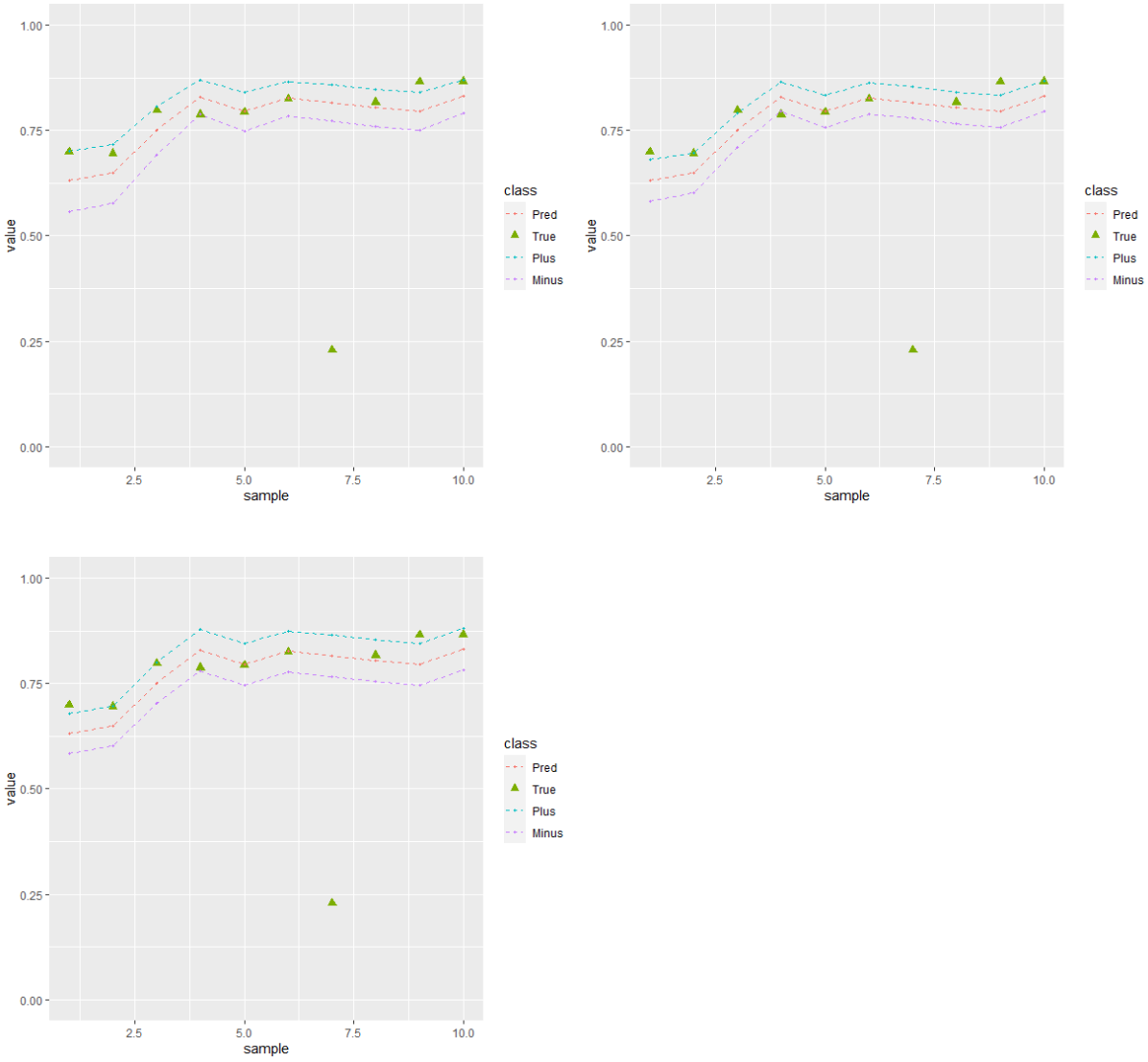

圖 4.10 模糊化兩個權重之結果

表 4.3 模糊化兩個權重之績效

模糊化之參數	Average Range	Raw Range	Hit Rate
1-1/3-6	0.09922887	91.4146	7/10
2-2/3-6	0.07904592	72.82106	5/10
3-5/3-6	0.09760638	89.91988	7/10

4.3.4 模糊化兩個閾值

接續 4.3.2 的實驗結果，可以看出模糊化 4-1 閾值所得到的命中率較高，因此判定

此閾值較適合作為改善的對象。實驗結果可以看出模糊化 3-1/4-1 與 3-3/4-1 可以達到相似的結果(圖 4.11)。

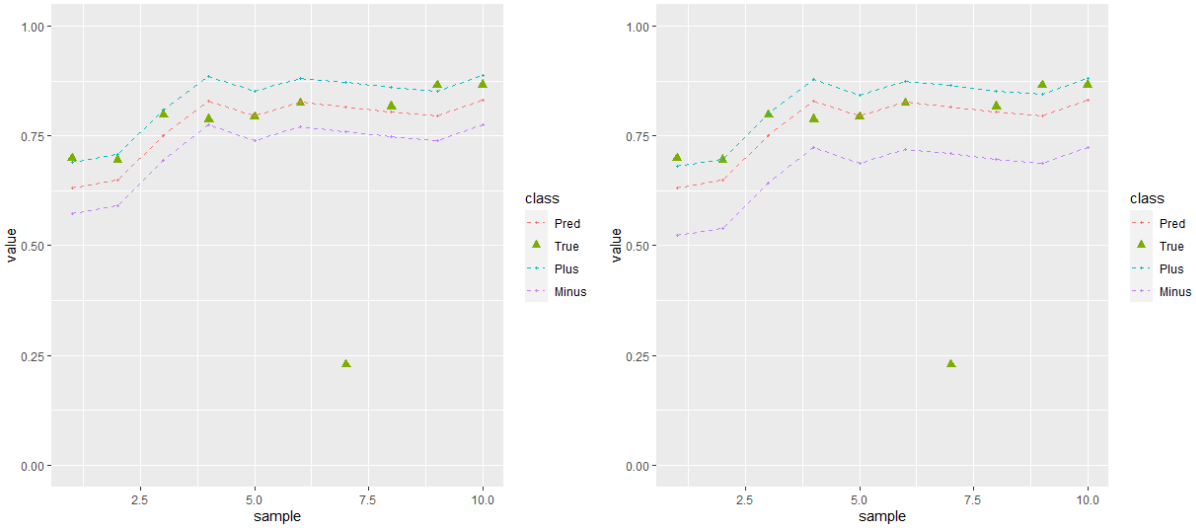

圖 4.11 模糊化兩個閾值之結果

表 4.4 模糊化兩個閾值之績效

模糊化之參數	Average Range	Raw Range	Hit Rate
3-1/4-1	0.1130451	104.1428	7/10
3-3/4-1	0.09652119	88.92014	7/10

4.3.5 模糊化一個權重與一個閾值

先前之實驗結果可以看出：模糊化單一權重會對結果產生較大的影響，而得到較寬的平均區間；模糊化閾值則對結果影響較小，而得到較窄的平均區間。因此本研究結合這兩種不同的特性，先對權重進行模糊化後，再對閾值進行模糊化，以在影響區間寬度最小的條件下對結果進行最佳化。權重的部分依舊選擇 3-6，閾值則是挑選先前實驗中表現較佳的 3-1、3-3 及 4-1 做搭配。結果可以看出在 3-6/4-1 搭配的表現最佳(圖 4.12 左下)，可以達到八成的命中率。

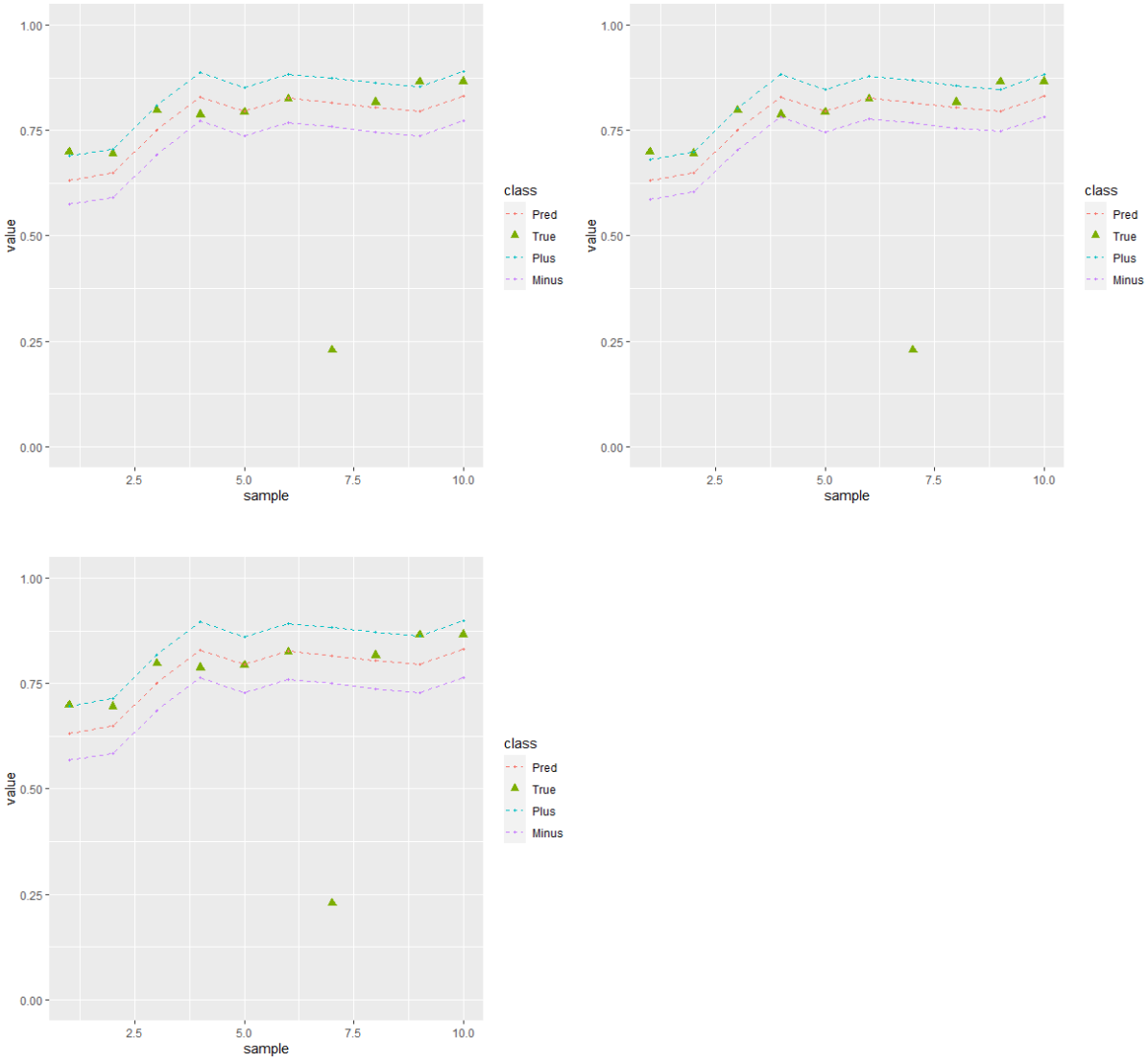

圖 4.12 模糊化一個權重與一個閾值之結果

表 4.5 模糊化一個權重與一個閾值之績效

模糊化之參數	Average Range	Raw Range	Hit Rate
3-1/3-6	0.1157227	106.6096	7/10
3-3/3-6	0.09923726	91.42232	7/10
3-6/4-1	0.1326159	122.1724	8/10

4.3.6 模糊化多個權重與閾值

最後，嘗試模糊化多個權重與閾值，實驗後可以達到九成的命中率(圖 4.13)，且與

在 4.4.1 模糊化單一權重的結果比較，平均區間之寬度減少了大約 20%左右，顯示出多個參數之模糊化結果會比單一參數模糊化的結果更加優異。

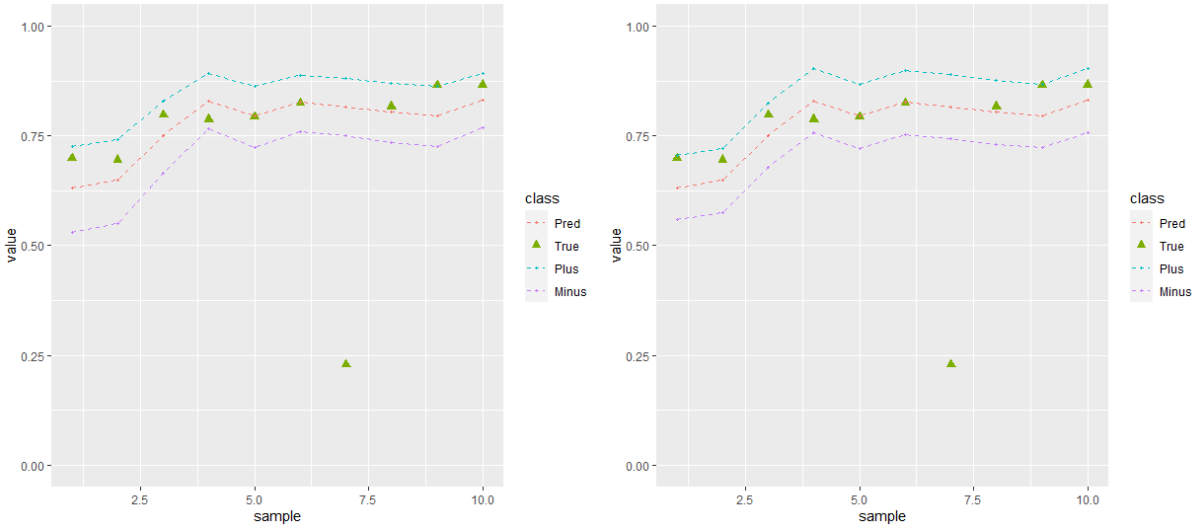

圖 4.13 模糊化多個權重與閾值之結果

表 4.6 模糊化多個權重與閾值之績效

模糊化之參數	Average Range	Raw Range	Hit Rate
1-1/3-1/3-6	0.147296	135.6964	9/10
3-1/3-5/3-6	0.1456823	134.2099	9/10
3-4	0.1818987	167.5742	9/10

4.4 結果與分析

根據實驗結果，進行了以下分析：

1. 模糊化單一權重：經過模擬實驗後可以發現，模糊化越靠近輸入層的權重對最後輸出值的影響越小；越靠近輸出層則反之。模糊化權重 3-4 可得到最高的命中率；模糊化權重 3-6 則是得到次高的命中率，但其模糊化後的平均區間較窄，因此可以在之後模糊化多個參數時優先考慮此權重。
2. 模糊化單一閾值：模糊化越靠近輸入層的閾值對最後的輸出結果之影響越小；越靠近輸出層則反之。此外，與模糊化權重相比，模糊化閾值所得出的模糊平均區間皆較

窄。模糊化閾值 3-1 與 4-1 得到的結果相似，都是較窄的平均區間且不高的命中率，因此可以在之後模糊多個參數時優先考慮權重與閾值的搭配。

3. 模糊化兩個權重：在模糊化單一權重時，模糊化權重 3-4 的結果僅未命中一真實值(離群值)，因此在模糊化兩個權重時則優先考慮改善空間較大的權重 3-6，希望可以透過模糊化另一權重使命中率上升。實驗結果顯示同時模糊化權重 3-5 與 3-6 可以得到最好的結果。
4. 模糊化兩個閾值：閾值 4-1 在模糊化單一閾值時的結果表現較佳，其命中率較高，因此希望透過模糊化另一閾值使命中率進一步提升，以包含更多真實值。實驗結果顯示同時模糊化閾值 4-1 與 3-3 可以得到最好的結果。
5. 同時模糊化一個權重與一個閾值：根據實驗結果顯示，模糊化閾值對輸出結果的影響較小，因此僅適合用於小範圍的調整。因此同樣以模糊化權重 3-6 為優先，再挑選一閾值進行模糊化。實驗結果顯示，模糊化權重 3-6 與閾值 4-1 能得到最高的命中率。
6. 模糊化多個權重與閾值：從先前實驗結果可看出在模糊化兩參數時得出的結果比模糊化單一參數有顯著的改善。因此，可以透過模糊化對輸出結果影響較小的參數(靠近輸入層之參數、閾值等)對結果進行改善。實驗結果顯示此方案在提升命中率的同時，也不會太過於增加區間的寬度。

下表列出命中率高於百分之九十的一些結果。傳統統計在常態分配的假設下，約有95%的抽樣結果會落在$\pm 2\sigma$之區間內。與此區間之長度做比較，可以看出透過模糊化參數建構出來的模糊區間的寬度相較於統計方法的區間大幅縮窄，且命中率表現相同。

表 4.7 模糊化參數之結果與比較

模糊化之參數	Average Range	Raw Range	Hit Rate
3-4	0.1818987	167.5742	**9/10**
1-1/3-1/3-6	0.147296	135.6964	**9/10**

3-1/3-5/3-6	**0.1456823**	**134.2099**	**9/10**
統計方法	0.8357581	816.1772	**9/10**

第五章 結論與未來研究方向

5.1 結論

生產週期時間受到許多不確定性因素影響，想百分之百預測正確是非常有難度的。在過去，預測晶圓批之生產週期時間多著重於提升預測的準確度。本篇研究提出一模糊類神經網路方法，在考慮準確度的同時，也進一步提升預測的精確度。在過去，晶圓廠通常會藉由過去經驗延後交期時間以確保能如期交貨，但這樣是無理論根據也未必可靠的。因此本研究透過模糊類神經網路的運算，得到出一個具有實質參考價值的生產週期時間區間，不僅對生產者本身，對客戶也是非常有幫助的。所提出之模糊類神經網路方法結合模糊理論與類神經網路之優點，可以在準確地預測生產週期時間之後，精確地估計期之範圍。與傳統之統計方法如點估計、區間估計等相比，本篇研究提出之方法無需太多的假設，也較有彈性。本方法透過模糊化類神經網路中的參數，以產生一晶圓批模糊生產週期時間的模糊週期時間預測值，也就是生產週期時間之區間。

根據實驗結果可整理出以下幾點結論：

(1) 模糊化越靠近輸入層的參數對最後的輸出值的影響越小；模糊化越靠近輸出層之參數則反之。

(2) 模糊化參數之數值越大，對最後的輸出值之影響越大，模糊化參數之數值越小則反之。

(3) 相較於模糊化連結權重，模糊化閾值對最後的輸出值之影響較小。

(4) 隨著模糊化參數的增加，模型的預測表現也會隨之提升。

(5) 本篇研究最終的結果為模糊化 3-1/3-5/3-6 這三個參數得出的預測績效為最佳。

(6) 與傳統之統計方法相比，本研究方法所得到的預測值區間之寬度大約縮減了 80%。

5.2 未來研究方向

本研究未來的研究方向包括：

(1) 對模型預測之績效進行交叉驗證(cross validation)：透過對資料集的切割並進行交叉驗證，以提高預測模型的可靠性。

(2) 使用更有效率的方法選擇欲模糊化之參數：本篇研究使用試誤法(try and error)挑選欲模糊化之參數，在之後的研究可以探討如何更有效率地挑選欲模糊化之參數以節省實驗時間，並提升預測精確度。

(3) 調整模糊化參數的幅度：本研究模糊化參數之方式統一為增加或減少該參數的 10%，後續研究可調整模糊化的幅度對輸出結果進行微調，使區間寬度在不被大幅拉長的情況下增加命中率。

(4) 增加模糊化參數之個數：本篇研究實驗結果顯示，隨著模糊化參數個數之增加，模型的表現也會提升。因此，後續研究可考慮模糊化更多個參數，或搭配不同模糊化幅度的組合以尋求更好的結果。

(5) 增加資料集資料筆數：可透過取得更多的公開資料來進行訓練與測試，用以測試模型在大量資料的情況下是否能夠保持良好的表現。

參考文獻

1 百度百科。自適應神經模糊系統。取自 https://reurl.cc/pmd7xb

2 科學 Online. (2011). 化學氣相沉積法(Chemical Vapor Deposition)。取自 https://highscope.ch.ntu.edu.tw/wordpress/?p=40948

3 財經新報(2020)。竹科下一個 40 年，高通提出三大產業趨勢嘉惠台灣。取自 https://finance.technews.tw/2020/12/17/qqualcomm-suggest-for-taiwan/

4 新浪新聞(2020)◦2021 年全球半導體市場規模預計達到 4694 億美元 同比增長 8.4%◦取自 https://news.sina.com.tw/article/20201202/37043912.html

5 經濟日報(2020)。積體電路、半導體封裝測試 1~9 月產值創歷年同期新高。取自 https://money.udn.com/money/story/5612/5047542

6 經濟日報(2020)。搞懂半導體產業，才能把台股當印鈔機！一張圖看如果晶圓代工是肥肉，材料就是蛋白質？取自 https://money.udn.com/money/story/5612/5089004

7 Integrated Circuits Manufacturing (IC)。取自 https://sites.google.com/site/icmanufacturing2204/

8 Abhishek, K et al. (2012). A rainfall prediction model using artificial neural network. 2012 IEEE Control and System Graduate Research Colloquium, 1, 82-87.

9 Aengchuan, P. (2018). Comparison of fuzzy inference system (FIS), FIS with artificial neural networks (FIS + ANN) and FIS with adaptive neuro-fuzzy inference system (FIS + ANFIS) for inventory control. Journal of Intelligent Manufacturing , 29, 905–923.

10 Alimissis, A. et al. (2018). Spatial estimation of urban air pollution with the use of artificial neural network models. Atmospheric Environment, 191, 205-213.

11 Chang, P. C. (2006). Fuzzy Delphi and back-propagation model for sales forecasting in

PCB industry. Expert Systems with Applications, 30(4), 715-726.

12 Chang, P. C. (2008). A fuzzy neural network for the flow time estimation in a semiconductor manufacturing factory. International Journal of Production Research, 46(4), 1017-1029.

13 Chen, J. C. et al. (2020). Capacity allocation with lot splitting in photolithography area using hybrid genetic algorithm based on self-tuning strategy. Computers & Industrial Engineering, 148, 106656.

14 Chen, T. (2001). Improvement of Projected-Out Date Accuracy in a Wafer Fab. National Science Council Project Report NSC 89-2213-E-275-005.

15 Chen, T. (2006). A Hybrid SOM-BPN Approach to Lot Output Time Prediction in a Wafer Fab. Neural Processing Letters, 24(3), 271-288.

16 Chen, T. (2007). An intelligent hybrid system for wafer lot output time prediction. Advanced Engineering Informatics, 21(1), 55-65.

17 Chen, T. (2008). A hybrid fuzzy-neural approach to job completion time prediction in a semiconductor fabrication factory. Neurocomputing, 71(16–18), 3193-3201.

18 Chen, T. et al. (2015). Fuzzy collaborative intelligence and systems, International Journal of Intelligent Systems, 30, 617-619.

19 Chen, T., & Wu, H. C. (2015). A new cloud computing method for establishing asymmetric cycle time intervals in a wafer fabrication factory. J Intell Manuf, 28, 1095-1097.

20 Concha, A. N. (2019). An Artificial Neural System to Predict Building Demolition Cost. 2019 IEEE 11th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM), 1, 1-4.

21 Ghasemi, A et al. (2020). Optimizing capacity allocation in semiconductor manufacturing photolithography area – Case study: Robert Bosch. Journal of Manufacturing Systems, 54,

123-137.

22 Gill, E. J. et al. (2010). Training back propagation neural networks with genetic algorithm for weather forecasting. IEEE 8th International Symposium on Intelligent Systems and Informatics, 1, 465-469.

23 Hu, Z. et al. (2019). Prediction of Fuel Consumption for Enroute Ship Based on Machine Learning. IEEE, 7, 119497-119505.

24 Idris, N. O. et al. (2020). Predicting the Selling Price of Cars Using Business Intelligence with the Feed-forward Backpropagation Algorithms. 2020 Fifth International Conference on Informatics and Computing (ICIC), 1, 1-6.

25 Joshi, S. (2016). Detection and Prediction of Diabetes Mellitus Using Back-Propagation Neural Network. 2016 International Conference on Micro-Electronics and Telecommunication Engineering (ICMETE), 1, 110-113.

26 Kadir, M. K. A. et al. (2014). Wheat yield prediction: Artificial neural network based approach. 2014 4th International Conference on Engineering Technology and Technopreneuship (ICE2T), 1, 161-165.

27 Khashei, M. (2008). A new hybrid artificial neural networks and fuzzy regression model for time series forecasting. Fuzzy Sets and Systems, 159(7), 769-786.

28 Kim, T. S. et al. (2002). Intelligent yield and speed prediction models for high-speed microprocessors. 52nd Electronic Components and Technology Conference 2002. (Cat. No.02CH37345), 1, 1158-1162.

29 Li, C. S. et al. (2010). Application of back-propagation artificial neural network to predict maintenance costs and budget for university buildings. 2010 Sixth International Conference on Natural Computation, 1, 1546-1551.

30 Moghaddam, A. H. et al. (2016). Stock market index prediction using artificial neural network. Journal of Economics, Finance and Administrative Science, 21(41), 89-93.

31 Pei Yi, Hao. (2020). Forecasting the Trends of Stock Price through Social Networks by Fuzzy Support Vector Machine with Possibility Measures. Proceedings of the 7th Multidisciplinary in International Social Networks Conference and The 3rd International Conference on Economics, 8, 1-7.

32 Rhee, T. G. et al. (2016). Predicting Movie Box Office Profitability: A Neural Network Approach. 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA), 1, 665-670.

33 Salah, M et al. (2018). Predicting Medical Expenses Using Artificial Neural Network. Knowledge and Information Systems, 2(10), 11-17.

34 Shenfield, A. et al. (2018). Intelligent intrusion detection systems using artificial neural networks. ICT Express, 4(2), 95-99.

35 Wu, L. (2010). Fuzzy neural network based yield prediction model for semiconductor manufacturing system. International Journal of Production Research , 48(11), 3225-3243.

36 Yang ,Liu. (2021). Foreign Trade Export Forecast Based on Fuzzy Neural Network. Complexity, 2021.

37 Yousif Alyousifi (2020). Predicting Daily Air Pollution Index Based on Fuzzy Time Series Markov Chain Model. Symmetry 2020, 12(2), 293.

38 Zadeh, L. A. (1965). Fuzzy sets. Information and Control, 8(3), 338-353.

39 Zhang, X. et al. (2019). Fundamentals of nano/microfabrication and scale effect. Molecular Sensors and Nanodevices (Second Edition), 43-111.

附錄一

```
library(neuralnet)   #呼叫 neuralnet 包
library(caret) #呼叫 caret 包


data <- read.csv("raw_data.csv") #讀取資料
maxmin <- function(x) ((x-min(x))/(max(x)-min(x))*0.8+0.1) #定義標準化函數
data <- as.data.frame(apply(data,2,maxmin)) #對資料進行標準化

####Outlier####
boxplot(data) #查看資料之箱型圖
data_new <- subset(data,!data$qn%in%boxplot(data$qn)$out) #去除 qn 欄位之離群值並指派為新資料
boxplot(data_new) #查看新資料之箱型圖
data_new <- subset(data_new,!data_new$Dn%in%boxplot(data_new$Dn)$out) #去除 Dn 欄位之離群值並指派為新資料
boxplot(data_new) #查看新資料之箱型圖


data <- data_new #將新資料指派給資料
data <- data_new[,-1] #將資料中的第一個欄位去除

boxplot(data) #查看處理離群值後的資料箱型圖

set.seed(81)
idx <- createDataPartition(data$an,p=3/4,list = F) #以四等分切割資料
data_tr <- data[idx,] #3/4 為訓練資料
data_ts <- data[-idx,] #1/4 為訓練資料

bpn <- neuralnet(formula = an~.,
                 data = data_tr,
                 hidden = c(3),
                 learningrate = 0.01,
                 threshold = 0.01,
                 stepmax = 500000,
                 linear.output = T
) #建構一初始類神經網路
```

```
pred_bpn <- compute(bpn,data_ts[,1:length(data_ts)-1]) #進行預測
pred_bpn <- as.data.frame(pred_bpn$net.result) #整理為資料集形式
colnames(pred_bpn) <- c("pred_bpn") #更改欄位名稱
result <- cbind(data_ts$an,pred_bpn) #將結果指派給 result

(MSE_bpn1 <- sqrt(sum((pred_bpn-data_ts$an)^2/length(data_ts$an)))) #計算模型預測結果之 MSE
(MAE_bpn1 <- sum(abs(pred_bpn-data_ts$an))/length(data_ts$an)) #計算模型預測結果之 MAE

model <- train(form = an~.,
                        data = data_tr,
                        method = "neuralnet",
                        tuneGrid = expand.grid(.layer1=c(1:5),
                                               .layer2=c(0:5),
                                               .layer3=c(0)),
                        learningrate = 0.005,
                        threshold = 0.01,
                        stepmax = 500000
) #調整類神經網路之參數

model #查看最佳結果

set.seed(52)
bpn2 <- neuralnet(formula = an~.,
                            data = data_tr,
                            hidden = c(1,3),
                            learningrate = 0.005,
                            threshold = 0.01,
                            stepmax = 500000,
                            linear.output = T
) #以最佳參數組合建構一類神經網路

pred_bpn2 <- compute(bpn2,data_ts[,1:length(data_ts)-1]) #進行預測
pred_bpn2 <- as.data.frame(pred_bpn2$net.result) #整理為資料集形式
colnames(pred_bpn2) <- c("pred_bpn") #更改欄位名稱
result2 <- cbind(data_ts$an,pred_bpn2) #將結果指派給 result2
```

```
####PLOT####
class <- rep(c("True","Pred"),each=length(data_ts$an)) #將真實值標註並指派給 class
value <- c(result2$`data_ts$an`,result2$pred_bpn) #將模型預測結果指派給 value
sample <- c(1:length(data_ts$an)) #將資料數量指派給 sample
rs <- data.frame(sample = sample,class = class,value = value) #將上述三種資料轉為資料集
形式並指派給 rs

ggplot(data = rs,mapping = aes(x=sample,y=value,color=class))+
  geom_line(aes(linetype = class))+
  geom_point()+
  scale_linetype_manual(values = c("dashed","blank"))+
  scale_y_continuous(limits = c(0,1)) #將結果視覺化


###########FUZZY PLUS###########模糊化上限###########

plot(bpn2) #查看類神經網路之架構
bpn2$weights #查看類神經網路之參數
bpn_puls <- bpn2 #將類神經網路之參數指派給新的 bpn_puls 作為模糊類神經網路

#Weight 模糊化權重
bpn_puls$weights[[1]][[3]][4,1] <- bpn_puls$weights[[1]][[3]][4,1] +
  0.1*bpn_puls$weights[[1]][[3]][4,1] #模糊化參數 3-6

bpn_puls$weights[[1]][[3]][3,1] <- bpn_puls$weights[[1]][[3]][3,1] -
  0.1*bpn_puls$weights[[1]][[3]][3,1] #模糊化參數 3-5

bpn_puls$weights[[1]][[3]][2,1] <- bpn_puls$weights[[1]][[3]][2,1] +
  0.1*bpn_puls$weights[[1]][[3]][2,1] #模糊化參數 3-4

bpn_puls$weights[[1]][[2]][2,1] <- bpn_puls$weights[[1]][[2]][2,1] -
  0.1*bpn_puls$weights[[1]][[2]][2,1] #模糊化參數 2-2

bpn_puls$weights[[1]][[1]][2,1] <- bpn_puls$weights[[1]][[1]][2,1] +
  0.1*bpn_puls$weights[[1]][[1]][2,1] #模糊化參數 1-1

bpn_puls$weights[[1]][[1]][5,1] <- bpn_puls$weights[[1]][[1]][5,1] +
```

```
  0.1*bpn_puls$weights[[1]][[1]][5,1] #模糊化參數 1-4

bpn_puls$weights[[1]][[1]][3,1] <- bpn_puls$weights[[1]][[1]][3,1] +
  0.1*bpn_puls$weights[[1]][[1]][3,1] #模糊化參數 1-2

#Thereshold 模糊化閾值
bpn_puls$weights[[1]][[2]][1,1] <- bpn_puls$weights[[1]][[2]][1,1] +
  0.1*bpn_puls$weights[[1]][[2]][1,1] #模糊化參數 3-1

bpn_puls$weights[[1]][[2]][1,3] <- bpn_puls$weights[[1]][[2]][1,3] -
  0.1*bpn_puls$weights[[1]][[2]][1,3] #模糊化參數 3-3

bpn_puls$weights[[1]][[3]][1,1] <- bpn_puls$weights[[1]][[3]][1,1] -
  0.1*bpn_puls$weights[[1]][[3]][1,1] #模糊化參數 4-1

pred_bpn_plus <- compute(bpn_puls,data_ts[,1:length(data_ts)-1]) #以模糊化後之類神經網
路模型進行預測
pred_bpn_plus <- as.data.frame(pred_bpn_plus$net.result) #整理為資料集形式
colnames(pred_bpn_plus) <- c("pred_bpn") #更改欄位名稱
result_plus <- cbind(data_ts$an,pred_bpn_plus) #將結果指派給 result_plus

####PLUS####
classP <- c("Plus") #將"Plus"字元指派給 classP
valueP <- c(result_plus$pred_bpn) #將模糊上限預測結果指派給 valueP
sampleP <- c(1:length(data_ts$an)) #將樣本個數指派給 sampleP
rsP <- data.frame(sample = sampleP,class = classP,value = valueP) #將上述三種資料轉為資
料集形式並指派給 rsP

############FUZZY MINUS############模糊化下限############

bpn_minus <- bpn2 #將類神經網路之參數指派給新的 bpn_minus 作為模糊類神經網路

#Weight 模糊化權重
bpn_minus$weights[[1]][[3]][4,1] <- bpn_minus$weights[[1]][[3]][4,1] -
  0.1*bpn_minus$weights[[1]][[3]][4,1] #模糊化參數 3-6

bpn_minus$weights[[1]][[3]][3,1] <- bpn_minus$weights[[1]][[3]][3,1] +
  0.1*bpn_minus$weights[[1]][[3]][3,1] #模糊化參數 3-5
```

```
bpn_minus$weights[[1]][[3]][2,1] <- bpn_minus$weights[[1]][[3]][2,1] -
  0.1*bpn_minus$weights[[1]][[3]][2,1] #模糊化參數 3-4

bpn_minus$weights[[1]][[2]][2,1] <- bpn_minus$weights[[1]][[2]][2,1] +
  0.1*bpn_minus$weights[[1]][[2]][2,1] #模糊化參數 2-2

bpn_minus$weights[[1]][[1]][2,1] <- bpn_minus$weights[[1]][[1]][2,1] -
  0.1*bpn_minus$weights[[1]][[1]][2,1] #模糊化參數 1-1

bpn_minus$weights[[1]][[1]][5,1] <- bpn_minus$weights[[1]][[1]][5,1] -
  0.1*bpn_minus$weights[[1]][[1]][5,1] #模糊化參數 1-4

bpn_minus$weights[[1]][[1]][3,1] <- bpn_minus$weights[[1]][[1]][3,1] -
  0.1*bpn_minus$weights[[1]][[1]][3,1] #模糊化參數 1-2

#Thereshold 模糊化閾值
bpn_minus$weights[[1]][[2]][1,1] <- bpn_minus$weights[[1]][[2]][1,1] -
  0.1*bpn_minus$weights[[1]][[2]][1,1] #模糊化參數 3-1

bpn_minus$weights[[1]][[2]][1,3] <- bpn_minus$weights[[1]][[2]][1,3] +
  0.1*bpn_minus$weights[[1]][[2]][1,3] #模糊化參數 3-3

bpn_minus$weights[[1]][[3]][1,1] <- bpn_minus$weights[[1]][[3]][1,1] +
  0.1*bpn_minus$weights[[1]][[3]][1,1] #模糊化參數 4-1

pred_bpn_minus <- compute(bpn_minus,data_ts[,1:length(data_ts)-1]) #以模糊化後之類神經網路模型進行預測
pred_bpn_minus <- as.data.frame(pred_bpn_minus$net.result) #整理為資料集形式
colnames(pred_bpn_minus) <- c("pred_bpn") #更改欄位名稱
result_minus <- cbind(data_ts$an,pred_bpn_minus) #將結果指派給 result_minus

####MINUS####
classM <- c("Minus") #將"Minus"字元指派給 classM
valueM <- c(result_minus$pred_bpn) #將模糊下限預測結果指派給 valueM
sampleM <- c(1:length(data_ts$an)) #將樣本個數指派給 smapleM
rsM <- data.frame(sample = sampleM,class = classM,value = valueM) #將上述三種資料轉為資料集形式並指派給 rsM
```

```
############Performance############
(MSE_bpn2 <- sqrt(sum((pred_bpn2-data_ts$an)^2/length(data_ts$an)))) #計算模型預測結果之 MSE

(MAE_bpn2 <- (sum(abs(pred_bpn2-data_ts$an))/12)) #計算模型預測結果之 MAE

(RMSE_bpn2 <-sqrt(sum((pred_bpn2-data_ts$an)^2)/length(data_ts$an))) #計算模型預測結果之 RMSE

(MAPE_bpn2 <- sum((abs(pred_bpn2-data_ts$an)/data_ts$an))/length(data_ts$an)*100) #計算模型預測結果之 MAPE

FinalData <- rbind(rs,rsP,rsM) #將預測結果、模糊上下限結果合併為 FinalData

ggplot(data = FinalData,mapping = aes(x=sample,y=value,color=class))+
  geom_line(aes(linetype = class))+
  geom_point(size=2)+
  scale_linetype_manual(values = c("dashed","blank","dashed","dashed"))+
  scale_y_continuous(limits = c(0,1)) #將結果視覺化


up <- subset(FinalData,class=="Plus") #將模糊上限資料指派給 up
down <- subset(FinalData,class=="Minus") #將模糊下限資料指派給 down

rawd <- read.csv("raw_data.csv") #讀取原始資料並指派給 rawd

raw_up <- ((up$value-0.1)*(max(rawd$an)-min(rawd$an)))/0.8+min(rawd$an) #將模糊上線反標準化並指派給 raw_up
raw_down <- ((down$value-0.1)*(max(rawd$an)-min(rawd$an)))/0.8+min(rawd$an) #將模糊上線反標準化並指派給 raw_down

(range_raw <- sum(up$value-down$value)/length(data_ts$an)) #計算原始區間寬度
(range <- (sum(raw_up-raw_down))/length(data_ts$an)) #計算區間寬度

mean(data$an)+2*sd(data$an) #統計兩倍標準差上界
mean(data$an)-2*sd(data$an) #統計兩倍標準差下界
(mean(data$an)+2*sd(data$an))-(mean(data$an)-2*sd(data$an)) #統計方法區間
```

mean(rawd$an)+2*sd(rawd$an) #統計兩倍標準差上界
mean(rawd$an)-2*sd(rawd$an) #統計兩倍標準差下界
(mean(rawd$an)+2*sd(rawd$an))-(mean(rawd$an)-2*sd(rawd$an)) #統計方法區間

附錄二

qn	Un	Qn	BQn	WIPn	Dn	FQn	an
11	0.83	104	52	156	143	110	1809
24	0.86	110	55	160	145	114	1853
24	0.79	114	59	167	261	115	1858
24	0.79	115	61	170	261	123	1401
24	0.79	114	61	172	59	122	1881
24	0.82	115	61	176	59	126	1408
24	0.82	117	59	177	59	126	1881
24	0.82	118	63	182	59	131	1889
24	0.82	119	64	186	59	134	1885
24	0.82	120	65	187	59	136	1447
24	0.82	121	65	188	59	137	1943
24	0.82	122	65	191	64	140	1949
24	0.82	127	65	197	64	145	1976
24	0.88	129	65	202	251	150	1979
24	0.88	129	66	203	251	150	1985
10	0.88	132	68	209	251	154	1496
24	0.88	135	68	214	251	161	1990
24	0.88	135	65	216	420	160	1990
24	0.88	137	66	217	420	161	1992
5	0.98	139	71	224	54	171	1899
24	1	140	72	229	44	174	1970
24	1	140	73	229	61	176	1481
24	1	141	72	230	61	180	1980
24	0.86	142	76	231	81	180	1976
24	0.86	144	74	231	28	176	1992
24	0.86	146	76	232	28	177	1992
24	0.97	145	78	233	92	184	1983
24	0.97	146	77	232	122	183	2004
22	0.95	152	81	235	171	183	1987
24	0.95	154	86	238	84	190	2005
24	0.91	156	84	240	84	186	1999
24	0.91	156	86	242	84	190	1456
24	0.91	154	88	242	197	189	2003
24	0.65	157	98	245	45	195	1971
24	0.65	158	93	245	189	193	1417
24	0.76	158	96	246	189	192	1974
24	0.76	159	96	245	165	193	1990
24	0.76	162	96	246	165	196	2017
24	0.86	163	96	247	440	195	2001
24	0.86	164	98	247	16	198	1997
24	0.86	165	94	248	16	198	2009
24	0.86	165	96	250	16	199	2005
24	0.98	168	96	251	65	198	2046
24	0.98	170	100	252	222	200	2034
24	0.84	170	102	256	222	208	2042
24	0.84	168	99	255	49	205	2037
24	0.84	167	100	256	230	207	2074
24	0.99	169	108	255	90	204	2046
24	0.99	169	102	256	218	202	2048
24	0.99	172	105	257	218	205	2043
24	0.95	174	109	258	637	208	2066
24	0.95	177	110	258	51	206	2068
24	0.95	178	110	258	676	208	1337
24	0.98	178	113	259	676	213	2062
24	0.98	179	114	260	676	210	2060
2	0.93	180	113	262	146	210	1939