Software Engineering and Applications 软件工程与应用, 2013, 2, 62-68
http://dx.doi.org/10.12677/sea.2013.23011 Published Online June 2013 (http://www.hanspub.org/journal/sea.html)

Application of General Regression Neural Network in Software Development Cost Estimation Based on Principle Component Analysis

Yu-Chia Chen, Jung-Hua Lo

School of Department of Applied Informatics, Fo Guang University, Yilan
Email: jessicachen0922@gmail.com, jhlo@mail.fgu.edu.tw

Received: Mar. 18th, 2013 revised: Mar. 29th, 2013; accepted: Apr. 10th, 2013

Abstract: The quality of project management and project cost are important factors affecting the success of software projects. These critical factors for software development contain time, quality and cost. Nowadays, the most popular method for software development cost estimation is judged by the project manager's experience. The project manager needs to estimate a reasonable software development cost according to the previous relevant data information of project while facing the problem. Therefore, in this research we propose a new cost estimation model based on the Principle Component Analysis (PCA) and General Regression Neural Network (GRNN) for software development project.

Keywords: Software Development Costs; Principle Component Analysis; General Regression Neural Network

基于主成分分析的 GRNN 在软件开发成本预测的应用

陈妤嘉，罗荣华

佛光大学信息应用学系，宜兰
Email: jessicachen0922@gmail.com, jhlo@mail.fgu.edu.tw

收稿日期：2013 年 3 月 18 日；修回日期：2013 年 3 月 29 日；录用日期：2013 年 4 月 24 日

摘　要：软件项目管理的质量与开发成本，决定了软件项目的成功与否，时间、质量与成本项目的考虑，成为了影响软件开发成本的关键性要素，项目经理在面对软件开发项目时，即需要预估合理的软件开发成本，现行软件产业最普遍使用的软件成本预估方法，多以项目经理的经验为基础，参考过去开发过的软件项目数据作为主观的项目预估值，本研究提出一个能运行于软件项目开发成本预测模型之研究，运用主成分分析与广义回归类神经网络结合的一种新的预测方法。

关键词：软件开发成本；主成分分析；广义回归类神经网络

1. 引言

软件开发流程的首要步骤在于软件项目的规划，软件开发过程通常是变动性、无法预测和动态的，其考虑的因素有开发成本、开发时程、风险评估、人力资源、软硬件技术与设备采购等管理活动，其中开发成本估算，将是影响企业组织抉择是否参与该项项目投资的关键要素，软件项目成本来自于开发成本与维护成本，Capers Jones，Olivier Bonsignour[1]提到软件质量牵动着企业组织能获得的商业价值和经济成本的回馈。因此，若能使用最快的开发速度与较低的成

本，发展出较高质量的软件，将会是引颈期盼；准确预估开发成本是有助于提升软件开发的成功率，减缓软件开发失败的风险，并增强企业组织的竞争力，软件开发成本管理在于决定项目期间的工作量、成本与相关资源的适宜性问题，管理期间所进行的各项活动，也将影响整体开发成本的配置，因此控制软件项目开发成本，需仰赖于成本的估算。

2. 软件开发成本的预测

工作量(Effort)、成本(Cost)、开发时程(Schedule)是软件成本衡量的重要因素，一般而言预估模式来自于软件开发工作量的评估，以工作量多寡来衡量软件规模(Software Size)大小，已是软件成本衡量指针(Software Cost Metrics)的通用模式，软件开发工作量为软件项目规划期间所耗用的人力资源，即所谓投入人月(Person Month, PM)，软件开发工作量评估方法依据需求目的的不同，可分为统计模式(Statistically Based Model)与人工智能模式(Artificial Intelligence Model)，以统计模式而言，参数式预估模式(Parametric Models)是软件开发工作量预估方法中最为常用的模式；依据过去已开发完成软件项目的历史数据，找出影响开发工作量的因素，再使用统计方法推论出软件工作量的估算方程式，其中最著名与普遍性使用的是 COCOMO 统计模式衡量方法，以 COCOMO 模式在相关文献的研究较为久远，本研究期许以 COCOMO II 模式之软件开发工作量做为软件成本衡量的指针。

3. COCOMO II 统计模式

COCOMO II (Constructive Cost Model II)统计模型以COCOMO模式做为基础修正后的成本预估模型，由 Boehm，Clark，Horowitz，Madachy，Shelby and Westland (1995)，依据 TRW 公司开发过的 63 个项目数据，并假设软件规模与发展时程属于非线性关系[2]，而且不同的开发环境有着不同的估算方式，这个方式可以使用在衡量时程的方法，并估算软件生命周期(Systems Development Life Cycle, SDLC)的每个阶段要如何分配人力，也可以利用在敏感度分析(Sensitivity Analysis)的使用上，因应软件项目的不同模式行为或变化情形，拟定不同的策略与调整成本因子[3]。

Boehm (1995)等人依据软件开发成本估算值的不同，区分成三种不同模型有基本模型(Basic Model)、中级模型(Intermediate Model)、详细模型(Detailed Model)，每一模型又根据软件项目的复杂程度，分为有机型(Organic Mode)、半分离型(Semi-Detached Mode)及嵌入型(Embedded Mode)等三种程度，COCOMO 模式复杂度系数[4]，如表 1。

Boehm, Reifer and Chulani (1998)将 COCOMO II 统计模型用于软件开发成本估算，此模式区分为三个阶段，有应用程序构成模型(Application Composition Model)[5]、早期设计模型(Early Design Model)、后期架构模型(Post-Architecture Stage)，依据 Tadayon (2005)完整模型采用应用程序构成模型与后期架构模型[6]，此模型包含 5 个衡量要素 SF (Scale Factor)与 17 个成本因子(Cost Drivers)指针，做为类神经网络的输入变量，表 2 提供了 COCOMO II 早期设计模型与后期结

Table 1. COCOMO model complexity factor
表 1. COCOMO 模式复杂度系数

模式类型	基本模式		中级模式		详细模式	
	a	b	a	b	a	b
有机型	2.4	1.05	3.2	1.05	3.2	1.05
半分离型	3.0	1.12	3.2	1.12	3.2	1.12
嵌入型	3.6	1.20	2.8	1.20	2.8	1.20

Table 2. The COCOMO II Early Design with Post-Architecture Stage structure model of the cost factor differences relationship (Data source: Boehm, 2000)
表 2. COCOMO II 早期设计模型与后期结构模型成本因子差异关系(数据源：Boehm，2000)

属性名称	早期设计模型成本因子	后期结构模型成本因子
产品属性	产品可靠性与复杂度(RCPX) 程序代码重复使用的要求(RUSE)	软件可靠性(RELY) 数据库大小(DATA) 产品复杂度(CPLX) 开发流程需求文件的合适度(DOCU) 可重复使用(RUSE)
平台属性	平台的困难度(PDIF)	运行时间的限制(TIME) 主要储存装置限制(STOR) 平台变动的程度(PVOL)
人员属性	人员能力(PERS) 人员经验(PREX)	分析师的能力(ACAP) 程序设计师的能力(PCAP) 员工的流动性(PCON) 实施上的经验(AEXP) 平台的经验(PEXP) 程序语言工具的经验(LTEX)
项目属性	熟练度(FCIL) 排程的执行进度(SCED)	软件工具的使用价值(TOOL) 多点开发(SITE) 发展时程表(SCED)

构模型成本因子差异关系；而在软件规模则是取用程序千行数(Kilo Lines Of Code, KLOC)方法，故此，本研究也以此方法做为遵循的方向。

4. 主成分分析方法

1901 年由 Karl Pearson 提出主成分(Principle Component)的概念，用以观察多个变量间的关系性进行的一种多元统计方法，1933 年由 Harold Hotelling 加以发展。在大部份所进行的研究过程里，收集到的数据可能都包含了多个变量，每个变量之间都会有一定的相关性，而当变量过多时，要透过每个变量去解释其对应的关系性，就会变得困难与复杂。

主成分分析(Principle Component Analysis, PCA)方法的主要功能，利用原相关性的数据变量去组合新的独立的线性组合变量，透过新的组合变量去解释大部份原始数据的变异性[7]，相关性较高的变量经由线性组合后，可获得成分中较高的变异数(Variance)，其中变异数就是利用特征值(Eigenvalue)与特征向量(Eigenvector)的方法，筛选出占最大变异数的数据，将多个变量进行转换成主成分后，即可利用较高的主成分去说明原始数据相关性，简而言之，即新变量的变异数愈大，解释数据的能力也就愈强，因此，主成分分析是最常被选用来寻找判断某种事物或某种现象的综合指标(Metrics)，对于综合指针所包含的讯息再给予适当的解释，也可称为是一种数据简化(Data Reduction)方法或是维度简化(Dimensional Reduction)的方法。

5. 广义回归类神经网络

广义回归类神经网络(General Regression Neural Network, GRNN)是由机率类神经网络(Probabilistic Neural Network, PNN)所演化而来，学习策略属于监督式学习网络(Supervised Learning Network)的一种。1988 年 D. F. Specht 提出机率类神经网络，属于无迭代式学习(One-pass Learning)算法，适合用于解决分类(Classification)问题，而无法解决连续变量(Continuous Variabled)问题。于 1991 年 D. F. Specht 再提出广义回归类神经网络学习算法，此网络不仅可以做分类(Classification)问题，更可学习一个动态模式作为控制使用或预测使用。广义回归类神经网络主要的特性是学习速度快，所需调整的参数也仅有一个即为平滑参数，并且在少量的训练样本数的情况下，也可以获得良好的预测效果。相较与其他监督式学习网络之优点：

1) 不用初始网络链接加权值。

2) 学习过程与回想(Recall)过程不相关。

3) 不用推测的输出向量(Output Value)与训练范例的目标输出向量的差距，来修正网络链接加权值。

4) 无迭代的学习过程(One-pass learning)。

5) 学习过程的目的是在于寻找优化平滑参数 σ 值。

6) 网络的神经元数与训练范例相关。

广义回归类神经网络架构上是一个四层神经元结构的网络模型，如图 1 所示，第一层为输入单元(Input Unit)，又称为分配单元，负责将输入向量 x 测量值分配给第二层型态单元(Patten Unit)内的所有神经元。第二层为型态单元，每一个型态单元都代表着一个训练范例或是一个群集中心(Cluster center)，一个新的输入向量 x 测量值进入网络后，此向量会减去训练范例的向量，两者差之平方值会进行加总，并输入到非线性的作用函数，而作用函数出来的值，就是型态单元的输出值。此输出值会被传送到总和单元(Summation Unit)，第三层为总和单元，型态单元有两个输出值，分别为 Y 的所有测量值乘以加权向量值的加总以及完成加权向量的加总，第四层为输出单元(Output Unit)，为总和单元的两个输出值相除，可得到 y 的估计值，如公式 1。

$$\hat{y}(x)=\frac{\sum_{i=1}^{n} y_i exp\left[-\frac{D_i^2}{2\sigma^2}\right]}{\sum_{i=1}^{n} exp\left[-\frac{D_i^2}{2\sigma^2}\right]} \tag{1}$$

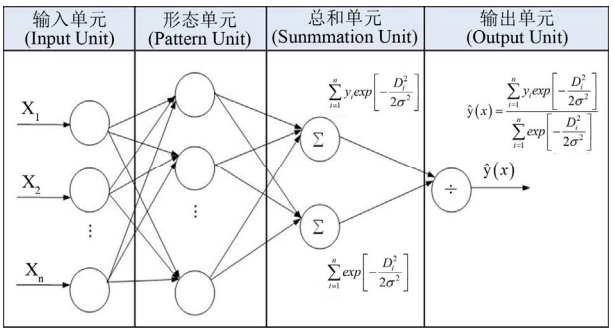

Figure 1. General regression neural network architecture
图 1. 广义回归类神经网络架构

6. 影响软件开发成本因子指标

透过历史数据的分析或利用实验研究的方法，找出影响软件开发成本的重要因子，本研究采用 COCOMO II 统计模式做为衡量方法，依据过去已开发完成软件项目的历史数据，找出影响开发工作量的指标因素。

在选用成本因子指标的考虑上，以软件可靠性(x_1)、数据库大小(x_2)、产品复杂度(x_3)、可重复使用(x_4)、开发流程需求文件的合适度(x_5)、运行时间的限制(x_6)、主要储存装置限制(x_7)、平台变动的程度(x_8)、分析师的能力(x_9)、程序设计师的能力(x_{10})、实施上的经验(x_{11})、平台的经验(x_{12})、程序语言工具的经验(x_{13})、员工的流动性(x_{14})、软件工具的使用价值(x_{15})、多点开发(x_{16})、发展时程表(x_{17})、程序行数(x_{18})等作为本研究的成本因子指标。

为了促使网络训练模式在进行预测时，不会因变量值域过大，而影响学习效果与预测的准确度，因此必须将值域过大的变量进行正规化(Scaling)处理，将值域范围对应到较小的值域上，通常以采用对数正规化方法。上述成本因子指标彼此间都具有一定的相关性，若将所有指针都作为网络训练的输入变量，必然会增加网络训练的复杂性，同时也有可能会影响到实验结果的可靠性与准确性，因此，利用主成分分析方法，将成本因子指标先进行降维处理，以提高网络运算的效率。

7. PCA-GRNN 预测模式

软件开发成本的影响因子至少存在一个以上数据变量量，本文提出一个基于主成分分析与广义回归类神经网络结合于软件开发成本预测的应用，经本研究应用于软件开发成本预估之实验结果中证明，所提出的预测模式除了在预测上有较良好的准确性，对于学习速度上也更为提升。

本研究于软件开发成本影响成本因子指标选用了 18 个成本因子指标，每一成本因子指标彼此间都具有一定的相关性，因此增加了网络训练的复杂性，同时也有可能影响到实验结果的可靠性。再者当指针变量过多时，也会影响到网络训练的学习效果，为了提高网络学习速度，利用主成分分析方法取得 m 个主成分作为广义回归类神经网络的输入变量，再与广义回归类神经网络结合，建构出新的预测模式，得以求出较佳的预测效果。以软件开发成本影响因子建构新的预测模式，利用主成分分析与广义回归类神经网络结合(PCA-GRNN)步骤，如图 2 所示。

PCA-GRNN 预测模式执行步骤，分述如下：

Step 1：将 m 个原始数据变量进行标准化。标准化函式如公式 2。

$$u_m = \frac{x_m - \bar{x}_m}{s_m} \tag{2}$$

$\bar{x}_m$ 为指标平均值；s_m 为指标的标准偏差；u_m 为标准化数值。

Step 2：取得相关系数的关系性。

以任意二个变数 x_1 及 x_m 的相关系数 r，如公式 3，经由相关系数计算，得到相关系数矩阵 R，如公式 4。

$$r = \frac{\sum_{i=1}^{n}\left(x_1 - \bar{x}_1\right)\left(x_m - \bar{x}_m\right)}{\sqrt{\sum_{i=1}^{n}\left(x_1 - \bar{x}_1\right)^2}\sqrt{\sum_{i=1}^{n}\left(x_m - \bar{x}_m\right)^2}} \tag{3}$$

$$R = \begin{bmatrix} 1 & r_{x_1x_2} & \cdots & r_{x_1x_m} \\ r_{x_2x_1} & 1 & \cdots & r_{x_2x_m} \\ \vdots & \vdots & \ddots & \vdots \\ r_{x_mx_1} & r_{x_mx_2} & \cdots & 1 \end{bmatrix} \tag{4}$$

Step 3：取得特征值(Eigen Value)与特征向量(Eigen Vector)。

特征值是反应主成分对原始变量的影响程度，简而言之，即是传递主成分可以作为解释原始变量的

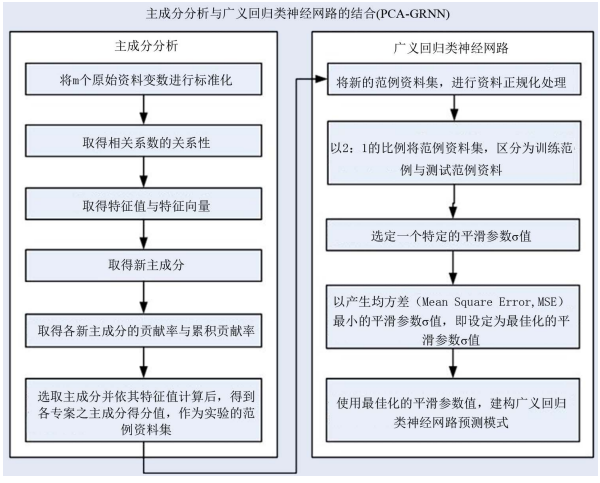

Figure 2. Principle component analysis and general regression neural network by step

图 2. 主成分分析与广义回归类神经网络结合步骤

讯息，一般情况，以特征值大于 1 时做为选用的标准。

使用特征公式$|\lambda_m - R| = 0$将相关系数矩阵 R 代入公式中，求出特征值 $\lambda_m (i = 1,2,3,\cdots,m)$ 后，再将特征值依照数值大小排序，即 $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \cdots \geq \lambda_m \geq 0$，依序求出相对应的特征向量 $y_m (i = 1,2,3,\cdots,m)$，即 $y_1 (y_{11}, y_{12}, y_{13}, \cdots, y_{1m})$。

Step 4：取得新主成分。

一般选用主成分的标准，为累积贡献率达到 80%以上即可做为选用的标准，但实际上仍需要视研究的主体情况，而本研究依累积贡献率达到 88.69%做为选用主成分的标准。

Step 5：取得各新主成分的贡献率与累积贡献率。

贡献率为新主成分在整个数值分析中的解释能力，贡献率愈大表示解释的能力愈强，而累积贡献率的大小则反应了可靠性强弱，累积贡献率愈大表示可靠性愈强。

计算各新主成分 z_m 的贡献率 ρ_m 与累计贡献率 t_m 公式。z_m 的贡献率为 $\rho_m = \lambda_i / (\lambda_1 + \lambda_2 + \lambda_3 + \cdots \lambda_m)$，主成分 $z_1, z_2, z_3, \cdots, z_m$ 的累积贡献率 $t_m = \rho_1 + \rho_2 + \rho_3 + \cdots + \rho_m$，然而 m 即表示为选取的主成分的个数。

Step 6：选取主成分并依其特征值计算后，得到各项目之主成分得分值，做为实验的范例数据集。

经由主成分分析方法得到 m (m < 18)个主成分，并称为第1…m 个主成分，主成分的贡献率愈大表示对变量量的解释讯息则愈强烈，将所挑选后的主成分作为广义回归类神经网络的训练范例数据集。

Step 7：将新的范例数据集，进行数据正规化处理。

Step 8：以 2:1 的比例将范例数据集，区分为训练范例与测试范例数据。

并将 m 个主成分视为网络输入变量，而软件开发成本衡量指标通常以投入人月做为基本的衡量方式。

Step 9：选定一个特定的平滑参数 σ 值。

Step 10：以产生均方差(Mean Square Error, MSE)最小的平滑参数 σ 值，即设定为优化的平滑参数 σ 值。

Step 11：使用优化的平滑参数 σ 值，建构广义回归类神经网络预测模式。

8. 范例数据收集限制

在台湾软件开发市场大部份仍属中小企业为最多，所承接的软件规模普遍性并不大，因此在预估软件开发成本时，几乎是采用专家判断或者是以企业组织成员最资深的项目经理的经验做为基准，然而对于软件开发厂商在软件项目原始码的部份，属于企业组织保护性资产，取得资料不易，本研究参考刘子扬[6]所提供台湾软件项目数据 22 笔数据作为本实验的范例数据，而数据源包含台北与高雄地区等三家软件厂商，以问卷访谈方式经由厂商自行填列，取得台湾软件项目数据后，再依照商业软件类别分类、汇整后，得到共 22 笔台湾软件项目数据。

本研究期许能以台湾软件项目数据，透过 COCOMO II 统计模式，找出影响软件开发成本因子指标，基于主成分分析方法减化复杂度，利用广义回归类神经网络预测模型，更加提升网络运算的速度与准确性，以建构出新的预测模式是适合用于台湾地区的软件现况。

9. 实例分析

以主成分分析方法将选用的 18 个成本因子指标先进行标准化处理，取得标准化后的数据相关系数的关系性，得以计算出各指标间相关系数矩阵的特征值、贡献率与累计贡献率，如表 3。

由表 3 可知，前 8 个主成分的累积贡献率已达 88.69%，因此，将这 8 个主成分 $(x_1, x_2, x_3, \cdots, x_8)$ 作为新的综合指标替代原始的 18 个成本因子指针，取得新的范例数据集，如表 4。

Table 3. Correlation coefficient matrix eigen value contribution rate and cumulative contribution rate
表 3. 相关系数矩阵特征值、贡献率与累计贡献率

PCA	Eigen value	Proportion	Cumulative
PCA1	5.383	29.91	29.91
PCA2	2.648	14.71	44.62
PCA3	2.245	12.47	57.09
PCA4	1.680	9.33	66.42
PCA5	1.071	5.95	72.37
PCA6	1.010	5.61	77.98
PCA7	0.972	5.40	83.38
PCA8	0.955	3.11	88.69
	(部份省略)		
PCA15	0.101	0.56	99.52
PCA16	0.057	0.32	99.84
PCA17	0.020	0.11	99.95
PCA18	0.009	0.05	100.00

Table 4. New composite indicator sample data
表 4. 新的综合指针范例数据

专案编号	Input Value				Out Value
	x_1	x_2	……	x_8	Log ACTMM
1	3.0033	3.3208		1.3956	1.9823
2	−4.9769	−0.0045		0.4153	1.1139
3	−5.2123	−0.5791		0.5452	1.0792
4	−2.4471	−0.2731		0.2137	1.1139
5	1.8144	1.9088	……(略)	0.8009	1.2553
6	3.3280	2.1738		0.8266	1.2553
7	3.4706	4.2245		−1.0335	1.5563
8	−2.7751	−0.3231		−0.7632	0.6021
9	4.7849	−2.9871		−1.1834	0.7782
10	4.4822	−2.1856		−0.1974	1.0792
			(部份省略)		
17	2.4957	−2.0333		0.7265	1.0000
18	1.5046	−3.1166		−0.4922	1.0792
19	4.3357	−3.0401	……(略)	0.0895	1.1461
20	−3.1494	−2.2821		−0.3751	1.0000
21	−5.6394	2.6631		0.5761	1.1461
22	1.9627	6.7597		−2.0013	2.0792

取得主成分数据后，即可将新的数据数据加入于广义回归类神经网络的预测模式中，藉以得到一种新的软件开发成本的预估方法，为了避免数据数据在进行网络训练初期即达到饱和的状态，本研究利用 Matlab 语法将变量数据进行处理，使变量数据的值域范围统一落在[−1,1]，Premnmx()函数是 Matlab 工具箱函数，其函数模型如公式 5：

$$\begin{aligned}&[\text{pn},\text{minp},\text{maxp},\text{tn},\text{mint},\text{maxt}]\\&= premnmx(p\_train, T\_train)\end{aligned} \tag{5}$$

pn 表示正规化后输入变量，minp 表示输入变量最小值，maxp 表示输入变量最大值，tn 表示正规化后输出变量，mint 表示输出数数最小值，maxt 表示输出变量最大值。

利用正规化进行数据处理后，即可建立广义回归类神经网络预测模式，将 22 个项目数据以 2:1 的比例随机选取，其中 15 组做为训练范例的学习，7 组做为训练结果的测试。在广义回归类神经网络预测模式中，唯一需要进行参数设定的即为平滑参数，平滑参数的范围一般都设定在 0~1 之间，在本实验中设定平滑参数值域范围[0.1,0.5]，每次训练以 0.1 做为递增数用以进行循环测试。

当广义回归类神经网络预测模式建构完成后，即可执行网络测试将 7 个测试范例循序进行测试，记录每个预测值与训练范例值间的均方差(Mean Square Error, MSE)，并将每次的 MSE 加总后计算出平均值，直到产生最小的平滑参数 σ 值，即为优化的平滑参数 σ 值。MSE 计算函式如公式 6。

$$\text{MSE} = \frac{\sum_{i=1}^{m}\left(\hat{x}_1 - x_i\right)^2}{n} \tag{6}$$

n 为样本值，$\hat{x}_1$ 为预测值，x_i 为实际值，MSE 数值愈小则代表所预测的精确度愈高。

取出优化平滑参数后，即可使用优化的网络训练参数建构广义回归类神经网络预测模式，以期许获得最好的预测结果，newgrnn()函数是 Matlab 工具箱函数，以下为函数使用方式：*net* = *newgrnn* (*desired_input*, *desired_output*, *desired_spread*)。

表 5 以不同预测方法与实际值进行模式比较，观察不同预测模式发现，使用平均相对误差(Mean Relative Errors, MRE)做为数据评估的基准，观察 PCA-GRNN 模式得到 MRE 为 0.107，COCOMO II 模式得到 MRE 为 0.656，倒传递网络类神经网络(Back-propagation Neural Network, BP)模式得到 MRE 为 0.185，而从实验结果中也证明了 PCA-GRNN 预测模式是比其它模式表现的更为良好。

图 2 将单位 LogPM 进行数据反处理后，得到不同预测模式与实际值比较之折线图，如图 3 所示。由图中可观察到在项目编号 12 与 20 的实际值为 28 与 10，使用 PCA-GRNN 模式预测值为 18.998 与 12.880，而使用倒传递网络类神经网络模式预测值为 22.761 与 10.351，是三种预测模式中表现最佳的，因此，将 PCA-GRNN 与倒传递网络类神经网络模式进行数据结果比较，显示 PCA-GRNN 预测能力仍有可成长的空间，而就整体网络预测能力仍是本研究可接受的结果。

10. 结论

本研究针对软件开发成本预估提出了一种新的预测模式，利用主成分分析与广义回归类神经网络的结合，应用于软件开发成本预测，经实验结果证明，此方法是可以提升了网络学习的速度，达到良好的预测效果，将本研究主要贡献归纳如下：

1) 提出了一种基于主成分分析与广义回归类神经网络于软件开发成本之研究。

Table 5. Different forecast model and the actual value
表 5. 不同预测模式与实际值比较

Project	ACTMM	MRE		
		PCA-GRNN	COCOMO II	BP
4	13	0.000	0.846	0.265
10	12	0.001	0.333	0.259
12	28	0.322	0.284	0.187
18	12	0.115	0.500	0.230
19	14	0.016	1.072	0.088
20	10	0.288	1.200	0.035
21	14	0.004	0.357	0.230
平均相对误差 MRE		0.107	0.656	0.185

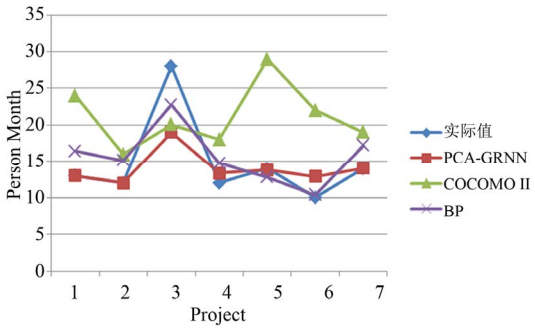

Figure 3. Comparison of different prediction model and the actual value of line chart
图 3. 不同预测模式与实际值比较之折线图

2) 结合主成分分析与广义回归类神经网络的优势，将影响成本因子指标的多量性原始指标转换为少数几个相互独立的综合性指标，并且能用以解释大部份的原始指标，改善了多量性指标的复杂性问题，提升了广义回归类神经网络的学习速度与效果。

3) 应用新的预测模式对软件开发成本预估进行研究分析，证明了此方法是适用于软件开发成本预估的。

4) 对台湾企业组织而言，大部份皆采用专家经验进行软件开发成本预估，并无一套可靠性及量化性的预估方法，藉由本研究所建构出的预测模式，可提供于实务应用之参考依据。

由于广义回归类神经网络特性，在函数逼近、与分类能力和学习效果的表现上有较佳的预测优势，相对的也代表在训练样本数据的选择上出现了较强烈的依赖性，因此预测误差若出现较大的悬殊，即有可能代表着来自于训练样本数据问题，因此，应该如何减缓预测误差值也是未来所要研究的方向。

11. 致谢

本研究承蒙国科会(计划编号：NSC100-2221-E-431-002)及佛光大学专题经费赞助，使本研究得以顺利完成，谨此志谢。

参考文献 (References)

[1] C. Jones, O. Bonsignour. The economics of software quality. Addison-Wesley Professional, 2011.
[2] 吴典璋. 以类神经网络预估软件开发成本之研究[D]. 国立中央大学信息管理研究所硕士论文, 1998.
[3] B. Boehm. Software engineering economics.englewood cliff. Upper Saddle River: Prentice-Hall, 1981.
[4] D. F. Specht. A general regression neural network. IEEE Transactions on Neural Networks, 1991, 2(6): 568-576.
[5] B. Boehm, E. Horowitz, R. Madachy, D. Reifer, B. Clark, B. Steece, W. Brown, S. Chulani and C. Abts. Software cost estimation with COCOMO II, englewood cliff. Upper Saddle River: Prentice-Hall, 2000.
[6] 刘子扬. 以类神经网络预估客制化软件开发成本之研究[D]. 中国文化大学信息管理研究所硕士论文, 2008.
[7] 许邦辉. 以主成分分析法为基础之文件自动分类模式[D]. 国立清华大学工业工程与工程管理学系研究所硕士论文, 2006.