2013 臺灣國際科學展覽會
優勝作品專輯

作品編號　**110018**

參展科別　**電腦科學科**

作品名稱　**AI 人工智慧-應用社群網站互動於類神經網絡訓練之研究**

得獎獎項　**四等獎**

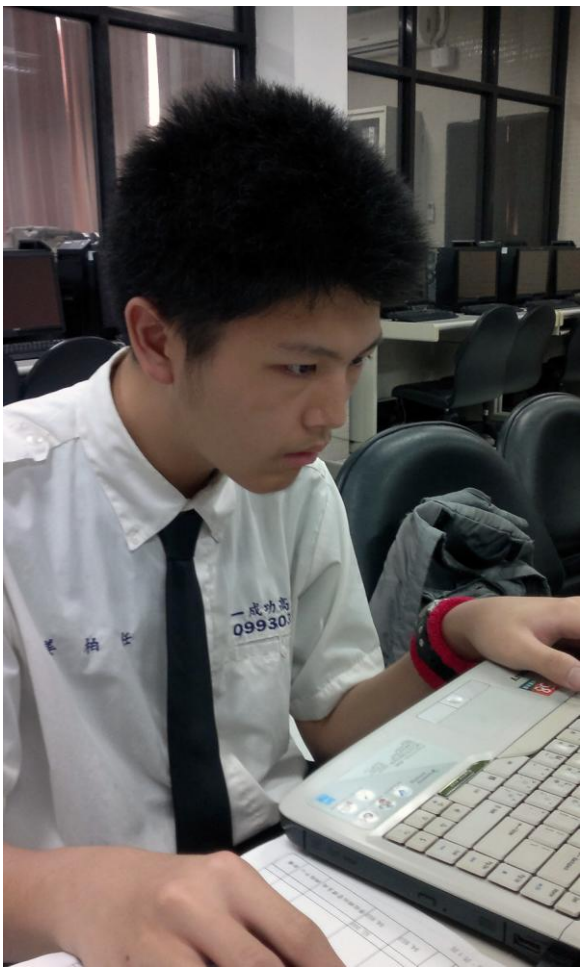
就讀學校　臺北市立成功高級中學

指導教師　杜玲均

作者姓名　姜柏任

關 鍵 字　人工智慧、社群網站、類神經網絡

作者簡介



誕於多雨的臺北，卻在風城新竹成長，在城鄉交界處播下己身對知識的信仰；國小二年級自學程式設計，在老師的引導下學習研究方法，自此長年以程式專案叩關科展。

熱愛語文及其所象徵的意義，同時追求英語文、公民、生物與資訊領域的卓越。希望能以自己跨領域的興趣與背景，讓科學與人文以不同樣貌交會，在研究中迸灑真理的光亮。

plurk.com/RSChiang

摘要

傳統的類神經網絡人工智慧多半是以受控訓練為主。然而在本次研究中，我們先建構出一套以類神經網絡模型為基礎的人工智慧，再利用社群網站噗浪（Plurk）上使用者與此系統的互動，訓練類神經網絡，以期驗證社群網站作為訓練來源的效率與準確度。我們利用分析詞、句的方式，促使系統做出自動的回應，同時並收集相關資料作為統計與修正之用。經過漸進式的調整與精進後，我們成功利用高度模組化的人工智慧系統，達成「利用社群網站資料自我修正」的目標，且其準確度呈現遞增的趨勢。我們相信只要充分掌握社群文化，社群網站做為資料來源對學術研究必有所裨益，且能為自然語言領域帶來更多可能性。

Abstract

Traditional ANN-based artificial intelligence is mostly based on supervised training. But in our research, we established an A.I. system on the basis of Artificial Neural Network, and instead made use of users' interaction with the system on social network website *Plurk* to train our neural network. In order to verify the efficiency and accuracy of using social websites as a training source, we designed our A.I. system to make auto replies to posts by analyzing words and sentences, collecting related data for statistics and correction. After series of progressive adjustments and refinements, we successfully accomplished our goal by deploying a highly-modular A.I. system, which is able to perform self-correction **relying only on social website data**, with accuracy measures increasing by time. We believe training A.N.N. using social websites is beneficial in the field of natural language processing, widening the possibilities of data collecting when performed in a community-cultural aware way.

壹、研究動機

大約從 2008 年開始，諸多「社群網站」挾帶著 Web 2.0 的傲人氣勢，一飛沖天登上各新聞媒體版面。僅在短短三年內，全球社群網站的使用者便超越八億[11]，而其中產生的龐大資料量更是令各國企業爭相投資。

在一次偶然的機會裡，我們討論到建立屬於自己的人工智慧的可能性。「要是能設計出如同電影《A.I.人工智慧》中能夠與我們對談的程式，應該會是件很有趣的事吧。」然而在搜尋資料後，我們發現既有的人工智慧設計大多需要大量人力協助訓練，才能在生活中實際應用。對於僅為學生的我們而言，是十分難以達到的門檻。

然而噗浪（Plurk）、臉書（Facebook）等社群網站在同儕間的風行，讓我們想到了嶄新的可能性。「若是能借助社群網站龐大的資料量以訓練我們的人工智慧系統，會不會有不一樣的表現呢？」充滿好奇心的我們，便在幾經規劃後，著手進行此次研究；希望能以最簡潔的模型，證明社群網站於人工智慧訓練上的價值。

貳、研究目的

（一）評估與建立利用社群網站作為訓練資料來源之類神經網絡人工智慧模型。

（二）分析與精進利用社群網站資料做來源的效益與正確性。

（三）彰顯社群網站於學術研究上的潛力，並能以研究資料貢獻語意分析等領域。

參、研究設備與器材

一、硬體設備

（一）伺服器（Intel i7™ 950 3.07GHz w/ 12GB RAM）

（二）筆記型電腦（AMD Turion™ 2.0GHz w/ 2GB DDR2）

（三）紙、筆、寬頻網路

二、軟體環境

（一）開發環境

1. Windows 7 Enterprise 32-bit SP1
2. Visual Studio 2010 Professional SP1
3. .NET Framework C♯ v4.0.30319.1
4. Python 2.7.2

（一） 執行環境

1. Windows Server 2008 Standard 32-bit SP2
2. Visual Studio 2010 Remote Debugger
3. .NET Framework C♯ v4.0.30319.1

肆、研究過程與方法

一、理論模型

我們希望證明社群網站可以做為人工智慧的訓練來源，但要如何做到這點呢？幾經思索後，我們發現社群的本質是***互動***，也就是應以「系統與一般人的相互反應」作為訓練的方式。我們於是設計了以下的情境流程：

（一）使用者於社群網站上發表一篇新動態（status update）

（二）人工智慧系統給予一篇評論（comment）

（三）使用者於動態底下討論與回應

（四）人工智慧系統收集彙整這些回應

（五）分析是否獲得正向意見，並調整人工智慧系統

而其中人工智慧系統的實作，我們則是參考了類神經網絡模型。別名人工神經網絡，類神經網絡全名 Artificial Neural Network，簡稱 A.N.N.，是一種以數學模擬生物神經網絡的模型架構；其中包含輸入節點、輸出節點、以及連繫兩者之間的隱藏層。在各節點連結之間設有不同程度的權重（weight），並可隨著網絡的運作更新權重、進而達到學習的效果。根據學習方式的不同可分為監督式學習（supervised learning）、無監督式學習（unsupervised learning）、以及強化式學習（reinforcement learning）等。

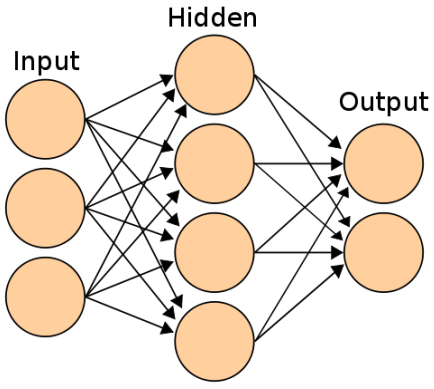


圖 4.1　類神經網絡示意圖

在此次研究中，我們採用簡化過的單層神經網絡模型，將欲回應的動態

解析成詞彙，並以詞為單位作為輸入節點，以撰寫好的成句作為輸出節點，其間以單一權重連結。由於一篇動態中常會有多個詞語，我們將各詞彙連結的節點彙整，以線性函數標準化後，乘上各詞彙於句中所佔的語意比重，經排序選取權重最高者作為最終的輸出結果。我們並沒有刻意設計隱藏層，而是以改變計算順序的方式模擬隱藏層的存在。雖然並非標準的神經網絡設計方式，但相對地也較容易分析。

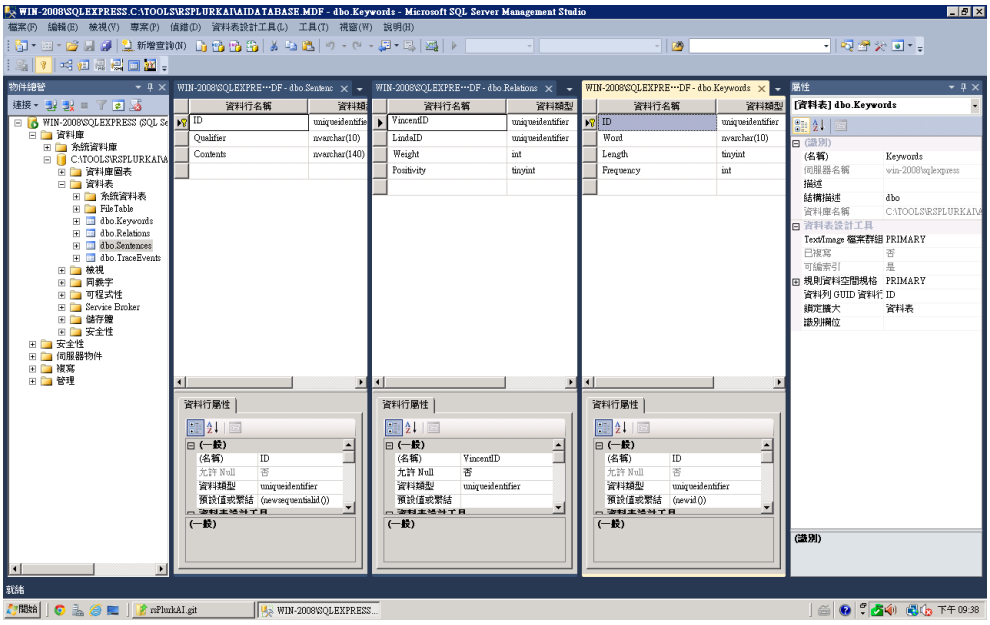


圖 4.2 建立研究所需的資料庫

而該如何從一篇社群網站的動態擷取出一個個的詞，供神經網絡解析呢？學術上有許多常用的斷詞方法，其中不泛牽涉到許多機器學習的範疇，甚至研究規模遠超出我們正在進行的研究。然而若是我們將範圍限縮在**已知**的詞彙，便能以簡單、有效、也是最廣為使用的長詞優先法（maximum matching algorithm）作為斷詞方式[3]。然而長詞優先法的準確率大幅取決於使用的辭典，因此我們也撰寫了概念程式，在不同辭典的組合間做出嘗試。

我們以 C♯ 為主要的程式語言，先建立一 List<KeyValuePair<string, int>> data，分別載入從中研院取得的詞彙庫、以及從開放原始碼的新酷音輸入法 1089 版取得的詞語庫，並先以字串長度、再以詞語庫所附之相對常用頻率作為排序，

如下：

```
data.Sort((KeyValuePair<string, int> x, KeyValuePair<string, int> y) =>
    {
        int len = x.Key.Length - y.Key.Length;
        if (len > 0) return -1; else if (len < 0) return 1; // Order by length desc
        int prior = x.Value - y.Value;
        if (prior > 0) return -1; else if (prior < 0) return 1; // Order by prior desc
        return StringComparer.CurrentCulture.Compare(x.Key, y.Key); // Order by form asc
    });
```

我們以此資料排序之順序進行取代測試。然而出乎意料之外，指導老師所取得、由中研院提供的詞彙庫解析效果並不甚理想，因此我們最終以新酷音輸入法之詞語庫作為斷詞依據。然而中研院的語彙庫僅三萬多詞，而新酷音之詞語庫原始數量多達五倍，共計 13,0173 個詞；我們於是除去六字以上的專有名詞（如「馬克希米林恩街」七字）、以及單詞罕用詞（如「穭」、「糮」、「鷙」等），精簡成 6,9745 字的辭庫。

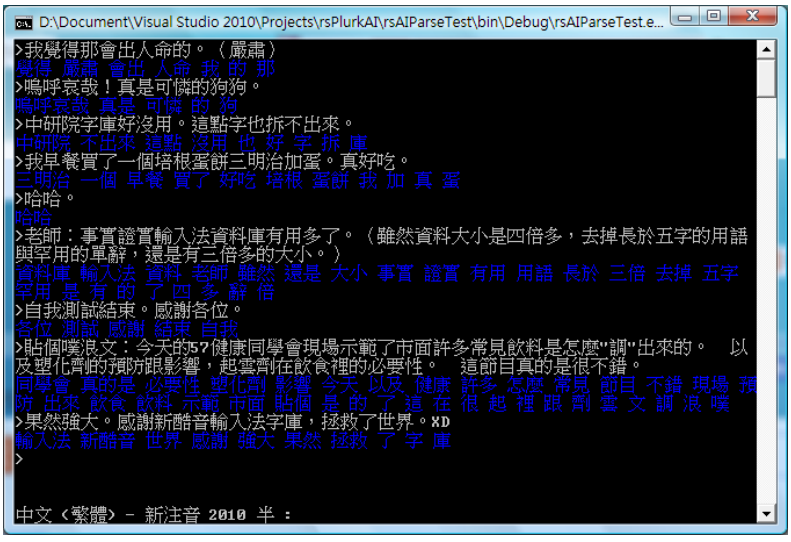


圖 4.3 研究初期實驗用的斷詞程式

現在系統得以自文句中解出詞語，也能透過網絡選取撰寫好的成句作為回應，接著必須面對的是分析使用者的反應（reaction），進而判斷該如何調整神經網絡。我們將使用者的反應歸類為正向（positive）或負向（negative），而其範疇如下：

（一）使用者直接向人工智慧系統對話，是部分正向的（＋）

（二）使用者直接對人工智慧表達正面意見，是正向的（＋）

（三）使用者回以代表「愉快」、「感到安慰」的表情符號，是正向的（＋）

（四）雖不直接與人工智慧對話，但底下造成熱烈討論，是部分正向的（＋）

（五）而負向範疇則可能有下列情況：

（六）使用者直接對人工智慧表達負面意見，是負向的（－）

（七）使用者回以代表「憤怒」、「傷心」的表情符號，是部分負向的（－）

（八）使用者回以「不理解」的表情符號，是負向的（－）

（九）該篇在人工智慧回應後無人理睬，是部分負向的（－）

而正負面意見的計算，則是統計各詞語與表情符號的正面傾向（positivity），取其方均根所得到的值。輸入法無法提供我們此資訊，因此我們利用課餘時間將建立好的辭庫匯出，並設計一 PHP 網站，讓班上同學、親友與師長們能在取得一組邀請碼（invitation code）後協助將詞語庫每一筆詞彙貼上「正向」、「負向」、「中性」的標籤。

圖 4.4 為了取得詞語正面傾向資料，設計出的協作標記網站

```
<?php
    require_once("conn.php");
    session_start();
    $code;
    if (isset($_POST["invitecode"])) $code = quoteit($_POST["invitecode"]);
    if (is_logged_in()) {
        $ad = new AIDatabase();
        $ad->Connect();
        $weight = ($_SESSION["prof"]) ? 3 : 1;
        if (isset($_GET["do"]))
            switch ($_GET["do"]) {
                case "positive": $uw = $weight; break;
                case "negative": $uw = -$weight; break;
                case "neutral": $uw = 0; break;
            }
        if (isset($uw) && isset($_SESSION["lastwdid"])) {
            $wdid = $_SESSION["lastwdid"]; $usr = $_SESSION["icid"];
            $ad->Query("INSERT INTO `ai_train_rels` (`word`, `modification`, `source`) " +
                        "VALUES ('$wdid', '$uw', '$usr')");
            $_SESSION["answ"]++;
        }
        $rand = mt_rand(0, 69745);
        $ad->Query("SELECT * FROM `ai_train_words` WHERE `id` >= $rand " +
                    "ORDER BY `id` ASC LIMIT 1");
        $wd = $ad->Rows();
        $_SESSION["lastwdid"] = $wd["id"];
?>
<!DOCTYPE html>
<html>
<head>
  <?php require("meta.php") ?>
  <link rel="stylesheet" href="train.css" />
  <title>類神經網絡訓練系統 - 訓練中</title>
</head>
<body>
  <div id="page" class="train">
    <div id="main">
      <p class="word" alt="<?php echo $wd["id"]; ?>"><?php echo $wd["word"]; ?></p>
      <div id="selector">
        <p>就客觀的判斷，這個字是正面、負面、抑或是中性詞？</p>
        <ul>
          <li class="positive"><a href="train.php?do=positive">正面</a></li>
          <li class="neutral"><a href="train.php?do=neutral">中性</a></li>
          <li class="negative"><a href="train.php?do=negative">負面</a></li>
        </ul>
      </div>
    </div>
    <div id="counter">已回答<span class="count">
                        <?php echo $_SESSION["answ"]; ?></span>題</div>
    <footer>
      <ul>
```

```
            <li><a href="about.htm">關於本研究</a></li>
            <li><a href="rank.php">排行榜</a></li>
            <li><a href="manage.php">維修通道</a></li>
            <li><a href="logout.php">登出</a></li>
          </ul>
        </footer>
      </div>
</body>
</html>
<?php
    } else {
        if (strlen($code) <= 0) header("Location: index.php");
        else {
            $ad = new AIDatabase();
            $ad->Connect();

            $ad->Query("SELECT * FROM `ai_train_invitations` WHERE `invitecode`='$code'");
            if ($ad->RowCount() <= 0) header("Location: index.php");
            else {
                $ic = $ad->Rows();
                $_SESSION["icid"] = $ic["id"];
                $_SESSION["prof"] = $ic["profession"];
                if ($ic["administration"]) $_SESSION["admin"] = true;

                $usr = $ic["id"];
                $ad->Query("SELECT COUNT(*) AS `count` FROM `ai_train_rels` WHERE `source` =
$usr");
                $tmpc = $ad->Rows();
                $_SESSION["answ"] = $tmpc["count"];
                header("Location: train.php");
            }
        }
    }
?>
```

對學校的國文、英文老師，我們另外發放了特殊的邀請碼加倍他們的權重，使師長們權威性的答案有機會彌平可能的誤差。我們在兩周的時間中獲得了 2,9962 筆標記，其中由學校各班自由登入協助者便占了 62.3%（1,8691 筆）。獲得這些資訊後，我們根據眾人的作答結果轉換成各詞語的正面傾向，進而能在人工智慧系統自我訓練時計算出需要修正的方向。

圖 4.5 設計及時排名頁面，使得協助的師長同學能夠有參與動力

二、架構設計

幾經複雜的準備，我們已經有使人工智慧能夠自我修正的運作模式。然而只有演算法並無法讓程式奇蹟似的執行起來，因此我們仍須建立一套可供研究使用的架構（Framework），使各部分的程式能夠偕同運作。經過資訊老師同意，我們使用學校裝有 Windows Server 2008 的伺服器，以 Windows 系統服務（service）的方式，使伺服器於復電後與其他網路服務並行執行。此伺服器也同時作為研究程式碼的版本控制庫（revision repository），而我們採用 Linux 之父 Linus Torvalds 所研發的 Git，作為本研究之原始碼控制系統（source code management system）。

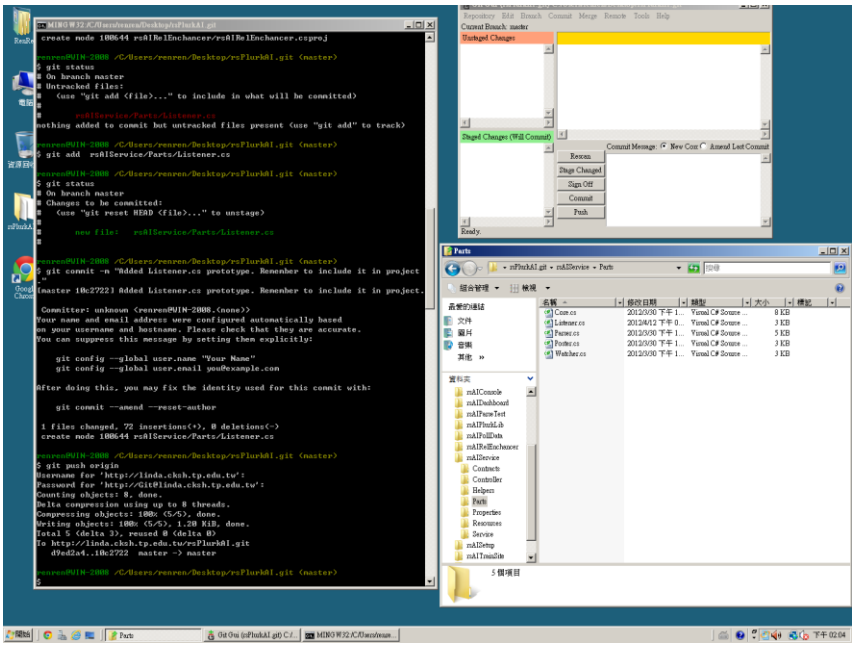


圖 4.6 利用 Git 於開發環境與伺服器間管理與同步程式碼

我們以 C# 為基礎程式語言，參考了設計模式（design pattern）中的 Actor model，將程式劃分為 8 個類別（class），分別運作於 9 條執行緒（thread）上。各類別間為鬆散耦合（loosely coupled），僅能透過共用的 Contracts.WorkItemBase 類別攜帶資料，於各執行緒的佇列（Queue）間傳遞工作項目。雖然撰寫起來相對繁瑣，但各類別間可以互相抽換，彼此也無需顧慮類別實作的演算法，並且能將發生的任何例外狀況（exception）隔離於單一執行緒上解決，不影響系統其他部分。根據類別間的互動，可繪製圖解如下：

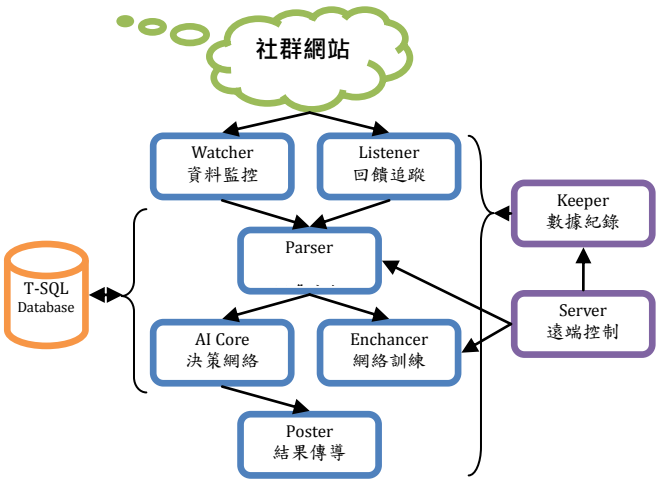


圖 4.7 人工智慧系統架構示意圖

左半部為主要的回應流程，由資料監控類別（Watcher）收集社群網站上任何新的動態，將其傳遞給語彙分析類別（Parser），再將分析出的詞彙訊息轉由核心類別（Core）進行網絡決策，最後由傳導類別（Poster）張貼回應到社群網站。此時右半部的回饋追蹤類別（Listener）會密切注意該篇動態的一舉一動，並交由語彙分析類別歸納使用者出現了「正向」或是「負向」的反應，最後由網絡強化類別（Enchancer）微調權重。

底下我們以至社群網站張貼回應的傳導類別（Poster）為例，展示我們的設計架構。一個類別上的執行緒會在註冊後獨立運作，藉由信號事件（signal event）與執行緒同步鎖（synchronization lock）等待與讀取佇列中由其他執行緒賦予的工作。工作因委職不同而異，此例中為繼承自 Contracts.WorkItemBase 的 Contracts.PosterWorkItem。

```
using System;
using System.Collections.Generic;
using System.Threading;
using RenRen.Plurk;

namespace RenRen.AI
{
  internal static class TimelinePoster
  {
      #region "Application Model"
      private static Thread postThread;
      public static void Start()
      {
          postThread = new Thread(PostProc);
          StatisticsKeeper.ConcurrentRegisterThread(Contracts.ThreadRole.Poster,
              new Func<ThreadState>(() => postThread.ThreadState));
          inputQueue = new Queue<Contracts.PosterWorkItem>();
          QueueNotEmptyEvent = new ManualResetEvent(false);
          postThread.Start();
      }

      public static void Stop()
      {
          postThread.Abort();
      }

      private static void PostProc()
      {
          while (true) {
```

```
                Contracts.PosterWorkItem item = DequeueTask();
                try {
                    PlurkHelper.AddResponse(Convert.ToInt64(item.Source),
                                                item.Content.Qualifier, item.Content.Text);
                }
                catch (Exception ex) {
                    StatisticsKeeper.WarnException(Contracts.ThreadRole.Poster, ex, null);
                }
                ItemPostedCount++;
            }
        }
        #endregion

        #region "Parse Queue"
        private static Queue<Contracts.PosterWorkItem> inputQueue;
        private static ManualResetEvent QueueNotEmptyEvent;
        public static void EnqueueTask(Contracts.PosterWorkItem item)
        {
            lock (inputQueue)
            {
                inputQueue.Enqueue(item);
                StatisticsKeeper.NotifyEnqueue(Contracts.ThreadRole.Poster,
                                    item.Source, inputQueue.Count);
                QueueNotEmptyEvent.Set();
            }
        }

        private static Contracts.PosterWorkItem DequeueTask()
        {
            QueueNotEmptyEvent.WaitOne();
            Contracts.PosterWorkItem item;
            lock (inputQueue)
            {
                item = inputQueue.Dequeue();
                if (inputQueue.Count <= 0) QueueNotEmptyEvent.Reset();
                StatisticsKeeper.NotifyDequeue(Contracts.ThreadRole.Poster,
                                    item.Source, inputQueue.Count);
            }
            return item;
        }
        #endregion

        #region "Statistics"
        /// <summary>
        /// Gets the count of posted (i.e. processed) item.
        /// </summary>
        public static long ItemPostedCount { get; private set; }
        #endregion
    }
}
```

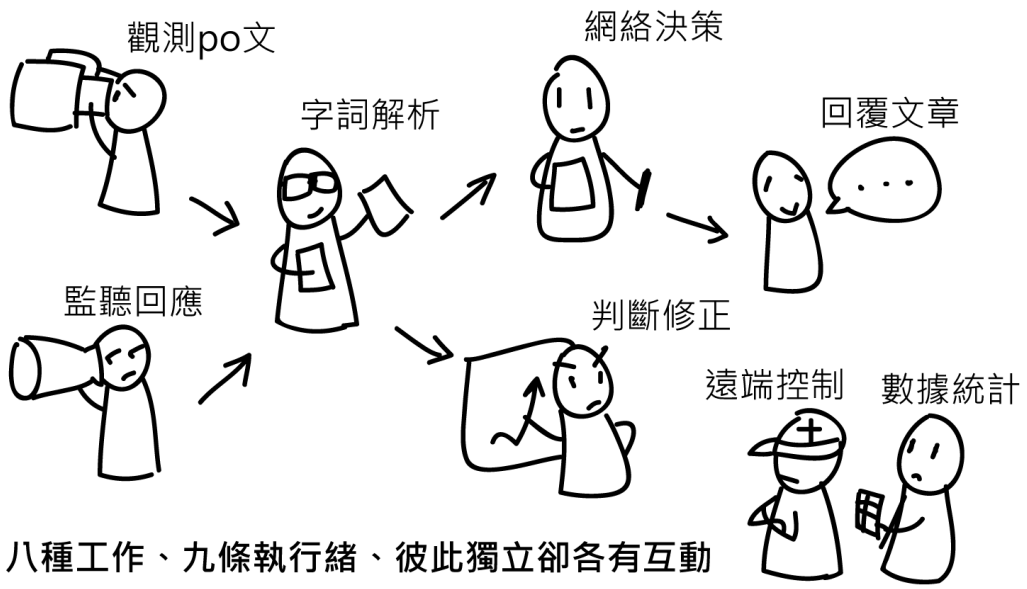


圖 4.8 人工智慧系統架構插畫圖

在人工智慧系統的六個類別外，尚存在兩個特殊的類別。數據統計（Keeper）由於同時負責記錄與備份統計資料至 SQL 資料庫，所以占用兩條執行緒。此類別並且未與其他類別鬆散耦合，是為了收集研究數據所設下的特例，但不影響其他類別之間的獨立性。遠端控制（Server）類別則讓我們得以自遠端命令列傳遞特殊指令，進而了解人工智慧系統的運作狀況，甚至檢視與匯入資料庫內容。當有執行緒因為不明原因發生錯誤時，遠端控制類別也可協助偵測與診斷問題，或是於無人控制時自動進行保護修復。

```
private static void ServerProc() {
    using (NamedPipeServerStream server =
            new NamedPipeServerStream("rsPlurkAI.Mgmt.v1", PipeDirection.InOut)) {
        while (true)
            try {
                server.WaitForConnection();
                StreamReader sr = new StreamReader(server, Encoding.UTF8);
                string buffer = sr.ReadLine();
                string response = OnServerCommand(buffer);
                StreamWriter sw = new StreamWriter(server, Encoding.UTF8);
                sw.AutoFlush = true;
                sw.WriteLine(response);
                server.Disconnect();
            }
            catch (IOException ex) {
                AIWindowsService.AIEventLog.WriteEntry(String.Format(
                    "An error occured while maintaining server IO pipe: {0}. " +
```

```
                        "Current failure {1}x.", ex.Message, ++fails));
                    if (fails >= 5) throw;
                }
        }
}
private static string OnServerCommand(string command) {
        if (String.IsNullOrEmpty(command)) return String.Empty;
        string[] args = command.Split(' ');
        switch (args[0].ToLower()) {
                case "ls": return CommandLoadSentence(ref args);
                case "lv": return CommandLoadVocabulary(ref args);
                case "parse": return CommandParse(ref args);
                case "listen": return CommandAppendListenerTask(ref args);
                case "train": return CommandManualTrain(ref args);
                case "cts": return CommandThreadStatus(ref args);
                case "flow": return CommandFlowControl(ref args);
                case "crs": return CommandRefreshStatistics(ref args);
                case "select": return CommandSelect(ref args);
                case "delete": return CommandDelete(ref args);
                case "config": return CommandConfig(ref args);
                case "version": return CommandVersion(ref args);
                default:
                        return "Unrecognized command.";
        }
}

#region "Commands"
private static string CommandParse(ref string[] args) .....
```

然而遠端控制類別的益處不僅止於偵錯便利而已。因為各執行緒獨立卻又皆與數據統計類別連結，我們得以於多處安插程式碼，將資料從類別內部調出並儲存進資料庫，於關鍵處掌握程式執行的動態。匯聚以上條件，我們甚至可以寫出用於監控人工智慧系統的專屬儀表板（Dashboard）程式！僅需藉由遠端控制類別定期查詢程式執行狀況，便能於前端即時呈現人工智慧系統各部分的互動狀態。

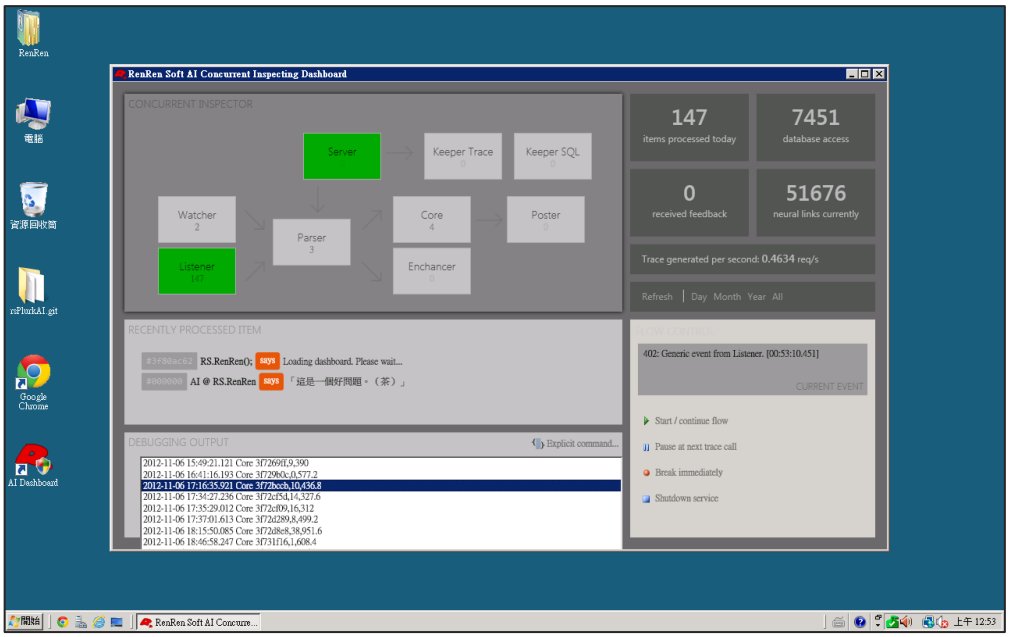


圖 4.9 能夠顯示執行緒間委派工作、回覆使用者等等情況的儀表板程式

圖 4.9 為「人工智慧並行檢驗儀表板」（A.i. Concurrent Inspecting Dashboard, ACID）的執行畫面。它能夠及時觀察各執行緒的狀態，並以不同的燈號：暗灰色—未啟動、銀色—等待中、綠色—執行中、紅色—錯誤，於中央區域分別指示系統各部分運作的情形。

藉由 WPF 中的項目樣板（item template）技術將資料具現化之便，儀表板程式還能實作更多適合研究的功能。右方區域能即時顯示今日回應過的動態數、目前類神經網絡訓練的狀態、資料庫存取的次數、以及每秒輸出的狀態資料數，甚至可以藉由下方四個按鈕控制全系統執行、暫停、切換偵錯或停止。左下方會顯示最新一次的回應，以及神經網絡決策的計時數據；如有需要，透過 [Explicit Command] 功能，可以輸入命令，額外自資料庫調用詳細數據。

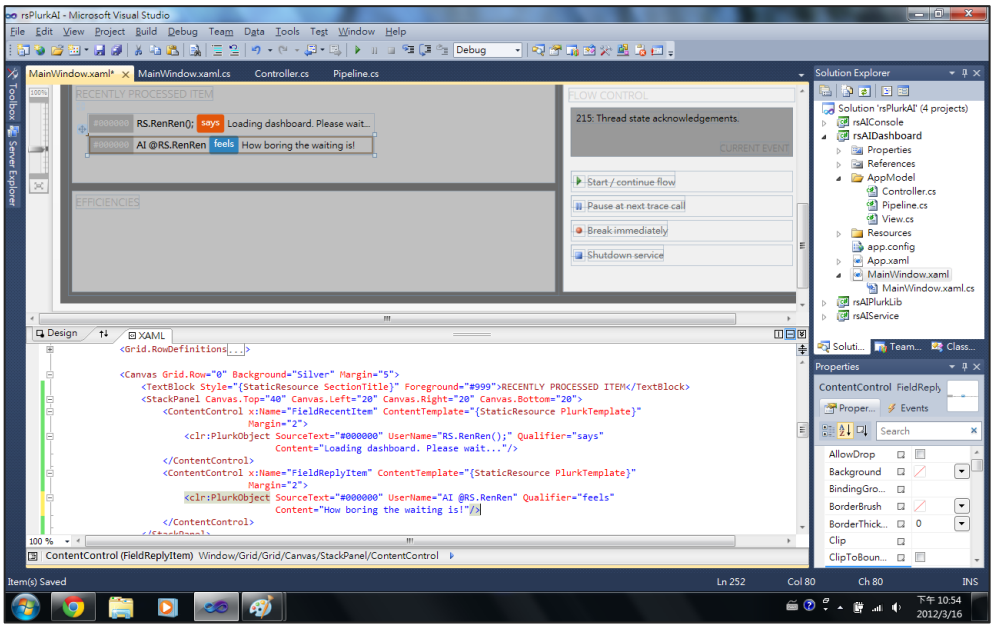


圖 4.10 修改內容控制項樣板，使系統資料自動具現化

至此，一套完善營運人工智慧及相關網路服務的架構儼然成形。我們將接著說明社群網站訓練來源的選擇。

三、社群網站

社群網站近年來有顯著性的成長，僅就台灣而言，較有知名度的社群網站便已有 Facebook（臉書）、Twitter（推特）、Plurk（噗浪）三者。然而 Facebook 雖然資料最為可觀，但程式開發介面（API）常常變動；而 Twitter 則多以英語系使用者為主，尚未在台灣造成風潮。考量集中研究精力與控制研究母體等因素，我們最終選擇了 Plurk 作為此次研究所採用的社群網站。

Plurk 相較前面兩者有以下優點：

（一）支持 Open Data，對於社群圖表（social graph）持開放態度，能夠在不侵犯使用者隱私的前提下調閱公開資料。

（二）對程式開發人員友善。我們於研究時發現不少 API 記載或實作的錯誤，Plurk 團隊都能於一至兩天內修復完畢並及時通知我們，甚

至信件往返討論。

（三）為台灣最常使用的的微網誌社群網站，占 Plurk 總流量約四成[9]。台灣以外地區的中文使用者（尤其是華僑）也十分活躍。

（四）有每日發文次數的限制，使用者發布動態前通常會稍作斟酌，單篇動態的質量通常都有不錯的表現。

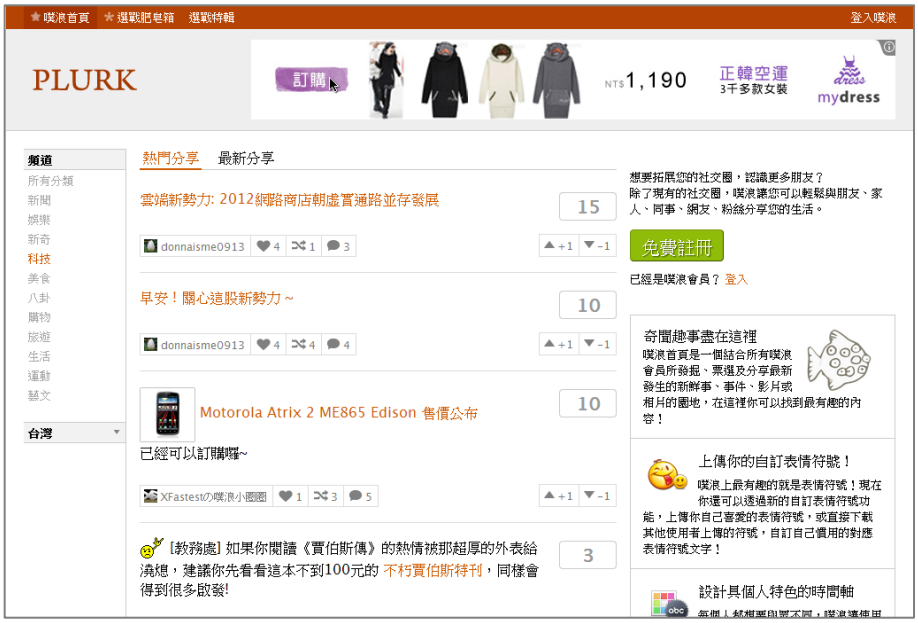


圖 4.11 Plurk 首頁

圖 4.12 時間軸可以追蹤不同使用者的訊息

Plurk 將網站上的每篇動態稱作一個「噗」(英文亦為 plurk),在每個噗前都置有發語詞(qualifier),如「說」、「想」、「喜歡」、「覺得」、「好奇」等。而其 API 設計採 OAuth 1.0a 協定、JSON 資料格式,需要較為繁瑣的編碼、簽章等程序,因此我們特地為此撰寫了可供 C♯ 程式引用的 Plurk library。其中 Entities 命名空間存放著 Plurk JSON 資料格式的對應類別,OAuthInstance 維護認證狀態與協助送出伺服器請求,PlurkHelper 則是由人工智慧系統使用、將背後複雜機制透過物件導向方式包裝起來的便利類別。

```
/// Provides functionalities of interacting with Plurk.
public static class PlurkHelper
{
    private static OAuthInstance Instance ...
    public static void AddPlurk(string qualifier, string message) ...
    public static void AddResponse(long plurk_id, string qualifier, string message)
    {
        NameValueCollection nvc = new NameValueCollection();
        nvc.Add("qualifier", qualifier);
        nvc.Add("content", message);
        nvc.Add("plurk_id", plurk_id.ToString());
        string req = Instance.SendRequest("Responses/responseAdd", nvc);
    }
    public static Entities.GetPlurkResponse GetPlurk(long plurk_id) ...
    public static Entities.GetResponseResponse GetResponses(long plurk_id) ...
    public static void MutePlurks(long[] plurk_ids) ...
    public static Entities.GetPlurksResponse GetUnreadPlurks() ...
    public static Entities.GetPlurksResponse GetPublicPlurks(int userId,
                                        DateTime offset, bool onlyResponded = false) ...
    public static IEnumerator<Entities.User> EnumerateFriends() ...
}
```

由於 OAuthInstance 為 OAuth 1.0a 的完整客戶端實作,包含特殊 RFC 逸脫編碼、字典排序、HMAC-SHA1 數位簽章、以及自訂 HTTP Authorization 標頭等,內容繁雜龐大;但也正因是系統最複雜之處,我們投入了可觀的時間撰寫、維護、除錯,謹將此類別的核心原始碼置於附錄,研究結束後也會開放全部原始碼,使他人皆能與吾等共享。

湊齊了理論、架構、資料三部分,我們接著介紹本研究目前累積的成果。

伍、研究結果

一、程式碼概要

人工智慧系統，包含分析與偵錯用的軟體，共分為 10 個專案。我們自 2012 年 2 月引入並建置完成版本控制，並於每次程式修改結束後，均以 Git 將程式碼上傳到伺服器統一管理，同時編譯並將伺服器上的系統更新至新版本。期間經過 93 次 pull commit，共增添過 1,4723 行、刪改過 1,851 行程式碼。目前最新版本編譯程序中的專案程式碼計有 7,455 行，其中不包含字詞庫與概念測試（proof-of-concept）用的暫時性程式。

二、社群網站概要

人工智慧系統發想於 2010 年暑假期間，2010/8/10 開始進行線上測試，並於當年 9 月中旬正式營運。我們自 Plurk 團隊取得了參與人工智慧系統測試的使用者、從 2008/6/3 開始，至 2012/10/13 為止，共計 5,3049 篇動態的統計資料，作為人工智慧系統上線前後、社群網站 Plurk 活躍度的參考。

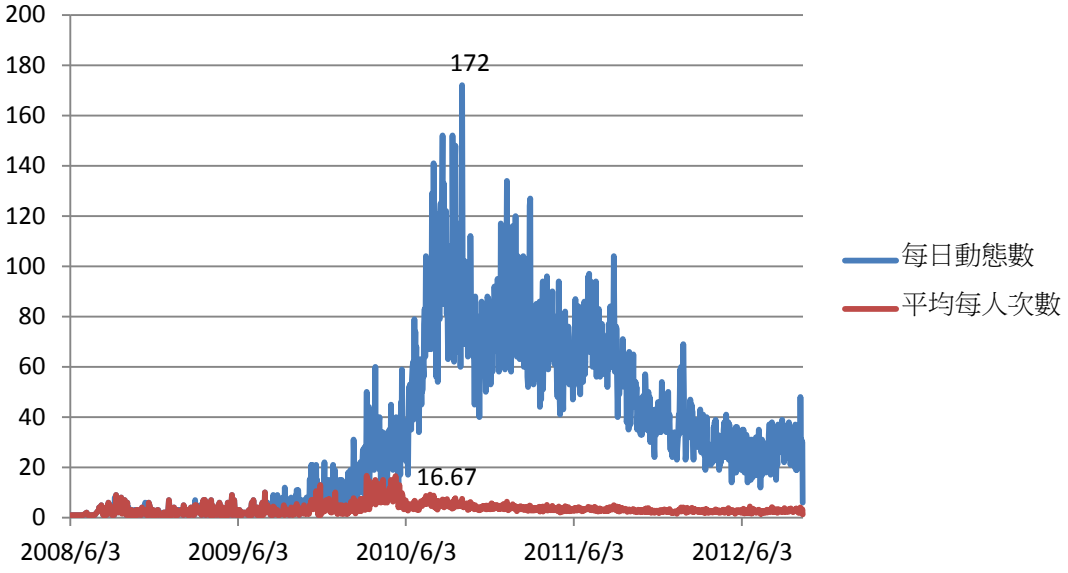


圖 5.1 參與人工智慧測試的使用者每日發布動態數量，折線圖

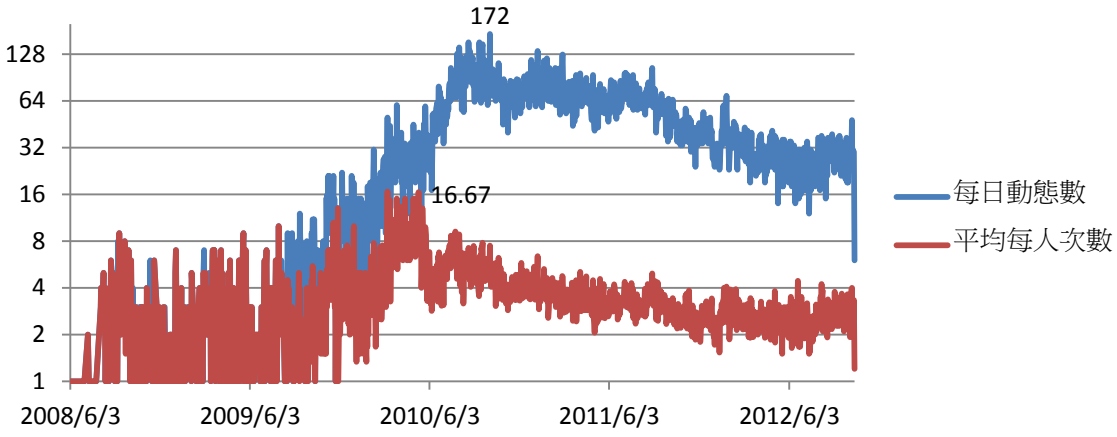


圖 5.2 參與人工智慧測試的使用者每日發布動態數量，對數刻度圖

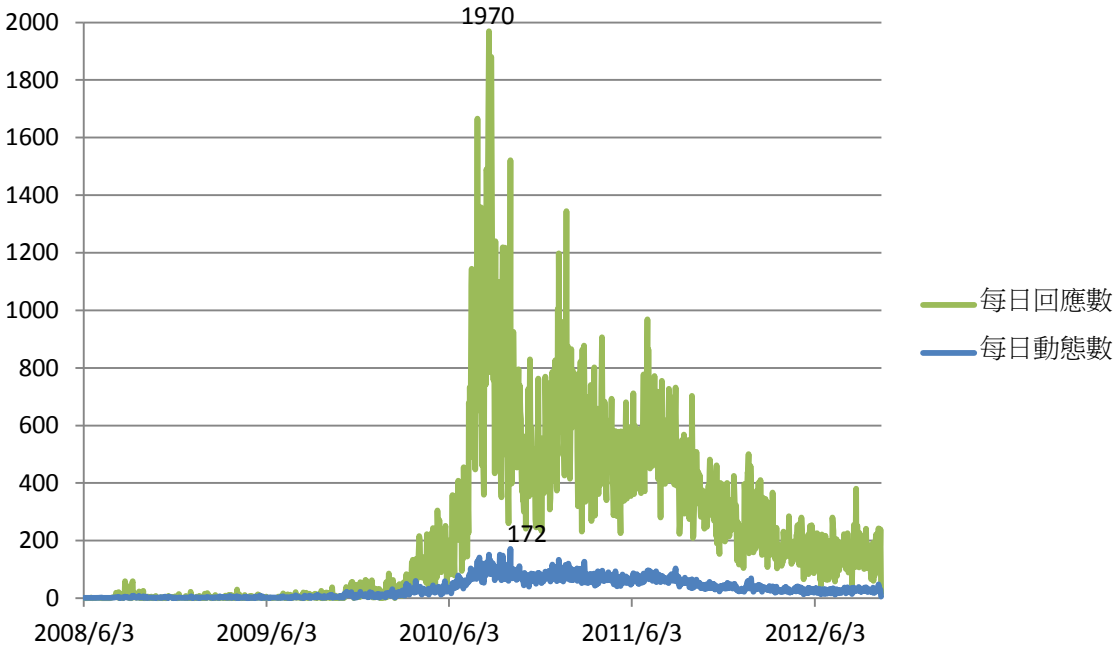


圖 5.3 每日發布動態數量與當日回應數，折線圖

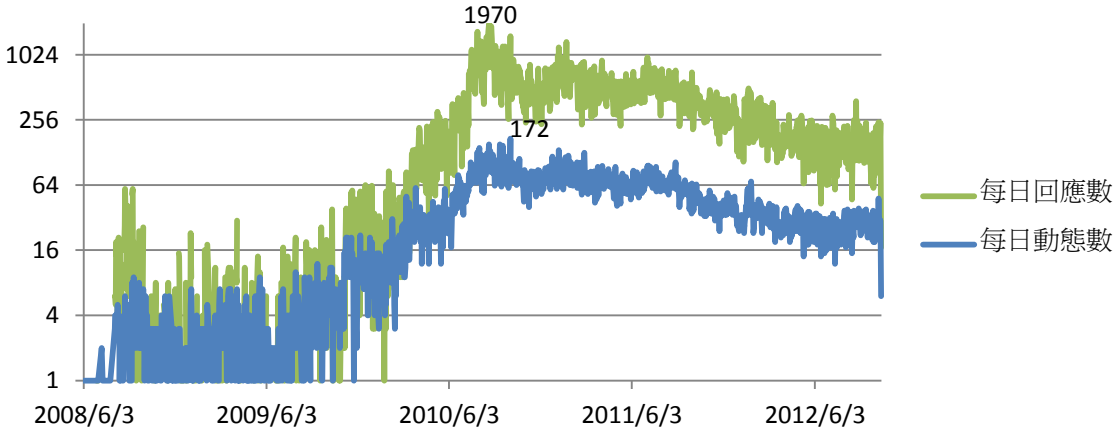


圖 5.4 每日發布動態數量與當日回應數，對數刻度圖

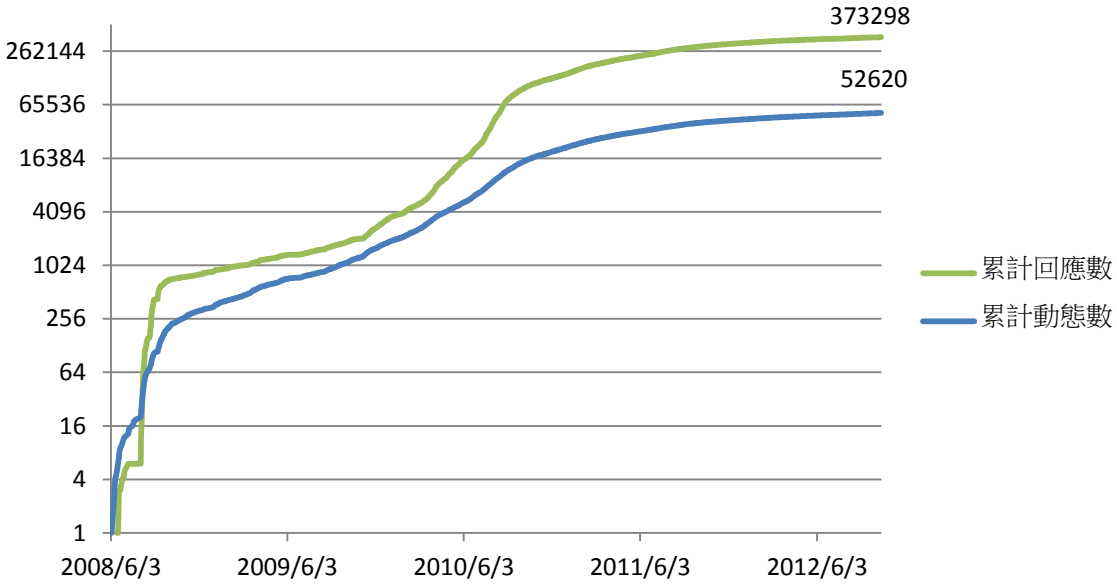


圖 5.5 每日發布動態數量與當日回應數，對數刻度累計次數圖

三、神經網絡概要

總計 6,9745 詞的辭庫、267 句的語句庫中，成功建立起網絡的詞彙計有 1,1323 個，共 5,0348 條網絡連結。其中詞對句的部分，以「的」(265 條)、「了」(246 條)、「我」(236 條) 三個詞建立的連結最多，平均每詞擁有 4.45 條連結，標準差 11.22。

而句對詞的部分，以語帶積極的 #119「想要追求更高、更遠的境界」(365 條)、出自水滸傳的 #164「相公也真是個癡漢」(359 條)、以及略顯神秘的 #120「不過我是不會說的。(笑)」(353 條)建立起的連結最多，平均每句擁有 189.61 條連結，標準差 69.69。

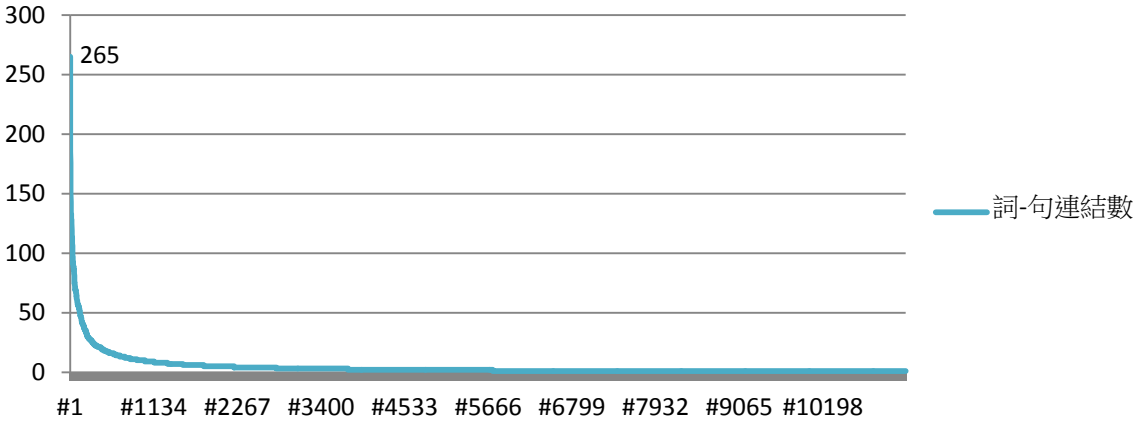


圖 5.6 詞對句的連結數，降冪排序圖

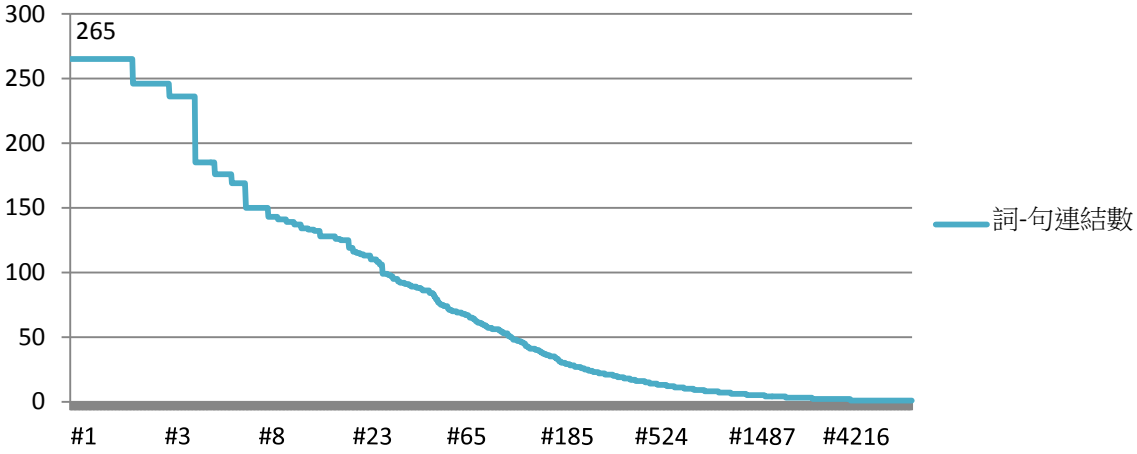


圖 5.7 詞對句的連結數，對數刻度降冪排序圖

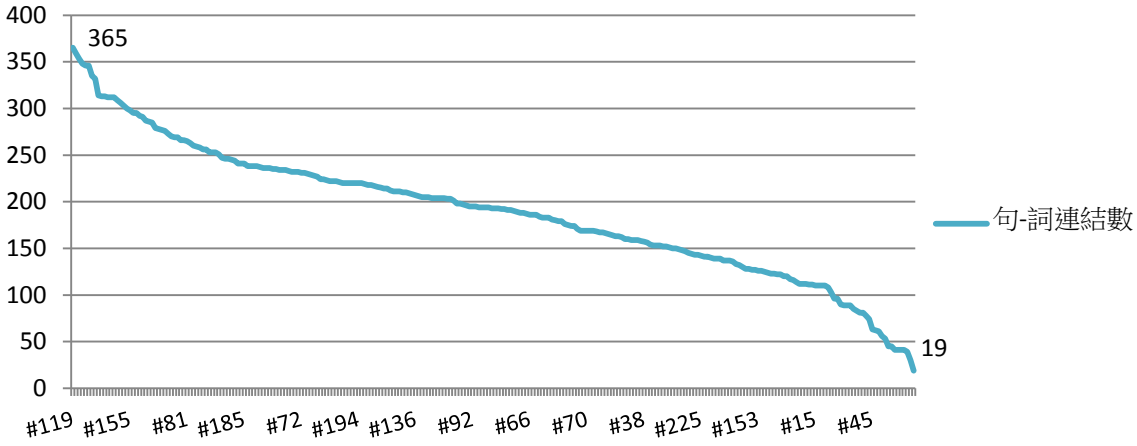


圖 5.8 句對詞的連結數，降冪排序圖

四、準確度測試

我們自前述 5,3049 篇動態中，隨機抽取人工智慧系統有回應的 400 篇動態，人工檢驗其準確度。我們定義一個有效的試驗為：

（一）人工智慧系統的回應切合該動態主題或情緒，或

（二）其他使用者針對該評論有正面迴響，引起多人討論。

依照此規則，我們抽樣估計人工智慧從上線至今，總體準確率約為 54.00%，且在 95%的信心水準下，誤差值為±4.98%。由圖可見人工智慧系統於上線後兩個月內便攀升至 46.8%高點，其後隨著參與計畫人數突增至 50 餘人而一度稍有起伏、隨後穩定回升。

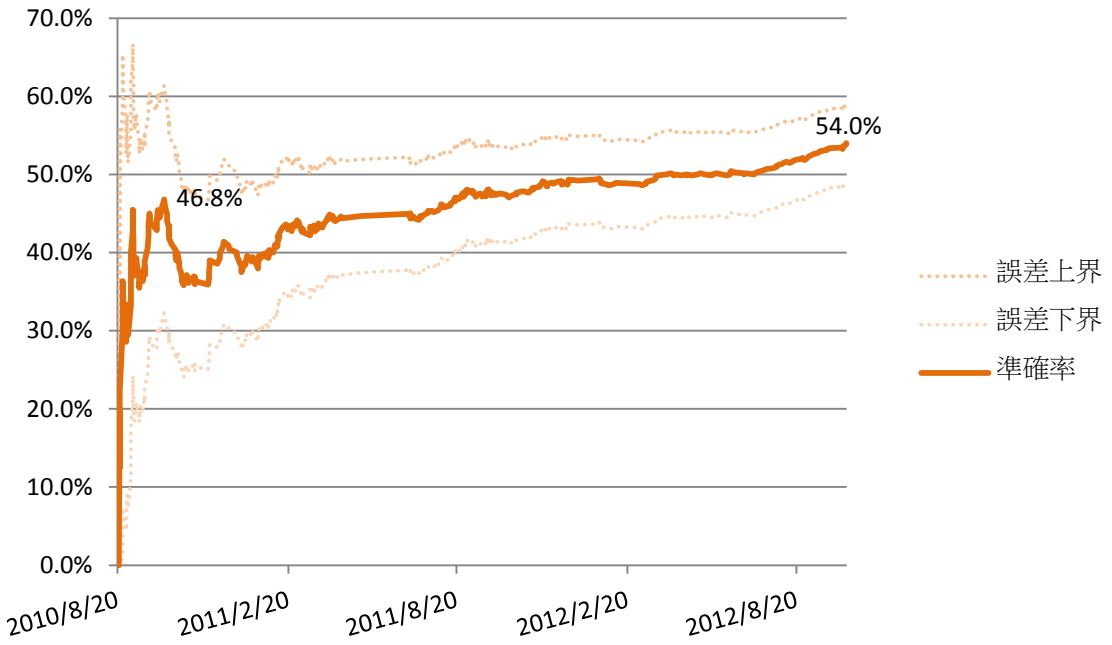


圖 5.9 隨機抽樣 400 篇動態測試準確度，累計百分比圖

我們可以看出人工智慧的準確率呈現穩定攀升趨勢，然而若將此抽樣依照日期先後分為 10 個區間，則能更明顯的看出在一般情況下人工智慧約略以每季 5%－10% 起伏成長。2012 年初我們鼓勵使用者多與人工智慧系統互動、並於計算修正係數時採列表情符號，更是使得抽樣中該區間（約 4 個月）的

準確率躍升至 85.0%。

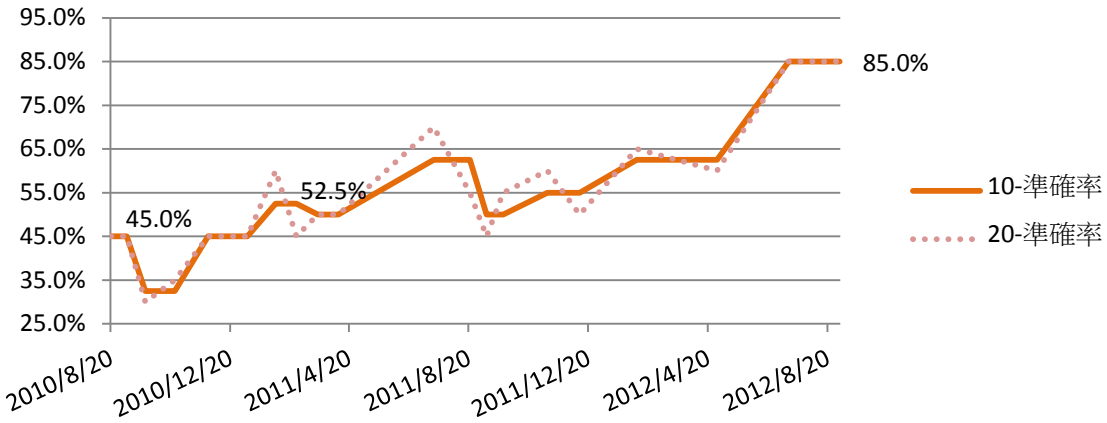


圖 5.10 隨機抽樣 400 篇動態測試準確度，分段折線圖

陸、討論

在研究結果中，我們再次確認了社群網站帶來的龐大資料量、以及其所應用於類神經網絡訓練的速度之快。然而在準確率的抽樣調查中，我們卻發現研究中期的準確率難以提升，遲至後期才有大幅度的躍進，這是為什麼呢？經由討論，我們認為是由於 2012 年 4 月翻修系統時一併加入「將表情符號納入正、負向意見計算」這個功能所致。我們當時發現，使用者對人工智慧系統習以為常後，便較少以中文與其對話，而多以簡單的表情符號帶過，使得系統中語彙分析類別無法做出準確判斷。在採計表情符號後準確率再次的躍升，提醒我們若要以社群網站進行長期訓練，需充分把握社群的生態與人際互動間任何的語意線索。

柒、結論

經過這次的研究後，對於利用社群網絡訓練與實作人工智慧動態回應系統，我們已經累積了大量程式碼基底（code base）可供使用。從研究軟體撰寫架構、自然語言分析、網路程式設計、使用者介面規劃、錯誤處理、作業系統核心服務開發……直到完成目前由多支旁系程式所共同支撐起的人工智慧系統，讓我們在研

究中獲得的許多樂趣和程式實作經驗，這是我所始料未及的。

雖然人工智慧系統平均並沒有石破天驚的表現，但透過社群網站的訓練、而能在短短兩個月內達到接近研究後期的準確率，讓我們相信若是能充分把握網路上人際互動的一切語意線索，其價值非常可能只是目前成果的冰山一角而已。社群網站能在短時間內累積可供分析的龐大數據資料，並能在輕鬆的談天間產生無數價值，對於學術上語意、字詞關係解析等研究，都將無疑是一份重要資產。

我們期待這次的研究成果，除了人工智慧系統的框架在開放原始碼後，能吸引更多有志於人工智慧研究的同儕之外；更能夠引起資訊科學領域對於使用社群網絡做為資料來源的興趣，共同探討如何從虛擬世界的閒話家常間找出語言與人類行為的蛛絲馬跡，進而拉近學術研究與日常生活的距離，為人類社會做出更多的貢獻。

捌、參考資料與附錄

[1] 亢世勇（民 88）。**計算機時代漢語語法研究的特點**。術語標準化與訊息技術，**4**。

[2] 巫沛倉（民 88）。**類神經網路簡介** [簡報]。民 100 年 12 月 9 日，取自：http://www.im.isu.edu.tw/faculty/pwu/expert/ann.ppt

[3] 林千翔、張嘉惠（民 99）。**基於特製隱藏式馬可夫模型之中文斷詞研究**。國立中央大學資訊工程學系論文。民 101 年 11 月 7 日，取自：http://aclweb.org/anthology-new/O/O06/O06-1019.pdf

[4] 哈雷爾（民 91）。**電腦也搞不定：計算機科學的罩門**。（李國偉譯）。台北市：天下文化。
（原著出版年：2000 年）

[5] 黃挺豪（民 98）。**應用於中文意見分析之詞內暨詞間語法結構自動擷取研究**。國立台灣大學資訊網路與多媒體研究所碩士論文，未出版，台北市。

[6] **人工智慧原理與意義**（無日期）。台北市：教育部。民 100 年 12 月 13 日，取

自：http://content.edu.tw/senior/computer/ks_ks/et/ai/chap1/index.htm

[7] **中文斷詞系統**（無日期）。台北市：中央研究院。民 100 年 12 月 13 日，取自：http://ckipsvr.iis.sinica.edu.tw/

[8] 人工智慧（無日期）。**維基百科**。民 100 年 12 月 9 日，取自：http://zh.wikipedia.org/wiki/%E4%BA%BA%E5%B7%A5%E6%99%BA%E8%83%BD

[9] 噗浪（無日期）。**維基百科**。民 101 年 11 月 9 日，取自：http://zh.wikipedia.org/wiki/Plurk

[10] Tony Northup, S. Wildermuth, B. Ryan (2006). .*NET Framework 2.0 Application Development Foundation*. Microsoft Press.

[11] Craig Kanalley (2011, January 9). *The Growth Of Social Media (INFOGRAPHIC)*. Retrieved November 8, 2012, from http://huffingtonpost.com/2011/09/01/growth-social-media-infographic_n_945256.html

rsPlurkLib 原始碼摘錄

研究所使用的 rsPlurkLib 原始碼摘錄如下。目前 Google、Facebook 等網站皆有實作 OAuth 協定，相信 rsPlurkLib 具有一定的平台可移植性與實用價值。完整的程式原始碼置於 GitHub 網站（https://github.com/rschiang/rsPlurkLib），讓對社群網站開發有興趣的社群成員能自由運用，能夠省去我們在閱讀規範文件、以及撰寫符合 OAuth 1.0a 通訊協定之函式庫時所花費的寶貴時間。

```
using System;
using System.Collections.Generic;
using System.Text;
using System.IO;
using System.Net;
using System.Security.Cryptography;
using System.Collections.Specialized;
using Newtonsoft.Json;
namespace RenRen.Plurk
{
    /// Provides OAuth authentication service. This class cannot be inherited.
    internal sealed class OAuthInstance
    {
        #region "Fields"
        private OAuthToken token = new OAuthToken();
        #endregion

        #region "Constant Fields"
        private static string cAppKey = "<protected>";
        private static string cAppSecret = "<protected>";
        private static string cReqTokenUrl = "http://www.plurk.com/OAuth/request_token";
        private static string cGrantUrl = "http://www.plurk.com/OAuth/authorize";
        private static string cExchangeUrl = "http://www.plurk.com/OAuth/access_token";
        private static string cApiBaseUrl = "http://www.plurk.com/APP/";
        private static DateTime cTimestampBase = new DateTime(1970, 1, 1, 0, 0, 0, DateTimeKind.Utc);
        #endregion

        #region "Methods"
        /// <summary>
        /// Sends a request to specified URL, using specified token and arguments.
        /// </summary>
        /// <param name="apiPath">Relative path of target API.</param>
        /// <param name="args">Additional arguments. Must not start with 'oauth'.</param>
        /// <exception cref="InvalidOperationException">Occurs when a valid request token
        /// hasn't been obtained.</exception>
        /// <exception cref="UnauthorizedAccessException">Occurs when the token has expired
        /// or the verifier specified is not valid.</exception>
        /// <exception cref="WebException">Connection problems occured.</exception>
```

```
public string SendRequest(string apiPath, NameValueCollection args)
{
	if (String.IsNullOrEmpty(token.Content))
		throw new InvalidOperationException("A valid token is required");
	var param = new NameValueCollection() { { "oauth_token", token.Content } };
	if (args != null)
		foreach (string key in args.Keys)
			if (!key.StartsWith("oauth_")) param.Add(key, args[key]);
	HttpWebRequest request = CreateRequest(cApiBaseUrl + apiPath, param);
	HttpWebResponse response = null;
	try
	{
		response = (HttpWebResponse)request.GetResponse();
		using (StreamReader sr = new StreamReader(response.GetResponseStream()))
			return sr.ReadToEnd();
	}
	catch (WebException ex) {
		WebExceptionHelper(ex); throw;
	}
}
#endregion

#region "Private Worker Functions"
/// Gets the OAuth standard encoding of a string.
private static string UrlEncode(string source)
{
	if (string.IsNullOrEmpty(source)) return null;
	byte[] chars = Encoding.UTF8.GetBytes(source);
	StringBuilder sb = new StringBuilder();
	for (int i = 0; i < chars.Length; i++)
	{
		if (chars[i] >= 'A' && chars[i] <= 'Z') sb.Append((char)chars[i]);
		else if (chars[i] >= 'a' && chars[i] <= 'z') sb.Append((char)chars[i]);
		else if (chars[i] >= '0' && chars[i] <= '9') sb.Append((char)chars[i]);
		else if (chars[i] == '_') sb.Append((char)chars[i]);
		else if (chars[i] == '-') sb.Append((char)chars[i]);
		else if (chars[i] == '.') sb.Append((char)chars[i]);
		else if (chars[i] == '~') sb.Append((char)chars[i]);
		else {
			sb.Append('%').Append(Convert.ToString(chars[i], 16).ToUpperInvariant());
		}
	}
	return sb.ToString();
}

/// <summary>
/// Creates a HttpWebRequest with specified OAuth parameters and URL.
/// </summary>
/// <param name="uri">The request target URI.</param>
/// <param name="param">The OAuth parameters to use.</param>
/// <returns>The created HttpWebRequest.</returns>
private HttpWebRequest CreateRequest(string uri, NameValueCollection param)
```

```
{
    HttpWebRequest request = (HttpWebRequest)WebRequest.Create(uri);
    request.Method = "POST";
    request.ContentType = "application/x-www-form-urlencoded";
    // Generate boilerplate parameters
    param.Add("oauth_consumer_key", cAppKey);
    param.Add("oauth_nonce", GetNonce()); // Must be unique
    param.Add("oauth_timestamp", GetTimestamp());
    param.Add("oauth_version", "1.0");
    param.Add("oauth_signature_method", "HMAC-SHA1");
    string sig = GetSignature(request.Method, uri, param); // Create signature
    param.Add("oauth_signature", sig);

    StringBuilder sb = new StringBuilder(); // Build up header
    sb.Append("OAuth realm=\"\"");
    foreach (string key in param.AllKeys)
        if (key.StartsWith("oauth_")) // Possible additional parameters
            sb.Append(", ").Append(key).Append("=\"")
              .Append(UrlEncode(param[key])).Append("\"");
    request.Headers.Add("Authorization", sb.ToString());

    // Build up POST body
    StreamWriter sw = new StreamWriter(request.GetRequestStream(), Encoding.ASCII);
    sb = new StringBuilder();
    string prefix = "";
    foreach (string key in param.AllKeys)
        if (!key.StartsWith("oauth_")) {
            sb.Append(prefix).Append(key).Append("=").Append(UrlEncode(param[key]));
            prefix = "&";
        }
    sw.Write(sb.ToString());
    sw.Flush(); sw.Close();
    return request;
}

/// <summary>
/// Generates a HMAC-SHA1 signature from the specified data.
/// </summary>
/// <param name="method">HTTP method.</param>
/// <param name="uri">Normalized service URI.</param>
/// <param name="param">Parameters used in the request.</param>
private string GetSignature(string method, string uri, NameValueCollection param)
{
    // Build up base string
    StringBuilder sb = new StringBuilder();
    sb.Append(method).Append('&').Append(UrlEncode(uri)).Append('&');
    // Sort the params
    List<string> keys = new List<string>(param.AllKeys);
    keys.Sort(StringComparer.InvariantCulture);
    string prefix = "";
    for (int i = 0; i < keys.Count; i++)
    {
```

```
                sb.Append(prefix).Append(keys[i]).Append("%3D")
                    .Append(UrlEncode(UrlEncode(param[keys[i]])));
                prefix = "%26";
            }
            string source = sb.ToString();
            string key = UrlEncode(cAppSecret) + "&" + UrlEncode(token.Secret);
            HMACSHA1 hashProvider = new HMACSHA1(Encoding.UTF8.GetBytes(key), true);
            byte[] result = hashProvider.ComputeHash(Encoding.UTF8.GetBytes(source));
            return Convert.ToBase64String(result);
        }

        /// Returns the number of seconds since January 1, 1970 00:00:00 GMT.
        private static string GetTimestamp()
        {
            return Math.Ceiling((DateTime.UtcNow - cTimestampBase).TotalSeconds).ToString();
        }

        /// Generates an unique string.
        private static string GetNonce()
        {
            // Must be 64bit, but Plurk didn't mention
            return new Random().Next(1, 99999999).ToString("d8");
        }

        /// Processes exception occured during authorization.
        private void WebExceptionHelper(WebException ex)
        {
            HttpWebResponse resp = ex.Response as HttpWebResponse;
            if (resp != null)
                if (resp.StatusCode == HttpStatusCode.Unauthorized)
                    throw new UnauthorizedAccessException(resp.StatusDescription, ex);
                else if (resp.StatusCode == HttpStatusCode.BadRequest)
                    try
                    {
                        string response;
                        using (var reader = new StreamReader(resp.GetResponseStream()))
                            response = reader.ReadToEnd();
                        Entities.ErrorResponse err =
                                JsonConvert.DeserializeObject<Entities.ErrorResponse>(response);
                        throw new PlurkException(
                            String.Format("Server rejected the request due to {0}.", err.error_text));
                    }
                    finally { resp.Close(); }
        }
        #endregion
    }

    /// Stores a OAuth token. This class cannot be inherited.
    [Serializable]
    public sealed class OAuthToken {
        public string Content { get; set; }
        public string Secret { get; set; }
```

```
        public OAuthTokenType Type { get; set; }
    }

    /// Represents the state of an OAuth token.
    public enum OAuthTokenType { Empty = 0, Temporary, Permanent }
}
```

評語

整體而言，此作品所用的方法，有很好的未來性，目前所需補強的是類神經網路的強度、層數，如此才能提升辨識的正確率。