第 5 章 自動摘要於文件分類上之應用

自動摘要可視為去除網頁雜訊（如版權宣告、廣告）的技術；也可視為一種特徵抽取的方式，將網頁重要的字句摘錄出來，用以訓練分類器來預測新網頁所屬的類別 [Shen *et al.* 2004] ，此外以摘要後的文件做索引，也可加速計算速度與減少儲存空間。

本章以自動摘要為基礎，提出主題混合模型分類器，用來做文件分類，並與 *K*-最近鄰（*K*-Nearest-Neighbor, *K*NN）做比較，並實驗在不同摘要比例下的結果。

自動文件分類，是根據文件內容或標題決定其類別的流程，其目的在於對文件進行分門別類的加值處理，讓文件易於管理、利用。例如，新聞文件可依其內容，給予「政治」、「經濟」、「社會」等類別資訊，方便使用者瀏覽取得其所需資訊。而類別的制定一般以使用者自定為主，傳統上的分類工作都藉由人工，然而在文件繁多、類別日益增大的情況下，此一工作變得日益艱難，是以經由文件分類就可節省大量的人力及提升分類的效率。

在進行文件分類時，需要瞭解文件的內文大意才能據此給予類別，這需要相當高階的知識處理，然而目前自然語言理解的技術，尚無法讓電腦瞭解任意的自由字句。 因此電腦在做文件分類時，常將文件分解成一個個語意較小的單位， 通常為文件的關鍵詞彙，再從這些詞彙與類別中找出對應的關係。有時分類的問題，簡單到只要文件的某個欄位中出現什麼特徵詞， 就分到什麼類別去。但在一般情況下，在進行文件分類前首先必須歸納出分類時的規則，如此電腦才能據以執行。而在類別龐大時，分類規則就難以用人工分析而得。所以，電腦在進行自動分類之前必須加以訓練，使其自動學習出人工分類的經驗與知識，此一流程即機器學習（Machine Learning）所欲達到的目標。

5.1 分類（Classification）與分群（Clustering）

文件分類（Document Classification）是將文件依據其內容指定為一個或多個事先

定義好的文件類別的過程。而文件分群（Document Clustering）為將相似的文件放置同一群中，且讓不相似的文件在另一群；然而分群無法給予所得到的類別（Class）一個綜合性的描述。文件分群與文件分類最主要的差異在於：文件分類是擷取文件特徵並與文件類別的特徵作比較，再依照其相關程度進行分類；而文件分群亦是擷取文件特徵並進行比對，但文件分群並不需要事先定義文件類別，而是依照各文件之間之相關程度進行分群。

本論文後續章節將繼續對文件分類的相關議題進行探討。

5.2 特徵抽取

在文件分類上常使用特徵抽取，以得到代表某一類別的詞彙，經由這些有鑑別力的詞彙我們可以加速運算、提高正確度及避免過度學習 [Joachims 1998]。

常見的方法有文件頻門閥值（Document Frequency Thresholding）[Yang *et al.* 1997] 、互斥資訊量（Mutual Information，MI）、條件式互斥資訊量（Conditional Mutual Information，CMI）[Wang *et al.* 2004]，敘述如下：

文件頻門閥值

計算每一詞在訓練語料庫中的文件頻率，並設定門閥值用以移除低於其值的字詞，其假設在於較少出現的字詞，對分類預測較無資訊，且不影響整體準確度；此方法也可移除那些干擾字詞。

互斥資訊量

計算每一詞 t，與一類別 c 之 MI 值

$$\mathrm{MI(t;c)}=\log\frac{p(t,c)}{p(t)p(c)}\approx\log\frac{A\times N}{(A+C)\times(A+B)} \tag{5.1}$$

其中 A 是 t 與 c 共同出現的次數，B 是 t 出現 c 不出現的次數，

C 是 c 出現 t 不出現的次數，N 是總文件數。

條件式互斥資訊量：$I(F_k;C\,|\,F_1,...,F_{k-1})$

其中 C 為某一類，F 為特徵值，經由 CMI 我們可以得到 JMI（Joint Mutual

Information）值為

$$I(F_1,...,F_k;C) = I(F_1,...,F_{k-1};C) + I(F_k;C \mid F_1,...,F_{k-1}) \quad (5.2)$$

其計算步步驟如下

開始	選取最大 MI 之項為特徵值
迭代	設已選取 *k*-1 特徵值，使得 JMI (Joint Mutual Information)值最大，選取使 CMI 最大的特徵值加入之，使 JMI 值最大

5.3 分類器（Classifier）

分類器，是一種文件歸納的處理過程，用以決定某個文件屬於某個分類[Sebastiani 2002]。對於某一類別 $c_i \in C$，給定某一文件 d_j，分類器計算對於每一類的類別狀態值（Categorization Status Value, CSV），不同類型的分類器，其 CSV 值域會有所不同。

$$CSV_i : D \rightarrow [0,1] \quad (5.3)$$

5.3.1 空間向量模型（Vector Space Model, VSM）

此模型將文件用一多維空間之向量來表示，藉由計算新進文件與分類規則向量，兩向量之間的相關程度函數，餘弦（Cosine），得到每一類別的 CSV 值，再經排序得到類別相關程度排名，最後可藉由設定門閥值判斷文件是否符合特定類型之文件，或者使用相關度最高的類別當作所屬類別。

5.3.2 單純貝式（Naïve Bayes, NB）模型

NB 模型假設索引單位之間相互獨立，在新進一篇新文件 $\vec{d}$ 時，估計給定特徵值下每個類別 C_i 的機率，以得到 CSV 值

$$p(C = c_i \mid \vec{d}) = \frac{p(\vec{d} \mid C = c_i)\, p(C = c_i)}{p(\vec{d})} \quad (5.4)$$

對於 $p(C = c_i)$ 簡化假設其為均勻分佈（Uniform distribution），也就是對於全

部的類別是相同的， $p(\vec{d})$ 不影響結果是以可以省略，此外在估計類別為 c_i 的情況下產生文件 $\vec{d}$ 的機率， $p(\vec{d} \mid C = c_i)$ ，假設各特徵值間是互相獨立的，是以可進一步簡化為：

$$p(\vec{d} \mid C = c_i) = \prod_{w \in \vec{d}} p(w \mid C = c_i) \tag{5. 5}$$

5.3.3 *K*-最近鄰（*K*-Nearest-Neighbor, *K*NN）

*K*NN 分類器在學習階段並不會像 Naive Bayes 分類器會產生或記錄每個類別的特徵，相對地只是簡單的將訓練文件中每筆資料以適當的表示法予以儲存，如此便完成其訓練工作。當有一筆測試資料集中的文件資料需要進行分類時，*K*NN 分類器會將欲進行分類的文件資料與所有訓練資料集中的文件資料逐一計算相關度，找出 *K* 筆最相近的訓練資料，再依據這 *K* 個訓練資料所屬的類別，來決定此測試資料最後所屬的類別 [V. Tam *et al.* 2002] 。*k*NN 法的過程可以下列公式表示

$$y(q, c_i) = \sum_{d_j \in kNN} sim(q, d_j) y(d_j, c_i) \tag{5. 6}$$

其中 $y(q, c_i)$ 代表類別 c_i 對新進文件 q 的 CSV 值、$y(d_j, c_i) \in \{\text{true}, false\}$ 用以表示文件 d_j 是否屬於類別 c_i ，而 $sim(q, d_j)$ 表示測試文件 q 與訓練文件 d_j 之問的相關程度（可利用餘弦或其它公式），$d_j \in KNN$ 代表與測試文件 q 最相關的 k 筆文件。

5.3.4 分類器比較

綜合上述，整理如下表所示

表 5. 1 分類器比較

分類器	優點	缺點
空間向量模型	容易計算、快速分類	準確度較不佳
單純貝式模型	計算容易、快速分類	簡化假設使準確度不足
*K*NN	訓練資料少時仍有不錯之分類準確度	訓練資料增加會造成算速度過慢

5.4 主題混合模型分類器

由 2.7 節關於主題混合模型的討論，由式(2.17)可得：

$$p(Q \mid D_i) \approx \prod_{n=1}^{N} \sum_{k=1}^{K} p(q_n \mid T_k) p(T_k \mid D_i)$$

由此模型，我們可得到機率值 $p(q_n \mid T_k)$ 與 $p(T_k \mid D_i)$。如果將每一潛藏主題視為一類別，並於分類時即時迭代更新，得到某一新進文件 N 的機率值，$p(T_k \mid N)$，代表類別的 CSV 值。最後由最大 $p(T_k \mid N)$ 值，所對應的類別代表所推薦的類別，即可完成分類流程，詳述如下：

訓練階段：

設定潛藏主題大小為類別的種類 K

給定一文件集，內含文件 D_i，並且每一文件已事先得知其所屬類別 T_k

(a) 初始化

$$p(T_k \mid D_i) = \begin{Bmatrix} 0.99999999 & \text{if Di} \in \text{T}_\text{k} \\ \dfrac{0.00000001}{\text{K-1}} & \text{if D}_\text{i} \notin T_k \quad \text{(close to 0)} \end{Bmatrix}$$

在類別內的文件，其文件產生類別的機率 $p(T_k \mid D_i)$ 為接近 1 的值，否則設為很小的值

$p(q_n \mid T_k)$ ，由主題單連語言模型而來

(b) 迭代

使用非監督式訓練式(2.23)、(2.24)，迭代更新 $p(q_n \mid T_k)$ 與 $p(T_k \mid D_i)$

測試階段：

新進一篇新文件 N

(a) 初始化

預設新文件 N 產生各類別的機率是均勻分佈，$p(T_k \mid N) = \dfrac{1}{\text{K}}$

(b) 迭代

$$\hat{P}(T_k \mid N) = \frac{\sum_{q_s \in N} n(q_s, N)\, p(T_k \mid q_s, N)}{|N|} \tag{5. 7}$$

$$p(T_k \mid q_s, N) = \frac{p(T_k \mid N)\, p(q_s \mid T_k)}{\sum_{l=1}^{K} p(T_l \mid N)\, p(q_s \mid T_l)} \tag{5. 8}$$

其中 $p(q_s \mid T_k)$ 由訓練階段所得的主題單連語言模型而來

(c)決策

由 $p(T_k \mid N)$ 的大小，找最大的值所對應的類別，視為新進文件 N 的類別

5.5 實驗設定

本實驗使用東森新聞做語料庫 [東森新聞報]，相關統計如表 5.2 所示：

表 5. 2 東森新聞語料相關統計

	發展集	測試集
類別	共十類 {政治、財經、社會、地方、兩岸、國際、生活、綜藝、資訊、運動}	
新聞時間	2003 年 1 月	2003 年 2 月
新聞數	5533 則	4632 則

在文獻中可發現 *K*NN 為目前在分類結果上較佳的分類器之一 [Yang *et al.* 1999]，是以在基礎實驗上使用 *K*NN 與所提出主題混合模型分類器做一比較。在 *K*NN 分類器中，使用向間向量模型來表達每一文件，並使用餘弦估測相關度。在實驗時，將發展集由向量空間模型模型做摘要，再利用固定測試集做測試，以觀察自動摘要對於分類器正確率的提升是否有所助益。

在實驗評估方面，使用 MicroF 以及 MacroF 值同時呈現分類的效果，其計算方式如下：

$$\text{MicroF}=\frac{2\times\sum_{i=1}^{C}TP_i}{2\times\sum_{i=1}^{C}TP_i+\sum_{i=1}^{C}FP_i+\sum_{i=1}^{C}FN_i} \quad (5.9)$$

$$\text{MacroF} = \frac{1}{C}\sum_{i=1}^{C}\frac{2\times TP_i}{2\times TP_i+FP_i+FN_i} \quad (5.10)$$

其中 C 是類別總數，i 代表某一類別，而 TP_i（True Positive）、FP_i（False Positive）、FN_i（False Negative），分別代表 ：是類別 i 而且也正確分為類別 i 的文件數、不是 i 類卻分為 i 類的文件數、是 i 類卻沒有分為 i 類的文件數。

由於 MicroF 是全部文件一起累加統計，不分類別，因此容易受到大類別（佔大多數文件）表現好壞的影響。相對的，MacroF 考慮每個類別的成效後再做平均，因此容易受到大量的小類別影響。將兩種平均數據都報告出來，可以瞭解大多數文件的分類效果（MicroF），以及大多數類別的分類效果（MacroF）。

5.6 實驗結果

在表 5. 3~表 5. 4 的 1~25、26~50 等，代表選擇 K 個文件的出處，如 26~50 代表由相關度排名介於 26~50 名的文件。

由表 5. 3~表 5. 4 實驗結果顯示：

1. 當 K 值選取愈大時，MicroF 與 MacroF 均有變好的趨勢，但過大時因雜訊的增多，其結果慢慢變差。
2. 經由自動摘要過後，其 MicroF 與 MacroF 值大都降低，這與預期結果相反，這可能是因為在做摘要的過程當中，因只保留對文件重要的資訊，而沒有考量與類別的關係，也就是說，有可能將對類別有高鑑別性的字句給去除，是以造成結果的下降。

表 5. 3 *K*NN 於測試集 MicroF 值

K	25				50		100
摘要比例	1~25	26~50	51~75	76~100	1~50	51~100	1~100
0.2	0.5782	0.5365	0.5214	0.4989	0.5868	0.5402	0.5827
0.3	0.5786	0.5412	0.5276	0.5060	0.5885	0.5443	0.5872
0.5	0.5883	0.5501	0.5348	0.5140	0.6013	0.5538	0.595
0.7	0.5874	0.5570	0.5432	0.5179	0.5984	0.5548	0.5978
1.0	0.5954	0.5745	0.5475	0.5352	0.6056	0.5715	0.6066

表 5. 4 *K*NN 於測試集 MacroF 值

K	25				50		100
摘要比例	1~25	26~50	51~75	76~100	1~50	51~100	1~100
0.2	0.5692	0.5251	0.5038	0.4817	0.5771	0.5206	0.5699
0.3	0.5712	0.5306	0.5113	0.4840	0.5788	0.5237	0.5732
0.5	0.5827	0.5378	0.5197	0.4942	0.5921	0.5365	0.582
0.7	0.5820	0.5452	0.5278	0.4988	0.5898	0.5360	0.5855
1.0	0.5891	0.5644	0.535	0.5182	0.5967	0.5563	0.595

由表 5. 5~表 5. 6 實驗結果顯示

1. TMM 於摘要比例 1.0（即不做自動摘要）迭代 1 次時，比 *K*NN 略差。但經由多次迭代後，不論在 MicroF 與 MacroF 均較 *K*NN 分類器來得好，且迭代的次數愈多，其結果愈見明顯。
2. TMM 在摘要後，其結果不論在 MicroF 與 MacroF 大致均較 *K*NN 分類器來得好，且迭代的次數愈多，其結果愈見明顯。
3. 經由自動摘要過後，*K*NN 與 TMM 分類器，其 MicroF 與 MacroF 值大都降低。

表 5. 5 TMM 與 *K*NN 分類器，於測試集 MicroF 值比較

摘要比例	*K*NN	TMM 1 次迭代	TMM 50 次迭代	TMM 100 次迭代
0.2	0.5868	0.5939	0.5954	0.5961
0.3	0.5885	0.5939	0.5941	0.5961
0.5	0.6013	0.6015	0.6051	0.6066
0.7	0.5984	0.5987	0.6058	0.6060
1.0	0.6056	0.6015	0.6077	0.6101

表 5. 6 TMM 與 *K*NN 分類器，於測試集 MacroF 值比較

摘要比例	*K*NN	TMM 1 次迭代	TMM 50 次迭代	TMM 100 次迭代
0.2	0.5771	0.5834	0.5871	0.5886
0.3	0.5788	0.5843	0.5864	0.5891
0.5	0.5921	0.5909	0.5962	0.5985
0.7	0.5898	0.5885	0.5978	0.5983
1.0	0.5967	0.5920	0.6006	0.6033

5.7 本章小結

文件自動摘要的目的，是將文件縮減濃縮成重要字句，並去除冗餘的訊息。此外，摘要後的文件，因資料量較少，也可提升後續文件處理的效率。基於這樣的觀察，自動摘要的技術，可能有助於文件的自動分類。然而經由實驗，結果並未如預期，自動摘要雖然提升了自動分類文件的效率，卻因損失一些分類資訊，使分類文件的精確度降低。

初步實驗結果顯示，主題混合模型分類器較常見 *K*-最近鄰（*K*-Nearest-Neighbor, *K*NN）分類器在 MicroF 與 MacroF 分類結果上，有些微的提升。