國立交通大學

資訊科學與工程研究所

碩 士 論 文

以序列標記方法解決古漢語斷句問題

Classical Chinese Sentence Division
by Sequence Labeling Approaches

研 究 生：黃瀚萱

指導教授：孫春在　教授

中 華 民 國　九 十 七　年 六 月

以序列標記方法解決古漢語斷句問題

Classical Chinese Sentence Division by Sequence Labeling Approaches

研 究 生：黃瀚萱　　Student：Hen-Hsen Huang

指導教授：孫春在　　Advisor：Chuen-Tsai Sun

國 立 交 通 大 學

資 訊 科 學 與 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Institute of Computer Science and Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

June 2008

Hsinchu, Taiwan, Republic of China

中華民國九十七年六月

以序列標記方法解決古漢語斷句問題

學生：黃瀚萱　　指導教授：孫春在

國立交通大學資訊科學與工程研究所碩士班

摘　　要

斷句是古漢語處理的特殊議題。在 20 世紀之前，中文的書寫系統，並沒有使用標點符號的習慣。在閱讀古籍的時候，讀者必須從文句中，辨別應該停頓或分隔的地方，而後才能理解文義。由於斷句並沒有明確的規則和方法，全憑讀者的語感和經驗來判斷，同一個句子，不同的讀者，往往會有不同的斷法，而不同的斷法，造成了不同的文義解讀。所以，在處理古籍的時候，斷句是重要而困難的第一步驟。

過去沒有理想的自動化斷句方法，斷句的工作，多半交由文史專家，以人力來處理。雖然常見的經史典籍，目前已有斷句標點過的版本，但隨著歷史文獻不斷地發掘出土，仍然有無數的古代文獻，尚待斷句處理。

在本研究中，我以 hidden Markov models（HMMs）和 conditional random fields（CRFs）等兩種序列標記模型，設計古漢文斷句系統，並在實驗中獲得不錯的斷句結果。同時，在實驗中也發現，只要 training data 的質量足夠，則具有跨文本、跨作者、跨體裁的適用性。例如，以《史記》作 training data，對於其他上古漢語的文本，都有頗佳的斷句表現。本研究的成果，展現了自動化古漢語斷句的可行性，並得以實用在數位典藏、文字探勘、資訊擷取等工作上，輔助人力，更快速地處理大量歷史文獻。

Classical Chinese Sentence Division by Sequence Labeling Approaches

Student: Hen-Hsen Huang Advisor: Chuen-Tsai Sun

Institute of Computer Science and Engineering

National Chiao Tung University

ABSTRACT

Sentence segmentation is a special issue in Classical Chinese language processing. To facilitate reading and processing of the raw Classical Chinese data, I proposed a statistical method to split unstructured Classical Chinese text into smaller pieces such as sentences and clauses. To build this segmenter, I transformed the sentence segmenting task to a character labeling task, and utilized two sequence labeling models, hidden Markov models (HMMs) and conditional random fields (CRFs), to perform the labeling work. My methods are evaluated on nine datasets from several eras (from the 5th century BCE to the 19th century). My CRF segmenter achieves an acceptable performance and can be applied on a variety of data from different eras.

誌謝

感謝我的指導老師，孫春在老師。從孫老師身上散發出來的活力與熱情，讓我深深感受到研究的奧妙。在我就讀碩士班的兩年之間，老師安排了一系列的訓練，透過實際的練習和摸索，讓我逐步了解作研究的門道和興味。老師非常注重題目的創新和價值，在實驗室的聚會中，老師不厭其煩地闡釋什麼樣的題目，才是好的題目。更重要的是，老師引領我去思考，好的題目爲什麼好，題目的背後，潛藏了哪些重要的價值和意義。經過這一連串的啓發，我對學術的本質有更深入的認識，也激起了繼續探索、繼續研究的動力。

感謝國立清華大學中國文學系劉承慧老師和蔡柏盈老師。在決定研究古漢語斷句之時，我對語言學，特別是古漢語的語言學一無所知，而劉老師在這個時候給我重要的指引。她提到斷句問題在古漢語語言學上的難處，介紹我找《馬氏文通》一書作爲古漢語言語學的入門，並推薦我由《左傳》、《孟子》、《莊子》作爲研究的起點。而蔡老師在研究的關節處，提出許多寶貴的意見，並出借相關書籍于我參考。

感謝中央研究院張復老師。在偶然的機會裡，我與張老師有一次短暫的會面。張老師在唔談當中，向我介紹最新的機器學習技術，讓我對這個完全陌生的領域，有了下手的方向。於是，我才知道 hidden Markov models 和 conditional randon fields 等序列標記模型，並順利地援用到我的題目上。

感謝國立台灣大學資訊工程系項潔老師。項老師大方提供了一批珍貴的清代語料，讓我的實驗更爲豐富完整。在口試和私下的會談間，項老師對古漢語斷句這個目題本身，提出很好的見解，讓我更深刻地了解這個研究的價值和貢獻。同時，也感謝另外兩位口試委員，國立清華大學資訊工程系張智星老師與國立交通大學資訊工程系梁婷老師，他們的意見，都成爲我修改論文的重要參考。

感謝實驗室的學長和同學。特別是謝吉隆學長，對我的研究相當關照，三番

兩次給我提醒和建言，讓我論文的呈現，更爲完整妥善。

再次感謝以上所有的師長和朋友，沒有他們的支持和幫助，我的論文不可能順利完成。在埋頭研究的過程中，想起師友的鼓勵和期許，我有了更堅強的信心和力量，向茫茫未知的世界繼續前進。

目錄

表目錄

圖目錄

一、 緒論

1.1 研究動機

現今通行的標點符號，是 20 世紀之後，由西方傳入轉化而成。在此之前，中文的書寫系統沒有使用標點符號的習慣。如圖 1 和圖 2，過去的書籍文本，段與段之間有所間隔，句子則是串連在一起，必須由讀者在閱讀時，依據經驗和語感斷句，將文本切成一段段句子（*sentences*）或子句（*clauses*），然後才能理解文義。從漢代開始，有些讀者在斷句之後，會在書上留下斷句的符號，通常以圈代表句子的結尾（類似現今的句號「。」的作用），以點表示句子中語氣的停頓（類似現今逗號「，」的作用）。這種斷句的過程，就稱做句讀或圈點。雖然句讀符號類似今日句號與逗號的作用，但這是讀者在閱讀時所標記，而不是作者或印刻出版者在寫作制版過程中，事先標記在文本上。宋人岳珂在《九經三傳沿革例》說道：「監蜀諸本皆無句讀，惟建監本始仿館閣校書式從旁加圈點，開卷瞭然，於學者為便。然亦但句讀經文而已。惟蜀中卞本與興國本並點注文，益為周盡。」可知宋代有少數刻本，在刻書製版時加上句讀符號。雖然如此，從宋代到清末，有斷句標點的刊物甚少，並沒有成爲風氣，更不是書寫習慣的一部份。直到民國以後，西元 1919 年胡適等人提出《請頒行新式標點符號議案》，引入西方的標點符號，漢語的書寫系統，才開始普遍使用標點符號。因此可知，在民國之前，絕大多數的漢語典籍文本，都沒有標點符號。《三字經》說：「詳訓詁，明句讀」，斷句實是閱讀典籍文章的第一步驟。

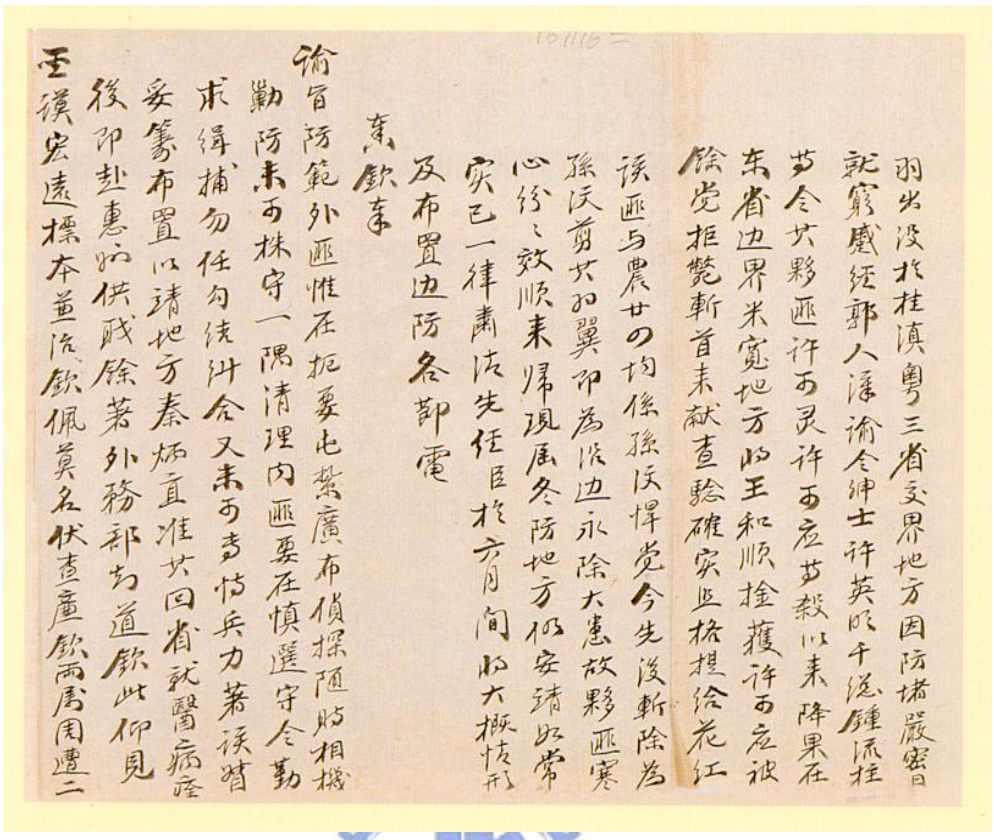

圖 1 清末奏摺

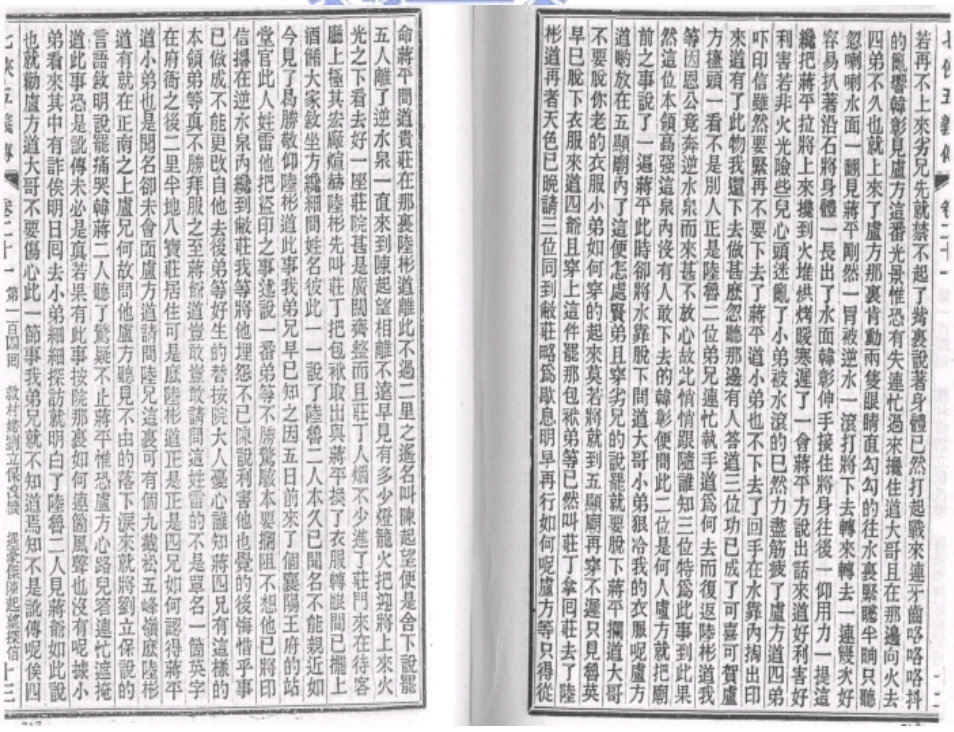

圖 2 清《七俠武義》刻本

古漢語的文本，在段落與段落之間有分隔，但在同一個段落裡，並沒有任何標點劃分句子與句子之間的界線。例如《莊子．逍遙遊》的開頭兩原文：

北冥有魚其名為鯤鯤之大不知其幾千里也化而為鳥其名為鵬鵬之背不知其幾千里也怒而飛其翼若垂天之雲是鳥也海運則將徙於南冥南冥者天池也

齊諧者志怪者也諧之言曰鵬之徙於南冥也水擊三千里摶扶搖而上者九萬里去以六月息者也野馬也塵埃也生物之以息相吹也天之蒼蒼其正色邪其遠而無所至極邪其視下也亦若是則已矣

而經過後人斷句之後，今日通行的文本則是：

北冥有魚．其名為鯤．鯤之大．不知其幾千里也．化而為鳥．其名為鵬．鵬之背．不知其幾千里也．怒而飛．其翼若垂天之雲．是鳥也．海運則將徙於南冥．南冥者．天池也．

齊諧者．志怪者也．諧之言曰．鵬之徙於南冥也．水擊三千里．摶扶搖而上者九萬里．去以六月息者也．野馬也．塵埃也．生物之以息相吹也．天之蒼蒼．其正色邪．其遠而無所至極邪．其視下也．亦若是則已矣．

由於斷句的符號並非原作者所加，沒有明確的規則與方法，全憑讀者自行判斷而來，因此同一篇文本，不同的讀者，往往有不同的斷法。舉例來說，《老子》中的第一句「道可道非常道名可名非常名」，有人斷作「道．可道．非常道．名．可名．非常名．」，也有人斷爲「道可道．非常道．名可名．非常名」。又如《論

文・八佾篇》中的「祭如在祭神如神在子曰吾不與祭如不祭」，常見的斷法是「祭如在・祭神如神在・子曰・吾不與祭・如不祭・」，有人卻認爲應斷作「祭如在・祭神如神在・子曰・吾不與・祭如不祭・」意義才明瞭 [1]。清人趙恬養則針對句讀沒有明確規則，摸稜兩可的特性，寫下「下雨天留客天留我不留」一例。[1] 這個句子共有七種斷法，每種斷法都通，但意義各不相同。由此可知，斷句對於文義的理解，不時有巨大的影響。從古至今，文人於對某些文本的斷句，時常有相歧的見解，對文義也因而有全然不同的解釋。所以，斷句不但沒有明確的規則，而且對熟讀詩書的古代文人來說，也是頗爲困難的消除歧義（*disambiguation*）程序。

由於斷句相當依賴人類的經驗和語感，到目前爲止，斷句的工作都以人力處理，並沒有自動化的斷句工具。儘管諸多古漢語的經典文本，都已經有人工斷句完成的版本，但隨著歷史文獻不斷地發掘，仍然有難以數計的文件，尚待斷句整理。因此，如果有自動化的工具，快速地處理大量的文件，將文本作初步斷句，後續再由專人修訂校對，自然可以大幅簡省時間和人力，並且增進斷句成果的正確性 [2], [3]。於此之外，在建立自動化斷句系統的過程當中，同時會釐清斷句的模式與特性，歸納出斷句的模型與基本規則。這將增進我們對古漢語的了解，有助於今人對古漢語的學習和研究，更可作爲日後研究古漢語處理的基礎。

字與字之間沒有空白，句與句之間沒有標點，這些都是中文等亞洲語言的傳統與特質。中文的自動化斷詞（Chinese word segmentation）研究，已經有多年歷史，在自動翻譯、中文辨識、中文輸入法等領域，也都有普遍的應用。相對於斷詞研究的豐碩成果，自動化古漢語斷句尚有很大的可能性，值得研究探索。

[1] 趙恬養，《增訂解人頤新集》

1.2 問題描述

在開始斷句工作之前，必須先定義斷句的目標和產物。以中文斷詞爲例，斷詞所得的產物就是詞（word），換句話說，斷詞是在連成一串的中文字元（漢字）中，辨別每一個詞的邊界。而古漢語斷句（或說句讀），目前在語言學上卻沒有明確的定義。其中，「句」是辨識句子（sentences）的邊界，類似句式邊界偵測（sentenece boundaries detection），這部份比較沒有疑問。而「讀」的部份，一般來說，是進一步將句子再切分爲「子句」（clauses）的單位。但是，除了子句需要斷開，中文在閱讀和傳述時，也習慣在「話題」（topics）之後稍作停頓。例如《莊子．逍遙遊》的句子「野馬也．塵埃也．生物之以息相吹也．」，其中「野馬也」、「塵埃也」都屬於話題，不是子句。因此，「讀」的工作，比較接近 shallow parsing（chunking）。綜合以上，斷句的產物，其實有多個層次，最大的單位是句子，其次是子句，而子句之中，有時又可以再按話題切開。換句話說，古漢語斷句工作，涵蓋了句子、子句、話題等三種邊界辨識的任務。

然而，中文句子和子句的分際，並不像英文那麼嚴謹。對一段很長的子句，在適當的地方插入逗點，便能將子句拆成多個較短的子句；對一句很長的句子，將部份逗點替換爲句點，長句子就變爲多個短句子。也就是說，句子或子句的邊界，其實有模糊的空間。不同的斷法，或多或少改變了文本的語氣，但在文法上都是合法的。在這樣的情況下，要明確地區別句子、子句、話題這三種層次，斷句工作勢必更爲複雜困難。而且，目前的數位化古漢語文獻，其中較嚴謹的版本，多以單一符號如「．」作爲斷句的分界（delimit）符號，並不細分句號和逗號，故而以 supervised learning 的技術，無法透過這樣的資料學習分辨斷開的地方，究竟是句子的結尾，還是子句的結尾。

所以，在本研究中，斷句的目標，是找出句子、子句、話題的結尾，將其斷開，但並不分辨斷出來的產物，屬於哪一種單位。也就是說，給定一段未經標點

的漢語文本，斷句系統在理想的情況下，會找出所有應該斷開的地方。至於斷開的地方，應該插入哪一種標點符號，是「讀」還是「句」，則不在本研究的範圍之內。

1.3 研究目標

自動化斷句系統是一個新的研究，沒有前人的標準可循，所以研究的首要工作，即是設定斷句成果的評估指標（*metrics*）。有了評估指標，計算斷句成果的精確度，而後才能以量化的方式，分析斷句系統的效能，評比系統的好壞，並作爲系統改進時的參考基準。因此，爲斷句問題尋找合用的效能評估方法，量化斷句結果「好」或「壞」的程度，是基本工作，也是本研究的重要環節。

其次，蒐集適當的語料（*corpus*），建立 *dataset*。Dataset 可以作爲 *machine learning* 系統的 *training data*，也可以作爲驗證斷句成果、評估系統效能的 *benchmark data*。Dataset 必須有一般性，才能使斷句系統在 dataset 上的效能表現，得以推廣到大多數的文本。作爲系統建立的 training data，這套 dataset 也必須有足夠的數量，和相當的代表性，才能幫助系統從中找出斷句通則，建構精確的斷句系統。

有了基礎的工具和素材，於是可以建立斷句模型。斷句是自然語言處理（natural language processing）的問題，在本研究中，我以 *empiricist* 的方法作爲基礎。斷句模型可以視爲一種機器學習器（*machine learner*），從大量斷句完成的 training data 中汲取斷句規則，調整參數；而後，面對未斷句的資料，便可以判斷哪些地方應該斷開。機器學習的基本架構，如圖 3 所示。截至目前，機器學習已有多年的研究，應用在諸多自然語言處理的問題，都有很好的成果。機器學習器的架構和實作方法也相當多樣，爲斷句系統設計一套適合的學習模型，是本研究的核心。

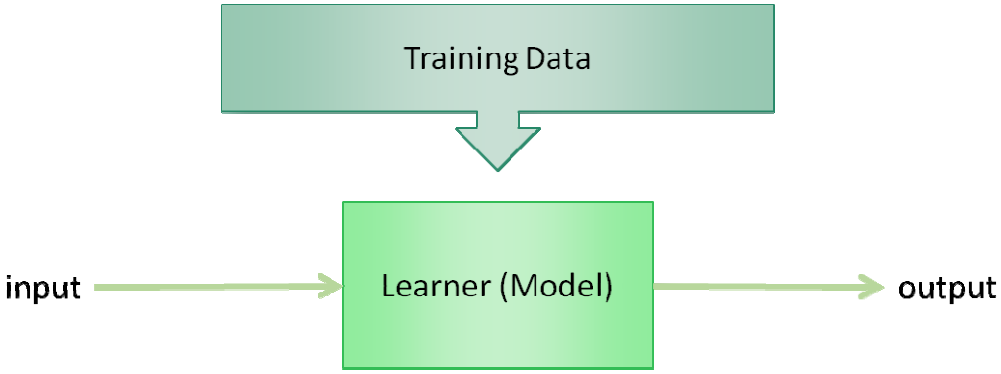

圖 3 Machine learning 示意圖

爲了簡化問題的複雜度，本研究鎖定在先秦兩漢的文本。古漢語隨著時代演進，有演化的現象，不同的時代，語言也有各自的特徵。將斷句系統鎖定在單一時代，會比找出符合所有時代的斷句系統來得實際。先秦兩漢的文本，句子比較短，句子中的子句也短，句型相對單純，而且許多詞彙（words）都以單一漢字表示，可以略微排除斷詞的需求。此外，漢語在先秦兩漢已經相當成熟，諸多經典著作，都完成於先秦兩漢。例如《四書》、《五經》、《老子》、《莊子》、《史記》，這些典籍都是後代文人研讀學習的素材，相當有代表性，是古漢語的根本基礎。所以，本研究聚焦在先秦兩漢的文本，從中選擇 datasets，作爲設計斷句系統的主要目標，並佐以清代的奏摺語料，實驗斷句系統，面對不同時代的文本，有哪些效能上的差異。

二、 相關研究

古漢語斷句是自然語言處理的問題，有幾項自然語言處理的典型議題，對本研究來說，頗有借鏡的價值。所以，我將討論中文斷詞（*Chinese word segmentation*）、句式邊界偵測（*sentence boundary detection*）、詞性標記（*part-of-speech tagging*）等議題目前的發展，以及這些領域的研究成果中，對斷句研究有所幫助的層面。

自然語言處理可以從方法上概分爲 *rationalist* 和 *empiricist* 這兩類。在 1960 到 1985 年之間，rationalist 方法是自然語言處理的主流。Rationalist 以專家系統（expert systems）的框架建立推論模型。首先，由語言學家（linguists）或語言專家（language experts）以其專業知識，針對目標問題，預先制定推論規則。然後，電腦程式便依循規則，進行推理判決。換句話說，電腦透過專家制定的規則，去重現人腦的判斷過程，所以電腦是在模擬、逼近專家的知識和經驗。而 empiricist 方法則是以機器學習的框架建立推論模型。人類不必具備太多語言方面的專業知識，只要提供資料與恰當的學習模型，用統計、樣式辨識（pattern recognition）、機器學習等技術，讓電腦直接從大量的資料上歸納出有意義的規則、調校模型參數，便可以進行推理。在此，電腦不再試著重覆人類的推理思維，而挾著其鉅大的記憶容量和快速的運算能力，用量化的觀點解決問題。由於 empiricist 方法是近年來自然語言處理的主流，在主要的議題上有很好的成果，因此，在本研究中，我也依循 empiricist 方法，以機器學習的架構，設計斷句模型。

古漢語斷句其實是一項 *text segmentation* 的工作，所以，我將特別關注相關領域中，序列處理的議題（即 *sequence labeling* 或 *sequence segmentation*），並在以下探討序列標籤化（*sequence labeling*）的典型方法 *hidden Markov models*（HMMs）和當前主流技術 *conditional random fields*（CRFs）。

由於古漢語本身的特性，頗不同於西方語言，也有別於現代漢語。在處理斷句之前，對古漢語要有基本的了解。我將從古漢語的特質中，找尋有助於人類斷句的特徵，並討論這些特徵是否合適應用在自動化的斷句系統中。

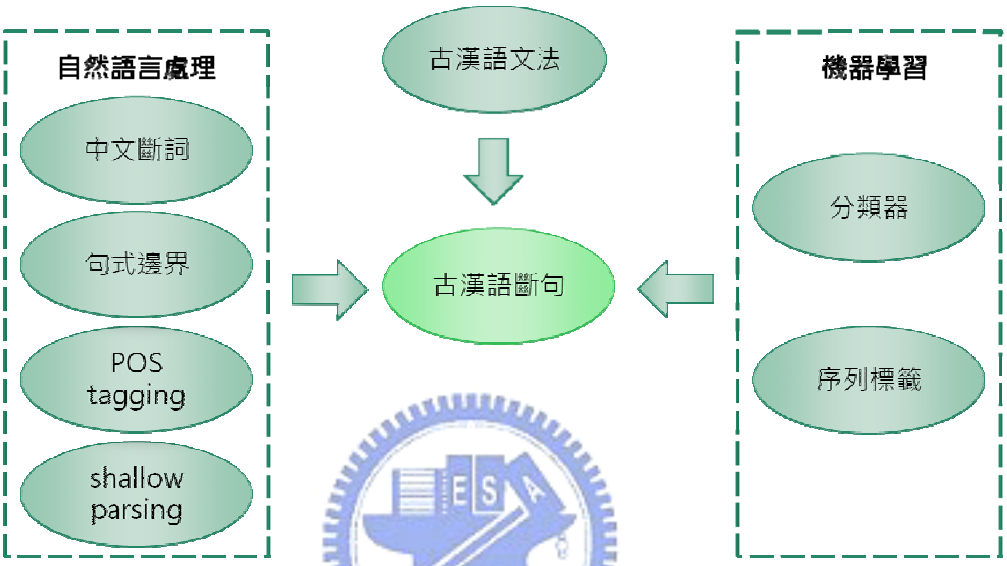

圖 4 古漢語斷句的相關研究

2.1 中文斷詞

西方語文在詞（word）與詞之間，通常有空白作爲分界（*delimit*），從文字之中，抓出每一個詞彙，並不是太困難的任務。但中文、日文、泰文等東亞語文，卻沒有使用 delimit 的習慣，在兩個標點符號之間，所有的字元（如漢字、*Chinese character* 或 *hanzi*）都連在一起，沒有空白或其他分隔符號，界定出詞的範圍。所以，在處理中文資料時，往往不能避開斷詞（word segmentation）的工作，必須決定句子之中，每一個詞彙的邊界，進而將句子中的所有詞彙辨識出來。

依據語言學上的定義，漢字的地位，介於西方語言中的字元（character）和詞彙（word）之間，相當接近 *morpheme*（詞索）的概念。一個或多個 morphemes 即可組成一個詞，以英文來舉例，*dog* 是 morpheme，同時也是由單一 morpheme 構成的詞。而 *dogs* 也是一個詞，但卻是由 *dog* 和 *s* 這兩個 morphemes 組成。英文等西方語言，morpheme 是連在一起的，但詞和詞之間有空白爲分界。而中文的詞彙既然由一個或多個 morphemes 構成，但缺乏詞與詞中間的分隔符號，故判斷中文斷詞的分界相當困難。

以這個句子爲例：

日文章魚怎麼說？

正確的斷法是：

日文　章魚　　怎麼　說　？

但卻也可以斷爲：

日　　文章　　魚　怎麼　　說　？

第一個斷法當然是對的。然而，就字面上來說，第二個斷法所斷出來的詞，也都是常見的詞彙，但整句話卻沒有意義。其中，「文」字可以和「日」字組成「日文」這個詞彙，卻也可以和「章」字組成另一個詞彙「文章」；「魚」字可以作爲單一字（morpheme）的詞彙，也可以和「章」字組成雙字的詞彙「章魚」。所以，斷詞的工作，必須在許多可能的斷法中，排除各式各樣的歧義（*ambiguities*），找出最合理的一種斷法。

中文斷詞的方法，主要分爲兩個大方向：字典式和機器學習式。字典式的方法（*dictionary-based approaches*），是預先建立詞彙的字典，然後由專家定義一套斷詞的規則，讓系統按照預定的規則，佐以字典上的資訊，作出斷詞判斷。但受限於人力，字典法很難訂立周延的規則和詞彙，應付各式各樣的字詞組合以及未知的新詞（*unknown words* 或作 *out-of-vocabulary*）。機器學習式的方法（*machine learning approaches*），則完全不依賴字典和專業知識，直接由大量經過斷詞的資料，統計歸納出斷詞的規則，藉此訓練 N-gram 或序列標記等模型，再用經過訓練的模型去斷詞。這兩種方向，可以互相搭配運用，這也是目前中文斷詞研究的主流。例如，Gao et al. [4] 將中文的詞彙分成四類，配合字典與詞類的資訊，利用 improved source-channel models 作斷詞；Zhang et al. [5] 使用階層式的 hidden Markov model，搭配專業的詞彙知識，建立斷詞系統。

在 Xue 的研究中，將中文斷詞轉化爲字元標記（character tagging）的問題，再以 maximum entropy model 來作序列標記 [6]。原本斷詞的工作，是在每一個漢字與漢字之間，決定是否要斷開。Xue 則定義了四種標籤：LL、RR、LR、MM，爲每一個漢字加上標記。被標示爲 LL 的漢字，就是某個詞的左界，是該詞的首字；被標示爲 RR 的漢字，則是某個詞的右界，是該詞的尾字；MM 表示該漢字是在詞的中間，不是首字也不是尾字；LR 則代表該漢字是單一字的詞，同時是詞的首字，也是尾字。舉例來說，下面這段文字：

汽車 用 時速 四十五 英里 瘋狂 地 穿過 空地

將每個字標記之後，成爲：

汽/LL 車/RR 用/LR 時/LL 速/RR 四/LL 十/MM 五/RR 英/LL 里/RR

瘋/LL 狂/RR 地/LR 穿/LL 過/RR 空/LL 地/RR

於是，LL 代表詞的開始，RR 表代詞的結束，LR 表代單一個字的詞，其他位於詞內部的字，則以 MM 表示；文字經由如此標記，斷詞就成爲很簡單的工作。換句話說，在 Xue 的方法中，斷詞的任務，轉變爲標記（tagging、labeling）的任務，於是便可以用諸多序列標記的演算法，如 Xue 所用的 *maximum entropy models*，來解決這個問題。

Peng et al. [7] 也用類似的策略處理中文斷詞問題。但與 Xue 在 [3] 的方法，有兩個主要的不同。第一，Xue 使用了 LL、RR、MM、LR 四種標籤，而 Peng et al. 只用 START 與 NON-START 兩種標籤，被標爲 START 的漢字，是詞的開始，其他漢字則是 NON-START。這樣的作法接近傳統的 segmentation，當某個漢字被標爲 NON-START，而下一個漢字是 START，就表示這兩個字中間應該斷開。第二，Xue 使用 maximum entropy model 來作 sequence labeling，而 Peng et al. 則以更新的技術 *conditional random fields* [8] 作爲標記的模型。

古漢語斷句和中文斷詞，都是中文特有的議題，並且都是 text segmentation 的工作。如前所述，Xue [6] 和 Peng et al. [7] 的研究，對本斷句研究有很大的幫助。Xue 使用 LL、MM、RR、LR 四種標籤，將 segmentation 的工作，轉變爲 labeling 的工作。Xue 所處理的單位是詞，但也可以擴大爲短語（chunks）或子句（clauses），LL 可以改爲標記子句的句首；RR 則是子句的句尾，MM 就是子句句中的字，LR 則是構成短句的單一字，如《孟子．告子下》裡的「告子曰．性．猶鯀柳也．義．猶桮棬也．」中的「性」字和「義」字。而 Peng et al.將 conditional random fields 這個在 2001 年由 Lafferty et al. [8]提出來的序列標記技術，首次應用在中文斷詞的領域，並獲得很好的效能。

2.2 句式邊界偵測（Sentence Boundary Detection）

句子（sentences）是構成文章的重要單位，有許多文件處理的應用，例如句式分析（syntactic parsing）、機器翻譯（machine translation）、自動摘要（document summarization）等，首先要將文本切分爲一連串的句子，才能進行後續的工作。在許多語言中，所謂句子，多半以句點（period, “.“）、問號（question mark, “?”）、驚嘆號（exclamation mark, “!“）作結尾。所以，這些符號可以視爲句子與句子之間的分界符號（delimit），並能藉此將文件分割成一連串的句子。然而，真實的情況並不是這麼單純。

大多數的西方語文，句點（period, “.“）符號固然常用在句子的最末，表示句子完結，但有時卻也用來表示縮寫（abbreviation），如 “No. 1”，有時又可以作爲刪節號（ellipsis, “...”）的一部份，或是數字中的小數點（如 ”3.1415926”）。所以，在文本中判斷句子的結尾，辨識句子的邊界，並不是簡單直觀的任務。在自然語言處理中，把這個問題稱爲句式邊界偵測（sentence boundary detection，又稱作 sentence boundary disambiguation 或 sentence boundary identification，也可稱爲 sentence segmentation）。

句式邊界偵測有很多種作法。最簡單是 *period-space-capital letter* 演算法 [9]。凡是句點之後緊接著一個空白字元，又接一個大寫的字母，則可以假定這個句點標示句子的結尾。這個方法實作簡單，可以寫成 regular expression [.?!][ ]+[A-Z] 的形式，簡便地運用在許多地方，但效果卻差強人意。舉例來說，在 *He stopped to see Dr. White . . .* 句中，*Dr.* 中的句點，是用來表示縮寫，但因爲其後接著一個空白，又接著大寫字母 *W*，在 period-space-capital letter 演算法的判斷下，會將這個句點視爲句子結尾，於是切出 *He stopped to see Dr.* 這樣錯誤的句字。對於這樣的錯誤，固然可以準備一套字典，羅列各種縮寫詞彙，讓 period-space-capital letter 盡量涵蓋各種例外情況，但必須耗費極鉅的人力和工時，而且還未必能應

付層出不窮的特殊情況。

目前主流的研究，捨棄了 rule-based 的框架，改以統計式（statistical）的思維切入這個問題。統計式的作法，就是機器學習的方法，將句式邊界偵測，視爲一個分類（classification）的問題。首先，利用機器學習的技術，建立適當的分類器，繼而從大量的語料中，訓練分類器。訓練完成之後，分類器對輸入文本中的每一個句點、問號、驚嘆號作二元分類，判斷是否爲句子的結尾，而判斷的依據，就是符號前後的文字。分類器的實作方式很多，前人研究中，使用了回歸樹（regression trees）[10]、類神經網路（artificial neural networks）[11], [12]、決策樹（decision trees） [13]、maximum entropy modeling [14] 等分類器，效能和最佳的 rule-based 演算法差不多，錯誤率在 0.8%-1.5%之間 [9]。

句式邊界偵測，乍看之下，和古漢語斷句頗有類似之處，但是現有的句式邊界偵測模型，卻無法直接套用到古漢語斷句的系統上。第一，傳統的句式邊界偵測所 disambiguating 的對象，是文本中的每一個句點、問號、驚嘆號，斷句所 disambiguating 的對象，卻包括所有字與字之間的間隙。在這點來說，古漢語斷句更接近中文斷詞。第二，只要用簡單的 rules，例如前述的 period-space-capital letter 演算法，就堪能辨識大多數的案例。但在古漢語斷句問題，卻沒簡單的規則，足以應付大多數的狀況。第三，句式邊界偵測的問題，主要出現在西方語言，相關的研究，自然都以西語（特別是英語）爲主。然而，許多西語的性質，爲中文所無，所以，許多有助於句式邊界辨識的語言特徵，不能完全套用在中文上。舉例來說，西文的大小寫（capitalization）、字首變化（prefix）、字尾變化（suffix）等特徵，在句式邊界判定中，有顯著的參考意義，但中文卻沒有相應的特性可以利用。

儘管無法直接從句式邊界偵測的現有研究中，找到合乎斷句需求的模型，但

卻可以借用此一領域的效能準則（performance metrics），來作古漢語斷句系統的評估指標。句式邊界偵測的常用指標是 *F-measure* 和 *NIST-SU error rate* [15]。

F-measure 是 *recall* 和 *precision* 這兩個數值的 *harmonic mean*。先不考慮問號和驚嘆號，假設某個文本，共有 k 個句點，其中的 m 個是實際的句尾標示（故該文本共有 m 個句子，且 $m \leq k$），而句式邊界偵測系統判定這 k 個句點中，有 n 個句尾標示，其中判定正確的有 c 個（故 $c \leq n, m \leq k$）。則 $recall = \frac{c}{m}$，即代表，所有的句尾當中，獲得正確判斷的比例； $precision = \frac{c}{n}$，代表句式邊界偵測系統所判定的句尾當中，真正是句尾的比例。這兩個量值的範圍都在 0%到 100%之間，數值越高，判斷的效能越好。F-measure 綜合了 recall 和 precision 這兩個指標，合成單一指標，方便評估比較。F-measure 也介於 0%到 100%之間，並接近 recall 和 precision 當中，較低的一邊，只有當 recall 和 precision 均高的時候，F-measure 才會高。

NIST-SU error rate 是另一個常見的指標，除了句式邊界偵測，這指標也經出現在其他 segmentation 的議題。NIST-SU error rate 計算是 segmentation 的錯誤率，所以其值越低，代表判斷的錯誤越少，最低爲 0%，最大值則可能超過 100%。

F-measure、recall、precision 這三個指標，在中文斷詞和諸多 classification 的研究經常出現，作爲評估效能的參考。而 NIST-SU error rate 則常見於各類 text segmentation 研究。我將在第三章中，詳細介紹這幾項指標，並作定義。進而使用這些指標，來作斷句效能的評量。並會對照實驗結果，檢討這些指標在斷句研究中的適用性。

2.3 詞性標記（Part-of-Speech Tagging）

詞性標記（part-of-speech tagging, POS tagging，簡稱 tagging）是自然語言處理的基礎課題。在研究如何讓機器理解（understanding）自然語言之前，必須先能解析（parsing）自然語言，而要達成解析的工作，則先要能辨識文本中每一個詞的詞性（syntactic category）。舉例來說，*The representative put chairs on the table* 這句話，可以加上詞性標記，得到：

The/AT representative/NN put/VBD chairs/NNS on/IN the/AT table/NN.

詞性的分類方式有許多種，有精有粗，但總不脫名詞（nouns）、動詞（verbs）、形容詞（adjectives）、副詞（adverbs）等幾個大類爲基礎。在上面的例子中，AT 爲冠詞（article）、NN 爲單數名詞、VBD 爲過去式動詞、NNS 爲複數名詞、IN 爲介繫詞（preposition）。

因爲一個詞，可能同時有許多種詞性，所以詞性標記也是 disambiguation 的工作。舉例來說，前述的句子，也可以標記爲：

The/AT representative/JJ put/NN chairs/VBZ on/IN the/AT table/NN.

在此，representative 被標記爲 JJ（形容詞，adjectives）、put 被標爲名詞、chairs 被標爲 VBZ（第三人稱現在式動詞）。第二種標記方法，在文法的角度上完全正確，但在文義上卻沒有意義。因此，詞性標記所面對的問題，是在各種可能的標記方案中，選擇最有可能的一種。

自動化的詞性標記，從 1950 年代開始。早期以 rule-based 爲主，依靠人工建立 disambiguation 的規則，如今則以 machine learning 的方式爲主流，達到

96%-97%以上的精確度。由於詞性標記是自然語言處理，基礎而重要的工具，在西語之外，中文、日文等許多語言也都有類似的研究，甚至還有古漢語的詞性標記 [16]。在 Huang et al. [16] 的詞性標記研究中，也有提到古漢語斷句的需求。在他們的系統中，古漢語文本必須先經過斷句，然後才能標記詞性。但 Huang et al. 並沒有處理古漢語斷句的問題，而是直接拿經由人工斷句完成的資料，來作測試。

詞性標記是典型的序列標記（sequence labeling）任務，對應到 Xue [6] 的中文斷詞研究，序列標記不只可以應用在爲詞性標記，也可以爲單一漢字標上其在詞中的位置（Xue 以 LL、RR、LR、MM 四種標籤來標記）。再以此推廣，序列標記的目的，也可以化爲對每一個漢字，標上其在子句中的位置（如句首、句中、句尾等），於是便達成斷句的工作。

2.4 Markov Model Taggers

典型的 machine learning 架構，包含 decision trees、Bayesian networks、support vector machines、neural networks 等方法 [17], [18]。這些基本的方式，通常假定每一次處理的 instance 互相獨立，也就是說，每個 instance 的處理結果，並不會影響到其他 instance。但在本研究中，處理的對象是一段一段連續的漢字字串，在一個長句中任何一個地方斷開，對整個句型和句意都有莫大的影響，必然牽連到下一處的斷句判定。在面對漢字字串這樣前後相依的序列性資料（sequence data）時，則很合適以序列標記（*sequence labeling*）的模型來解決。

目前，序列標記技術，在資料探勘（data mining）、樣式辨識（pattern recognition）（特別是語音辨識）等領域，皆有廣泛的應用。近十年來，除了 maximum entropy models [19]、conditional random fields [8] 等典型的序列標記模型之外，也有人將一般性的 large margin 分類器，如 support vector machine

和 AdaBoost 化用在序列標記的工作上，並得到不錯的效果 [20], [21]。但是，截至今日，最典型，最簡單而好用的序列模型，仍然是具有數十年歷史的 hidden Markov models [22]。

Hidden Markov models 是 *generative models*。當給定模型參數 λ 之後，對於 observation sequence 的隨機變數 $\mathbf{X}$ 和其所對應的 label sequence 隨機變數 $\mathbf{Y}$，具有 joint 機率分布 $p(\mathbf{X}, \boldsymbol{Y})$。於是可以利用此特姓，處理三種典型的 sequence 問題：

1. 估測（evaluation）：給定模型λ，計算某一特定的 observation sequence X 的出現機率 $P(X|\lambda)$。

2. 解碼（decoding）：給定模型λ，和某個特定的 observation sequence X，找出 X 背後出現機率最高的 label sequence $Y = \arg\max_Y P(X, Y|\lambda)$。在此，通常使用 Viterbi 演算法實作 [22], [23]。

3. 學習（learning）：在未知 λ 的情況下，給定許多組 observation sequence X，作爲 training set。從中找出最符合 training set 的模型 $\lambda = \operatorname{argmax}_\lambda P(X|\lambda)$。

古漢語斷句轉化爲序列標記的問題之後，便很合適運用 hidden Markov models 來處理 labeling 的工作。首先，以統計的方式，從眾多的古漢語典籍中，建立出斷句的序列標記模型參數 λ。得到 λ 之後，再利用問題 2 的方式，爲尚未斷句的文本，找出最佳的斷句決定。由於 λ 可以直接從 training data 中統計得知，而不必透過問題 3 的方法，從 unsupervised 的資料中模索調整，所以這個方法

又可稱爲 *Visible Markov Model Tagger* [24]。

2.5 Conditional Random Fields

2.5.1 簡介

Hidden Markov models 是經典的 sequence models，然而，最近幾年，效能最好，獲得許多領域廣泛使用的序列模記模型，則是 conditional random fields (CRFs)。Lafferty et al. [8] 提出 conditional random fields，主要針對 hidden Markov models 和 maximum entropy Markov models [25] 的缺點，改進而來。相較於傳統的 hidden Markov models，以 conditional models 爲基礎的 conditional random fields 沒有 generative models 的限制，所以可以如 maximum entropy models 或 maximum entropy Markov models，自由地在 model 中，增加各種形式的 features。而相較於同樣是 conditional probabilistic based 的序列模型的 maximum entropy Markov models，conditional random fields 迴避了惡名昭彰的 *label bias problem* [8], [26], [27]。除了理論上的優點，conditional random fields 也確實在實驗中，超越 hidden Markov models 和 maximum entropy Markov models，而迅速成爲主流的序列標記模型。目前已經應用在詞性標記（part-of-speech tagging） [8]、部份短語標記（shallow parsing） [28]、中文斷詞 [7]、語音資料的句式邊界偵測 [29] 等問題，並且展現優秀的效能。

2.5.2 模型定義

在一般情況下，conditional random fields 是一種 undirected graphical models（無向圖模型），但在作序列標記時，則只使用其中的特例：鏈狀結構

（chain-structured）無向圖，如圖 5 所示。

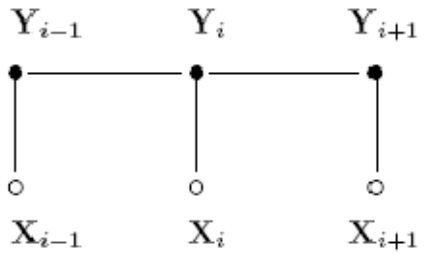

圖 5 鏈狀結構的 conditional random fields 圖形

設 $\mathbf{X}$ 爲 observation sequence 的隨機變數，$\mathbf{Y}$ 是對應於 $\mathbf{X}$ 的 label sequence 的隨機變數，在此假設 $\mathbf{X}$ 和 $\mathbf{Y}$ 有相同的長度。$\mathbf{Y}$的每一個項目 $\mathbf{Y}_i$ 的內容都是一個 label，例如詞性標記中的詞性，或是中文斷詞中，漢字在詞中的位置（舉 [6] 的例子，就是 LL、RR、MM、LR 這些標籤）。Conditional random fields 定義 $p(\mathbf{Y}|\mathbf{X})$ 是給定輸入序列 $\mathbf{X}$ 之後，label sequence $\mathbf{Y}$ 的條件機率分布。

定義：設無向圖 $G = (V, E)$，且 $\mathbf{Y} = (\mathbf{Y}_v)_{v\in\mathbf{V}}$，故 $\mathbf{Y}$ 裡的每一項，都是 G 上的頂點。當 $\mathbf{X}$ 條件成立，且隨機變數 $\mathbf{Y}_v$ 符合 *Markov property* $p(\mathbf{Y}_v|\mathbf{X}, \mathbf{Y}_w, w \neq v) = p(\mathbf{Y}_v|\mathbf{X}, \mathbf{Y}_w, w\sim v)$，則 $(\mathbf{X}, \mathbf{Y})$ 是 *conditional random fields*。[2]

由於我們在此只考慮鏈狀結構（chain-structured）的 conditional random fields，因此，G 可以簡化爲鏈狀形式：$G = (V = \{1, 2, 3, \ldots m\}, E = \{(i, i+1)\})$，其中 $1 \leq i < m$。

設輸入的 observation sequence $X = x_1, x_2, \ldots x_m$，其對應的 label (state) sequence $Y = y_1, y_2, \ldots y_m$，則可以透過 conditional random fields 計算給定 X

[2] 在此，$w \sim v$ 表示 w 和 v 是 G 上的相鄰點

時，Y 的條件機率：

$$P_\lambda(Y|X) = \frac{1}{Z_\lambda(X)} \exp\left(\sum_{t=1}^{m}\sum_{k} \lambda_k f_k(y_{t-1}, y_t, X, t)\right) \qquad (1)$$

其中，$Z_\lambda(X)$是正規化因數（normalization factor），使所有 $P_\lambda(Y|X)$對所有 Y 值的機率總合爲 1。而 $\boldsymbol{\lambda} = \{\lambda_1, \lambda_2, \ldots\}$ 和 $\boldsymbol{f} = \{f_1, f_2, \ldots\}$ 則分別是參數（parameters）和特徵函數（feature functions）。

特徵函數 $f_k(y_{t-1}, y_t, X, t)$ 通常是二元值（binary-valued）函數，而參數 λ_k 則可以視爲是這個函數 f_k 的 *weight*。由於特徵函數的格式，考慮了標籤前後的相依（$y_{t-1} \rightarrow y_t$）、整個 observation sequence X、以及目前在序列中的位置 t，所以在使用 conditional random fields 時，可以針對問題特質，設計各式各樣的特徵函數。以古漢語斷詞來說，被標記爲 RR 的字（詞的尾字），下一個字很可能就是 LL（詞的首字）。所以對應的特徵函數就是：

$$f_k(y_{t-1}, y_t, X, t) = \begin{cases} 1, & \text{if } y_{t-1} = RR \text{ and } y_t = LL \\ 0, & \text{otherwise} \end{cases}$$

由於在 RR 後面接著 LL 的可能性極高（另一個可能是 RR 之後接 LR，但 LR 是極少數），所以這個特徵函數所對應的參數λ_k經過訓練之後，理應有相當高的權重。

而像「汽車」這個詞，「汽」字是 LL，「車」字是 RR，則可以用這個特徵函數來描述：

$$f_k(y_{t-1}, y_t, X, t) = \begin{cases} 1, \text{if } y_{t-1} = LL \text{ and } y_t = RR \text{ and } x_{t-1} = \text{汽} \text{ and } x_t = \text{車} \\ 0, \text{otherwise} \end{cases}$$

將 feature function 改寫爲比較簡短的形式，得到：

$$F_k(Y,X) = \sum_{t=1}^{m} f_k(y_{t-1}, y_t, X, t) \qquad (2)$$

再將 (1) 式改寫爲：

$$P_\lambda(Y|X) = \frac{1}{Z_\lambda(X)} \exp\left(\sum_k \lambda_k F_k(Y,X)\right) \qquad (3)$$

特徵函式可以由人工設計、從外部的字典檔產生，也可以透過自動化的方法，從 training data 中汲取。而每一個特徵函式 f_k 所對應的 weight λ_k 值，無法用分析式的方法計算，而必須要用 parameters estimation 求出。

有了特徵函數 $\boldsymbol{f}$ 以及參數 $\boldsymbol{\lambda}$，對於給定的 input sequence X，其最有可能的 label sequence 就是：

$$\hat{y} = \arg\max_y P_{\boldsymbol{\lambda}}(y|X)$$

此時，只要套用典型的解碼演算法 Viterbi [22], [23]，便可以很有效率地算出 label sequence $\hat{y}$ [8]，和 hidden Markov models 的解碼過程相似。

2.5.3 參數評估

對 conditional random fields 而言，*training* 就是對**λ**作參數評估（parameter estimation）。

假設總共有 n 筆 training data，data i 以 (x^i, y^i) 表示，其中x^i是 observation sequence，y^i是 label sequence。則 training 的目的，要找到一組$\boldsymbol{\lambda}$，使得 training data 的 log-likelihood 最大化，即 maximum log-likelihood estimation：

$$L(\lambda) = \sum_{i=1}^{n} \log P_\lambda\left(y^i \middle| x^i\right) = \sum_{i=1}^{n} \left[\log \frac{1}{Z(x^i)} + \sum_k \lambda_k F_k\left(y^i, x^i\right) \right]$$

由於 L 是 concave function，必然收斂，而且可以找到 global maximum，而收斂的點在 L 微分爲 0 的地方：

$$0 = \nabla L = \sum_{i=1}^{n} \left\{ \left[\sum_k \lambda_k F_k\left(y^i, x^i\right) \right] - \left[\sum_y \sum_k F_k\left(\mathrm{y}, x^i\right) P_\lambda\left(\mathrm{y} \middle| x^i\right) \right] \right\}$$

參數值 λ 無法用分析式的方法計算，必須用數值方法迭代逼近。在 Lafferty et al. [8] 原始的論文中，沿用 maximum entropy models 的研究成果 [19], [30]，用 improved iterative scaling algorithms（IIS）作 conditional random fields 的 training。雖然 improved iterative scaling algorithms 實作簡單，也保證必定收斂，但速度並不理想。Sha et al. [28] 在 shallow parsing 的研究裡，測試了許多種優化演算法（optimization algorithms），包括 preconditioned conjugate-gradient (CG)、limited-memory quasi-Newton (L-BFGS)、generalized iterative scaling (GIS)、non-preconditioned CG，並指出 L-BFGS 收斂速度最快，而且時間只有 GIS 的

2.27%，是目前最理想的評估方法。而後，其他的研究也都以 L-BFGS 作爲主流 [7], [29], [31]。

2.5.4 Averaged Perceptron Training

在 Sha et al. [28] 的研究中，將 Collins [32] 的 *averaged perceptron* 視爲 conditional random fields 的變形，也將 averaged perceptron 當作參數評估的方法之一，和其他評估方法比較。

原始的 perceptron 演算法由 Rosenblatt [33] 年提出，是一種簡單的類神經網路。Freund et al. [34] 依據 perceptron 的精神，提出 voted perceptron 和 averaged perceptron，並證明這兩種演算法的效能，足以和 support vector machines 等主流的分類器相提並論。而後，Collins 再將 Freund et al. 的成果擴展，用 voted perceptron 和 averaged perceptron 處理序列標籤的問題 [32], [35]。

Collins [32]用於 parsing 和 tagging 的 perceptron 演算法，機率模型與 conditional random fields 完全一樣，不同之處在於，傳統的 conditional random fields 用 maximum likelihood estimation 求參數最佳化，而 Collins 改以 perceptron 的方法作參數逼近。Perceptron 的 training 概念是，儘量使特徵函數在 training data 上的適用度 $F_k(y^i, x^i)$，接近特徵函數與系統預測結果的適用度 $F_k(\hat{y}_k, x^i)$。舉例來說，在第 t 回合時，對於每一筆 training data i，用以下的方法更新所有參數：

$$\lambda_k^{t+1} = \lambda_k^t + F_k(y^i, x^i) - F_k(\hat{y}_k, x^i) \text{ for all } k$$

其中，$\hat{y}_k$是以目前的λ^t 爲基礎，透過 Viterbi 演算法求得：

$$\hat{y}_k = \arg\max_y P_{\lambda^t}(y|x^i)$$

Perceptron 不像其他參數評估法，擔保必定收斂。但 Collins [32]和 Sha et al. [28] 的實驗指出，perceptron 以非常快的速度逼近 global maximum，通常只要將整個 training data 掃過十數個回合，就能得到很好的成果。但是，經過前幾回合快速地逼近之後，成長速度會大幅衰減，甚至還有可能退步。根據 Collins 的實驗，將每一代的參數平均，作爲最終的模型參數，比只取最後一代的參數有更好效能。這個方法，Collins 便稱之爲 *averaged parameters*，而後又有人稱呼此版本的 perceptron 爲 *averaged perceptron*。在 Sha et al. [28] 的研究中，averaged perceptron 僅僅訓練兩個回合，shallow parsing 的 F-measure 已超過 93%，但再經過許多回合之後，得到最佳的成績是 94.09%，成長極微。即使如此，只訓練兩個回合的成績，其實已經逼近以 L-BFGS 優化，傳統 conditional random fields 的成績（94.38%）。

Collins [32] 的 averaged perceptron 演算法，和 conditional random fields 有相同的機率模型，雖然並不更精確，但實作相對容易，訓練的速度也快，卻又能達到逼近 conditional random fields 的表現。因此，averaged perceptron 在自然語言社群，受到廣泛的歡迎 [36]。

2.6 古漢語的語言特徵

清末馬建忠所著的《馬氏文通》，是第一部中文的文法書。馬建忠挪用西洋拉丁語系的文法概念，分類歸納古漢語的語法，其中，也討論到了句讀的問題，並整理出一部份粗略的斷句規則。民國初年的語言學家楊樹達，繼馬建忠之後，

建立詞類劃分爲中心的體系，研究古漢語語法。其著作《詞詮》 [37]，收錄古書中常用的 470 多個虛詞，分門別類，舉例說明用法。楊樹達的另一著作《古書句讀釋例》 [1] 則以「句讀之事，視之若甚淺，而實則頗難」爲由，從「誤讀的類型」、「誤讀的貽害」、「誤讀的原因」、「特殊的例句」四個層面，舉例探討句讀錯誤的因素。該書假定讀者預先具備斷句的基礎，講述細微而容易誤讀的案例。因此，雖然《古書句讀釋例》是斷句的重要知識來源，畢竟無法單憑書中的條例與片面的規則，建立真正具備斷句能力的 rule-based 系統。

前面介紹了許多種序列標籤化的方法，然而，無論是哪一種方法，都必須從 data 中汲取有意義的特徵（features），利用這些特徵作 training，也利用這些特徵資訊，爲未知的 sequence 作 labeling。所以，如果能從 data 中粹取（extraction）越多特徵，將有助於增進分類器的效能。在本研究中，training data 是一段一段經過斷句的文字。漢字本身固然可以當作一種特徵，但藉由中文的特性，由漢字出發，還能找到更多有助於斷句的間接特徵，這些特徵包含聲韻、詞性、對句等。

以聲韻來說，某些音韻很少連在一起使用，當相鄰兩個字的聲韻屬於這種情況時，表示這兩個字很可能必需斷開。楊樹達 [1] 提到「因不識古韻而誤讀」，可知聲韻也是斷句辨識的線索之一。在詞性則有助於分析句子的結構：某些虛詞常用在句首，如「夫」、「蓋」，而某些常用在句末，如「也」、「矣」，這些都可以作爲斷句的識別符號。對句在古漢語中，有頗高的出現頻率，這種特殊的性質，也有助於斷句。比如說，當我們發現上下兩段文字的句型結果類似，如「賢者以其昭昭，使人昭昭，今以其犷犷，使人昭昭」，或者有明顯的詞對，如「堯舜，性之也，湯武，身之也」（「堯舜」對「湯武」、「性」對「身」。以上兩句出自《孟子．盡心篇》），都可以從中把句子切爲對稱的兩半。

然而，這些由字面連結出去的間接特徵，都需要相關資料庫的配合，才能附加到 training data 上。目前，台灣大學中國文學系和圖書資訊學系提供線上音韻

字典[3]，供一般大眾查詢漢字的聲韻，收有中古音、中原音韻、中州音韻、吳語等幾種聲韻資料，並以反切和擬音等記號標示發音。雖然，這個資料庫並沒有上古漢語的資訊，但仍然可以借用中古音來作現階段暫時性的材料。至於詞性和對詞，目前則尚未找到合用的資料庫。雖然有《幼學瓊林》、《馬氏文通》、《高等國文法》、《詞詮》等包含詞性和對詞資料的書籍，但將紙本文件數位化和結構化需要人工處理，曠日費時，只能留待將來解決。

[3] http://moodle.lips.tw/~yinyun

三、 系統設計

本研究的主要目標有三：

1. 找尋適合評估斷句成效的準則（metrics）
2. 蒐集語料、建立 datasets
3. 以 machine learning 技術，建立**古漢語斷句模型**

這三項工作，在古漢語斷句研究上，如圖 6 所示，其實環環相扣，互相依賴。少了任何一部份，古漢語斷句研究便無法進行。斷句模型，居於斷句系統的核心位置，也是本研究的核心議題。Dataset 在本研究中，有兩種功能，訓練（training）和測試（testing）。所謂 training，是讓斷句模型從中汲取有意義的語言特徵，並統計出每一個特徵的權重，建立模型的參數（parameters），然後，斷句模型始能運作。而 testing 則在斷句模型上線之前，測試系統的功能，評估斷句模型的表現。當 test data 經過斷句之後，成爲一篇篇有斷句符號的文件，此時固然可以用肉眼分析斷句的成果，觀察斷句的效能，但若 test data 相當龐大，成千上萬，單純用肉眼判斷，不但費時耗力，而且也很難從繁浩的文件中，理出有意義資訊。這時，就必須藉由評量準則（metrics），將文字性的測試成果，轉換成量化的數據，並將龐大的數據統計成少數幾項指標，人類再透過這些量化的指標去了解測試結果，觀察斷句模型的特性，評量斷句系統的效能。

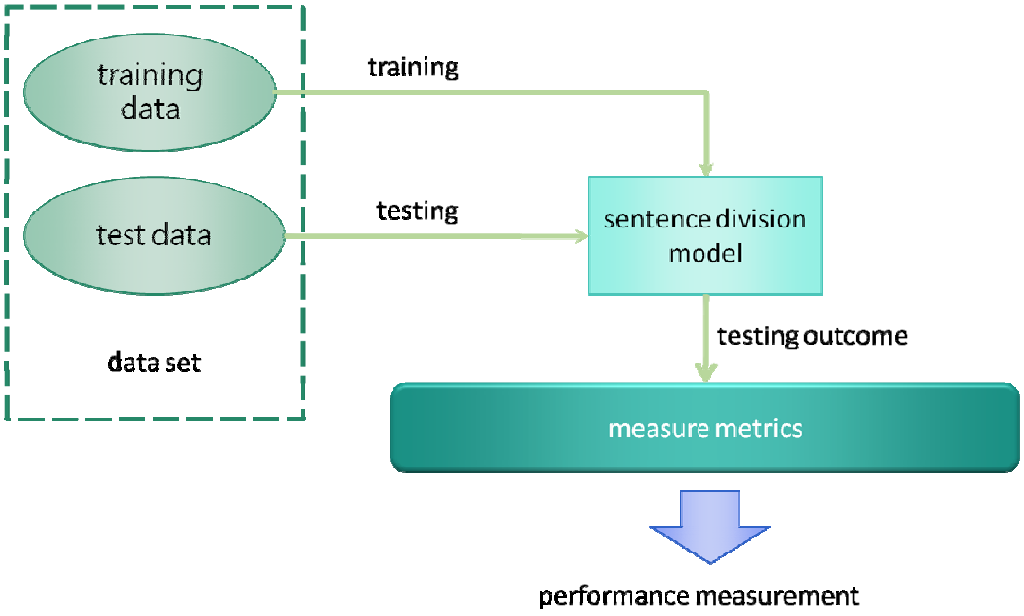

圖 6 古漢語斷句研究系統架構圖

典型的自然語言處理議題，都各有其常用的 datasets 和評估指標。但古漢語斷句是新的研究，沒有前人的標準可用，所以，本研究雖然以斷句模型的研究爲核心，但爲了訓練斷句模型和檢驗斷句模型的效能，所以也將擬定 metrics 和 datasets 的工作，納入本研究的目標。

本章以下的篇幅，將逐項介紹這三項工作的設計內容，並重心放在古漢語斷句模型設計，詳細探討我的設計觀點和援用技術。

3.1 評量準則（metrics）

古漢語斷句是自然語言處理（natural language processing）的問題，牽涉到機器學習（machine learning）、樣式辨識（pattern recognition）、語音辨識（speech recognition）、資料探勘（data mining）、資訊擷取（information retrieval）等領域的技術。所以，我先從這些領域中的相關議題下手，找尋通用的、適當的評量準則。

在分類器（classification）的評量上，最常見的測量方式是精確度（accuracy measure），也就是正確判斷的次數，除以判斷的總次數。在古漢語斷句來說，斷句模型決定了每兩個相鄰的漢字之間，是否應該斷開。倘若一段文字，共有 n 個漢字，斷句模型則需作 $n-1$ 次「斷或不斷」的判決。若這 $n-1$ 個判決當中，有 m 個是正確的（$0 \leq m \leq n-1$），則整體斷句的精確度就是 m/n。

用精確度來作評量，雖然簡單方便，但應用在斷句的評估上，卻有盲點。以《孟子》爲例，不含標點和篇名，總字數爲 35392 字，而用來將句子斷開的標點則有 7091 個[4]。平均來說，每五個字爲一斷，相鄰的字與字之間，不斷的機率遠高於斷，斷句模型即使對所有的 case 一概作出「不斷」的判斷，也能達到 80% 的精確度。在這種度量標準下，一個幾乎完全不斷句，但每次斷都斷錯的分類器，也可以得到不錯的精確度，然而，這種斷句成果在實質上並沒有用處。反之，積極斷句，但有時會錯斷的分類器，對人類或許有參考價值，卻會在精確度的評量上吃虧。由此可見，精確度可以參考，但不能作爲評量斷句成果的單一準則。

字與字之間，「斷」與「不斷」的機率分佈懸殊，這樣的情況叫做 *class imbalance problem* [17]，許多醫療診斷的問題也是如此。在檢驗疾病時，健康無病的受驗者，通常會遠多於真正患了該病的受驗者。所以，在評量測驗工具時，偏向陰性（無病）判斷的測驗工具，在精確度上會有較佳的表現，但此種較爲樂觀的判斷，可能忽略真正得病的患者，而延誤就醫時機。所以，在 class imbalance 的情況下，評估分類器效能，在精確度之外，會合併考慮 recall（又作 *sensitivity*，敏感度）和 *specificity*（特異度）這兩個準則，作爲評比的參考。Recall 是指所有患了某病的人，被正確檢驗出患病的機率；specificity 和 recall 互補，指是所有的沒有患該病的人，被正確判斷出無病的機率。所以，如果將這兩個指標應用在斷句評估，則 recall 就是該斷的地方，有正確斷開的機率；specificity 則是不

[4] 統計自中央研究院漢籍電子文獻之《斷句十三經經文・孟子》

該斷開的地方，沒有被錯斷的機率。recall 和 specificity 是相對的，如果某種分類器的 recall 和 specificity 都相當高，則表示此分類器有相當高的準確性，而不像單一的精確度容易受類別分佈懸殊的影響。

在評估系統的 recall-specificity 的關係時，常配合 *ROC curve*（receiver operating characteristic curve）輔助 [17], [24]。ROC curve 是二維的圖，縱軸通常是 recall，橫軸是 *fallout*（= 1 – specificity）。對於某個分類器，每次的效能表現，便可以取 recall 和 specificity 兩項指標，約化爲二維圖型上的點，最後由(0, 0)開始，沿著橫軸，將每一個點連線，最後再接到(100%, 100%)，便得到一條曲線。這條曲線的涵蓋面積（AUC，area under the curve）越大，代示該分類器的整體 recall-specificity 表現較好。如圖 7 的範例，A 與 B 是兩個分類器，各有兩次表現。由圖上所示，A 的 ROC Curve 涵蓋面積，明顯大於 B。因此可以推論，整體來說，A 在 recall-specificity 關係上，效能勝於 B。

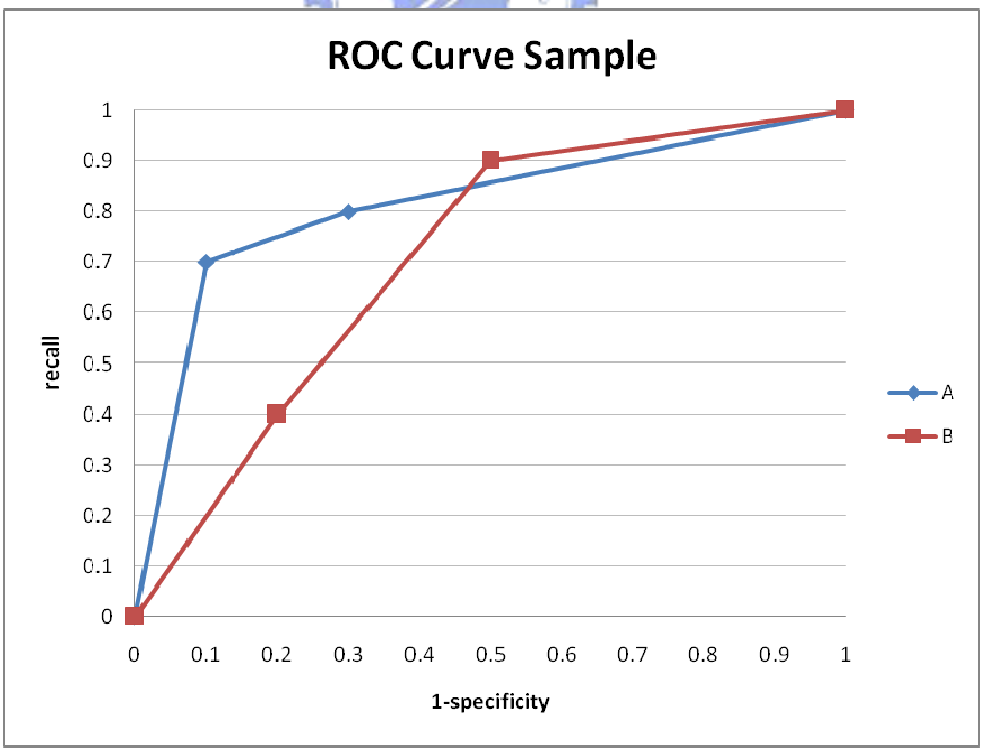

圖 7 ROC Curve 範例

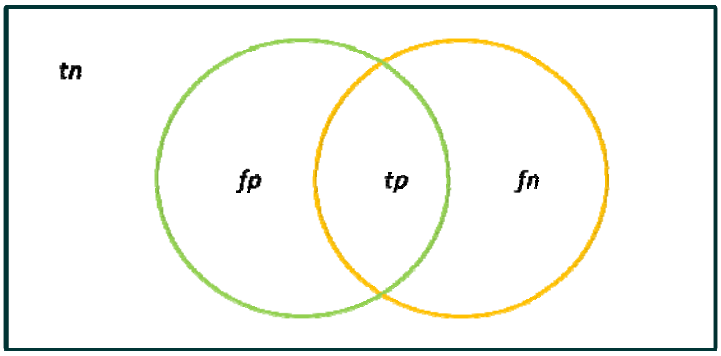

圖 8 分類結果示意圖

在本研究中，將 recall 與 specificity 作爲重要的評量準則，所以以下用更精確的方式定義。如圖 8 所示，二元分類器的分類結果，共有四種可能：真陽性（*true positive, tp*）、真陰性（*true negative, tn*）、僞陽性（*false positive, fp*）、僞陰性（*false negative, fn*）。再以醫療檢驗爲例，確實患了某個病，也被正確檢查出來的人，是真陽性（*tp*）；實際上沒有病，也正確地判斷爲無病的人，是真陰性（*tn*）；實際上有病，但被判斷爲無病的人，是僞陰性（*fn*）；實際上沒病，但被誤認爲有病的人，是僞陽性（*fp*）。在圖中，黃色的圈，是真正有病的；而綠色的圈，則是被檢驗中認爲有病的。這也可以用一個表格來表示：

表格 1 二元分類器分類結果

分類器判定	實際情況	
	positive	Negative
positive	tp	fp
negative	fn	tn

於是，利用這四種分類結果，重新定義 accuracy、recall、specificity 這三個評量指標：

$$\text{accuracy} = \frac{tp + tn}{tp + tn + fp + fn}$$

$$\text{recall} = \frac{tp}{tp + fn}$$

$$\text{specificity} = \frac{tn}{fp + tn}$$

在 shallow parsing、named-entity extraction、中文斷句等議題，recall 獲得廣泛使用，但是通常會與另一個指標 precision 一起出現，互相對照：

$$\text{precision} = \frac{tp}{tp + fp}$$

Precision 表示，被檢驗判定爲有病的人，其真正罹病的機率。類似 recall 與 specificity，recall 和 precision 也有互補的關係，所以這個準則經常同時考慮，如果 recall 和 precision 都高，則表該系統的效能不錯。也因爲經常將這兩個數字一起考慮，所以有時會將這兩個值，取 harmonic mean，合併爲單一量值 F-measure（或叫 F-score、F1）：

$$\text{F-measure} = \frac{2 \times \text{r} \times \text{p}}{\text{r} + \text{p}}$$

其中，r 爲 recall，p 爲 precision。

在句式邊界偵測的研究中，除了使用 F-measure、recall、precision 這三個

指標，也還會考慮 NIST-SU error rate。假設文本中共有 k 個句點，其中有 m 個是實際的句尾標示（該文本共有 m 個句子，且$m \leq k$），而句式邊界偵測系統判斷錯誤的次數是 w，則 NIST-SU error rate 定義爲：

$$NIST - SU\ error\ rate = \frac{w}{m}$$

錯誤的次數 w 越少，NIST-SU error rate 則越低，所以 NIST-SU 越低越好，最爲 0%。當系統將所有的句點一律判斷爲不斷句時，$w = m$，故 NIST-SU error rate 爲 100%。但若系統的錯誤更多，則$w \geq m$，此時 NIST-SU error rate 可能超過 100%。

由於我在本研究中，將斷句的問題，轉化爲 sequence labeling 的問題，所以應用在 labeling 或 tagging 上的準則，也可以應用在本研究中。典型的 tagging 問題，詞性標記，通常使用 *labeling accuracy* 這個指標。其實，labeling accuracy 和前面提到的 accuracy measure 有些類似。對斷句的問題來說，accuracy measure 計算的是字與字之間，正確地判斷出該斷或不該斷的機率；labeling accuracy 則是計算斷句模型以 sequence labeling 的方式處理 sequence 的過程中，sequence 上的每一個字，受到正確 labeling 的機率。由於 sequence labeling 是本研究處理斷句問題的切入方法，所以援用 labeling accuracy，從 labeling 的角度來衡量系統的效能，或許也有其意義。

綜合這一小節的論述，我提出了 accuracy、recall、specificity、precision、F-measure、NIST-SU error rate、labeling accuracy 等七種評量指標，並將用這七項指標來量化實驗結果，衡量斷句模型的效能。除此之外，本研究也會檢討這些指標，應用在古漢語斷句評估的效果，以歸納出最符合斷句需求，最貼近人類

對斷句品質認知的評估方法。

3.2 Datasets

3.2.1 語料選擇

古漢語（Classical Chinese）依據時代，細分爲上古漢語（Old Chinese、Archaic Chinese）、中古漢語（Middle Chinese）、近代漢語（Proto-Mandarin）。所謂上古漢語，包含了商朝到西漢數百年之間的漢語。這個時期的漢語文本，字句較短，結構單純，多以單字詞爲主。我以這些因素爲考量，而先鎖定上古漢語的文本，作爲古漢語斷句研究的處理對象。

在上古漢語中，《論語》、《孟子》、《莊子》、《春秋三傳》這些籍典具有代表性的地位，可以說是上古漢語的範本。《論語》記載孔子和弟子的語錄，內容全以對話的方式呈現。《孟子》結構類似《論文》，以對話爲主，但夾雜敘事，篇幅較《論語》爲長。《春秋三傳》是《左傳》、《公羊傳》、《穀梁傳》的合稱，《公羊傳》、《穀梁傳》以解釋《春秋》經文爲主，《左傳》則以記敘經文中相應的事件爲主 [38]。其中以《左傳》最爲重要，篇幅也最長。《左傳》分爲「經」與「傳」兩個部份，每一段春秋經文隨接一段「傳」，解釋經文的內容，「經」、「傳」循環接替。「經」的部份用字精省，按照「年・時節・月・日・事件」的體裁記事。而「傳」的部份沒有固定的格式，但敘事仍然相當簡練，文字純淨典雅。《莊子》以敘事爲主，由一篇篇寓言故事構成，語言奇麗，跌宕不羈，是中國文學上的奇葩，同時也是現今最受中外矚目的上古漢語作品，因此也很合適納入斷句研究的datasets。此外，《史記》是中國第一部紀傳體通史，描寫人物深刻，敘述事件精彩，把歷史寫得像故事一樣生動 [39]，是漢語文本中，寫人敘事的典範。同時篇幅既長，卷帙浩繁，共計一百三十篇，五十餘萬字，可以視爲一套鉅量的古漢

語語料庫，作爲斷句系統的 dataset，有很高的價值。

在上古漢語之外，我們和台灣大學資訊工程系數位典藏與自動推論實驗室合作，得到一批清代的奏摺 [40]。這批奏摺有一部份，目前已經過數位化和斷句標點的整理，但仍有一大部份尚未處理。所以，本研究也將這些已經斷句標點的奏摺文文，作爲 dataset，以實驗斷句系統面對不同時代、不同格式的文本，是否依然適用，並比較其中的異同，深入了解斷句模型的特性。並將探討如何針對清代奏摺調整斷句系統，以期實際應用在數位典藏的工作上。

3.2.2 資料蒐集與處理

我所選定的文本，除了清代奏摺之外，都能在中央研究院歷史語言研究所的〈漢籍電子文獻資料庫〉中，找到經過嚴謹處理，精工校對，並已加上標點符號的優良版本。[5] 這些文本雖然公開在網站上，但只能以小節或段爲單位，分段閱讀，不能直接下載全文。爲了快速的下載資料，我撰寫工具程式，擷取網站上的全文。但從網頁上抓取的資料原始檔，充滿了許多 HTML 標籤等雜訊，爲此，又撰寫了另一支程式清理。最後，我將清理好的文本，以原文的最小分隔單位「段」爲單位，以一段爲一筆資料，存入資料庫。

在原文中，「段」的長度有很大的出入。有的段非常短，如《孟子・盡心上》中的：

> 孟子曰・人不可以無恥・無恥之恥・無恥矣・

扣除斷句符號，只有十六個字。但同樣是《孟子》裡的段落，有的卻很長，如〈滕文公上〉的第四篇，扣除斷句符號之後，尚有 1117 個字。由於段落是古

[5] http://140.109.138.249/ihp/hanji.htm

文原典裡，最小的切分單位，所以爲了模擬斷句的真實情境，我也以原典的段落，作爲我的 dataset 資料單位，dataset 裡的每一筆 entry，就是原典上的一段。

《史記》分爲〈本紀〉、〈表〉、〈書〉、〈世家〉、〈列傳〉五個主題，由於其中〈表〉的部份，以表格的方式，排列歷史事件的次序，而沒有落段的結構。所以，我在建立 dataset 時，將〈表〉的內容省略。除此之外，其他先秦兩漢的文本，都是全文收錄。

古籍中有許多特殊的字型，超出 Big-5 字碼集。儘管我已經使用 Unicode 編碼（UTF-8 格式）來儲存資料，但仍有不少罕見字不在 Unicode 標準之中，在〈漢籍電子文獻資料庫〉中，也僅以圖片顯示，或甚至缺字。對於這樣的問題，由於少見字出現機會少，原不足以對統計式的斷句模型造成太大的影響，所以我以雜訊（noise）看待，直接忽略那些少見字，而不做額外補救。

除此之外，中央研究院歷史語言研究所的〈漢籍電子文獻資料庫〉追求嚴謹，對於脫字或有存疑的字，並不套用常見校本的選字，而用問號標記，或以括號夾註。這些額外的訊息，人類閱讀不成障礙，但對自動化斷句系統而言卻是干擾。爲此，我以人工的方式校對，找出所有存疑的文字，再參考其他數位化的版本[6]，以最常用的字取代。雖然所參考的版本，不如中研院版嚴謹可靠、有憑有據，但畢竟脫字與疑字也是少數的情況，即使參考的版本有差錯，也不易影響斷句系統的表現。

台灣大學資訊工程系數位典藏與自動推論實驗室所提供的清代奏摺語料，其中經過斷句標點的奏摺，計有 12,721 件，總字數超過 100 萬字，頗有份量。這批文件同樣也有脫字、疑字、夾註的情況，在此，我將有雜訊的奏摺濾除，得到 11,072 件完整無疑義的奏摺，再按照文件的格式，除去檔頭資訊，將一整篇奏

[6] 主要參考裴明龍所編之《錦繡中華之一頁》（http://www.chinapage.com）以及維基文庫（http://zh.wikisource.org）所收錄的文本。

摺依段落爲單位，一段一段個別儲存。最後，從中隨機取出 1000 個段落，共計 111,739 字，約佔所有的奏摺的十分之一，作爲清代奏摺的 dataset。

表格 2 Dataset 的統計資料

Dataset	年代	段落數	總字數	用字數	子句數	平均段落長數	平均子句長度
《論語》	戰國	500	15,982	1,368	4,015	31.964	3.981
《孟子》	戰國末期	260	35,392	1,916	7,351	136.123	4.815
《莊子》	戰國至西漢	1,128	65,165	2,936	12,574	57.770	5.183
《春秋左傳》	春秋至戰國	3,381	195,983	3,238	47,281	57.966	4.145
《春秋公羊傳》	戰國	1,804	44,352	1,638	11,151	24.585	3.977
《春秋穀梁傳》	戰國至西漢	1,801	40,711	1,585	10,946	22.605	3.719
《史記》	西漢	4,778	503,890	4,788	99,792	105.460	5.049
清代奏摺	清	1,000	111,739	3,147	15,521	111.739	7.199
上古漢語混合	先秦至西漢	1,250	97,476	3,489	20,573	77.981	4.738

本研究所採用的 dataset，整理於表格 2，並列出基本的統計資料。如表格 2 所示，本研究總共採用了 9 款 dataset。前 8 款 dataset 已經在前文介紹，而「上古漢語混合」是從《論語》、《孟子》、《莊子》、《春秋左傳》、《史記》五部上古漢語典籍中，各隨機取出 250 個段落，混合而成。

表中的「總字數」，是 dataset 全文，扣除標點符號和夾註等雜訊之後的字數總合。「用字數」則是曾在文本出現的漢字的種類數。「子句數」，就是文本經過斷句之後，所斷出來的「子句」和「短句」個數。從統計資料來看，隨著年代推進，平均子句長度確有增加。上古漢語的文本，平均四到五個字一斷，而清代奏摺則明顯增加到平均七字一斷。在字數方面，《史記》字數最多，超過 50 萬字，其次是《春秋左傳》，逼近 20 萬字。一般來說，training data 越大，涵蓋的字數越多，訓練出來的模型會有越好的效能。

3.3 古漢語斷句模型

3.3.1 序列標籤化方法

Xue [6] 以序列標籤化的方法，處理中文斷詞問題，啓發了我對古漢語斷句的靈感。在 Xue 的研究中，使用了四種標籤標記漢字在詞中的位置。LL 表示詞的首字，如「汽車」的「汽」字；RR 表示詞的尾字，如「汽車」的「車」字；MM 表示這個字在詞的中間，例如「教科書」中的「科」字；LR 則是單字詞，如「雨中的貓」中的「貓」字，既是一個詞的首字，也是詞的尾字。

表格 3 古漢語斷句標籤

標籤	說明	範例
LL	子句或短句的首字	「告子曰」的「告」字
RR	子句或短句的尾字	「告子曰」的「曰」字
MM	子句或短句中間的字	「告子曰」的「子」字
LR	構成單字短句的字	「性，猶鰉柳也」的「性」字

既然可以用標籤的方法，解決斷詞問題，那麼，把 LL、RR、MM、LR 這四個標籤的意義推廣，就可以用來解決漢語斷句的問題。所以，我重新定義了這四個標籤，並整理在表格 3。如果一個漢字被標爲 LL，則表示這個字是一個子句或短句的首字；標爲 RR 的字，則是子句或短句的尾字；MM 是在句子中間的所有字；LR 則表示這個字，單獨構成一個短句。

舉例來說，以下的段落，

告子曰，性，猶鰉柳也，義，猶桮棬也，

按照前述的方法，標記之後得到：

告/LL 子/MM 曰/RR 性/LR 猶/LL 鰥/MM 柳/MM 也/RR 義/LR 猶/LL 梧/MM 桮/MM 也/RR

每當 RR 或 LR 出現，就表示到了子句或短句的結尾，於是就可以在此斷句。由上例可以觀察到，LL 後面往往接著 MM，而 MM 之後，可能接 RR，或是再接一個 MM，但卻絕不可能接 LL 或 LR。根據這些性質，得到了古漢語斷句標籤的 Markov 鏈，如圖 9 所示。

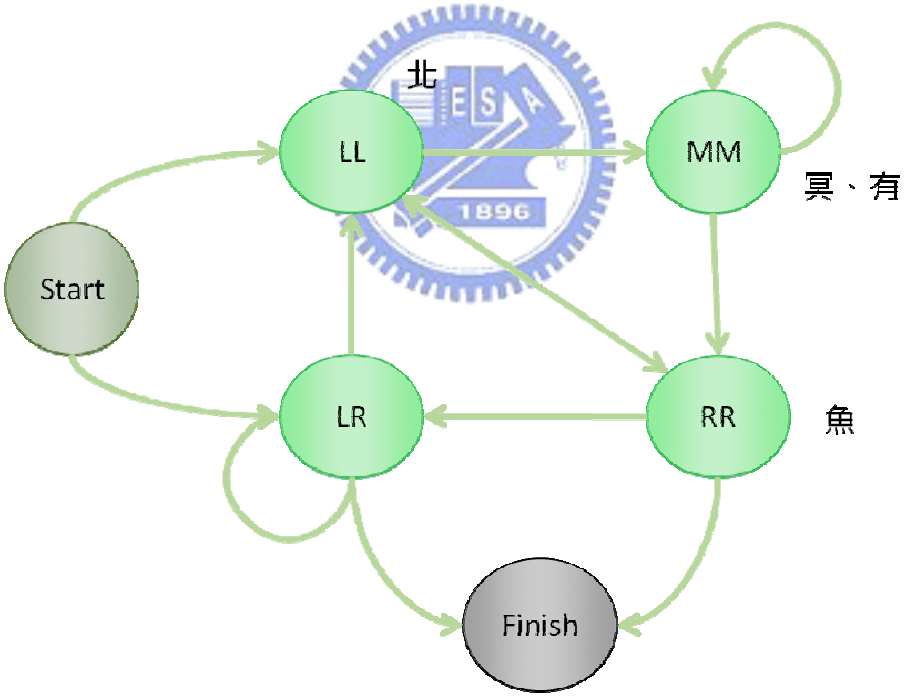

圖 9 中文斷句標籤的 Markov 鏈，以 北/LL 冥/MM 有/MM 魚/RR 爲例。

文本經過斷句之後，將其中每一個漢字加上斷句標籤是很容易的事情；將一串有斷句標籤的漢字，標上斷句的記號，也同樣是很簡單的事情。所以，只要經

過簡單的前後處理，就可以把斷句的問題，轉化爲序列標籤的問題，使用技術已經相當成熟的 sequence labeling 模型來解決。

3.3.2 斷句系統基本架構

根據我在第二章裡，對當前相關技術的探討，我預計採用 Markov model taggers 和 conditional random fields 這兩種 sequence labeling 的方法，作爲我的斷句模型。

這兩種方法，都是採用 supervised machine learning 的概念，從大量的預先標記好的資料中，訓練歸納出模型的參數，繼而以此模型去標記未知的資料。因此，斷句的模型，其實可以概分爲兩種階段：訓練階段（training stage）和上線階段（online stage）。

訓練階段比較複雜，又可以依序再細分爲資料準備、特徵粹取（feature extraction）、參數評估三個步驟。

所謂資料準備，是將原始的 dataset，轉變爲斷句模型比較容易操作的格式。舉例來說，原本 dataset 裡的項目，都是已經斷句完成的文本段落。而本研究要用 sequence labeling 的方法來處理斷句，所以要事先將斷句的資訊（即文本段落中的斷句符號），置換爲前一小節所述的 label 的資訊。除此之外，dataset 中的每一筆資料，都是以 UTF-8 編碼的中文字串。爲了增進效率，所以在資料準備時，我將每一個中文字編碼爲整數，此後便以整數表示中文字元。原本一個資料項目是由一列中文字串來表示，如今則變成一個整數陣列。

特徵粹取和參數評估這兩個步驟，就是所謂的訓練（training），也是本研究所核心的工作。而訓練的方法，則和所使用的 sequence labeling 模型有關。Markov model taggers 和 conditional random fields 在這兩個步驟有很大的不同，將下面

的小節中各別說明。

相較於訓練階段，上線階段的工作比較單純，可以概分爲資料準備和解碼（decoding）這兩個步驟。由於訓練的工作已經完成，上線階段不再需要 dataset，而以使用者輸入的資料作爲處理對象。使用者輸入的資料，預設是一段沒有斷句符號的古漢語字串，由於在運算時，一概以整數來表示中文字元，所以資料準備的工作，就是將輸入的中文字元，置換爲一列整數陣列，這個整數陣列，就是斷句模型的 observation sequence。

而所謂解碼，就是利用斷句模型，對 observation sequence 找出最符合的對應 label sequence，也就是對使用者所輸入古漢語字串中每一個漢字，標上斷句標籤（在此，漢字都已經轉成整數來表示）。

得到 label sequence 之後，還不能直接輸出，因爲使用者預期看到的 output，是經過斷句的古漢語字串。所以，此時要利用 label sequence 的資訊，將輸入的古漢語字串加上斷句符號。這個部份並不困難，所有被標上 LR 或 RR 的漢字，就是子句或短句的最後一個字，所以只要在這些漢字後面，插入斷句符號，就得到了斷句完成的古漢語字串。

3.3.3 Markov Model Tagger

Hidden Markov models 有相當多年的歷史，原理簡單，實作不難，訓練的速度很快，同時往往有不錯的表現。所以，在本研究裡，我把 Markov model taggers 當作 baseline，不方面作爲斷句研究的基礎方法，同時也和新的主流技術 conditional random fields 比較印證。

因爲 dataset 裡已經有預先標記好的 training data，Markov model tagger 不必透過迭代法訓練，直接從 training data 中統計，就可以得到 model 所須要的

所有的特徵和參數 $\lambda = (A, B, \Pi)$。

解碼的部份，使用傳統的 Viterbi 演算法。Viterbi 是 *dynamic programming* 演算法，在很有效率的時間內，找出最符合 observation sequence 的 label sequence。但此時得到的 sequence，是以斷句標籤組成的串列，必須再經過一次處理，將標籤轉爲斷好句的文本，才能輸出。

3.3.4 Conditional Random Fields

儘管 conditional random fields 在理論上有相當優秀的條件，但要有效率地實作，並充份發揮其威力，卻不是容易的事情。而決定其威力和效率的關鍵，在於特徵粹取和參數評估這兩個環節。

表格 4 搭配 conditional random fields 使用的特徵模版

樣版	說明
y_{i-1}, y_i	前一個 label 與目前的 label
x_i, y_i	目前的字
$\mathrm{x_{i-2}}, \mathrm{y_i}$	上兩個字
$\mathrm{x_{i-1}}, \mathrm{y_i}$	上一個字
$\mathrm{x_{i+1}}, \mathrm{y_i}$	下一個字
$\mathrm{x_{i+2}}, \mathrm{y_i}$	下兩個字
x_{i-2}, x_{i-1}, y_i	過去的兩個字
x_{i-1}, x_i, y_i	上一個字和目前的字
x_i, x_{i+1}, y_i	目前的字和下一個字
x_{i+1}, x_{i+2}, y_i	未來的兩個字

Conditional random fields 提供了非常自由的特徵函數介面，讓用家各憑需求和專業領域知識去設計特徵函數。徵特函數或者由人類手動編排，或者從 training data 中自動化吸收，也可兩種方法混合，截取 rule-based 和 statistical

雙方之長。在本研究的範圍裡，並不打算使用人工制訂的方式來設計特徵函數，而以完全以 empiricist 的方法，直接從 training data 中粹取。由於特徵函數的介面有很大的自由度，所以我設計 10 種特徵模版（feature templates），在 training data 中找尋相符的 patterns，再從中取出現次數較多，較爲顯著的，作爲斷句模型的特徵函數。我所使用的 10 種特徵模版，羅列於表格 4。

傳統的 conditional random fields，以 maximum log-likelihood estimation 來評估參數，目前實作的主流是 quasi-Newton (BFGS)法。但是，在本研究中，我改以 Collins [32] 提出的 averaged perceptron 作參數評估。相較於傳統的訓練方法，averaged perceptron 流程簡潔，效率也高，只要迭代幾個回合，就能找到不錯的參數組合。唯一的缺點是，averaged perceptron 不能如其他 maximum log-likelihood estimation 方法，保證收斂到 global maximum。即使如此，averaged perceptron 的效能已經非常接近目前最佳的評估方法 limited-memory quasi-Newton (L-BFGS) [28], [41]。權衡訓練的效率，averaged perceptron 的缺點是可以接受的。Averaged perceptron 的原理和作法，已經在第二章介紹，本研究中的實作演算法，如圖 10。

解碼的部份，conditional random fields 和 Markov model tagger 幾乎完全一樣，先用 Viterbi 演算法，找出最佳的 label sequence $\hat{y} = \arg\max_{y} P_{\lambda}(y|x)$，再將此 label sequence 轉爲斷句後的文本即可 [8], [22], [23]。

Inputs:

The training data (x^i, y^i) for $i = 1, 2, \ldots n$.

A parameter T being the number of iterations.

The feature functions $f = \{f_1, f_2, \ldots f_m\}$

Initialization:

Set initial parameters $\alpha = \{\alpha_1, \alpha_2, \ldots \alpha_m\}$ to 0

Procedure:

for $t = 1 \ldots T$ **do**

for $i = 1 \ldots n$ **do**

$z = \arg\max_v P_\alpha(v|x^i)$

if $z \neq y^i$ **then**

for $k = 1 \ldots m$ **do**

$\alpha_k = \alpha_k + F_k(y^i, x^i) - F_k(z, x^i)$

$\gamma_k^{t,i} = a_k$

for $k = 1 \ldots m$ **do**

$\lambda_k = \sum_{t=1 \ldots T, i=1 \ldots n} \frac{\gamma_k^{t,i}}{nT}$

Return:

The parameters λ

圖 10 Averaged Perceptron 學習演算法

四、 實驗

4.1 實驗設計

本研究的實驗分爲三個部份，首先檢驗斷句模型的效能，其次是評比上古漢語的 datasets，作爲 training data 的成效，最後探討 training data 是否具有跨越時代的通用性。

在本研究中，採用了兩種序列標記模型，hidden Markov models 與 conditional random fields，在實驗一中，我將針這兩種模型設定參數，以第三章所介紹的九款 dataset 作訓練和驗證，比較 hidden Markov models 與 conditional random fields 的效能，進而探討這兩個 model 應用在古漢語斷句問題中的特性。

接下來，我將在實驗二中，試驗《論語》、《孟子》、《莊子》、《左傳》、《史記》等五種 dataset，作爲 training data 的效果。驗證的方法是，每次以一種 dataset 作 training data，練訓 hidden Markov model 以及 conditional random fields 兩個模型，再以這兩個 models 對其他四個 dataset 作斷句。最後，可以整理出每一個 dataset 作爲 training data 的效能，而後再分析比較。

最後，在實驗三裡，我所要了解是，以上古漢語的 datasets 作爲 training data，所訓練的模型，是否能適用在清代奏摺的斷句工作上。相對的，我也想了解，以清代奏摺作爲 training data，是否能讓斷句模型斷好上古漢語的 datasets。從上古漢語到近代漢語，歷經 2000 年，語文有相當的演化。如果 training data 和 test data 是兩個不同時空的產物，卻還能有良好的斷句效能，即表示句讀背後隱含的規則和邏輯，並不太受漢語演化的影響。反之，則表示斷句模型以某時代的 dataset 作訓練，即有時代的侷限性，只能處理該時代的文本，而無法適用在時空差距太大的資料上。

本研究所有的實驗，都在 Linux/Debian 環境下測試。系統 CPU 爲 Pentium 3.4GHz，主記憶體 1G。所有的 dataset 都以 Unicode（UTF-8 編碼）格式，以 MySQL 5.0.22 資料庫系統儲存。考量資料庫存取、Unicode 字串處理、實驗結果呈現等需求，主程式以 PHP 語言實作。然而，PHP 語言處理大量數值運算的效能並不理想，所以計算量龐大的 conditional random fields，改以 C++語言實作。而 hidden Markov models 相較之下，計算量並不大，即使以 PHP 語言實作，大多數的訓練和標記任務，都能在 10 分鐘之內完成。

4.2 實驗一：斷句模型效能

4.2.1 實驗方法

在本實驗中，將依序測試 hidden Markov models 與 conditional random fields 用於古漢語斷句的效能。在此，採用全部的 dataset，包含《論語》、《孟子》、《莊子》、《春秋左傳》、《春秋公羊傳》、《春秋穀梁傳》、《史記》、清代奏摺、上古漢語混合等九種。測試和驗證使用 10-fold *cross-validation*，亦即，測試一項 dataset 時，先將該 dataset 隨機均分爲 10 份 data，一次取一份作爲 test data，而其餘九份作爲 training data，以此訓練和測試，並重複這個過程，使得 10 份 data 都輪流測試過。

Conditional random fields 和 average perceptron 演算法有幾項參數，必須在實驗之前給定。其一是特徵門檻（feature count cut-offs），其二是 perceptron 演算法的訓練回合數。

隨著文本長度和複雜度的增加，從 dataset 中找出來的特徵函數可能相當多，數量龐大。理論上，特徵函數越多，conditional random fields 的效能越好，然而卻也必須付出龐大的練訓時間。爲此，如果在訓練之前，濾除罕見而較不顯著

的特徵，則可以減少練訓的時間。由於特徵的頻率分布，近接 *Zipf's law*：一小部份重要的特徵，有相當高的出現頻率，而剩下的眾多特徵，出現頻率總合只佔一小部份。換句話說，一小部份重要的特徵函數，就足以掌握絕大多數的影響力。因此，只要考慮顯著的特徵函數，將其餘不重要的特徵函數濾除，對 conditional random fields 的效能影響不大，然而卻可以大幅縮減訓練時間 [32]。過濾特徵函數的方法，通常是設定一個門檻值（feature count cut-offs），特徵在 dataset 中出現的次數，必須大於等於這個門檻，才將其加入模型的特徵函數；出現次數低於門檻值的特徵，則直接捨去。

在本實驗中，由於 dataset 的長度差異頗大，最長的《史記》有 50 餘萬字，最短的《論語》只有一萬多字，字數相去數十倍，特徵數量和其出現次數也會有極大的差距。如果採用單一固定的特徵門檻，對於較短的 dataset，則將濾除太多特徵，而對於較長的 dataset，卻可能又濾得不夠多。對此，在本實驗中，不採用固定的特徵門檻，改爲固定特徵的總數，無論總共有多少特徵、出現頻如何，只選用前 100,000 個出現頻率最高的特徵。如果前 100,000 個特徵有平手的情況，例如，第 99,900 名到第 100,020 名的特徵，其出現次數都是 5 次，則將這 100,020 個特徵全部加入特徵函數。因此，實際的特徵函數總量，往往稍大於 100,000。

Averaged perceptron 的練訓回合數，就是圖 10 中的參數 T。這個參數，關係到訓練完成之後，模型的效能，也同時影響訓練的時間。根據 Collins [32]和 Sha et al. [28]的實驗，T 越大，並不一定會有更好的效能，反而可能形成 over-training，導致反效果。相反的，Sha et al. 在實驗中發現，只要兩個回合的練訓，就可以有很好的效果；而 Collins 的實驗則表示，在 averaged perceptron 的情況下，只要 10 個回合左右的訓練，就達到了最佳情況。此外，根據個人之前的經驗，對古漢語斷句的問題而言，T 在 3 到 10 之間，所訓練出來的模型，就已經逼近的收斂，而且相當穩定，並不會隨著 T 的不同，而有太大的落差。因

此，在本實驗中，設定 $T = 5$。

4.2.2 實驗結果與分析

Hidden Markov models 的斷句效能如表格 5，而表格 6 則列出各項效能指標的標準差。Conditional random fields 斷句效能列於表格 7，各項效能的標準差在表格 8。圖 11 是這兩個斷句模型的 recall-specificity ROC Curve 比較。

表格 5 Hidden Markov Models 斷句效能

Dataset	Accuracy	Precision	Recall	Specificity	F-Measure	NIST –SU Error Rate	Labeling Accuracy
《論語》	87.21%	78.25%	71.99%	93.01%	73.84%	49.84%	76.92%
《孟子》	88.63%	76.57%	67.08%	94.49%	70.86%	54.35%	78.79%
《莊子》	88.69%	73.47%	69.64%	93.72%	70.48%	57.25%	78.76%
《春秋左傳》	91.63%	84.42%	84.52%	94.57%	83.55%	33.70%	84.66%
《春秋公羊傳》	94.07%	89.16%	89.72%	95.78%	88.52%	24.22%	88.89%
《春秋穀梁傳》	93.23%	88.15%	87.91%	95.37%	86.92%	27.10%	87.20%
《史記》	84.93%	64.64%	59.48%	91.68%	60.87%	75.60%	72.28%
清代奏摺	92.70%	77.94%	72.27%	96.70%	73.19%	50.68%	86.65%
上古漢語混合	87.03%	73.41%	67.55%	93.00%	69.30%	59.05%	76.22%
平均	89.79%	78.45%	74.46%	94.26%	75.28%	47.98%	81.15%

表格 6 Hidden Markov Models 斷句效能標準差

Dataset	Accuracy S.D.	Precision S.D.	Recall S.D.	Specificity S.D.	F-Measure S.D.	NIST –SU Error Rate S.D.	Labeling Accuracy S.D
《論語》	8.99%	18.11%	19.88%	6.23%	17.19%	32.55%	15.05%
《孟子》	5.29%	14.73%	15.60%	3.63%	13.88%	25.87%	9.63%
《莊子》	7.51%	19.86%	21.71%	5.36%	19.29%	37.74%	13.87%
《春秋左傳》	8.39%	17.12%	17.95%	6.66%	16.07%	34.64%	14.99%

《春秋公羊傳》	7.45%	16.28%	15.82%	7.13%	14.75%	33.05%	13.84%
《春秋穀梁傳》	7.69%	16.58%	17.27%	7.23%	15.39%	33.82%	14.34%
《史記》	6.88%	18.51%	18.95%	4.66%	17.05%	35.41%	12.15%
清代奏摺	6.71%	21.32%	25.62%	3.91%	22.04%	41.92%	11.34%
上古漢語混合	8.22%	20.14%	20.96%	5.72%	18.98%	37.63%	14.41%
平均	7.46%	18.07%	19.31%	5.61%	17.18%	34.74%	13.29%

表格 7 Conditional Random Fields 斷句效能

Dataset	Accuracy	Precision	Recall	Specificity	F-Measure	NIST –SU Error Rate	Labeling Accuracy
《論語》	89.02%	80.16%	78.29%	93.23%	78.52%	42.63%	80.71%
《孟子》	89.01%	72.71%	78.32%	91.95%	75.08%	52.90%	80.48%
《莊子》	90.62%	76.73%	77.30%	94.13%	76.37%	48.20%	82.88%
《春秋左傳》	93.83%	86.87%	91.02%	95.16%	88.25%	25.60%	88.83%
《春秋公羊傳》	96.78%	93.92%	94.29%	97.83%	93.60%	13.53%	93.96%
《春秋穀梁傳》	95.82%	92.73%	92.65%	97.23%	92.12%	16.19%	92.25%
《史記》	88.58%	70.85%	75.82%	92.02%	72.69%	Ā8.02%	79.41%
清代奏摺	94.57%	88.92%	73.63%	98.68%	78.54%	35.24%	90.20%
上古漢語混合	87.49%	70.82%	78.00%	90.56%	73.44%	58.93%	77.78%
平均	91.75%	81.52%	82.15%	94.53%	80.96%	39.03%	85.17%

表格 8 Conditional Random Fields 斷句效能標準差

Dataset	Accuracy S.D.	Precision S.D.	Recall S.D.	Specificity S.D.	F-Measure S.D.	NIST –SU Error Rate S.D.	Labeling Accuracy S.D
《論語》	9.15%	17.89%	18.36%	6.41%	16.86%	33.85%	15.22%
《孟子》	6.19%	15.40%	13.60%	4.76%	13.84%	30.71%	10.82%
《莊子》	6.99%	18.58%	18.30%	4.98%	17.24%	36.17%	12.49%
《春秋左傳》	7.21%	16.01%	13.01%	6.38%	13.48%	31.23%	12.78%
《春秋公羊傳》	5.47%	12.41%	11.56%	4.79%	11.06%	25.01%	10.23%
《春秋穀梁傳》	6.40%	13.02%	12.99%	5.43%	11.99%	25.47%	11.69%

《史記》	6.67%	16.57%	15.68%	4.84%	15.05%	36.61%	11.60%
清代奏摺	6.12%	15.29%	26.49%	2.23%	21.50%	34.13%	10.14%
上古漢語混合	8.74%	20.67%	17.83%	7.01%	18.16%	43.20%	15.03%
平均	6.99%	16.20%	16.42%	5.20%	15.46%	32.93%	12.22%

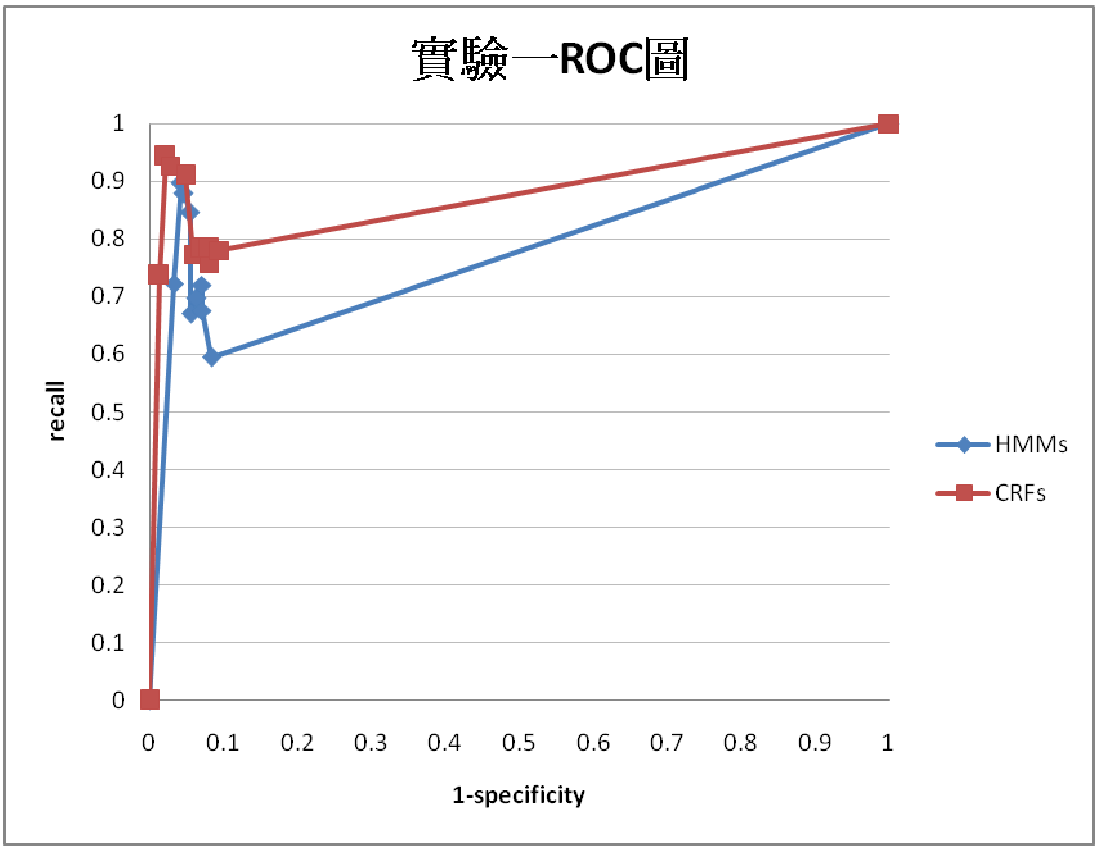

圖 11 Hidden Markov Models 與 Conditional Random Fields 的 ROC Curve 比較

觀察表格 5 和表格 7 的實驗數據，conditional random fields 的斷句能力明顯優於 hidden Markov models。除了 specificity 平均來說低了 0.5 個百分點之外，conditional random fields 所有的指標都比 hidden Markov models 好，特別是 recall，高出了 6.65 個百分點，差距相當明顯。雖然 conditional random fields 的 specificity 的平均值較低，但分項來看，除了上古漢語混合這個 dataset，此

外表現並沒有差 hidden Markov models 太多。甚且，面對《春秋公羊傳》、《春秋穀梁傳》、清代奏摺這三個 dataset，更到達了 97%-99%。

再看表格 6 和表格 8，兩個模型斷句的各項效能，標準差都相當接近。然而，無論是哪一項指數，conditional random fields 的標準差都以微幅差距低於 hidden Markov models。這個結果表示，conditional random fields 的斷句表現，比 hidden Markov models 來的穩定，面對各式各樣的 input 資料，表現比較一致，起伏不定的幅度較小。不過，由於兩個模型在各項評估指標的標準差都很相似，所以兩者的穩定程度，其實差別不大。

圖 11 中的 ROC curve，以視覺化的方式呈現兩個模型的表現。Conditional random fields 的 recall 指標遠勝過 hidden Markov models，所以在 recall-specificity ROC curve 上，呈現明顯的優勢。

觀察實際的斷句結果，《莊子．逍遙遊》的開頭第一段：

> 北冥有魚．其名為鯤．鯤之大．不知其幾千里也．化而為鳥．其名為鵬．鵬之背．不知其幾千里也．怒而飛．其翼若垂天之雲．是鳥也．海運則將徙於南冥．南冥者．天池也．

經過 hidden Markov models 斷句之後，成為：

> 北冥有魚其名為鯤·鯤之大不知其幾千里也·化而為鳥其名為鵬·鵬之背．不知其幾千里也·怒而飛其翼·若垂天之雲·是鳥也·海運則將徙於南冥．南冥者．天池也．

而 conditional random fields 則將這段斷為：

北冥有魚，其名為鯤鯤之大，不知其幾千里也，化而為鳥，其名為鵬鵬之背，不知其幾千里也，怒而飛，其翼若垂天之雲，是鳥也海，運則將徙於南冥南冥者，天池也，

在這個例子中，兩個模型的斷句效能在量化評估指標上幾乎完全相同（$F\ measure = 80.00\%, NSIT - SU = 35.71\%$），但斷錯的方式截然不同。《莊子》的另一個段落「連叔曰，其言謂何哉，」，其中「曰」和「其」是非常典型的邊界字元，「曰」經常用在 RR 的位置，「其」則常出現在 LL 的位置，因此兩個模型的斷句結果都完全正確。Hidden Markov models 和 conditional random fields 都是機率模型，對於「也」、「曰」、「其」這些經常出現在句首或句尾的漢字很少斷錯，但遇到少見的漢字組合，斷句的行爲則難以預料，這也是以機率爲核心的分類器比較難突破的部份。再看另一個同出於《莊子》的段落：

子綦曰，夫大塊噫氣，其名為風，是唯无作，作則萬竅怒呺，而獨不聞之翏翏乎，山林之畏佳，大木百圍之竅穴，似鼻，似口，似耳，似枅，似圈，似臼，似洼者，似污者，激者，謞者，叱者，吸者，叫者，謼者，宎者，咬者，前者唱于而隨者唱喁，泠風則小和，飄風則大和，厲風濟則红竅為虛，而獨不見之調調，之刁刁乎，

這個段落比較複雜，句型和用字都不太規則，hidden Markov models 將這個段落斷爲：

子綦曰，夫大塊噫氣，其名為風是，唯无作作，則萬竅怒呺而獨不聞之翏翏乎山，林之畏佳大木百圍之竅，穴似鼻似口似耳，似枅似圈似臼似洼者，

似污者・激者謞 者叱者・吸者叫者譹者实者咬者・前者唱于・而隨者唱喁泠風・則小和飄風・則大和・厲風濟則红竅為虛・而獨不見之調調之刁刁乎・

而 conditional random fields 則斷作：

子綦曰・夫大塊噫氣・其名為風・是唯无作作・則萬竅怒呺・而獨不聞之・翏翏乎山林之畏佳・大木百圍之竅・穴似鼻似口似耳・似枅似圈似・臼似洼者・似污者・激者謞者叱者・吸者叫者・譹者实者・咬者前者・唱于而隨者・唱喁泠風・則小和飄風・則大和厲・風濟則红竅為虛・而獨不見之・調調之・刁刁乎・

兩個模型對於這個段落，表現都很不理想。特別是「似鼻・似口・似耳・似枅・似圈・似臼・似洼者・ 似污者・激者・謞者・叱者・吸者・叫者・譹者・实者・咬者・」這幾個句子，hidden Markov models 幾乎不作任何斷句，而 conditional random fields 則以兩個爲一組作斷句，成果同樣不佳。

以《春秋公羊傳》爲 dataset 時，hidden Markov models 與 conditional random fields 都有最佳的效能表現。觀察實際的斷句結果，兩個模型對《春秋公羊傳》中多數的段落都有一致的表現，而且都能斷出不錯的結果。只有少部份有些差異，以其中某一段爲例：

天王使仍叔之子來聘・仍叔之子者何・天子之大夫也・其稱仍叔之子何・譏・何譏爾・譏父老・子代從政也・

在此，hidden Markov models 的表現差強人意：

> 天王使仍．叔之子來聘．仍叔之子者何．天子之大夫也．其稱仍．叔之子何．譏．何譏爾．譏父老子代從政也．

而 conditional random fields 則有幾乎完全正確的表現：

> 天王使仍叔之子來聘．仍叔之子者何．天子之大夫也．其稱仍叔之子何．譏．何譏爾．譏父老子代從政也．

再看另一個同樣出於《春秋公羊傳》，但是較長也較複雜的段落：

> 秋．七月葬鄭莊公．九月．宋人執鄭祭仲．祭仲者何．鄭相也．何以不名．賢也．何賢乎祭仲．以為知權也．其為知權奈何．古者鄭國處于留．先鄭伯有善于鄔．公者．通乎夫人．以取其國而遷鄭焉．而野留．莊公死已葬．祭仲將往省于留．塗出于宋．宋人執之．謂之曰．為我出忽而立突．祭仲不從其言．則君必死．國必亡．從其言．則君可以生易死．國可以存易亡．少遼緩之．則突可故出．而忽可故反．是不可得則病．然後有鄭國．古人之有權者．祭仲之權是也．權者何．權者反於 經．然後有善者也．權之所設．舍死亡無所設．行權有道．自貶損以行權．不害人以行權．殺人以自生．亡人以自存．君子不為也．

在此，hidden Markov models 產生了荒腔走板的斷句結果：

> 秋．七月．葬鄭．莊公．九月．宋人執鄭．祭仲祭仲者何．鄭相也．何以

不名・賢也・何賢乎祭仲以為知權也・其為知權奈何・古者・鄭國處于留先鄭伯有善于鄧公者・通乎夫人以取其國而遷鄭焉而野留莊公死已・葬祭仲將往省于留塗出于宋・宋人執之謂之曰・為我出忽而立突祭仲不從其言則君必死國必亡從其言則君可以生・ 易死國可以存易亡少遼・緩之則突可・故出而忽可・故反是不可得則病然後有鄭國・古人之有權者祭仲之權是也・權者・何權者反・於經然後有善者也・權之所設舍死亡無所設行權有道自貶損以行權不害人以行權殺人以自生亡人以自存君子不為也・

特別是段落的最後幾句，hidden Markov models 放過了許多理應斷開的地方，留下了接連 41 字而無任何斷句的長句子。相形之下，conditional random fields 的斷句成果則好上一截：

秋・七月・葬鄭莊公九月・宋人執鄭祭仲祭仲者何・鄭相也・何以不名・賢也・何賢乎祭仲以為知權也・其為知權奈何・古者・鄭國處于留先・鄭伯有善于鄧・公者 通乎夫人・以取其國・而遷鄭焉而野留・莊公死・已葬祭仲將往省于留・塗出于宋・宋人執之謂之曰・為我出忽而立突・祭仲不從其言・則君必死・國必亡從其言・ 則君可以生易死・國可以存易亡少・遼緩之・則突可故出而忽可・故反是不可得・則病然後有鄭國・古人之有權者・祭仲之權是也・權者何・權者反於經・然後有善 者也・權之所設舍死・亡無所設行・權有道自貶・損以行權不害人・以行權・殺人以自生・亡人以自存・君子不為也・

除了上古漢語的文本，這兩個模型對於清代奏摺也有不錯的表現。對於比較短的段落，例如「道光十六年二月十四日奉硃批・刑部議奏・欽此・」或「硃批・欣慰覽之・」兩個模型的斷句結果幾乎都完全正確。然而，對於比較長而文句複

雜的段落，conditional random fields 則稍佔優勢。例如這個段落：

竊福建台灣水師協標中營都司黎振湘升任遺缺·先因台灣酌改營制案內·改為台灣鎮標左營陸路都司·經臣璟會同前署福建撫臣葆亨恭摺奏准·以軍功留閩儘先補用·陸路都司徐正邦補授·茲台灣新改各營·業經奏准·仍復舊制·將徐振邦撤歸原班另補·所有原設台灣水師協標中營都司員缺·係台灣水師題補之缺·准到兵部咨·輪用儘先人員·行令照章請補·等因·

其實 hidden Markov models 的斷句結果並不太差：

竊福建台灣水師協標中營都司黎振·湘升任遺缺·先因台灣酌改營制案·內改為台灣鎮標左營陸路都司經臣璟會同·前署福建撫臣葆亨·恭摺奏·准以軍功留閩儘先補用·陸路都司徐正邦補授·茲台灣新改各營·業經奏·准仍復舊制·將徐振邦撤歸原班另補·所有原設台灣水師協標中營都司員缺·係台灣水師題補之缺·准到兵部咨輪用·儘先人員行令照章請補等因·

然而 conditional random fields 卻往往能把較長、較爲複雜的段落斷得更好：

竊福建台灣水師協標中營都司黎振湘升任遺缺·先因台灣酌改營制案內·改為台灣鎮標左營陸路都司經臣璟會同前署福建撫臣葆亨恭摺奏准·以軍功留閩儘先補用陸路都司徐正邦補授·茲台灣新改各營·業經奏准·仍復舊制·將徐振邦撤歸原班另補·所有原設台灣水師協標中營都司員缺係台灣水師題補之缺·准到兵部咨輪用儘先人員行令照章請補·等因·

透過實際觀察斷句結果得知，對於超過半數的文字段落，hidden Markov

models 和 conditional random fields 這兩個斷句模型有相近的表現。Hidden Markov models 斷的好的，conditional random fields 通常也能斷的好。然而，面對比較複雜的文句段落，conditional random fields 往往還能維持某個水準以上，差強人意的表現，而 hidden Markov models 則比較不穩定，動輒有荒腔走板的演出。這樣的觀察結果，也吻合表格 5 到表格 8 的統計數據：平均而言 conditional random fields 有較佳而且較爲穩定的表現。

再觀察 dataset 對斷句模型的影響，同樣可以發現，兩個斷句模型有相當高的一致性。這兩個斷句模型，對於《春秋三傳》和有最好的表現，對《史記》的表現最差，但總體來說，conditional random fields 幾乎在所有的 dataset 上，效能都勝於 hidden Markov models。有趣的是，上古漢語混合是由《論語》、《孟子》、《莊子》、《左傳》、《史記》等五個 dataset 混合而來，而 conditional random fields 個別面對這五個 dataset 時，效能都明顯勝於 hidden Markov models（僅面對《孟子》時，幾個指標微幅落後，而對其他 dataset 都超前），爲何當五個 dataset 混合之後，conditional random fields 效能卻降低許多（特別是 specificity），表現接近於 hidden Markov models 呢？

上古漢語混合是由五個 dataset 中，各隨機抽取 250 個段落，混合而來。而篇幅較長的文本《史記》和《左傳》，各有三四千個段落，從中只取出 250 個，就比例來說太少，資料的代表性不夠。而在這樣的條件下，conditional random fields 的表現受到影響，而 hidden Markov models 或許對此一情況有較佳的免疫力。

本研究針對上古漢語爲設計目標，本來並不預期應用在清代奏摺，會有理想的斷句表現。然而，實驗的結果卻顯示，同樣的斷句模型，應用在不同時代的文本，仍然有一致的表現。兩個模型面對清代奏摺，都有不錯的表現，尤其 conditional random fields 有亮眼的數據。這樣的結果，代表本研究所設計的斷

句模型具有一般性，對於句子更長，雙字詞、多字詞大量出現的近代漢語，也能夠適用，而且效能並不遜色。

4.3 實驗二：Training Data 評比

4.3.1 實驗方法

在實驗二中，將以 dataset 爲主角，試驗這九款 datasets，作爲 training data 的效能。測試與驗證的方法，是每次以一個 dataset 作爲 training data，分別以 hidden Markov models 和 conditional random fields 兩種模型訓練，再以其餘八款 dataset 作 test data，驗證斷句的效果。藉此，可以觀察該 training data 是否有足夠的代表性和一般性，使得訓練出來的模型，面對不同時代、不同作者、不同體裁的文本，也能準確斷句。如果，某個 dataset 所訓練出來模型，對諸多文本都有很好的效能，則將來實際應用古漢語斷句的系統時，這個 dataset 就非常合適作爲最終的 training data。

然而，上古漢語混合這個 dataset，由《論語》、《孟子》、《莊子》、《左傳》、《史記》等五個 dataset 組合而成，內容有重疊之處，拿來互相作訓練和測試，就犯了 training data 與 test data 重覆的問題。所以，上古漢語混合只與清代奏摺互相驗證，而不涉入其他上古漢語文本的訓練和測試。

本實驗中，conditional random fields 的兩項參數，依循實驗一的設定,。

4.3.2 實驗結果與分析

經過驗實發現，《春秋三傳》的同質性相當高，無論用於 training data 或是 test data，三者的表現幾乎一致。爲了便於呈現和分析，我僅取篇幅最長，最重

要的《左傳》作爲三傳的代表，而略去《公羊傳》和《穀梁傳》。表格 9 和表格 10，列出在 hidden Markov model 和 conditional random fields 這兩種模型下，以《論語》、《孟子》、《莊子》、《左傳》、《史記》這五個 dataset，分別作爲 training data，而以其餘四個 dataset 爲 test data 的平均效能表現。

表格 9 以各 dataset 作爲 training data，訓練 hidden Markov models 的平均斷句效能

Training Data	Accuracy	Precision	Recall	Specificity	F-Measure	NIST –SU Error Rate	Labeling Accuracy
《論語》	82.18%	62.36%	56.12%	90.36%	57.61%	80.96%	67.48%
《孟子》	82.93%	68.56%	54.52%	92.57%	59.10%	72.99%	68.49%
《莊子》	83.45%	69.95%	55.76%	92.77%	60.68%	70.50%	69.56%
《春秋左傳》	84.47%	64.73%	64.26%	90.32%	63.57%	73.77%	71.62%
《史記》	86.31%	75.56%	65.48%	93.56%	68.86%	57.97%	74.46%
平均	83.87%	68.23%	59.23%	91.92%	61.96%	71.24%	70.32%

表格 10 以各 dataset 爲 training data，訓練 conditional random fields 的平均斷句效能

Training Data	Accuracy	Precision	Recall	Specificity	F-Measure	NIST –SU Error Rate	Labeling Accuracy
《論語》	80.75%	57.95%	62.12%	86.80%	58.93%	88.16%	66.26%
《孟子》	82.92%	64.91%	65.27%	89.12%	63.90%	75.12%	69.81%
《莊子》	84.00%	68.61%	63.80%	91.00%	65.08%	68.97%	71.33%
《春秋左傳》	84.21%	62.56%	73.84%	87.37%	67.00%	75.93%	72.12%
《史記》	87.93%	76.93%	73.46%	93.23%	74.11%	51.19%	77.96%
平均	83.96%	66.19%	67.70%	89.50%	65.80%	71.87%	71.50%

比較這兩張表格，可以觀察到，除了《史記》之外，兩個模型的表現相當接近。然而，以《史記》去訓練的 conditional random fields，卻是所有組合中表現最佳的。與實驗一相較，conditional random fields 在大多數的 dataset 上並沒有優勢，和 hidden Markov models 表現當相。個人推測，這和實驗一當中，

conditional random fields 面對「上古漢語混合」表現較差，可能有類似的原因。如果 training data 數量不夠多，內容不夠一般性，訓練出來的 conditional random fields 面對變化多、複雜度高的 test data 時，效能就會明顯下降。而 hidden Markov models 對於這個情況，較具免疫力，不致於受到太大的影響。

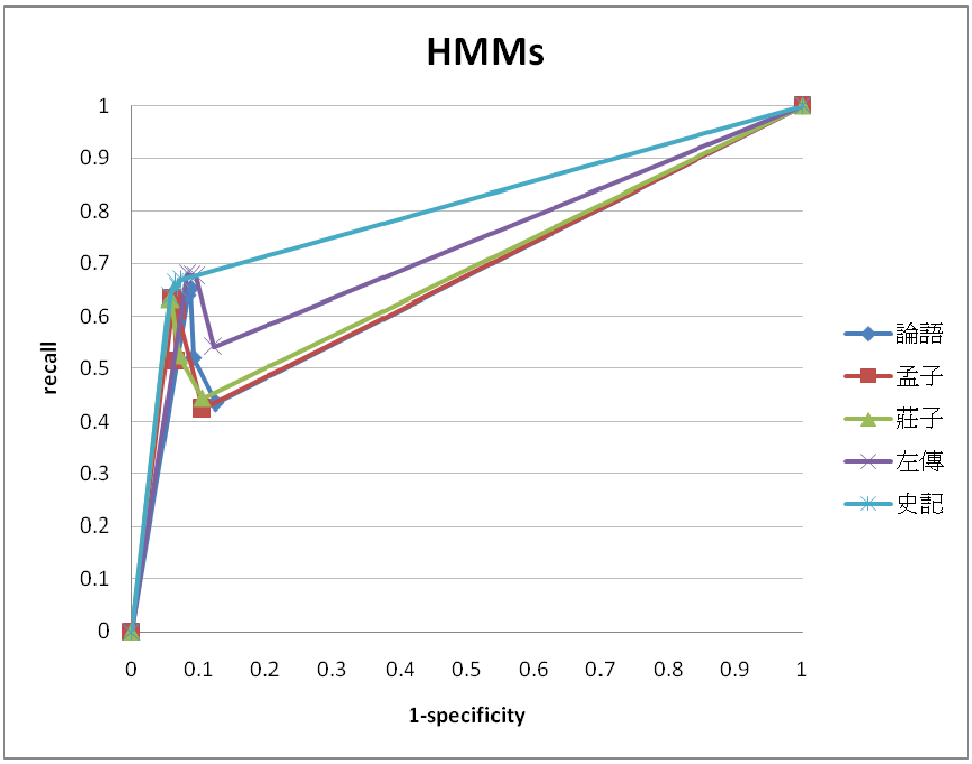

圖 12 五種 training data 之斷句效能比較，使用 hidden Markov models。

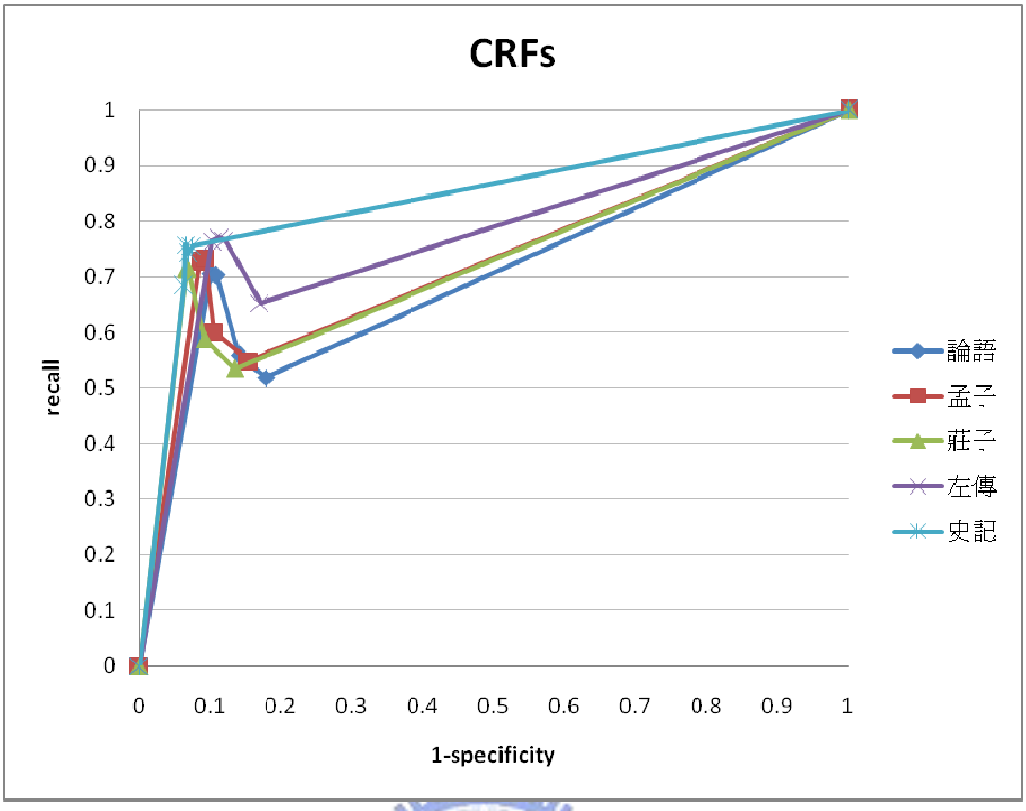

圖 13 五種 training data 之斷句效能比較，使用 conditional random fields。

儘管兩種模型的效能表現有所差異，但各個 dataset 作爲 training data 的效能表現，在兩種模型上呈現一致。在兩個模型上，《史記》都有最好的效能，《論語》的效能最差。在 hidden Markov models 上，兩者相差 4.13 個百分點，在 conditional random fields 上，相差更是明顯，達到 6.2 個百分點。圖 12 和圖 13 以 ROC curves 示顯五個 dataset 的斷句效能。在兩種模型下，ROC curves 其實非常類似，《史記》的效能明顯較好，《左傳》居次，最差的《論語》、《孟子》、《莊子》，線段幾乎重疊，呈現近似的曲線。再與第三章的表格 2 相互參照，由此推得，dataset 作爲 training data 的效能，和 dataset 的長度，呈現正相關，這也符合預期的假設。

4.4 實驗三：Training Data 跨時代的適用性

4.4.1 實驗方法

《論語》等五個 datasets，同屬於上古漢語的範疇，時代相隔不超過 500 年，相互之間，作爲 training data 和 test data，其效能表現都有一定的水準，特別是以《史記》作爲 training data 時，斷句效能甚好，接近在同一個文本內作 k-fold cross-validation。接下來，在實驗三中要再將 training data 與 test data 的差距增加，探討 training data 和 test data 的時代相隔千年以上，斷句效能如何。

實驗三又分 a 與 b 爲兩個部份。實驗三之 a，以各種上古漢語的 datasets，訓練 hidden Markov models 和 conditional random fields 等兩種模型，測試模型斷清代奏摺的效能，如圖 14 所示。實驗三之 b 則改以清代奏摺爲 training data，測試兩種模型斷上古漢語文本的效能，如圖 15。

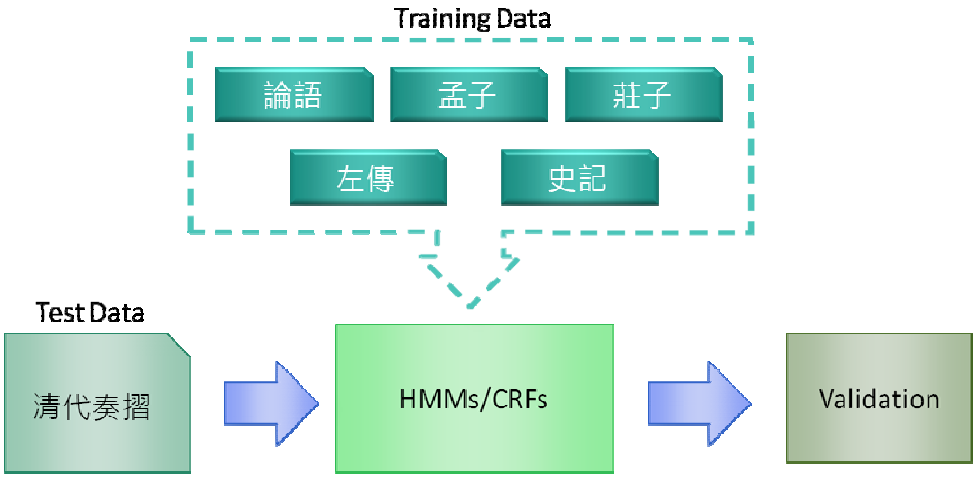

圖 14 實驗三之 a 示意圖。以 hidden Markov models 和 conditional random fields 兩模型，配合上古漢語文本爲 training data，爲清代奏摺斷句。

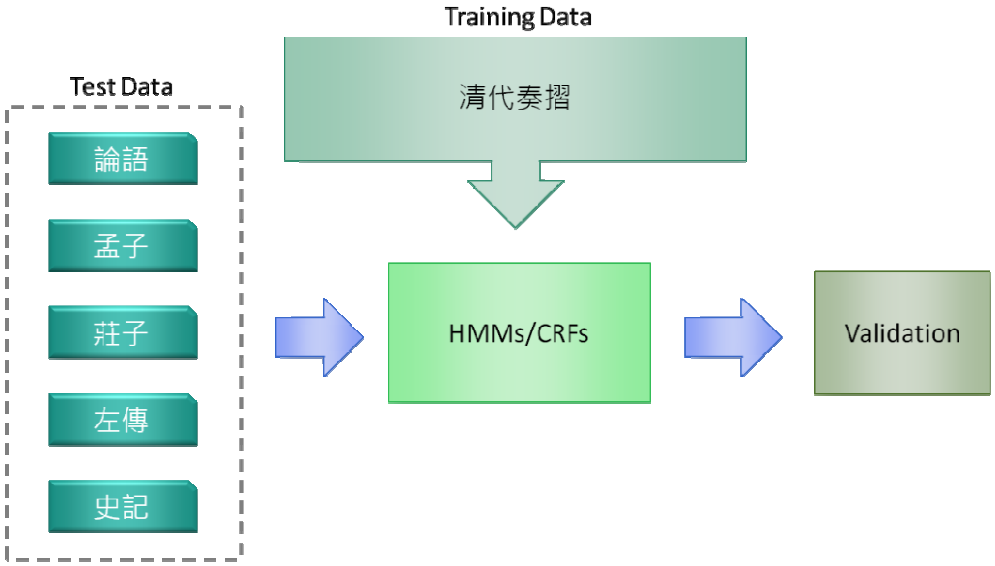

圖 15 實驗三之 b 示意圖。以 hidden Markov models 和 conditional random fields 兩模型，配合清代奏摺爲 training data，爲上古漢語文本斷句。

4.4.2 實驗結果與分析

表格 11 各種上古漢語 datasets 作爲 training data，訓練 hidden Markov models 對清代奏摺斷句的效能。

Training Data	Accuracy	Precision	Recall	Specificity	F-Measure	NIST –SU Error Rate	Labeling Accuracy
《論語》	81.91%	44.35%	42.56%	90.40%	41.05%	123.42%	66.82%
《孟子》	82.59%	46.42%	43.44%	91.20%	42.00%	119.78%	68.08%
《莊子》	82.31%	45.74%	44.36%	90.70%	42.50%	120.34%	67.71%
《春秋左傳》	76.71%	34.11%	42.91%	84.31%	34.36%	168.87%	57.68%
《史記》	80.38%	40.69%	46.36%	88.00%	40.35%	138.59%	63.87%
上古漢語混合	79.44%	39.32%	45.72%	87.06%	38.85%	146.95%	62.41%
平均	80.56%	41.77%	44.23%	88.61%	39.85%	136.33%	64.43%

表格 12 各種 datasets 作爲 training data，訓練 conditional random fields 對清代奏摺斷句的效能。

Training Data	Accuracy	Precision	Recall	Specificity	F-Measure	NIST –SU Error Rate	Labeling Accuracy
《論語》	73.44%	33.55%	50.25%	79.17%	37.26%	195.26%	52.17%
《孟子》	73.38%	33.12%	51.20%	79.03%	37.22%	194.35%	52.06%
《莊子》	76.94%	37.80%	51.59%	83.13%	40.75%	166.21%	58.00%
《春秋左傳》	71.65%	32.19%	56.42%	76.16%	37.66%	209.51%	48.84%
《史記》	77.33%	38.55%	53.61%	83.24%	41.82%	165.20%	58.55%
上古漢語混合	75.10%	35.01%	52.54%	80.88%	38.77%	182.38%	54.71%
平均	74.64%	35.04%	52.60%	80.27%	38.91%	185.49%	54.06%

以上兩個表格，是實驗三之 a 的斷句成效。表格 11 的數據，是分別以《論語》、《孟子》、《莊子》、《春秋左傳》、《史記》、上古漢語混合等 6 個 datasets，訓練 hidden Markov models，再對清代奏摺斷句的效能。表格 12 則改以 conditional random fields 爲模型，作相同測驗的效能。然而，在這樣的試驗裡，兩個模型的效能都不理想。

「以古鑑近」的成效不彰，再看實驗三之 b「以近鑑古」的狀況。表格 13 是以清代奏摺爲 training data，訓練 hidden Markov models，再對《論語》、《孟子》、《莊子》、《春秋左傳》、《史記》、上古漢語混合等 6 個 datasets 作斷句試驗的效能。表格 14 是相同的檢驗，唯斷句模型改爲 conditional random fields。

表格 13 以清代奏摺爲 training data，訓練 hidden Markov models 對各 datasets 斷句的效能

Test Data	Accuracy	Precision	Recall	Specificity	F-Measure	NIST –SU Error Rate	Labeling Accuracy
《論語》	77.77%	EREF _	39.90%	91.32%	47.15%	87.23%	60.27%
《孟子》	80.80%	56.88%	41.26%	91.55%	46.95%	90.74%	65.05%
《莊子》	80.75%	58.59%	43.67%	91.31%	48.35%	91.24%	64.52%

《春秋左傳》	78.79%	75.77%	41.59%	95.16%	50.41%	73.65%	60.55%
《史記》	80.29%	53.83%	31.32%	93.18%	37.73%	97.01%	63.19%
上古漢語混合	79.86%	61.77%	40.06%	92.57%	46.59%	87.16%	63.14%
平均	79.71%	61.63%	39.63%	92.52%	46.20%	87.84%	62.79%

表格 14 以清代奏摺爲 training data，訓練 conditional random fields 對各 datasets 斷句的效能

Test Data	Accuracy	Precision	Recall	Specificity	F-Measure	NIST –SU Error Rate	Labeling Accuracy
《論語》	78.43%	70.19%	32.85%	94.67%	43.12%	83.79%	59.92%
《孟子》	81.69%	64.42%	32.49%	95.01%	42.13%	86.50%	65.49%
《莊子》	82.32%	66.12%	38.29%	94.63%	46.52%	83.46%	66.82%
《春秋左傳》	79.83%	83.96%	38.53%	97.73%	49.01%	68.96%	61.69%
《史記》	81.80%	63.96%	25.77%	96.49%	34.35%	88.89%	65.41%
上古漢語混合	80.81%	69.39%	33.44%	95.76%	42.72%	82.50%	63.81%
平均	80.81%	69.67%	33.56%	95.72%	42.98%	82.35%	63.86%

「以近鑑古」的效能仍然不佳，但略勝於「以古鑑近」。最顯著的差異是 NIST-SU error rate 指標，在「以近鑑古」的情況下，在 80%-90%之間，不若「以古鑑近」的測試，error rate 動輒破百。特別是 conditional random fields，在「以古鑑近」的情況下，表現相當差，而在此卻又有勝於 hidden Markov models 的效能。對於這樣的差異，或許可以這麼解釋：語言文字，隨著時空遞嬗不斷演化，越晚期的語言，越複雜多變，涵蓋大多數前期語言的特色。所以，近代漢語，可以視爲是上古漢語的 super set，包含了許多上古漢話的特色。因此，以代近漢語作 training data，「以近鑑古」，會有相對爲佳的成效，但效能仍然頗差，recall 只有三成左右，無法實用。縱觀實驗三的結果，二千年前的上古漢語的經典和近代的清朝奏摺，都不是夠好的 training data，得以跨越時代的鴻溝，幫助歲月彼端的文本作好斷句工作。對照實驗一的結果，斷句模型本身並不受時代限制，只

要有恰當的training data，對於上古漢語或是清朝奏摺，都能有不錯的斷句成果。然而，training data 卻有時代性，以上古漢語文本構成的 training data，對同時代的文本有適用性，但不宜用在近代漢語上，反之亦然。

4.5 評量指標的討論

在實驗當中，使用 accuracy、precision、recall、specificity、F-measure、NIST-SU error rate、labeling accuracy 等七種評量指標來量化斷句模型的表現。然而，從實驗的結果發現，並不是每個指標都非常合適用來評估斷句的效能。特別是 accuracy 和 specificity 這兩個指標，無論斷句的表現再差，總是有 70%-80% 以上。舉例來說，在表格 11 與表格 12 裡，「以古鑑近」的斷句成果非常不理想，然而，accuracy 和 specificity 卻有 70%以上，在表格 11 裡 specificity 甚至高達 90%。而同樣表格裡的 recall 和 precision 卻只有四成上下，NIST-SU error rate 也超過了一百。在此，也顯示出 accuracy 這個指標的盲點。雖然 accuracy 很方便，單一數字就可以全面性地評估整個系統的效能，但面臨古漢語斷句這種"imbalance problem"，該斷的地方，遠少於不斷的地方，單看 accuracy 實難以判斷系統的好壞。所以，如果想以單一數值來概括斷句的效能，recall 和 precision 的調合平均數（harmonic mean） F-measure 是更合宜的選擇。NIST-SU error rate 在理想的情況，數值多半在 50%之內，而當數值超過 100%，則表示系統已經荒腔走板，效能嚴重不良。因此，對斷句系統而言，要快速評估效能，最簡便的方式就是參考 F-measure 和 NIST-SU error rate 這兩個指標。而想了解系統斷句的行爲，則可以再參考 recall 和 precision。Recall 高，表示斷句模型儘可能地找出了所有該斷開的地方；precision 高則表示斷句模型很謹慎，不會隨便把不該斷開的地方錯斷。除此之外，若要同時比較多個斷句系統、多次測試的總體效能，ROC curves 則相當實用。

五、 結論

在二十世紀，西方的標點符號傳入中國之前，中文的書寫，並沒有使用標點符號的習慣。自古以來，絕大多數的漢語的典籍，段落與段落之間有所分隔，同一段落內的文字，則串連在一起，沒有任何分隔句子和子句的符號。因此，斷句的工作，或稱爲「句讀」，必須交由讀者，在閱讀典籍時自行判斷。然而，斷句並沒有固定的章法，也沒有明確的規則可循，全憑讀者依賴經驗和語感判定。因此，面對同一篇文本，不同的讀者，往往有不同的斷法，而不同斷法，影響了讀者對文義的理解。由此可知，古漢語斷句是閱讀古籍時，困難而重要的第一步驟。

儘管諸多古文典籍，在今天都已經有經過斷句和標點的版本，但其實仍然有更多古漢語文獻，至今尚未經過斷句作業。目前有許多數位典藏計劃，利用文字辨識的技術，將紙本上的古漢語文本數位化，然而，斷句的工作，仍然必須交付專人，耗費極鉅的精力和時間處理。因此，如果有自動化的古漢語斷句工具，快速準確的爲大批古文獻斷句，將能大幅減省時間和人力，並將整個數位藏典，文獻處理的流程，推向全自動化的理想。

在本研究中，我提出了自動化古漢語斷句的可能，並從相關領域中，找尋適當的素材和工具，實現了古漢語斷句系統。在探索斷句問題的過程當中，我處理了三個子問題。第一，爲古漢語斷句系統的效能評量，尋找適當的評估指標。第二，蒐集上古漢語語料，並透過實驗，驗證這些語料作爲 training data 的效能。第三，將斷句的問題，轉化爲序列標記的問題，再使用 hidden Markov models 和 conditional random fields 這兩種序列標記模型，以統計式的方法設計古漢語斷句系統。

我從自然語言處理、機器學習、資料探勘、樣式辨識等相關領域中，提取可

能合適的評估方式。然後，在實驗之中，實際審核這些指標對斷句成果的評估能力。最後，挑選 specificity、f-measure、NIST-SU error rate 等三項作爲斷句研究的主要評估指標，並以 ROC curves 比較多個斷句模型的效能。

爲了簡化問題的複雜性，我鎖定上古漢語的文本，作爲斷句系統的 datasets。這些 datasets 同時具備 training data 和 test data 的功用，必須有足夠的數量和相當的代表性，才能訓練出一般化的斷句模型，適用在各種文本上。在本研究中，我蒐集七種上古漢語的典經文本，逐一測驗這些文本作爲 training data 的效能。其中，司馬遷的《史記》有最佳的成效，以《史記》訓練出來的模型，對其他文本都有不錯的效能表現。

在上古漢語之外，我和台大資工數位典藏與自動推論實驗室合作，取得一批經過專人校對標點的清代奏摺。我將這批奏摺視爲特殊的 dataset，與其他上古漢語文本比對。在實驗中發現，適用於上古漢語的斷句架構，如果改以清代的 dataset 作訓練，也能適用在近代漢語上，而且有不錯的斷句效能。由此推論，本研究提出的斷句模型，有跨時代的適用性，只要配合恰當的 training data，就能處理各時代的文本，並不侷限於上古漢語。

雖然模型可以適用各時代的文本，然而在實驗中也發現，training data 會受時代的侷限。《史記》對上古漢語的文本來說，是很好的 training data，但卻不能斷好清代的奏摺。反之，以清代的奏摺作 training data，同樣無法斷好上古漢語的文本。由此可知，古漢語斷句系統，必須針對處理的對象，挑選時代相近，數量足夠的 training data，才能發揮最好的效能。

斷句系統的核心，我以 hidden Markov models 和 conditional random fields 這兩種模型實作，並在實驗中，比較這兩種模型的效能和特性。Hidden Markov models 是行之有年的經典方法，效率高，學習速度非常快。應用在斷句工作上，

有不錯的效能。Conditional random fields 是 2001 年，由 Lafferty et al. 提出的序列標記模型，也是當前處理各類序列問題效能最好的方法之一。傳統的 conditional random fields，在學習的階段，必須用數值方法作參數評估，複雜度頗高，需要較長的訓練時間。我在本研究中，採用 Collins 的 averaged perceptron 演算法，取代傳統的數值方法，訓練 conditional random fields。Averaged perceptron 無法保證收斂到 global optimal，但訓練出來的模型，效能逼近傳統的參數評估法，效率卻很高，大幅減少訓練時間。

Conditional random fields 應用在斷句系統中，有很好的成效，從幾項指標來看，其效能明顯地優於 hidden Markov models。然而，conditional random fields 對 training data 比較挑剔，份量太少、不夠有代表性，都可能使其斷句效能顯著下降。反之，hidden Markov models 對 training data 較不敏感，在 training data 數量有限，品質不確定的情況，使用 hidden Markov models 有較爲穩定的表現。

自動化古漢語斷句是有待拓展的研究議題，在本研究處理的範圍之外，我還有許多想法，預計在未來逐一試驗。舉例來說，上古漢語還有許多文本，值得繼續實驗，以找尋更好的 training data。甚而，或能援用語言學知識，將眾多文本截長補短，組成更大、更有代表性的 training data。更進一步，或許可以融合各時代的文本，建構跨越時代的泛用型 training data，這對古漢語文獻處理和數位典藏，將更有實用價值。

在此之外，斷句系統經過 training 之後所得到的斷句模型，其實也可以視爲該 training data 在斷句層面上的語言模型。透過訓練所得的諸多特徵（$\boldsymbol{f}$）和其參數（$\boldsymbol{\lambda}$），可以由斷句的角度去了解 training data 的寫作風格。因此，這些特徵與其參數本身就是頗有價值的資訊，可以運用到語料語言學、語文教學、文本考證、作者辨識等領域的研究當中。

參考文獻

[1] 楊樹達，《古書句讀釋例》（上海：上海古籍出版社，2007）。

[2] 李鐸、王毅，〈關於古代文獻信息化工程與古典文學研究之間互動關系的對話〉，《文學遺產》，頁 126-160，2005 第一期。

[3] 林爾正、林丹紅，〈計算機應用於古籍整理研究概況〉，《情報探索》，頁 28-29，2007 第六期。

[4] J. Gao, M. Li, and C. Huang, " Improved Source-Channel Models for Chinese Word Segmentation," in *Proceedings of the 41st Annual Meeting of Association of Computational Linguistics (ACL)*, Japan, 2003.

[5] H. Zhang, Q. Liu, X. Cheng, H. Zhang, and H. Yu, "Chinese Lexical Analysis Using Hierarchical Hidden Markov Model," in *Proceedings of the Second SIGHAN Workshop*, Japan, 2003, pp. 63-70.

[6] N. Xue, "Chinese Word Segmentation as Character Tagging," *International Journal of Computational Linguistics and Chinese Language Processing,* vol. 8, no. 1, pp. 29-48, 2003.

[7] F. Peng, F. Feng, and A. McCallum, "Chinese Segmentation and New Word Detection using Conditional Random Fields," in *Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004)*, 2004, pp. 562-568.

[8] L. Lafferty, A. McCallum, and F. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data," in *Proceedings of the 18th International Conference on Machine Learning*, 2001, pp. 282-289.

[9] R. Mitkov, *The Oxford Handbook of Computational Linguistics*. New York.: Oxford University Press, 2003.

[10] L. Breiman, J. H. Friedman, R. Olshen, and C. J. Stone, *Classification and Regression Trees*. Belmont, CA.: International Group, 1984.

[11] S. M. Humphrey, "Research on Interactive Knowledge-Based Indexing: The Medindex Prototype," in *Symposium on Computer Applications in Medical Care*, 1989, pp. 527-533.

[12] D. D. Palmer and M. A. Hearst, "Adaptive Sentence Boundary Disambiguation," in *Proceedings of the 1994 Conference on Applied Natural Language Processing (ANLP)*, Stuttgart, Germany, 1994, pp. 78-83.

[13] M. D. Riley, "Some Ap plications of Tree-Based Modeling to Speech and

Language Indexing," in *Proceedings of the DARPA Speech and Natural Language Workshop*, Morgan Kaufmann, 1989, pp. 339-352.

[14] J. C. Reynar and A. Ratnaparkhi, " A Maximum Entropy Approach to Identifying Sentence Boundaries," in *Proceedings of the 5th Conference on Applications of Natural Language Processing*, 1997, pp. 16-19.

[15] S. Cuendet, D. Hakkani-Tür, and E. Shriberg, "Automatic Labeling Inconsistencies Detection and Correction for Sentence Unit Segmentation in Conversational Speech," in *Proceedings of MLMI 2007*, Brno, Czech Republic., 2007.

[16] L. Huang, Y. Peng, H. Wang, and Z. Wu, "Statistical Part-of-Speech Tagging for Classical Chinese," in *Text, Speech, and Dialogue: 5th International Conference (TSD 2002)*, 2002, pp. 115-122.

[17] P. N. Tan, M. Steinbach, and K. V., *Introduction to Data Mining*: Pearson Education, Inc., 2006.

[18] E. Alpaydin, *Introduction to Machine Learning*: The MIT Press, 2004.

[19] A. Berger, S. Della Pietra, and V. Della Pietra, "A Maximum Entropy Approach to Natural Language Processing," *Computational Linguistics,* vol. 22, no. 1, pp. 39-71, 1996.

[20] S. Abney, R. E. Schapire, and Y. Singer, "Boosting Applied to Tagging and PP Attachment," in *Proceedings of the Joint SIGDAT Conference on EMNLP and VLC*, 1999, pp. 38-45.

[21] Y. Altun and H. T., "Large Margin Methods for Label Sequence Learning," in *Proceedings of the 8th European Conference on Speech Communication and Technology (EuroSpeech)*, 2003.

[22] L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Application in Speech Recognition," *Proceedings of the IEEE,* vol. 77, no. 2, pp. 257-286, 1989.

[23] A. J. Viterbi, "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm," *IEEE Transactions on Information Theory,* vol. 13, no. 2, pp. 260-267, 1967.

[24] C. D. Manning and H. Schütze, *Foundations of Statistical Natural Language Processing*. MA, US: The MIT Press, 1999.

[25] A. McCallum, D. Freitag, and F. Pereira, "Maximum Entropy Markov Models for Information Extraction and Segmentation," in *Proceedings of International Conference on Machine Learning 2000*, Stanford, California, 2000, pp. 591-598.

[26] H. M. Wallach, "Conditional Random Fields: An Introduction," University of Pennsylvania CIS Technical Report 2004.

[27] R. Feldman and J. Sanger, *The Text Mining Handbook*. New York, US.: Cambridge University Press, 2007.
[28] F. Sha and F. Pereira, "Shallow Parsing with Conditional Random Fields," in *Proceedings of Human Language Technology Conference and North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*, 2003, pp. 134-141.
[29] Y. Liu, A. Stolcke, E. Shriberg, and H. M., "Using Conditional Random Fields for Sentence Boundary Detection in Speech," in *Proceedings of the 43rd Annual Meeting of Association of Computational Linguistics (ACL)*, 2005, pp. 451-458.
[30] S. Della Pietra, V. Della Pietra, and J. Lafferty, "Inducing Features of Random Fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 19, no. 4, pp. 380-393, 1997.
[31] A. McCallum, "Mallet: A machine learning for language toolkit," 2002.
[32] M. Collins, "Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2002, pp. 1-8.
[33] F. Rosenblatt, "The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain," *Psychological Review,* vol. 65, pp. 384-408, 1958.
[34] Y. Freund and R. E. Schapire, "Large Margin Classification using the Perceptron Algorithm," *Machine Learning,* vol. 37, no. 3, pp. 277-296, 1999.
[35] M. Collins and D. Nigel, "New Ranking Algorithms for Parsing and Tagging: Kernels over Discrete Structures, and the Voted Perceptron," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2002, pp. 263-270.
[36] A. McCallum and C. Sutton, "An Introduction to Conditional Random Fields for Relational Learning," in *Introduction to Statistical Relational Learning* MA, US: The MIT Press, 2007, pp. 1-35.
[37] 楊樹達，《詞詮》（上海：上海古籍出版社，2006）。
[38] 朱自清，《經典常談》（上海：復旦大學出版社，2004）。
[39] S. W. Durrant, *The Cloudy Mirror: Tension and Conflict in the Writing of Sima Qian*. Albany: State University of New York Press, 1995.
[40] S. Chen, J. Hsiang, H. Tu, and M. Wu, "On Building a Full-Text Digital Library of Historical Documents," in *Asian Digital Libraries. Looking Back 10 Years and Forging New Frontiers*, 2007, pp. 49-60.
[41] T. A. Cohn, "Scaling Conditional Random Fields for Natural Language

Processing," in *Department of Computer Science and Software Engineering, Faculty of Engineering*. vol. Doctor of Philosophy: University of Melbourne, 2007.