國立臺灣大學文學院語言學研究所
碩士論文
Graduate Institute of Linguistics
College of Liberal Arts
National Taiwan University
Master Thesis

文字部件爲本的語料分析：一個子字詞層次的中文語料庫工具
Glyph-based Corpus Analysis: A Toolkit for Sub-character Analysis of Chinese Corpora

廖永賦
Yongfu Liao

指導教授: 謝舒凱 博士
Advisor: Shu-Kai Hsieh Ph.D.

中華民國 111 年 1 月
January, 2022

doi:10.6342/NTU202200369

國立臺灣大學碩士學位論文
口試委員會審定書

文字部件為本的語料分析：
一個子字詞層次的中文語料庫工具
Glyph-based Corpus Analysis:
A Toolkit for Sub-character Analysis of Chinese Corpora

本論文係廖永賦君（R08142002）在國立臺灣大學語言學研究所完成之碩士學位論文，於民國 111 年 1 月 19 日承下列考試委員審查通過及口試及格，特此證明

口試委員： 謝舒凱 （簽名）
（指導教授）

陳正賢 張瑜芸

doi:10.6342/NTU202200369

致謝辭

這本論文能夠順利完成，有賴許多人的幫助，但礙於篇幅，這邊無法列出過程中幫助過我的每一位，在此先表達我的歉意。首先，我要感謝舒凱老師對於論文的主要架構與發展提供許多指引。我也要特別感謝 Sean 在這個過程中 (以及這三年多以來) 給予我的諸多建議。此外，Sean 先前開發的套件 CompoTree，爲這本論文搭起了鷹架，讓我得以安穩地走在上面逐步完成我的論文。謝謝三位口試委員——瑜芸老師、正賢老師以及舒凱老師——提供了諸多修改論文的方向與建議。對於三位老師們在春節將近時仍撥出時間擔任口委，我由衷感激。

在論文主題發想、實驗與撰寫時，我也受到 LOPE 夥伴們的諸多協助。Don 開發程式的創意與想法一直以來都對我有很大的啓發，這本論文中的語料庫程式的原型便源自於 Don 於 Hocor 2020 的 tutorial。沒有 Jessy 細心整理的漢語歷時語料 (包含語料的爬取以及文本時代的標記)，許多實驗將無法順利進行。另外，謝謝貿昌、Amber、Derek、Yulin 以及品而在聽完口試的 rehearsal 之後給予的回饋與建議 (甚至還有筆記！)，讓我得以修正之前忽略的盲點。

謝謝開源社群，沒有先前的這些開放資源，我將無法完成論文；沒有開源的概念與理想，我當初更不可能愛上寫程式，習得足夠的技能面對論文的挑戰。我也要特別感謝 Yihui，在我牙牙學「程式」時爲我帶來強大的信心，感謝你的主動以及謙遜，即使未曾謀面，信件與留言中的文字仍藏不住你的溫暖。最後，謝謝 Pandoc、LaTeX 以及 Google Docs 所提供的強大寫作工具，讓我得以在極度緊迫的時間內完成論文寫作。

謝謝爸媽無私地給予了我你們所能給的，並放任我自由地吸納你們無法給的。謝謝 Ivo，一直以來作爲我的明鏡，直言不諱地指出我的缺失以及長處；謝謝你教我如何信任、如何同理、如何被愛。謝謝晴方，讓我學習承擔與面對，並教我如何愛人。

doi:10.6342/NTU202200369

doi:10.6342/NTU202200369

摘要

中文書寫系統在世界書寫系統中具有獨特的地位，因爲絕大多數的漢字爲語素文字 (logogram)。因此，漢字本身即攜帶語義訊息，而不像許多其他書寫系統需透過拼音對應至詞彙來攜帶語意訊息。此外，漢字通常可以被分解成更小的元素，這些元素常攜帶著與該漢字相關的語意和發音。然而，由於漢字的編碼方式 (encoding)，電腦使用者不容易取得這些豐富的資訊——一個漢字對應到電腦中的一個編碼 (code point)，這讓使用者無法進一步取得漢字的內部結構訊息，因爲編碼本身並不會記錄這些資訊。例如，中文使用者會知道，「淋」和「霖」這兩個字的發音相同，因爲它們有共同的部件「林」。但是我們無法從「淋」和「霖」的編碼中取得這個共同的部件——在 Unicode 中，「淋」與「霖」分別對應到 U+6DCB 與 U+9716，但這些編碼並無法表徵這兩個字具有關聯的事實。面對這個局限，我們開發了一個可分析子字詞層次的中文語料庫工具。這個語料庫工具讓使用者能夠取得漢字豐富的部件資訊 (包含部首與非部首)，例如，這讓使用者可以根據漢字共有的部件進行檢索 (舉例來說，透過共同部件「林」，可以取得「淋」、「霖」、「琳」、「箖」與「惏」)，並且讓使用者能夠透過這類訊息來進行語料的量化分析。除了語料庫工具之外，我們還進行了一項個案研究，以透過實徵資料驗證子字詞層次的資訊是否有用，並同時探索此階層與更高階層的語意關聯。結果顯示，某些特定的漢字部首語義訊息與詞彙的語義訊息具有顯著的關聯，然而多數的部首與詞彙類型並無明確的對映關係。論文最後，我們指出了漢字內部的高度遞迴結構對於當前研究的一些影響，並討論了解決相關困境的潛在可能。

關鍵字：語料庫工具、書寫系統、漢字、部件、語料庫語言學

doi:10.6342/NTU202200369

doi:10.6342/NTU202200369

Abstract

The Chinese writing system is exceptional among the world's writing systems in that Chinese characters are predominantly logograms that denote words or morphemes. Hence, Chinese characters carry semantic information directly without reference to pronunciations as in other writing systems. Furthermore, Chinese characters are often decomposable into smaller elements that hint at their meaning and pronunciation. This rich internal information of the Chinese characters, however, is not easily accessed in computers nowadays due to the way characters are encoded—a Chinese character is mapped onto a single code point in computers, which makes it impossible to access the internal structures of the character since the code point does not provide such information. For instance, users of Chinese characters would know the characters 淋 and 霖 are pronounced identically due to their common phonetic component 林. But we have no way to access this common component from the encoding of 淋 and 霖, which is U+6DCB and U+9716 in Unicode representation, respectively. Facing this limitation, the current work sets out to develop a software toolkit for analyzing Chinese corpora at and below the level of characters. This corpus toolkit provides access to the rich character internals (radicals and non-radical components) of Chinese characters that were previously unavailable to users, which, for instance, would allow users to search characters based on their common components (e.g., 淋, 霖, 琳, 箖, and 惏 could be retrieved by their common component 林) and enable users to quantitatively analyze corpus data with sub-character information such as this. In addition to the introduction of the corpus toolkit, a case study is also conducted to collect empirical evidence for the usefulness of character internals and to explore their relationships to larger units of the Chinese writing system. The results indicated that semantic information encoded in certain character radicals has a reliable relationship with word semantic types, although most of them have no clear one-to-one correspondence. Finally, several complications due to the highly recursive internal structure of Chinese characters have been pointed out. Potential solutions to these complications are discussed.

doi:10.6342/NTU202200369

Keywords: corpus toolkit, writing system, Chinese character, character components, corpus linguistics

doi:10.6342/NTU202200369

Contents

doi:10.6342/NTU202200369

doi:10.6342/NTU202200369

List of Figures

doi:10.6342/NTU202200369

doi:10.6342/NTU202200369

List of Tables

doi:10.6342/NTU202200369

doi:10.6342/NTU202200369

Chapter 1

Introduction

1.1 Background and Motivation

Throughout history, writing has only been independently —without the influence of another writing system—invented four times. One was invented in Mesopotamia by the Sumerians around 3200 BC, one in Egypt at roughly the same time, one in China around 1200 BC, and the last one in Meso-America around 300 BC (Woods, 2010). These pristine writing systems started out as *logographic* representations that denote real-world entities through drawings but later introduced symbols that reflect pronunciations rather than meanings, ending up as a mixture of *logograms* and *phonograms* (Sproat & Gutkin, 2021). Three of these four pristine writing systems died out, and the Chinese writing system is the only one in use today. Other writing systems nowadays rarely use logograms and instead adopt phonographic representations in their orthographies. This renders the Chinese writing system exceptional among the world's writing systems in that Chinese characters are predominantly *logograms* that denote words or morphemes (Rogers, 2004; Sproat, 2000; Sproat & Gutkin, 2021). Hence, Chinese characters carry semantic information directly without reference to pronunciations as in other

doi:10.6342/NTU202200369

writing systems. Furthermore, Chinese characters are also often decomposable compounds that have components hinting at their meaning and pronunciation. This rich information of the Chinese writing system, or more precisely the Chinese characters, suggests that it is a valuable resource for computational and computer-assisted research of/based on Chinese characters. Unfortunately, this is hindered by several obstacles. One of the biggest arises from the encoding of characters in computers.

A character—including Hanzi, Japanese Kana, and Roman letters—is represented as a single (Unicode) code point in computers nowadays. For instance, the hanzi 我 is represented in Unicode as U+6211; the Kana わ is represented as U+308F; the Roman capital letter *A* is represented as U+0041. This necessary design (from the perspective of general usage of computers), however, quietly poses significant obstacles to computational and computer-assisted research of Chinese characters since it renders the internal structure and components of a character untouchable. This fundamental problem of character representation seems to have repercussions for computational research based on writing systems, such as computational linguistics and corpus linguistics. Both the fields of computational and corpus linguistics originated in communities that adopt an alphabetic writing system such as English. The basic representational unit in the computer for the English writing system—*letters*—seems to be fine for English linguistic research, as this unit is much smaller than the unit of *words* in English. This representational unit even accords pretty well with the unit of *phones* in English, as a letter or bi-/tri-gram would usually map nicely onto a phone. Hence, researchers of English and other languages with similar writing systems have a flexible toolkit, readily provided by computers, that

doi:10.6342/NTU202200369

allows them to operate on various levels of linguistic units. Take the recent advances in word representation learning for example. Vector representations of words are usually trained by using word-level information (Mikolov et al., 2013). Recent work has found that, by incorporating subword information during training, the learned vector representations could be improved (Mikolov et al., 2018). In English, this is easily accomplished through the incorporation of letter-level n-grams during training. This same level of flexibility, however, is not available to researchers of Chinese since words in Chinese predominantly consist of only one or two manipulable characters/syllables. In addition, the place where abundant information sits —the character internal—is masked by the encoding of Chinese characters, making it hardly accessible to NLP researchers of Chinese.

In the current work, we set out to develop a software toolkit for analyzing Chinese corpus data. The aims are to, first, make available to users the information locked within Chinese characters, and second, provide a set of functions that could be deployed to analyze the corpus data at the character and sub-character levels, which, to our knowledge have never been provided by any corpus toolkit. In addition to the provided corpus toolkit, a case study is also conducted to explore and gauge the importance of character components to the (linguistic) unit of words in the Chinese writing system, in terms of the semantic information they carry.

1.2 Contributions

The corpus toolkit developed is provided as an open-source Python library *hgct*[1], which can be easily installed on personal computers. Provided in the toolkit is a set of functions for corpus search and quantitative analyses. What distinguishes

[1]https://yongfu.name/hgct

doi:10.6342/NTU202200369

this toolkit from other existing ones is that it focuses on the character and sub-character levels rather than the word level. Hence, *hgct*'s search function can locate characters by their sub-character features. For instance, the characters 泉, 漿, and 棨 could be located by their common component 水; the characters 打 and 擎 could be located by their common radical 手. These character components may carry interesting semantic or phonetic information that could be important for research depending on character internals. Traditional corpus analysis functions are also extended for analyzing Chinese corpora at these new levels. For instance, frequency distributions of character components or even character internal organizations as described by the IDCs[2] could be computed. Quantitative analyses of morphological productivity (Baayen, 1993, 2009) are also extended to character components in *hgct* such that users could, for instance, quantify and explain why certain components look more familiar than others. The full set of functions shipped with *hgct* is described in detail in Chapter 3, and two simple tutorials of *hgct* are found in the appendices of the thesis. We expect this rich set of functions in *hgct* would foster creative analyses of Chinese corpora and bring about new insights from these new levels of analysis.

1.3 Organization of the Thesis

The rest of the thesis is structured as follows. Chapter 2 reviews the literature on writing systems, traditional analyses of Chinese characters, available resources for representing decomposed Chinese characters, and previous work on Chinese character decomposition. Chapter 3 presents the design, functionalities, and usage of the corpus toolkit *hgct*. Chapter 4 reports a case study on the relationship between

[2]Ideographic Description Characters (IDCs) are described in details in Section 2.3.2.

doi:10.6342/NTU202200369

semantic information encoded at the level of lexical items and the level of character components. Finally, Chapter 5 summarizes and concludes this work by pointing out current limitations and potential directions for future research.

doi:10.6342/NTU202200369

doi:10.6342/NTU202200369

Chapter 2

Literature Review

This chapter reviews the literature to provide the necessary background and theoretical frameworks to situate the current study. We first review the classification of the world's writing system, focusing specifically on the notion of *phonography* and *logography* and their relations to the writing system of modern Chinese. We then turn to Chinese characters—the basic unit of the Chinese writing system—and look at how they are traditionally classified and analyzed. Next, we focus on the internal structures of Chinese characters and see how they can be systematically decomposed by leveraging currently available databases. Finally, we review previous works on decomposing Chinese characters in computers.

2.1 Typology of Writing Systems

The writing system can be defined in a very broad sense as a notational system of visual communication (Sampson, 1985). Thus, writing systems representing spoken languages as well as other notational systems, such as mathematical notations, musical notations, traffic signs, and even the recently invented emoji[1], are all subsumed under this definition. From a linguistic perspective, this broad definition

[1]https://www.unicode.org/reports/tr51/

doi:10.6342/NTU202200369

of the writing system is not very useful, as it renders classifying writing systems extremely difficult. Hence, it is theoretically more interesting, in terms of the field of linguistics, to define the writing system narrowly, by tying it to natural languages. Various scholars have provided roughly similar definitions of the writing system in this narrow sense (Daniels, 1996; Rogers, 2004; Unger & DeFrancis, 1995). Taking Daniels (1996) for example, he defines a writing system as "a system of more or less permanent marks used to represent an utterance in such a way that it can be recovered more or less exactly without the intervention of the utterer". By tying the writing system with language, the world's writing systems could be described and hence classified in terms of how they are related to the language they represent. To draw the connection between a writing system and the language it represents, a description of the writing system's components is needed first. Below we describe these components by adopting Handel (2019)'s terminology.

The basic unit of a writing system is called a *graph*. A set of graphs sharing formal and functional features comprises a *script*. A *writing system* is a collection of one or more scripts and the rules for applying the scripts to represent a language (Handel, 2019). Take the English writing system for example. The letter *k* is a graph, and the set of all letters, i.e., the Roman alphabet, forms the script. The Japanese writing system is more complex, as it is a mixed system of three scripts —the *hiragana* (平仮名), *katakana* (片仮名), and *kanji* (漢字) scripts. Japanese could be written using a mixture of graphs from these three different scripts, such as カレーライスが好きです 'I like curry rice'. The first six graphs カ, レ, ー, ラ, イ, and ス are graphs of the *katakana* script, the graphs が, き, で, and す are graphs of the *hiragana* script, and 好 is a graph of the *kanji* script.

doi:10.6342/NTU202200369

Given the definitions above, a linguistically motivated method of classifying writing systems would be based on the mapping between the graphs of the writing system and the linguistic units of the language represented by the system. Phonological units are often referenced as the linguistic units for the classification of writing systems.

Phonologically, linguistic units ordered from small to large are the *segments* (consonants and vowels), *morae* (sequences of syllable partials), and *syllables*. Writing systems around the world are found to have their graphs mapped onto different phonological units. For example, the Finnish writing system (and various other writing systems that use the Roman alphabets such as Spanish, English, French, etc.), has its graphs (i.e., letters) mapped onto consonants or vowels in Finnish. Writing systems of this type are known as *alphabetic writing systems*. A graph in a writing system could also be mapped onto a larger phonological unit. In a *moraic writing system*, a graph is mapped onto a *mora* in the represented language. A mora is a phonological unit intermediate between the segment and the syllable. A syllable can be thought of as consisting of an onset, a nucleus, and a coda, while a mora is a syllable partial consisting of either an onset-nucleus sequence or a coda (Rogers, 2004). The Japanese *kana* writing system is moraic. A (*katakana* or *hiragana*) graph in the Japanese writing system corresponds to a mora in Japanese. For instance, "Japan" can be written in *katakana* as ニホン [n^{j}ihoN], which consists of three graphs: ニ [n^{j}i], ホ [ho], and ン [N]. The first two graphs ニ and ホ are mapped onto morae consisting of an onset-nucleus sequence, and the last graph ン is mapped onto a mora consisting of a coda. Finally, in a *syllabic writing system*, a graph is mapped onto a syllable of the language. The Chinese writing system is a classic example of

doi:10.6342/NTU202200369

this type. A graph is a character (a.k.a hanzi 漢字) in the Chinese writing system, which maps onto a syllable in Mandarin. For instance, “Japan” is written as 日本 [ʐʅ^{4}pən^3], with each of the two graphs 日 and 本 corresponding to the syllable ʐʅ^4 and pən^3, respectively.

By viewing writing systems in this phonological perspective, a graph in the system could be regarded as a *phonogram*, since it maps onto and hence carries the phonological information of a linguistic unit it represents. Note that, however, this by no means suggests that the mappings of *all* graphs in a system onto the phonological units are perfect. Nor does it suggest that a graph in a writing system cannot at the same time map onto non-phonological units. In fact, there rarely exists *purely phonographic* writing systems in the sense that the pronunciation of a word always unambiguously predicts the orthographic representation of that word (Sproat & Gutkin, 2021). In the English writing system, for example, *there* and *their* are both pronounced [ðɛr]. To pick out the correct representation for [ðɛr], a writer needs to access the *meaning*, or morphological information, of the word, which could be recovered when the word is spoken with other words in context. Writing systems vary greatly with respect to the degree of morphological information encoded in the representations of words, and this property is independent of the phonological classification of the writing system (Rogers, 2004; Sproat, 2000; Sproat & Gutkin, 2021). For instance, the writing systems of Finnish, Spanish, and English are all alphabetic writing systems, in which the letters map to phonological segments of the language in general. These three writing systems, however, differ in the degree of phonological transparency in the mapping. The Finnish writing system has very high transparency in the mapping between orthography and pronunciation,

doi:10.6342/NTU202200369

hence pronunciation nearly always predicts the spelling. Spanish is also highly transparent, but to a lesser degree, in its spelling. Users of the English writing system are aware of commonplace irregular relationships between pronunciation and orthography. These irregularities, however, often provide additional information about the morphology and meaning of the words. For instance, the graph sequence <*telegraph*> common to the words *telegraph* [ˈtɛləɡɹæf], *telegraphy* [təˈlɛɡɹəf-], and *telegraphic* [tɛləˈɡɹæf-] is pronounced differently in the three words. The presence of this graph sequence in representing different words, however, provides hints at the common morphologies, and thus meaning, of these words. This property is related to the notion of *logography*, which is traditionally thought of as a categorical property —a writing system is considered either to be logographic or not. We dive in further to the notion of *logography* next.

In its most intuitive (but rather vague) sense, *logography* is described with reference to the representation of words or morphemes in a writing system. A writing system is considered *logographic* if the graphs in the writing system represent words or morphemes, or, equivalently, whether *logograms* are the basic units of the writing system. The Chinese writing system is a classic example of a *logographic writing system* under this traditional notion of logography. In the Chinese writing system, a character is argued, and justified in most cases, to be a logogram, as a character often maps onto a morpheme or a word, such as the following examples: 人 'human', 牛 'cow', and 電腦 'computer' (literally 'electronic brain'). Problems arise when it is shown that Chinese characters are sometimes used purely as phonological syllabograms. These cases are commonly found in loanwords such as 加拿大 [tɕa^{1}na^{2}ta^{4}] 'Canada', 柏林 [p^{w}o^{2}l^{j}in^{2}] 'Berlin', and 歐巴馬 [əu^{1}pa^{1}ma^{3}] 'Obama'. In these words, characters function as syllables to help pronounce a loanword without carrying any meaning or morphological information of the word. Hence, it is incorrect to claim that all Chinese characters are logograms. It is even incorrect to claim that a *specific* Chinese character is a logogram since whether a character represents a morpheme or not depends on the context in which it occurs. For instance, the character 馬 is a logogram denoting 'horse' in the verb 騎馬 'ride the horse', but the same character is not a logogram in the loanword 歐巴馬 as it is used here solely for the pronunciation of the last syllable *ma* in Oba**ma**. Based on observations of this kind across various writing systems, authors (Rogers, 2004; Sproat, 2000; Sproat & Gutkin, 2021) have proposed a modified notion of logography, which views logography as being *graded* rather than *categorical.* Thus, considered as a whole, a writing system would be high/low in logography if a large/small degree of morphological information is represented in the graphs of the system. This graded notion of logography could be readily applied to the Chinese writing system, as well as the analogous situations of the alphabetical writing systems (the degree of morphology encoded in the spellings of Finnish, Spanish, and English discussed in the previous paragraph). Therefore, the Chinese writing system would be regarded as *highly logographic*, and the Finnish writing system as *highly non-logographic*, under this graded notion of logography.

Based on the notion of graded logography and the traditional phonological classification of writing systems, a two-dimensional planar taxonomy of writing systems is proposed by Sproat (2000) and Rogers (2004), as shown in Figure 2.1. From this figure, we can see the traditional phonological classification of writing systems on the horizontal axis, as an ordinal dimension. The vertical axis depicts the dimension of the graded logography or the amount of morphology in Rogers (2004) terms. These

doi:10.6342/NTU202200369

two dimensions together provide a typology of the world's writing systems. Based on this typology then, the Finnish writing system, for instance, is an alphabetic writing system that has a low degree of logography, whereas the Chinese writing system is syllabic and, at the same time, highly logographic.

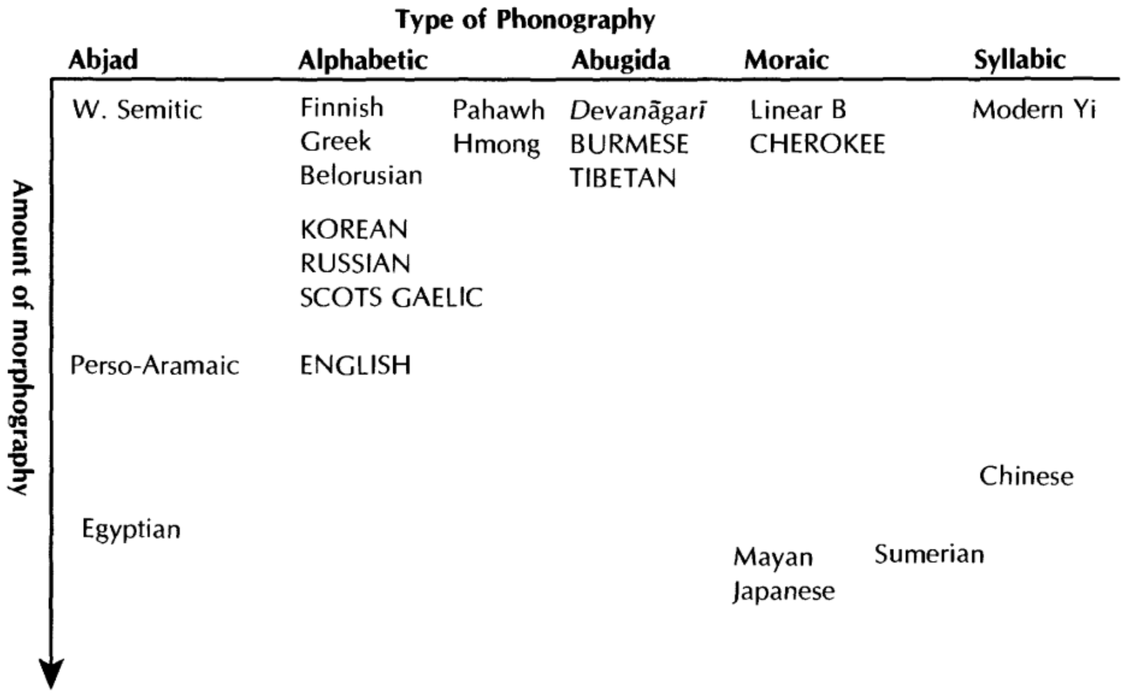


Figure 2.1: Classification of writing systems by Rogers (2004) (Figure 14.5, p. 274), which is based on Sproat (2000). Note that two types of writing systems—the *abjad* and *abugida*—are not reviewed here. These two writing systems could be thought of as slightly different from the alphabetical writing system. An *abjad* writing system has only graphs that map onto the consonants in a language. The vowels are left out, and readers have to infer the vowel by themselves. An *abugida* writing system has both consonants and vowels written but the graphs corresponding to them have different statuses. Vowels are marked secondary around the graphs of the consonants. Interested readers could refer to Rogers (2004).

2.2 Chinese Character Structure

The review in the previous section situates the Chinese writing system in the typology of the world's writing system—a syllabic and highly logographic writing system. The literature on writing systems, however, concerns mainly with graphs and larger units in the system. Units smaller than the graphs are often not the focus

doi:10.6342/NTU202200369

Table 2.1: The six categories of Chinese characters based on 六書 *liushu* .

Name	Graph Type	Examples
象形 *xiangxing*	pictogram	日, 月, 馬
指事 *zhishi*	pictogram	上, 下, 刃
形聲 *xingsheng*	semantic-phonetic compound	泥, 鬆, 媽
會意 *huiyi*	semantic-semantic compound	林, 信, 炎
假借 *jiajie*	phonetic adaptation (Handel, 2019)	北, 長, 令
轉注 *zhuanzu*	unclear	(考, 老)

of discussion since different systems of the graphs (e.g., Roman letters vs. Chinese characters) differ so greatly such that they can hardly be compared. To better understand Chinese character structure, we review analyses of Chinese characters in this section, starting with the classical analysis in Xu Shen's (許慎) *Shuo wen jie zi* (說文解字, hereafter *Shuowen*).

2.2.1 Traditional Analysis of Chinese Characters

Shuowen is a dictionary of Chinese characters written by Xu Shen and compiled around 100 CE during the Han Dynasty in China. Included in *Shuowen* are 9,353 characters along with glosses explaining their meaning and, sometimes, categories. In *Shuowen*, characters are classified into six categories based on the idea of 六書 *liu⁴shu¹* (hereafter *liushu*), which states six different processes that a Chinese character could be created. Table 2.1 lists the six character categories.

The first two categories 象形 *xiangxing* and 指事 *zhishi* are pictograms that are created through iconic processes. For *xiangxing*, the process is through physical similarity. *Zhishi* often expresses ideas through metaphors (Myers, 2019). Characters of these two categories could also be regarded as indivisible **unit graphs** in that there exist no meaningful subcomponents in such characters that could be used as independent characters to recombine and form other characters (Handel, 2019). The next two character categories 形聲 *xingsheng* (hereafter semantic-phonetic compound/

doi:10.6342/NTU202200369

character) and 會意 *huiyi* (hereafter semantic compound character) are compound characters, the opposite of unit graphs. Characters of these two categories could be made up through the compounding of characters, including characters from the above-mentioned four categories[2]. Semantic-phonetic characters are composed of a semantic component that is related to the meaning represented by the character and a phonetic component that hints at the pronunciation of the character. Semantic compound characters are composed of components that each carries semantic information. The meaning of a semantic compound character is suggested by the components' semantics and their arrangement inside the character. Characters of the category 假借 *jiajie* are formed through taking an identically-pronounced character to denote a target spoken word that currently has no existing character to denote it. This process is termed phonetic adaptation in Handel (2019), in which it is shown that subsequent processes could lead to semantic expansion of the chosen character (p.39-47). Finally, for the character category 轉注 *zhuanzu*, *Shuowen* only presents a single pair of examples (考, 老) but does not further elaborate nor are any examples found elsewhere in the book. The exact nature of *zhuanzu* is unclear, and we kept *zhuanzu* in Table 2.1 for the sake of completeness. In subsequent sections, we focus only on the first four character categories, i.e., unit characters and compound characters.

Among the first four character categories in Table 2.1, the semantic-phonetic category is the most frequent in the modern Chinese writing system (estimates range

[2]A possible complication is whether a variant of a character that cannot function independently as a character should be regarded as a character. For instance, 氵 is the variant of the character 水, but it cannot function as a character alone and only exists in compound character. In the current work, we treat these character variants as representing their original forms. Hence, we view, for example, the character 沐 as a character compound of the characters 水 and 木. The variant 氵 is thus treated as a pointer to the character 水.

doi:10.6342/NTU202200369

Table 2.2: Huang (2003)’s survey of the composition of character categories in historic Chinese texts, simplified from his Table 1.

	Oracle Bone Script 甲骨文	Western Zhou Bronze Script 西周金文	Warring States scripts 戰國文字	Small Seal Script 小篆	Regular Script 楷書
象形 xiangxing	28.28	12.78	9.06	3.71	2.07
指事 zhishi	4.29	3.25	1.84	1.25	0.53
形聲 xingsheng	29.10	59.95	69.76	86.29	93.87
會意 huiyi	37.50	19.00	11.36	8.75	3.53
Unknown	0.82	5.02	8.00	0.00	0.00

from 80% to 90%)[3]. In addition, the semantic-phonetic category also seems to be the most productive (Myers, 2019) in terms of character-creation, as semantic-phonetic characters seem to be coined throughout history whereas the percentage of other categories continues to drop. This could be seen in the survey of Huang (2003), which is shown in Table 2.2.

2.2.2 Character Radicals

In addition to the *liushu* classification of characters, another method of classification is based on the common components across characters. These components are called radicals (部首), which were first introduced by *Shuowen* to group characters for indexing characters in the dictionary. Contemporary Chinese dictionaries continue to adopt the radical system for indexing purposes. However, modern dictionaries do not adopt the radical system of *Showen*, which consists of 540 radicals. Instead, the 214 Kangxi radicals, which were established in the *Kangxi Dictionary* (康熙字典) compiled in the Qing Dynasty (1716 CE) of China, are chosen as the standard. The *Kangxi Dictionary* continues to be influential to date, and it is adopted by the Unicode Standard[4].

[3]Zhou (1978) estimates around 80% of characters to be semantic-phonetic; D. K. [黃德寬]. Huang (2003) estimates around 90% of characters to be semantic-phonetic.

[4]See Chapter 17 of The Unicode Consortium (2003).

doi:10.6342/NTU202200369

Radicals are not simply arbitrary components of the characters. Instead, they were systematically chosen such that many of them carry semantic information associated with the characters they participate. In semantic-phonetic characters, the semantic component of a character is often also the radical of the character. For instance, the character sets 燒, 炮, 燙, 煮 and 澆, 泡, 湯, 漿 are assigned the radical 火 (灬) and 水 (氵), respectively. The radical 水 and 火 are the semantic component in these semantic-phonetic characters, and each radical suggests the semantic content of the character they participate. Thus, radicals could be regarded as a semantic classifier, or semantic taxogram (Handel, 2019), for characters in many cases. However, complications arise when unit characters are concerned. As mentioned above, unit characters could not be further decomposed into functional components, but for the purpose of indexing, many of the unit characters are still assigned a radical by the common stroke patterns within the unit characters. For instance, the *xiangxing* (象形) unit character 丁 has no meaningful semantic subcomponents but is nonetheless assigned the radical 一, which is the horizontal stroke on the top portion of 丁. Still, radicals are reliable hints to the semantic content of characters in the modern Chinese writing system since it is known that the predominant Chinese characters used today are compound characters (see Table 2.2). Later in Chapter 3 and Chapter 4, we introduce an application of leveraging semantic information of the radicals in analyzing data.

2.3 Resources for Chinese Character Decomposition

In this section, we review currently available resources for the decomposition of Chinese characters in computers. Decomposing characters and representing their

doi:10.6342/NTU202200369

decomposed glyphs in computers is a crucial step as it is a prerequisite that makes computational research below the level of characters possible.

Before we turn to the details of the review, it is necessary to first clarify the terminology. Figure 2.2 summarizes the terminologies related to the Chinese writing system used throughout the thesis. At the top-most is the level of the *word*. Below the word is the level of the *character*. Since characters are also the basic units of the Chinese writing system, a parenthesized *graph* is shown under "character" to indicate this. Although the literature on writing systems focuses mostly on the level at or above Chinese characters (graph), we know from the traditional analyses of Chinese characters that there are rich structures within them. Hence, there exists a *sub-character* level below the character level. Chinese characters are often decomposable into smaller elements. We call these decomposed elements *components*, or more generally, *glyphs*. Finally, a component at this level could further be classified as either a *radical* or not.

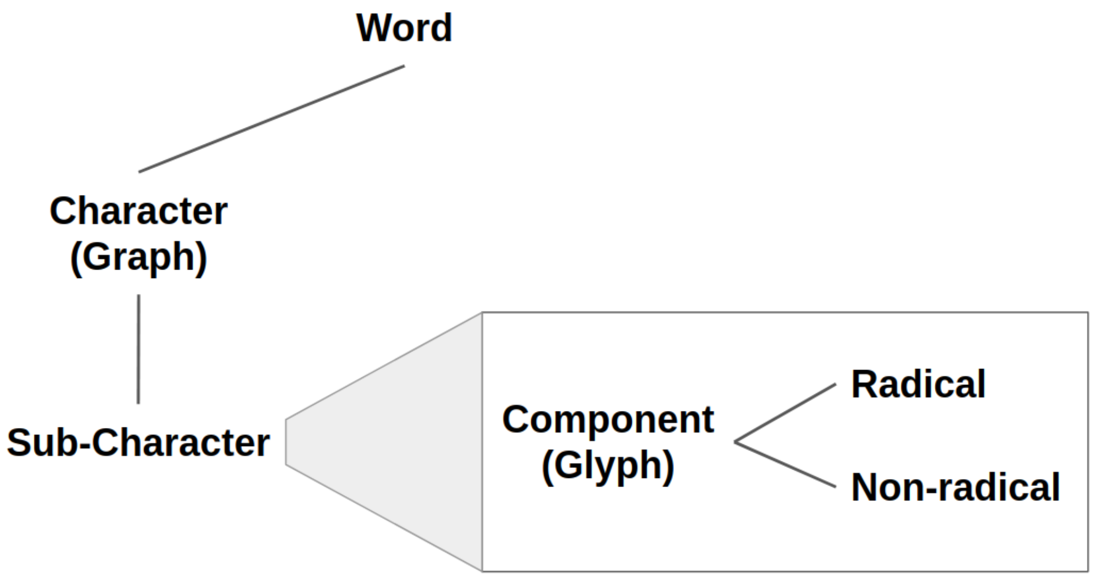


Figure 2.2: A hierarchical view of the units in the Chinese writing system.

doi:10.6342/NTU202200369

2.3.1 Kangxi Radicals

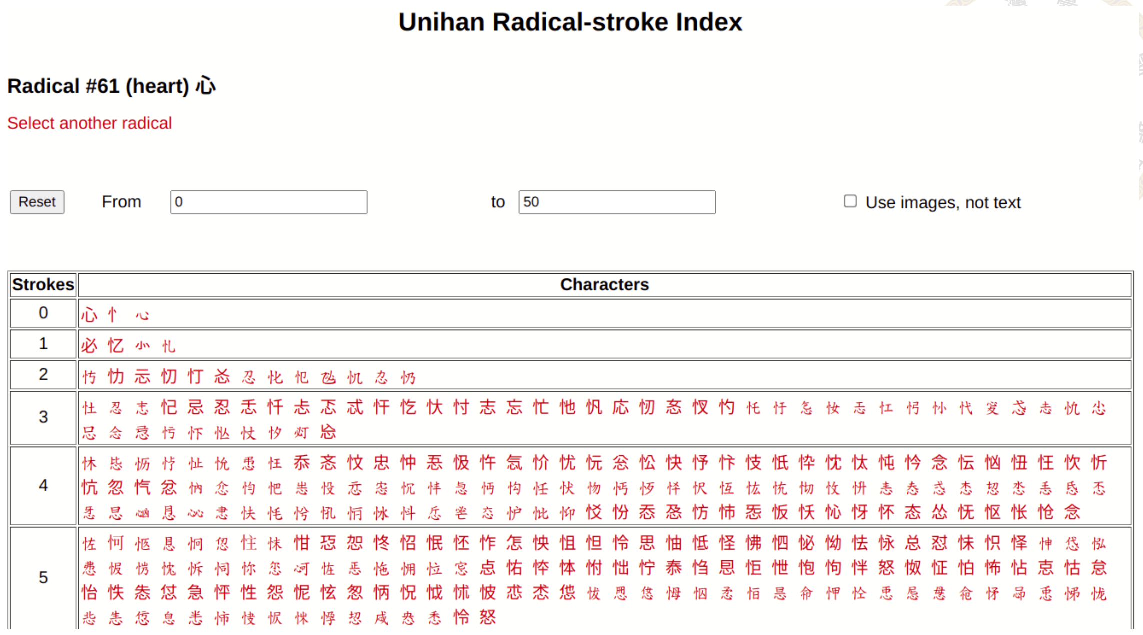
The most intuitive way of decomposing a Chinese character is by its Kangxi radical since the radicals are often introduced along the process of learning Chinese characters. The advantage of Character decomposition with the Kangxi radicals is that it is unambiguous, as the location of the radical inside a character is well-defined. In addition, the wide coverage of the *Kangxi Dictionary*, containing more than 47 thousand characters, ensures that almost all characters can be assigned a radical. Information derived from the *Kangxi Dictionary* is included in the Unicode Standard and the Unihan Database[5], which contains Kangxi radical information for the Chinese characters in Unicode. For instance, the Unihan Radical-stroke Index[6] provides a web interface to the Unihan Database for finding Unicode Chinese characters according to their Kangxi radicals (see Figure 2.3). Based on the publicly available Unihan Database, software tools providing similar functionalities have also been built, such as Tseng (2021).

2.3.2 Ideographic Description

Focusing on the internal organization of the components in Chinese characters, it is not hard to see that there seem to be consistent "rules" for assembling the components into characters. For instance, the character 燒 can be thought of as two components (火 + 堯) concatenated horizontally. Similarly, the characters 炮 (火 + 包), 澆 (氵 + 堯), and 泡 (氵 + 包) can also be considered forming from the same rule. Another common structural pattern is two vertically stacked components, as can be seen in the characters 燙 (湯 + 火), 煮 (者 + 灬), and 漿 (將 + 水). Based on these

[5]https://www.unicode.org/charts/unihan.html
[6]https://www.unicode.org/charts/unihanrsindex.html

doi:10.6342/NTU202200369



Unihan Radical-stroke Index

Radical #61 (heart) 心

Select another radical

Reset From 0 to 50 ☐ Use images, not text

Strokes	Characters
0	心 忄 㣺
1	必 忆 ⺗ 㣻
2	忇 㣼 忈 忉 忊 忎 㣽 㣾 忋 㣿 㤀 忍 㤁
3	忕 忍 忐 忋 忌 忍 忝 忏 忐 忑 忒 忓 忔 忕 忖 志 忘 忙 忚 忛 応 忉 忞 忟 忣 忶 忸 忹 忺 忻 忼 忾 忿 怂 怃 怄 怅 怆 忌 念 忥 忬 忭 忪 忮 忯 忰 忿
4	怀 忠 怄 怍 怔 怕 怨 怪 忝 忞 忟 忠 忡 忢 忣 忤 忥 忦 忧 忨 忩 忪 快 忬 忭 忮 忯 忰 忱 忲 忳 忴 念 忶 忷 忸 忹 忺 忻 忼 忽 忾 忿 怀 态 怂 怃 怄 怅 怆 怇 怈 怉 怊 怋 怌 怍 怎 怏 怐 怑 怒 怓 怔 怕 怖 怗 怘 怙 怚 怛 怜 思 怞 怟 怠 怡 怢 怣 怤 急 怦 性 怨 怩 怪 怫 怬 怭 怮 怯 怰 怱 怲 怳 怴 怵 怶 怷 怸 怹 怺 总 怼 怽 怾 怿 恀 恁 忟 忡 忢 忣 忤 忥 忦 忧 忨 忩 忪 快 忬 忭 念
5	怇 怐 怄 怎 怐 怨 性 怢 怗 怘 怎 怒 怊 怋 怍 怎 快 怚 怛 怜 思 怞 怟 怪 怫 怬 怭 怮 怯 怰 总 怼 怽 怾 怿 怦 怱 怲 㤈 㤉 㤊 㤋 㤌 㤍 㤎 㤏 㤐 㤑 㤒 㤓 㤔 㤕 点 怙 怦 怵 怸 怹 怺 泰 怡 恩 怇 怈 怉 怊 怋 怒 怓 怔 怕 怖 怗 怘 怙 怚 怛 怡 怢 怣 怤 急 怦 性 怨 怩 怪 怫 怬 怭 怮 怯 怰 怳 怴 怵 怶 怷 怸 怹 怺 㤖 㤗 㤘 㤙 㤚 㤛 㤜 㤝 㤞 㤟 㤠 㤡 㤢 㤣 㤤 㤥 㤦 㤧 㤨 㤩 㤪 㤫 㤬 㤭 㤮 㤯 㤰 怜 怒

Figure 2.3: The Unihan Radical-stroke Index provides a search interface for locating Chinese characters according to their radicals and stroke counts.

commonly observed patterns in Chinese characters, the Unicode Consortium (2003) introduced the Ideographic Description Characters (hereafter, IDCs). IDCs are a set of twelve graphical characters used for describing the internal structuring of the components in Chinese characters. For instance, the horizontal structuring of two components is described by the IDC ⿰ (left to right), and the vertical structuring of two components is described by the IDC ⿱ (above to below). Table 2.3 lists the twelve IDCs along with some example characters that are described by each of the IDCs.

With the introduction of the IDCs, it is then possible to describe any Chinese character by an Ideographic Description Sequence (IDS). An Ideographic Description Sequence describes how a character is assembled from its parts by laying out the character's IDC and components in a specific order. IDS employs a syntax, which helps deal with the ubiquitous recursive structures in Chinese characters[7], for laying

[7]For instance, the character 燙 can be decomposed into 湯 + 火 at the first level. The component 湯 can further be decomposed into 氵 and 昜 at the next level, and 昜 into 旦 and 勿 at the third level, and so on.

doi:10.6342/NTU202200369

Table 2.3: The twelve Ideographic Description Characters (IDCs) in Unicode.

IDC	Unicode Point	Name in *hgct*	Examples
⿰	U+2FF0	horz2	錤, 根, 敢, 祈, 液
⿱	U+2FF1	vert2	草, 筏, 纂, 羅, 罕
⿲	U+2FF2	horz3	衛, 辨, 弼, 徹, 讎
⿳	U+2FF3	vert3	器, 褒, 尋, 哀, 褻
⿴	U+2FF4	encl	困, 囪, 因, 四, 井
⿵	U+2FF5	surN	開, 風, 同, 向, 冈
⿶	U+2FF6	surU	函, 興, 鼎
⿷	U+2FF7	curC	匾, 匿, 區, 叵, 匹
⿸	U+2FF8	surT	雁, 厚, 底, 庫
⿹	U+2FF9	sur7	匍, 可, 司, 島, 式
⿺	U+2FFA	surL	趕, 尷, 毬, 廷, 爬
⿻	U+2FFB	over	必, 吏, 丰, 丸, 卅

out the IDCs and glyphs. To describe the syntax, we take 燙 and 燕 for example. The IDS for 燙 and 燕 are ⿱湯火 and ⿳廿⿰北口灬, respectively. These IDSs may not be easy to read at first glance. To make sense of an IDS, one can regard the IDC inside the IDS as a function and the components following the IDC as the arguments to the function. Thinking in this way, the IDS of 燙, ⿱湯火, can be expressed in a more readable fashion ⿱ (湯, 火). These notations are easily extended to IDS with nested structures like ⿳廿⿰北口灬. The IDC ⿳ takes three arguments, hence ⿳廿⿰北口灬 can be re-expressed as ⿳ (廿, ⿰北口, 灬). Going further down a level in the second argument of ⿳, ⿰北口, gives us ⿰ (北, 口). Replacing ⿰北口 with ⿰ (北, 口) inside ⿳ (廿, ⿰北口, 灬) gives ⿳ (廿, ⿰ (北, 口), 灬). This “function” expression has the advantage of emphasizing the recursive structures in IDSs and may help readers to understand the IDSs upon the first encounter.

Based on the Unicode’s IDC and IDS standard, the CJKV Ideographic Database[8] provides Ideographic Description Sequences for 88,937 Unicode characters[9]. Tseng (2021) builds on top of the IDS dataset and provides a programmatic

[8]CJKV stands for Chinese, Japanese, Korean, and Vietnamese.

[9]The dataset is available at https://github.com/cjkvi/cjkvi-ids, which derived from the IDS

doi:10.6342/NTU202200369

interface to access the dataset, making it possible to analyze Chinese characters' internal structures according to the IDS information. The corpus analysis toolkit developed in the current study is built on top of Tseng (2021), which is the subject of the next chapter.

2.4 Previous Work on Chinese Character Decomposition

2.4.1 Hantology

Chou & Huang (2006) have noticed the rich conceptual information encoded in Chinese characters and their components. This information could be a unique resource for Chinese language processing as the writing systems of other languages mostly carry far less logographic information than the Chinese. An ontology for Chinese characters[10] is thus built by mapping the senses of Chinese characters onto the SUMO ontology (Niles & Pease, 2001). In addition to the senses of the characters, the radical—based on *Shuowen*'s 540 radicals—of each character is also analyzed and mapped onto the SUMO ontology. This idea of utilizing semantic information carried in radicals is similar to what we have done in the current work, which we introduce later in Section 3.3.2.

2.4.2 Chinese Characters Information Database

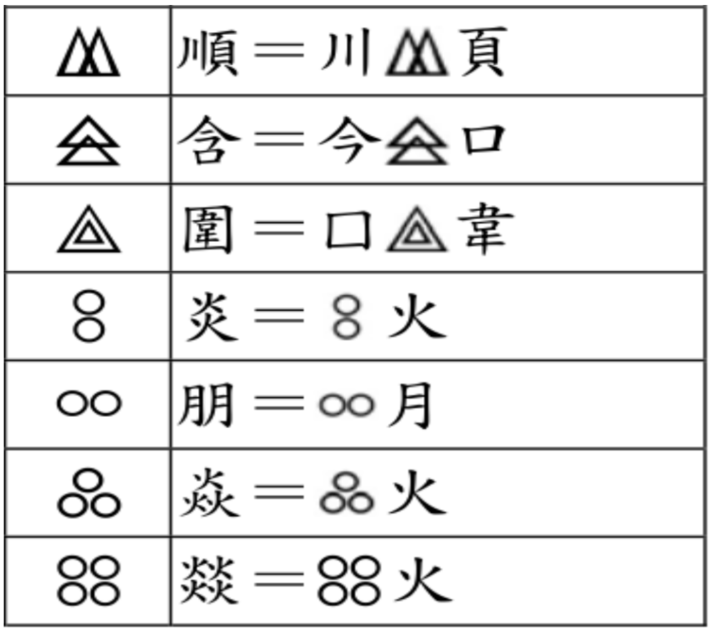
Starting off from a practical purpose to deal with missing characters in computers (i.e., characters not assigned Big5 code points), the Chinese characters in-

dataset in the CHISE project. See Section 2.4.3 for detail.

[10]The search interface of the Chinese character ontology is available at https://hantology.sinica.edu.tw.

doi:10.6342/NTU202200369

formation database 漢字構形資料庫 (Chuang & Hsieh, 2005), created since the 1990s, is one of the earliest work on Chinese character decomposition. One of the biggest contributions of this database, relevant to the current study, is its analyses of character compositions. Taking the 214 Kangxi radicals as a basis, a character is decomposed into components and the components into subcomponents and so on until a (sub)component is itself a Kangxi radical. For instance, the character 量 is first decomposed into 里 (the radical of 量) and 旦. Since 里 is a Kangxi radical, it is not further decomposed. On the other hand, 旦 is further decomposed into 日 (the radical of 旦) and 一. Since 日 and 一 are both Kangxi radicals, no further decompositions are carried out. In addition to setting up a decomposition scheme for characters, a set of operators are also introduced for describing the relations between components in a character. Figure 2.4 shows a selection of these operators. These operators are conceptually similar to the Ideographic Description Characters in Unicode introduced above.



	順＝川 頁
	含＝今 口
	圍＝口 韋
	炎＝ 火
	朋＝ 月
	焱＝ 火
	燚＝ 火

Figure 2.4: Operators for assembling a character from components. Modified from Table 12 in Chuang & Hsieh (2005).

doi:10.6342/NTU202200369

2.4.3 Character Information Service Environment (CHISE)

One of the most comprehensive collections of data for Chinese character decomposition is provided by the CHISE project (Morioka, 2008). Following Unicode's standard of Ideographic Description Character/Sequence, the CHISE project provides IDS data for all characters in Unicode's CJK Unified Ideographs (including Extension A and B). These IDS data were derived partly from the Chinese characters information database (Chuang & Hsieh, 2005) and CBETA[11]—an organization for digitizing Buddhist texts that created its own character composition scheme to deal with missing characters. A query interface is available[12] for searching character components based on the IDS dataset in CHISE. The CJKV Ideographic Database mentioned in Section 2.3.2 derives directly from CHISE's IDS dataset.

[11]https://www.cbeta.org/data/cbeta/rare.htm
[12]https://www.chise.org/ids-find

doi:10.6342/NTU202200369

Chapter 3

Design of System

This chapter describes the design, functionalities, and usage of the system, *hgct* (Hanzi Glyph Corpus Toolkit). *hgct* is compiled as an open-source Python3 package and is distributed on the Python Package Index (PyPI)[1] for easy access by users.

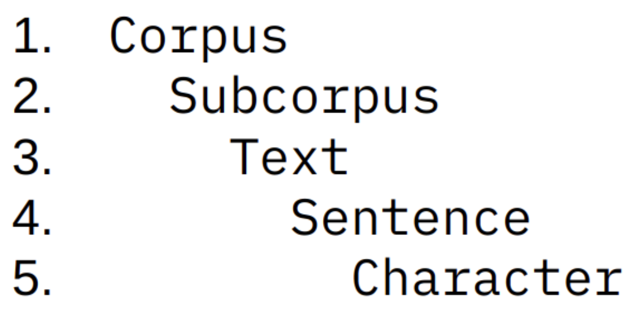
3.1 Corpus Structure and Input Data

The hierarchical structure of *hgct*'s input corpus representation is shown in Figure 3.1. Note that this hierarchy refers to the *data structure* representation. Hence, although linguistic labels are used here (*corpus*, *subcorpus*, *text*, *sentence*, and *character*), they serve only as pointers to the often used hierarchical level in corpus linguistic but do not posit linguistic definitions on these levels. The definitions are left to the users such that flexible and creative organizations of the corpus are possible. This hierarchical structure only specifies that a *corpus* consists of a list of subcorpora, a *subcorpus* consists of a list of text, a *text* consists of a list of sentences, and a *sentence* consists of a sequence of characters.

In this hierarchical structure, probably the most noticeable feature is the absence of a word level in the representation. Since the focus of *hgct* is on characters

[1]https://pypi.org/project/hgct

doi:10.6342/NTU202200369



```
1. Corpus
2.   Subcorpus
3.     Text
4.       Sentence
5.         Character
```

Figure 3.1: The hierarchical structure of *hgct*'s corpus representation.

and glyphs, *hgct* discards the step of tokenizing strings of characters into words (i.e., Chinse Word Segmentation in the case for Chinese corpora) that is usually deployed in other corpus analysis toolkits. Hence, there is no word-level representation in *hgct*'s representation of a corpus. A sentence is simply represented as a sequence of characters in *hgct*. Another level worth mentioning is the subcorpus level. The subcorpus level is designed in the corpus representation to allow flexible analyses of the corpus data. For instance, users interested in conducting diachronic corpus analyses could utilize this subcorpus level to organize the input corpus as an ordered sequence of diachronically subcorpora, and users interested in contrasting or comparing corpora could also utilize this level to group texts according to specific attributes, such as text genres or genders of the authors.

A finer-grained view of the corpus representation can be found in Code Block 3.1 at the end of this chapter, which shows the JSON representation of an example input corpus. This is also the internal corpus representation used by *hgct*. The JSON string in Code Block 3.1 represents an input corpus (an array of objects, or a list of dictionaries in Python) with two subcorpora (an object, or dictionary in Python), each of which contains three texts (an array of objects) and each text containing two sentences (an array of strings). *hgct* provides a user-friendly interface,

doi:10.6342/NTU202200369

`hgct.PlainTextReader()`[2], for converting a raw input corpus (a directory of UTF-8 encoded text files structured in a particular fashion) into the corpus structure required by *hgct*. Code Block 3.2 illustrates the directory structure of the input corpus that can be represented as JSON in Code Block 3.1.

After loading the corpus, *hgct* provides various functionalities that could be utilized to analyze the corpus data. Below, we group these functionalities into two broad categories, "Searching" and "Corpus Analysis", and describe each of them in Section 3.3 and Section 3.4 respectively. However, before introducing these functionalities, a description of the query language used by *hgct* to search the corpus is necessary.

3.2 Corpus Query Language

The query language used in *hgct* has a syntax similar to the influential and widely used Corpus Query Language (CQL), originally supported by the concordance system Corpus Workbench (CWB) developed at the University of Stuttgart (Christ, 1994; Evert & Hardie, 2011). The query language in *hgct* is in fact a strict subset of the CQL, which we name CQLS (Corpus Query Language Subset) here. The parser of CQLS in *hgct* is provided by the *cqls*[3] Python package.

In CQLS, a query describes a sequence of *tokens*, each of which could be further specified with some predefined set of *attributes* associated with the token. In *hgct*, *tokens* correspond to *characters*, not *words* as in most corpus systems. For instance, if a user is to search the corpus for the character sequence "我們", the query string would be `[char="我"] [char="們"]`[4]. Each pair of square brackets

[2]https://yongfu.name/hgct/corpusReader.html#hgct.corpusReader.PlainTextReader
[3]https://github.com/liao961120/cqls
[4]Since the default token attribute in *hgct* is char, this query could be written more succinctly

in the query string specifies a single token. Attributes can be given in the form "`attribute=value`" inside the brackets to specify the token attributes. In the example here, we add the attribute "`char`" (character form) and constrain its value to be "我" in the first token and "們" in the second. It is also possible to give abstract patterns described by regular expressions (hereafter, RegEx) to values of the attributes. For instance, if instead, a user wants to search the corpus for both "我們" and "你們", a RegEx pattern `我|你` can be given to the char attribute in the first token as `[char="我|你"] [char="們"]`. Multiple attributes may be given to jointly constraint a token. In *hgct*, it is possible to search the characters by their components. For instance, a query `[compo="木" & idc="horz2"]` may return characters such as 櫸, 松, 柏, 檜 since they all have the component "木" (specified in `compo="木"`) and the two components in the character (e.g., 木 and 公 for 松) are organized horizontally (specified in `idc="horz2"`). Details of these character components are described in Section 3.3.3.

3.3 Searching the Corpus

In *hgct*, a search function is provided to search the corpus. The function takes a CQLS string and returns matching results as a sequence of concordance lines for inspection and further analyses. Appendix A provides a tutorial on using the search API in *hgct*. The remainder of this section describes the design of the search function, along with the background knowledge to make sense of the provided functionalities.

as `"我" "們"`.

doi:10.6342/NTU202200369

3.3.1 Searchable Properties

The search of the corpus is performed on the level of characters, where several properties associated with Chinese characters could be specified. Among these searchable character properties, the simplest one is the character form (specified by the token attribute “char” in CQLS). It is also possible to search with sound properties of a character. *hgct* incorporates two systems of sound information. The first system is sound information of the Modern Chinese, which is constructed from the dictionary compiled by the Ministry of Education in Taiwan[5]. The second system is sound information of the Middle Chinese, which is based on Guangyun (廣韻)[6]. Thus, phonetic properties used by the two systems, such as Zhuyin (注音), IPA, Pinyin (漢語拼音), onset (聲母 in Guangyun), and tone (聲調 in Guanyun), could be used to search the corpus. Table 3.1 lists the searchable properties along with examples for each search type in *hgct*.

Chinese characters are also prominent in their rich internal structures. Since the majority of Chinese characters have internal structures and can be further decomposed into smaller components or “glyphs”, it is also possible to perform a search based on these decomposed features. *hgct* decomposes Chinese characters based on the *CompoTree*[7] Python package, which provides two different methods to decompose Chinese characters—by radicals (部首) or by Unicode’s Ideographic Description Characters. The following sections describe the two methods respectively.

[5]Moe dictionary (萌典): https://github.com/g0v/moedict-data/blob/master/dict-revised.json

[6]The data of Guangyun (廣韻) is taken from a publically available repository at https://github.com/nk2028/qieyun-data/blob/main/韻書/廣韻.csv, which is manually coded and maintained by volunteers on the internet.

[7]https://github.com/seantyh/CompoTree

doi:10.6342/NTU202200369

Table 3.1: Supported properties for corpus search in *hgct*. Attributes of a search type prefixed with an asterisk are the required attributes of that search type.

Search Type	**Attribute**	**Value Space**	**RegEx Support**	**Examples**	

Character Form	*char	hanzi	Yes	`[char="你 | 他 | 我"] [char="們"]`	{我們} 只採取;
					坐視 {他們} 走錯路;
					{你們} 能死在對面;

Radical	*radical	hanzi	No	`[radical="广"]`	國會大 {廈};
					入，坐於 {床} 東;
					設 {庠} 序以教之;

Ideographic Description	compo	hanzi	No	`[compo="水" & idc="vert2" & pos="1"]`	心如淵 {泉};
					吾食於十 {漿};
					一軍臨 {滎} 陽;

	idc	curC, encl, horz2, horz3, over, sur7, surL, surN, surT, surU, vert2, vert3	No		
	pos	-1, 0, 1, 2, 3	No		

Semantic Tag	*semtag	Refer to Table 3.2	Yes	`[semtag="植物"]`	獸伏就 {穢};
					{葦} 席以爲屋;
					乃百慮之 {筌} 蹄;
					{飈}，颮，其紹飈;

Sound (MOE)	phon	phonetic symbols (either Zhuyin, Pinyin, or IPA, controled by the ”tp” attribute)	Yes	`[phon="ㄨㄛ" & tone="4" & sys="moe"]`	入其 {幄} 中;
					{臥} 則夢之;
					旋蟲謂之 {斡};

	tone	1, 2, 3, 4, 5	No		
	tp	bpm, ipa, pinyin	No		
	*sys	moe	No		

Sound (Guangyun)	聲母	hanzi (refer to Guangyun)	No	`[韻母 ="東" & 聲調 ="平" & sys="廣韻"]`	{烘}，燎也;
					骨肉都 {融};
					{雄} 雄赫赫;
					道 {隆} 則從而隆;

	韻母	hanzi (refer to Guangyun)	No		
	攝	hanzi (refer to Guangyun)	No		
	聲調	平, 上, 去, 入	No		
	開合	開, 合	No		
	等第	hanzi (refer to Guangyun)	No		
	反切	hanzi (refer to Guangyun)	No		
	*sys	廣韻	No		

doi:10.6342/NTU202200369

3.3.2 Radicals and Derived Semantic Types

The most notable way of decomposing a Chinese character is by its radical (部首). As reviewed in Section 2.2.2, Chinese characters are assigned radicals that are used in dictionaries for grouping and indexing characters. In *hgct*, we adopt the 214 radicals based on the *Kangxi Dictionary* (康熙字典) published in the 18th century. These radicals are treated as glyphs that could be used for searching. For instance, searching the corpus with the radical 水 would return all characters with this radical, such as 澆, 泡, 湯, and 漿, in the corpus.

Also noted in Section 2.2.2, the radical of a character often provides information related to the semantic content of the character. Although not all 214 radicals in the Kangxi Dictionary carry such semantic information (since some of the radicals are used solely for indexing purposes), the radicals of most Chinese characters do and can be regarded as semantic radicals. For instance, Ma (2016, pp. 61–71) assigns 22 semantic types to 171 radicals (accounting for roughly 80% of the 214 Kangxi radicals) based on the semantic contents often associated with the radicals, as shown in Table 3.2. These radicals and semantic types derived from them are potential resources that could be leveraged in Chinese corpus and computational linguistics since traditionally only information at and above the word level is utilized. Semantic information encoded at the sub-word or the sub-character level is neglected and under-utilized. *hgct* incorporates Ma (2016)'s semantic types, making it possible to search the corpus with these predefined semantic types.

doi:10.6342/NTU202200369

Table 3.2: Ma (2016)’s semantic classification of 171 Kangxi radicals.

	Semantic Type	Kangxi Radical
1	人	人, 乙, 儿, 入, 士
2	人體精神	心
3	人體頭部	口, 頁, 首, 面, 目, 舌, 牙, 齒, 音, 臼, 耳, 鼻
4	人體四肢	肉, 手, 又, 支, 寸, 爪, 足, 走, 止, 辵, 廴, 立, 行, 工, 力
5	人體內部	骨, 血
6	人體性質	比, 舛, 黽, 臣, 釆, 鬥
7	人倫關係	父, 毋, 子, 女, 匕, 厶, 氏
8	禮樂	文, 攴, 豆, 鼓, 龠, 卜
9	住宿	門, 户, 宀, 穴, 广, 厂, 尸
10	廚房器物	皿, 缶, 鬲, 鼎, 斗, 凵, 彐
11	生活器物	舟, 車, 耒, 瓦, 网, 黹, 貝, 疋
12	穿著器物	毛, 衣, 糸, 幺, 革, 韋, 巾
13	武器	弓, 矢, 戈, 弋, 矛, 刀, 斤, 殳
14	顏色	白, 黑, 赤, 黃, 色, 彡
15	城鄉	田, 里, 阜, 邑, 囗, 爿, 片
16	家畜	牛, 馬, 羊, 犬, 豕
17	野獸	犬, 鹿, 豸, 鳥, 隹, 魚, 鼠, 龍, 虍, 虫
18	動物軀體	毛, 角, 皮
19	植物	生, 艸, 屮, 竹, 木, 禾, 麻, 米, 麥, 瓜, 豆, 韭, 黍, 干
20	性質	生, 乙, 尸, 歹, 鬼, 无, 高, 勹, 用, 辰, 食, 隶
21	生命性質	大, 小, 老, 方, 口, 玄, 長, 襾, 辛, 甘, 鹵, 艮
22	無生命	日, 月, 夕, 金, 水, 雨, 而, 冫, 心, 火, 土, 石, 玉, 風, 几, 山, 谷, 爻, 示, 文, 攴

3.3.3 Ideographic Description Characters

In *hgct*, users can use the information of the IDCs to search the corpus. There are three types of information of a character concerning the IDC that describes it:

A. IDC (idc): the IDC itself

B. Component (compo): the components making up the character

C. Position (pos): the mapping of the components to the IDC’s slots

For example, the character 徹 is described by the IDC ⿲ (a). The three components of 徹—彳, 育, and 攵 (b), map to position 0 (left), 1 (center), and 2 (right) in ⿲ (c). Hence, each of these types of information, or the combinations of them, could

doi:10.6342/NTU202200369

be used to search the corpus. For instance, the query `[compo=" 水" & idc="vert2" & pos="1"]` exemplified in Table 3.1 uses all three types of information. This query corresponds to finding all characters with the above-to-below structure (⿱) that have the component 水 at the bottom position (position 1). If the attribute "`pos`" is left out (i.e., the position of the component 水 is unspecified), as in `[compo=" 水" & idc="vert2"]`, characters such as 沓 are also retrieved since the component 水 can now also appear at the top position (position 0).

Note that many Chinese characters have complicated nested structures. For instance, the character 燙 can be decomposed into 湯 + 火 at the first level. The component 湯 can further be decomposed into 氵 and 昜 at the next level, and 昜 into 旦 and 勿 at the third level, and so on. Currently, the search function in *hgct* only deals with the components decomposable at the first level. This means, for instance, the query `[compo=" 氵 "]` would not be able to retrieve the character 燙 since 氵 is nested at the second level, inside the first-level component 湯.

Another issue worth noting concerns the non-uniqueness of decomposing characters. That is, a character can be decomposed in different ways and is described by multiple distinct Ideographic Description Sequences (IDS). For instance, the character 丟 can be described as ⿱王厶 or as ⿱一去 according to the IDS database[8]. This non-uniqueness also highlights the fact that Unicode's IDC and IDS are used purely for descriptive purposes and do not in themselves present a working theory of the creating processes of the characters. Nonetheless, the framework provided by IDCs and IDSs is still valuable as it provides a consistent and large-scale method to decompose Chinese characters into a set of common components. Furthermore, though non-uniqueness cases of decomposing characters do exist, the bulk of Chinese char-

[8]https://github.com/cjkvi/cjkvi-ids/blob/master/ids.txt

doi:10.6342/NTU202200369

acters do not suffer from this, at least at the first level of the decomposition, since left-to-right (⿰) and above-to-below (⿱) organizations account for most Chinese characters[9], and characters with these organizations have clear-delineated internal structures making them less likely to have non-unique decompositions at the first level.

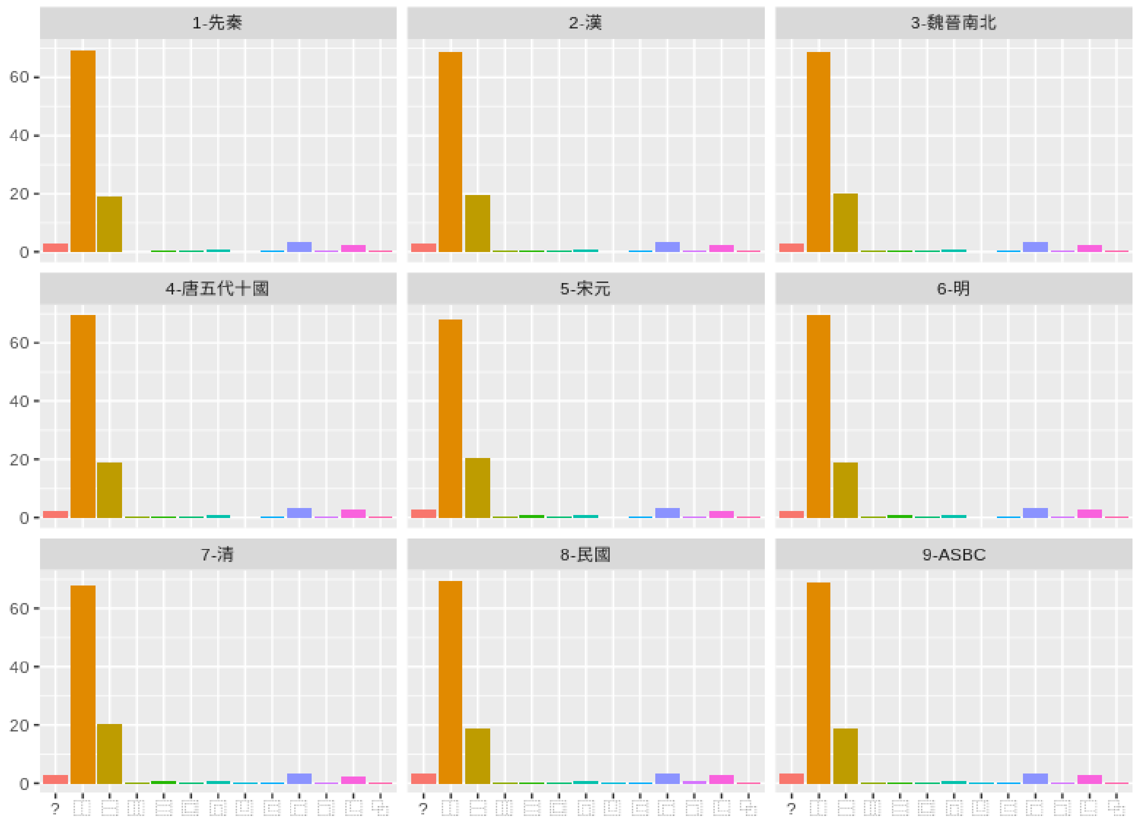


Figure 3.2: Proportion of Chinese character organizations (in terms of IDC) across time. The vertical axes indicate the percentage of the characters with a specific IDC at a particular time in history. The texts at different historical times, except the last one (ASBC), are collected from the Chinese Text Project (https://ctext.org)

[9]In a corpus of 458,738 characters sampled from the Academia Sinica Balanced Corpus, there are 4110 character types, among which, 2454 (59.7%) belong to the left-to-right (⿰) organization and 1019 (24.8%) to the above-to-below (⿱) organization. Together, left-to-right (⿰) and above-to-below (⿱) account for 84.5% of the character types.

We arrive at a similar picture even when historical texts are used. As shown in Figure 3.2, starting from the ages of pre-Qin (先秦) to modern Chinese (ASBC), characters with the left-to-right (⿰) organization make up nearly 70% of all characters, and characters with the above-to-below (⿱) organization make up nearly 20% of all characters. Note that token frequencies instead of type frequencies are used here.

3.4 Corpus Analysis

In addition to searching the corpus, a suite of functions is provided for quantitative analyses of the corpus (or subcorpus). This suite of functions includes traditional frequency-based corpus analysis functions (frequency lists/distributions, dispersion, character n-grams, and character collocations) but is tailored to deal with analyses below the unit of the word (i.e., characters and its components). Also included in the suite is a tentative function built upon the notion of productivity, which was originally applied to words to quantify morphological productivity (Baayen, 1993, 2009), is applied here to character glyphs (either radicals or components). Below, we describe each of the functions in detail. Appendix B provides a tutorial with code to use these functions.

3.4.1 Frequency Lists

Frequency distributions (and frequency lists[10]) at various levels can be obtained from *hgct*'s `CompoAnalysis.freq_distr()`. The top-most level is the character level, below which are the levels of the glyph—the radical level and the IDC level. The character-level distribution lists each type of character along with its (relative) frequency in the corpus. The radical-level and the IDC-level distributions are derived from the character-level distribution. The radical-level distribution lists the frequencies of the Kangxi radicals of the characters in the corpus, and the IDC-level distribution lists the frequencies of the IDCs of the characters in the corpus. Note that there are two ways to derive the distributions of radicals and IDCs from the character distribution—the first one uses the raw character distribution, and

[10]A frequency list is easily derived from a frequency distribution by sorting the items in the distribution according to their frequencies.

doi:10.6342/NTU202200369

the second discards the frequency of each character and considers types only. For instance, in a toy corpus with four characters: 你, 你, 們, and 它, the radical distribution derived by the first method (character token count) is `[人（亻）: 3, 宀: 1]`, and the radical distribution derived by the second method (character type count) is `[人（亻）: 2, 宀: 1]`. This is controlled by a parameter, `use_chr_types`, in `CompoAnalysis.freq_distr()`.

Glyphs can also be used to restrict the range of characters in a character distribution. For instance, to obtain the distribution of characters with a particular radical, such as 人 (亻), one can pass the radical 人 to the parameter `radical` in `CompoAnalysis.freq_distr()`. Similarly, the distribution of characters that are described by a particular IDC could also be retrieved by passing the name of the IDC (e.g., "vert2") to the parameter `idc` in `CompoAnalysis.freq_distr()`.

3.4.2 Dispersion

Measures of dispersion quantify the degree to which occurrences of a particular unit (e.g., word) are distributed evenly throughout the corpus (Gries, 2020). In *hgct*, the unit of occurrences can be defined flexibly. As long as the unit is definable in CQLS patterns, dispersion of the unit could be computed from the corpus. To obtain the dispersion measures, one simply needs to pass the results of a CQLS query (a sequence of concordance lines) into `Dispersion.pattern_dispersion()`. For instance, to compute the dispersion of characters with the component 冫, one needs to first write the CQLS query `[compo="冫"]`, pass it to the search function, and pipe the search results to `Dispersion.pattern_dispersion()`. The dispersion of a more complex unit, such as a sequence of two characters with the common radical 艸, can also be retrieved by defining the unit with the query `[radical="`

doi:10.6342/NTU202200369

`艸"] [radical=" 艸"]`.

To compute dispersion measures, it is required to first chunk the corpus into (more-or-less even) parts. Different dispersion measures, then, quantify the tendency of the occurrences of a particular unit to occur evenly (or unevenly) in these corpus parts. In *hgct*, the text level is used for chunking the corpus into parts. Taking the corpus structure in Code Block 3.2 as an example, the subcorpora each would be chunked into three parts, and the corpus as a whole would be chunked into six parts.

Six dispersion measures are provided in *hgct*[11]. They are all computed from these common definitions below. Suppose here the unit α is the target of computing dispersion[12]:

1. corpus size in characters (N)
2. number of corpus parts (n)
3. percentages of the sizes of the n corpus parts (s)
4. the frequency of the unit α in the corpus (f)
5. frequencies of the unit α in each corpus part (v)
6. percentages of the unit α in each corpus part (p)

Equation (3.1) lists the six dispersion measures (Gries, 2020) provided in *hgct*, each of which are expressed by the notations defined in the list above.

[11] https://github.com/liao961120/hgct/blob/main/hgct/dispersionStats.py
[12] The following definitions are taken and modified from Gries (2020, p. 102).

doi:10.6342/NTU202200369

$$
\begin{aligned}
&Range = \text{number of corpus parts containing } \alpha \\
&DP = 0.5 \times \sum_{i=1}^{n} \left| \frac{v_i}{f} - s_i \right| \\
&DP_{norm} = \frac{DP}{1 - min(s)} \\
&KL\ divergence = \sum_{i=1}^{n} \frac{v_i}{f} - log_2(\frac{v_i}{f \times s_i}) \\
&Juilland's\ D = 1 - \frac{sd_{population}(p)}{mean(p)} \times \frac{1}{\sqrt{n-1}} \\
&Rosengren's\ S = (\sum_{i=1}^{n} \sqrt{s_i \cdot v_i})^2 \times \frac{1}{f}
\end{aligned}
\tag{3.1}
$$

3.4.3 Character N-gram and Collocation

An n-gram is a sequence of *n* consecutive tokens appearing in the corpus. Thus, an n-gram in *hgct* corresponds to a sequence of *n* consecutive characters since a token corresponds to a character, not a word, in *hgct*'s corpus. Frequency distributions of n-grams can be obtained with the function `Concordancer.freq_distr_ngrams()`.

A collocation is a pair of tokens that co-occurs near each other (i.e., within a certain window size) in texts greater than chance. Various association measures are available to quantify the strengths of attraction between a pair of tokens. A strong attraction indicates a large positive deviation from the expected co-occur

doi:10.6342/NTU202200369

frequency based on the frequencies of the two tokens in the corpus. That is, the pair of tokens co-occur more often than when tokens are assumed to occur randomly in texts. Association measures can be computed for a given pair of tokens based on the contingency table of the two tokens' co-occurring frequencies. Table 3.3 illustrates a schematic contingency table for the token pair α and β. Based on contingency tables such as Table 3.3, the expected frequencies could be computed with the empirical marginal frequencies (R_1, R_2, C_1, C_2, and N), as shown in Table 3.4. After obtaining the observed (O_{11}, O_{12}, O_{21}, O_{22}) and the expected (E_{11}, E_{12}, E_{21}, E_{22}) frequencies, various association measures can be computed by arithmetically manipulating these values. Equation (3.2) lists the seven association measures provided in *hgct*.

doi:10.6342/NTU202200369

Table 3.3: Schematic contingency table of the token α and β.

	α	Other than α	Total
β	O_{11}	O_{12}	R_1
Other than β	O_{21}	O_{22}	R_2
Total	C_1	C_2	N

$$MI = log2(\frac{O_{11}}{E_{11}})$$

$$\begin{aligned}\chi^2 &= \sum_j \frac{(O_j - E_j)^2}{E_j} \\ &= \frac{(O_{11}-E_{11})^2}{E_{11}} + \frac{(O_{12}-E_{12})^2}{E_{12}} + \frac{(O_{21}-E_{21})^2}{E_{21}} + \frac{(O_{22}-E_{22})^2}{E_{22}}\end{aligned}$$

$$\begin{aligned}G^2 &= 2\sum_j O_j log\frac{O_j}{E_j} \\ &= 2(O_{11}log\frac{O_{11}}{E_{11}} + O_{12}log\frac{O_{12}}{E_{12}} + O_{21}log\frac{O_{21}}{E_{21}} + O_{22}log\frac{O_{22}}{E_{22}})\end{aligned} \tag{3.2}$$

$$Dice = \frac{2\ O_{11}}{O_{11} + O_{12} + O_{11} + O_{21}}$$

$$\Delta P_{2|1} = \frac{O_{11}}{O_{11}+O_{12}} - \frac{O_{21}}{O_{21}+O_{22}} \qquad \Delta P_{1|2} = \frac{O_{11}}{O_{11}+O_{21}} - \frac{O_{12}}{O_{12}+O_{22}}$$

$$Attract_{Fisher} = -log\rho_{Fisher\ Exact\ Test}$$

Similar to n-grams, collocations in *hgct* are character-based. *hgct* implements two different types of collocation extracting functions. The first one derives from character bigrams. After bigrams are retrieved from the corpus, association mea-

doi:10.6342/NTU202200369

Table 3.4: Expected frequency of co-occurrences based on empirical marginal frequencies (R, R, C, C, and N)

	α	Other than α	Total
β	$E_{11} = \frac{R_1 C_1}{N}$	$E_{12} = \frac{R_1 C_2}{N}$	R_1
Other than β	$E_{21} = \frac{R_2 C_1}{N}$	$E_{22} = \frac{R_2 C_2}{N}$	R_2
Total	C_1	C_2	N

sures are computed for each bigram type. Collocations are thus bigram types that score high in association measures. Collocations can be retrieved this way via the function `Concordancer.bigram_associations()`. The second one conceptualizes each collocation as consisting of a node and a collocate. Given a node of interest, the aim is to retrieve a set of tokens, occurring within a certain window size of around the node, that is most strongly attracted to the node. This set of strongly attracted tokens is considered to be the collocates of the node. This is implemented in the function `Concordancer.collocates()`, which takes a CQLS query to define the node and returns a list of collocates along with node-collocate associations.

3.4.4 Productivity of Chinese Character Glyphs

Productivity is often discussed in the context of words. Indeed, the study of statistical measures of productivity in corpus linguistics begins with the notion of morphological productivity (Baayen, 1993, 2009). Below, we briefly review the statistical measures of morphological productivity.

3.4.4.1 Measuring Morphological Productivity

Many words in a language have internal structures. For instance, English words such as *childhood*, *motherhood*, and *parenthood* contain the suffix *-hood*. The sets of words sharing formatives such as *-hood* above and *-ness* as in *happiness*, *hardiness*, and *robustness* are referred to as morphological categories (Baayen, 1993, 2009).

doi:10.6342/NTU202200369

These morphological categories differ in how productive they are. The morphological category *-ness*, for instance, is more productive than the category *-hood* in that the former has a larger membership and is more likely to attract new members (new words with *-ness* are coined more often than *-hood*).

Based on these intuitions, three corpus-based measures for quantifying different aspects of productivity are proposed (Baayen, 1993, 2009). The first measure, ***realized productivity*** (P_{realized}), quantifies a morphological category C's extent of use. P_{realized} is estimated by the type count of the category C in a corpus of size N, denoted as $V(C, N)$[13]. The second measure, ***expanding productivity*** ($P_{\text{expanding}}$), looks at the rate at which a category C attracts new members. This is based on the notion that, even if a category has a small membership, it can still be productive in the sense that many new words are being formed according to the productive rule of this morphological category. The rate at which new words are formed is known as the growth rate of the vocabulary (Baayen, 2001) and is estimated by the ratio of the number of hapax legomena (words that occur once) in a corpus of N tokens to the size of the corpus. In Baayen (2001)'s notation, this is $V(1, N)/\mathrm{N}$. $P_{\text{expanding}}$ is then defined as the contribution of the morphological category C to the growth rate of the vocabulary, which is $P_{\text{expanding}} = V(1, C, N)/V(1, N)$. $V(1, C, N)$ is the number of hapax legomena that belong to the category C in a corpus of N tokens. The third measure, ***potential productivity*** ($P_{\text{potential}}$), reflects the degree to which a category C is saturated. A category C is saturated if its membership hardly grows. This is estimated by the growth rate of the vocabulary of the category C itself, which is $P_{\text{potential}} = V(1, C, N)/\mathrm{N}(C)$. $\mathrm{N}(C)$ is the number of tokens belonging to category C in the corpus. Equation (3.3) summarizes the

[13]$V(C, N)$ is read as the vocabulary size of the category C in a corpus of N tokens.

doi:10.6342/NTU202200369

three measures of productivity introduced above.

$$
\begin{gathered}
P_{realized} = V(C, N) \\
P_{expanding} = \frac{V(1, C, N)}{V(1, N)} \\
P_{potential} = \frac{V(1, C, N)}{N(C)}
\end{gathered}
\tag{3.3}
$$

3.4.4.2 Applying Productivity Measures to Chinese Character Glyphs

Similar to words, Chinese characters have internal structures. As discussed in Section 3.3.2 and Section 3.3.3, Chinese characters can be decomposed into glyphs that are shared across different characters. Therefore, an analogy between the internal structure of words and the internal structure of characters can be drawn. Characters sharing a common glyph can be considered as members of the same "morphological" category. Thus, if we take the Kangxi radical system, we can partition all Chinese characters into 214 categories. We can also define categories by the shared IDC components in characters. For instance, 靜, 蜻, 氰, and 情 all belong to the category of 青. Even more abstract categories can be defined based on the IDCs themselves. Members of a category, then, are characters sharing a similar internal organization. For example, characters having a left-to-right organization, such as 錤, 根, 敢, 祈, and 液, would belong to the category of the IDC ⿰. Given these definitions of category membership, the three measures of productivity in Equation (3.3) can be readily applied. In *hgct*, this is handily provided in the

doi:10.6342/NTU202200369

Table 3.5: Table 4.2 in Myers (2019)

Radical position	$P_{expanding}$	Sample size	Proportion in sample
Left	0.7	2,338	0.7
Right	0.05	221	0.07
Top	0.12	445	0.13
Bottom	0.12	349	0.1

function `CompoAnalysis.productivity()`, where the user can compute productivity measures for a given category. The category can be defined by a Kangxi radical, a character component, or an IDC.

A similar idea of applying productivity measures to characters is also seen in Myers (2019). Myers (2019, pp. 122–125) conducts two case studies on Chinese characters to explore the productivity of different character types. The first study computes the productivity score of semantic-phonetic character and semantic compound character, showing that the semantic-phonetic category has a higher score of *expanding productivity* than the semantic compound character category. The second study, which is more related to the current work, explores the productivity of four different radical positions—top, bottom, left, and right of the semantic-phonetic characters. The result shows that the category with the "left" radical position scores the highest, and by a large margin to the second-highest position, in *expanding productivity*. Table 3.5, taken from Table 4.2 in Myers (2019), summarizes the productivity of the four radical positions.

The high productivity of the left radical suggests that it is related to the left-to-right organization of the Chinese characters. Based on this intuition, we computed productivity measures for the twelve IDCs from a sampled corpus (404,924 characters) constructed from Academia Sinica Balanced Corpus. Table 3.6 reports the results. We can see that indeed, the IDC ⿰ has the largest scores in all three mea-

doi:10.6342/NTU202200369

Table 3.6: Productivity of the twelve IDCs in a sampled corpus of ASBC.

Name	Shape	$P_{realized}$	$P_{expanding}$	$P_{potential}$	Proportion in sample
horz2	⿰	2454	0.6715	0.0022	0.4141
vert2	⿱	1019	0.1895	0.0009	0.2964
horz3	⿲	26	0.0054	0.0015	0.0049
vert3	⿳	39	0.0000	0.0000	0.0100
encl	⿴	18	0.0072	0.0007	0.0142
surN	⿵	45	0.0108	0.0008	0.0193
surU	⿶	6	0.0000	0.0000	0.0040
curC	⿷	12	0.0018	0.0019	0.0013
surT	⿸	176	0.0578	0.0017	0.0457
sur7	⿹	41	0.0108	0.0014	0.0109
surL	⿺	123	0.0271	0.0011	0.0332
over	⿻	32	0.0018	0.0001	0.0255
	NULL	119	0.0162	0.0002	0.1203

sures of productivity. Our results are consistent with Myers (2019), showing that the most productive structure in Chinese characters, according to the IDC, is the left-to-right organization.

doi:10.6342/NTU202200369

Code Block 3.1 JSON representation of an example input corpus. Refer to the next code block for the corresponding directory structure.

```
[
   {
      "id": "01",
      "text": [
          {
              "c": ["這是一個句子。", "這是第二個句子。"],
              "id": "01/text1.txt",
          },
          {
              "c": [
                  "這是第二篇裡的一個句子。", "這是第二個句子。"],
              "id": "01/text2.txt",
          },
          {
              "c": ["這是第三篇裡的一個句子。", "這是第二個句子。"],
              "id": "01/text3.txt",
          }
      ]
   },
   {
      "id": "02",
      "text": [
          {
              "c": ["這是一個句子。", "這是第二個句子。"],
              "id": "02/text1.txt",
          },
          {
              "c": ["這是第二篇裡的一個句子。", "這是第二個句子。"],
              "id": "02/text2.txt",
          },
          {
              "c": ["這是第三篇裡的一個句子。", "這是第二個句子。"],
              "id": "02/text3.txt",
          },
      ]
   }
]
```

doi:10.6342/NTU202200369

Code Block 3.2 Directory structure of an example input corpus. Refer to the previous code block for the corresponding representation in JSON.

```
corpus
  01
     text1.txt
     text2.txt
     text3.txt
  02
      text1.txt
      text2.txt
      text3.txt
```

doi:10.6342/NTU202200369

doi:10.6342/NTU202200369

Chapter 4

Case Study: Radical Semantic Information in Words

In Chapter 3, it is shown how *hgct* can be used to decompose Chinese characters and peek inside their internal structures. By systematically teasing characters apart into glyphs, new possibilities of quantitatively exploring the writing system of Chinese are opened up. Given these opportunities, we are interested in asking the question: to what degree is the semantic information of a word related to the character glyphs comprising the word? Intuitively, users of Chinese characters recognize that radicals or components of Chinese characters carry semantic information that is often related to the word. For instance, plant-related terms, such as 花, 草, 樹, 薔薇, and 橄欖, often consist of characters with the radical 艸 and 木. Previous work (Chou & Huang, 2006, 2010; C.-R. Huang & Hsieh, 2015) on the analysis of the conceptual organization encoded in Chinese characters and *Shuowen*'s radical system suggests that such relationships are prominent, at least in radicals such as 艸, 刀, and 口. It is assumed in the work that the 540 *Shuowen* radicals each represent a basic concept, and characters with a particular radical are conceptually dependent on that concept (C.-R. Huang & Hsieh, 2015). Based on these assumptions, exten-

doi:10.6342/NTU202200369

sive analyses were carried out on more than 3000 frequently used characters (Chou & Huang, 2010). An interesting finding is that characters with a given radical, such as radicals representing natural objects as in 艸 'grass', 牛 'cow', 馬 'horse', and 木 'wood', often show conceptual clusterings similar to the Qualia structure proposed by Pustejovsky (1995).

In this chapter, we report our quantitative exploration—by leveraging existing data—of the relationship between semantic information encoded at the level of words and the level of character glyphs (radicals in this case). The assumption is that there exists some correlation between the two. However, the strength of the correlation and, even more, how to operationalize the relation between the two, is unclear. In the remainder of this chapter, the details of our study are reported. We first describe the data and how the semantic information encoded in radicals and words is operationalized. Subsequent sections then report the relationships between the radicals and the words revealed in the data. The code for reproducing the study is documented in an IPython notebook accessible through GitHub[1].

4.1 Data

The semantic information encoded at the radical level and at the word level is operationalized by introducing two independent sources of semantic type information. The first semantic type information is taken directly from Ma (2016) described in Chapter 3, which provides a semantic classification system for 171 Kangxi radicals. Table 3.2 lists the 22 semantic types of these 171 Kangxi radicals in Ma (2016)'s system. The second semantic type information derives from Cilin[2] (Mei,

[1]https://github.com/liao961120/cilin/blob/main/etc/radical_semantic_classification.ipynb

[2]The electronic version of Cilin is compiled and maintained by the Research Center for Social Computing and Information Retrieval at Harbin Institute of Technology, which can be found at

1983), which is a hierarchically organized thesaurus (synonym dictionary) that arranges words into sets of synonyms at five levels of abstraction. At the first level of Cilin, words are divided into twelve categories (or sets of synonyms)—*Human* (A), *Objects* (B), *Time and Space* (C), *Abstract Entities* (D), *Traits* (E), *Actions* (F), *Mental Activities* (G), *Physical Activities* (H), *Phenomena and States* (I), *Relations* (J), *Auxiliaries* (K), and *Honorifics* (L). Each of these first-level categories is further divided into subcategories, which comprise the synonym sets at the second level. The third-level synonym sets derive similarly from splitting up each of the second-level synonym sets. This partition process is repeated until the fifth level to arrive at the five-level organization of Cilin. Table 4.1 lists the first two levels of Cilin's hierarchy.

In Section 4.2, the twelve categories at the first level of Cilin are taken as the semantic types of words to explore the correlation between word-level and radical-level semantic information. In Section 4.3, the first level and third level of Cilin are used in a classification task to help elucidate the importance of each radical semantic type for predicting lexical classes. Radical semantic type information derived from the third-level word categories are used as features to predict the first-level class labels of these word categories.

4.2 Association of Radical and Word Semantic Types

As an initial exploration of the radical-word semantic relationship, we focus here on single-character words. Thus, we take all single-character words in Cilin and partition them into twelve word categories (see Table 4.1) based on Cilin's first hierarchy level. Table 4.2 lists the word counts of single-character words, two-

http://ir.hit.edu.cn/117.html.

doi:10.6342/NTU202200369

Table 4.1: The first two levels in Cilin (Mei, 1983).

A 人	B 物	D 抽象事物	H 活動
Aa: 泛稱	Ba: 統稱	Da: 事情/情況	Ha: 政治活動
Ab: 男女老少	Bb: 擬狀物	Db: 事理	Hb: 軍事活動
Ac: 體態	Bc: 物體的部分	Dc: 外貌	Hc: 行政管理
Ad: 籍屬	Bd: 天體	Dd: 性能	Hd: 生產
Ae: 職業	Be: 地貌	De: 性格/才能	He: 經濟活動
Af: 身份	Bf: 氣象	Df: 意識	Hf: 交通運輸
Ag: 狀況	Bg: 自然物	Dg: 比喻物	Hg: 教衛科研
Ah: 親人/眷屬	Bh: 植物	Dh: 臆想物	Hh: 文體活動
Ai: 輩次	Bi: 動物	Di: 社會/政法	Hi: 社交
Aj: 關係	Bj: 微生物	Dj: 經濟	Hj: 生活
Ak: 品行	Bk: 全身	Dk: 文教	Hk: 宗教活動
Al: 才識	Bl: 排泄物/分泌物	Dl: 疾病	Hl: 迷信活動
Am: 信仰	Bm: 材料	Dm: 機構	Hm: 公安/司法
An: 醜類	Bn: 建築物	Dn: 數量/單位	Hn: 惡行
	Bo: 機具		
	Bp: 用品		
	Bq: 衣物		
	Br: 食品/藥品/毒品		
E 特徵	F 動作	G 心理活動	C 時間與空間
Ea: 外形	Fa: 上肢動作	Ga: 心理狀態	Ca: 時間
Eb: 表象	Fb: 下肢動作	Gb: 心理活動	Cb: 空間
Ec: 顏色/味道	Fc: 頭部動作	Gc: 能願	
Ed: 性質	Fd: 全身動作		
Ee: 德才			
Ef: 境況			
I 現象與狀態	J 關聯	K 助語	L 敬語
Ia: 自然現象	Ja: 聯繫	Ka: 疏狀	La: 敬語
Ib: 生理現象	Jb: 異同	Kb: 中介	
Ic: 表情	Jc: 配合	Kc: 聯接	
Id: 物體狀態	Jd: 存在	Kd: 輔助	
Ie: 事態	Je: 影響	Ke: 呼嘆	
If: 境遇		Kf: 擬聲	
Ig: 始末			
Ih: 變化			

character words, and all word types in these word categories.

For each of the twelve word categories, the radical semantic types from Ma (2016) are assigned to words/characters[3]. The joint frequency of each pair of word and radical semantic types is then counted. This results in a 12-by-23 matrix of

[3]For characters with radicals not in Ma (2016)'s system, the tag of NULL is given as the radical semantic type.

doi:10.6342/NTU202200369

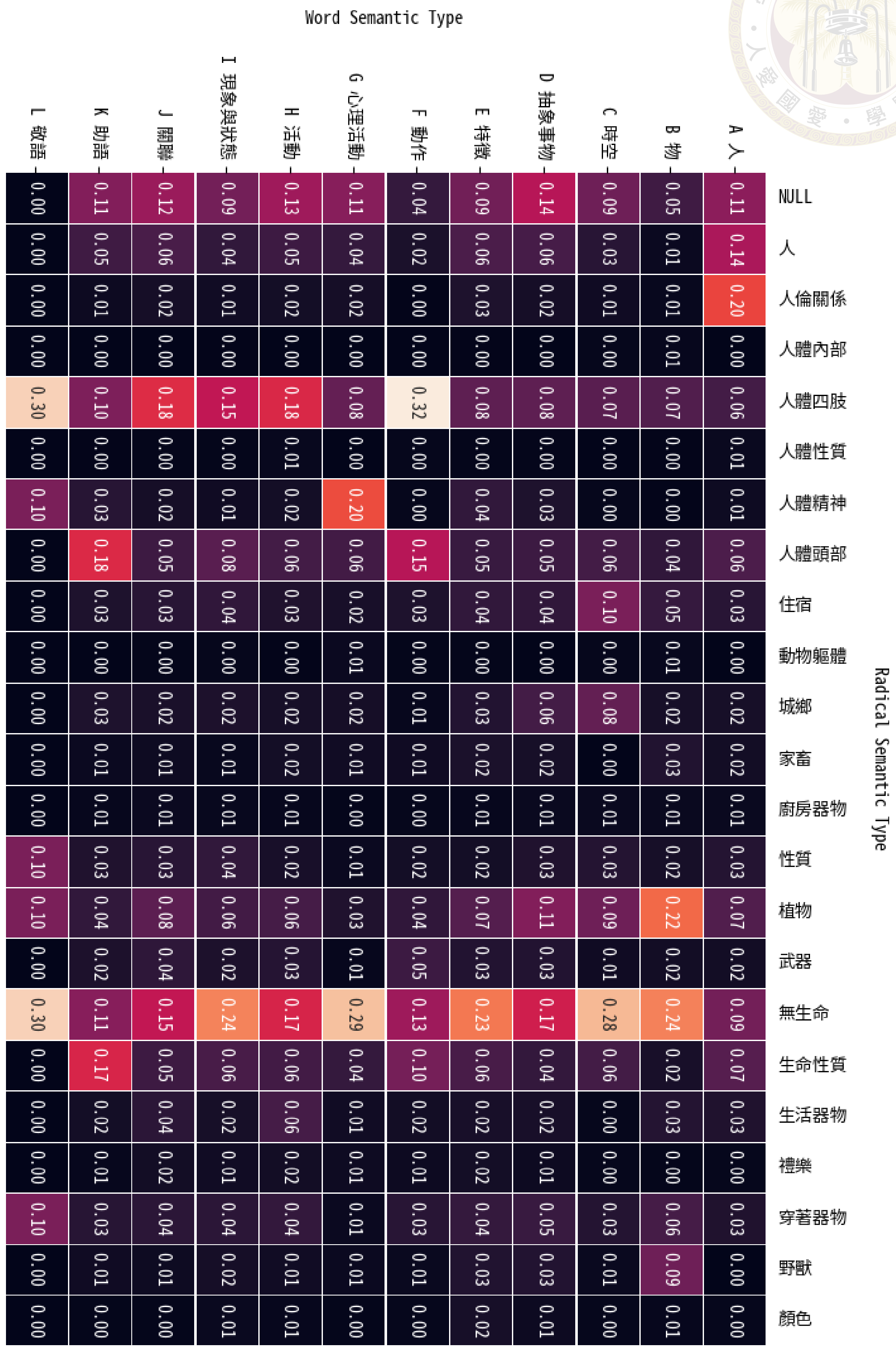
Table 4.2: The number of words, two-character words, and single-character words in each of the word categories at Cilin’s first hierarchy level.

	A 人	B 物	C 時空	D 抽象事物
Single-character	245 (4.51%)	1651 (9.14%)	218 (4.99%)	1069 (5.12%)
Two-character	2903 (53.48%)	10661 (59.03%)	2647 (60.64%)	9802 (46.98%)
All	5428	18060	4365	20865
	E 特徵	F 動作	G 心理活動	H 活動
Single-character	809 (8.99%)	622 (30.45%)	251 (9.37%)	1072 (8.85%)
Two-character	4311 (47.92%)	1215 (59.47%)	1449 (54.09%)	8686 (71.68%)
All	8996	2043	2679	12117
	I 現象與狀態	J 關聯	K 助語	L 敬語
Single-character	558 (10.39%)	297 (15.46%)	436 (22.11%)	9 (9.68%)
Two-character	3272 (60.95%)	1248 (64.97%)	1202 (60.95%)	71 (76.34%)
All	5368	1921	1972	93

joint frequencies, with the rows corresponding to the twelve semantic types of the words, the columns corresponding to the twenty-three (22 radical semantic types + 1 null type) semantic types of the radicals, and the cells corresponding to the joint frequencies of particular combinations of word and radical semantic types. Based on this matrix of joint frequencies, two matrices of conditional probabilities can be derived. The first is a matrix of probabilities of a radical semantic type given a word semantic type, $p(Radical\ Type|Word\ Type)$. These probabilities could tell us about the internal composition of a word category in terms of the semantic types derived from the radicals. This matrix is visualized as a heatmap in Figure 4.1. The second matrix, conversely, holds the conditional probabilities of a word semantic type given a radical semantic type, $p(Word\ Type|Radical\ Type)$. These probabilities tell us about the certainty of predicting a word category from its radical semantic type. The heatmap visualizing the matrix is shown in Figure 4.2. The interpretation of these conditional probabilites cannot be based on absolute values of the probabilities but depends on the comparison between them. Below, we discuss what these two

doi:10.6342/NTU202200369

matrices reveal about the word-radical associations.

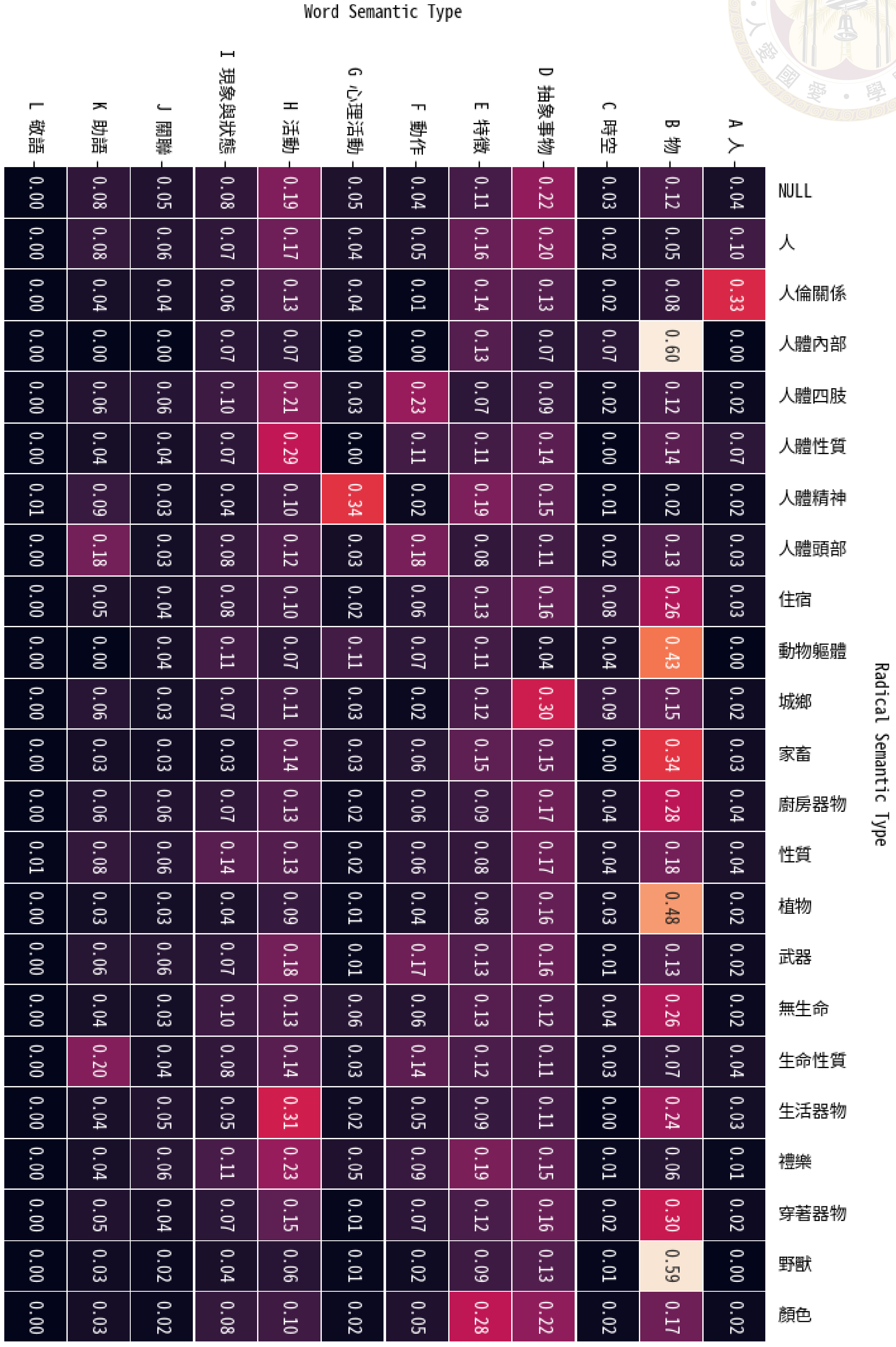


	L 敬語	K 助語	J 關聯	I 現象與狀態	H 活動	G 心理活動	F 動作	E 特徵	D 抽象事物	C 時空	B 物	A 人
NULL	0.00	0.11	0.12	0.09	0.13	0.11	0.04	0.09	0.14	0.09	0.05	0.11
人	0.00	0.05	0.06	0.04	0.05	0.04	0.02	0.06	0.06	0.03	0.01	0.14
人倫關係	0.00	0.01	0.02	0.01	0.02	0.02	0.00	0.03	0.02	0.01	0.01	0.20
人體內部	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00
人體四肢	0.30	0.10	0.18	0.15	0.18	0.08	0.32	0.08	0.08	0.07	0.07	0.06
人體性質	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.01
人體精神	0.10	0.03	0.02	0.01	0.02	0.20	0.00	0.04	0.03	0.00	0.00	0.01
人體頭部	0.00	0.18	0.05	0.08	0.06	0.06	0.15	0.05	0.05	0.06	0.04	0.06
住宿	0.00	0.03	0.03	0.04	0.03	0.02	0.03	0.04	0.04	0.10	0.05	0.03
動物軀體	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.01	0.00
城鄉	0.00	0.03	0.02	0.02	0.02	0.02	0.01	0.03	0.06	0.08	0.02	0.02
家畜	0.00	0.01	0.01	0.01	0.02	0.01	0.01	0.02	0.02	0.00	0.03	0.02
廚房器物	0.00	0.01	0.01	0.01	0.01	0.00	0.00	0.01	0.01	0.01	0.01	0.01
性質	0.10	0.03	0.03	0.04	0.02	0.01	0.02	0.02	0.03	0.03	0.02	0.03
植物	0.10	0.04	0.08	0.06	0.06	0.03	0.04	0.07	0.11	0.09	0.22	0.07
武器	0.00	0.02	0.04	0.02	0.03	0.01	0.05	0.03	0.03	0.01	0.02	0.02
無生命	0.30	0.11	0.15	0.24	0.17	0.29	0.13	0.23	0.17	0.28	0.24	0.09
生命性質	0.00	0.17	0.05	0.06	0.06	0.04	0.10	0.06	0.04	0.06	0.02	0.07
生活器物	0.00	0.02	0.04	0.02	0.06	0.01	0.02	0.02	0.02	0.00	0.03	0.03
禮樂	0.00	0.01	0.02	0.01	0.02	0.01	0.01	0.02	0.01	0.00	0.00	0.00
穿著器物	0.10	0.03	0.04	0.04	0.04	0.01	0.03	0.04	0.05	0.03	0.06	0.03
野獸	0.00	0.01	0.01	0.02	0.01	0.01	0.01	0.03	0.03	0.01	0.09	0.00
顏色	0.00	0.00	0.00	0.01	0.01	0.00	0.00	0.02	0.01	0.00	0.01	0.00

Figure 4.1: Heatmap of $p(Radical\ Type|Word\ Type)$, conditional probabilities of a radical semantic type given a word semantic type.

Focusing first on the word-type conditioned matrix in Figure 4.1, what immedi-

doi:10.6342/NTU202200369



Radical Semantic Type	A 人	B 物	C 時空	D 抽象事物	E 特徵	F 動作	G 心理活動	H 活動	I 現象與狀態	J 關聯	K 助語	L 敬語
NULL	0.04	0.12	0.03	0.22	0.11	0.04	0.05	0.19	0.08	0.05	0.08	0.00
人	0.10	0.05	0.02	0.20	0.16	0.05	0.04	0.17	0.07	0.06	0.08	0.00
人倫關係	0.33	0.08	0.02	0.13	0.14	0.01	0.04	0.13	0.06	0.04	0.04	0.00
人體內部	0.00	0.60	0.07	0.07	0.13	0.00	0.00	0.07	0.07	0.00	0.00	0.00
人體四肢	0.02	0.12	0.02	0.09	0.07	0.23	0.03	0.21	0.10	0.06	0.06	0.00
人體性質	0.07	0.14	0.00	0.14	0.11	0.11	0.00	0.29	0.07	0.04	0.04	0.00
人體精神	0.02	0.02	0.01	0.15	0.19	0.02	0.34	0.10	0.04	0.03	0.09	0.01
人體頭部	0.03	0.13	0.02	0.11	0.08	0.18	0.03	0.12	0.08	0.03	0.18	0.00
住宿	0.03	0.26	0.08	0.16	0.13	0.06	0.02	0.10	0.08	0.04	0.05	0.00
動物軀體	0.00	0.43	0.04	0.04	0.11	0.07	0.11	0.07	0.11	0.04	0.00	0.00
城鄉	0.02	0.15	0.09	0.30	0.12	0.02	0.03	0.11	0.07	0.03	0.06	0.00
家畜	0.03	0.34	0.00	0.15	0.15	0.06	0.03	0.14	0.03	0.03	0.03	0.00
廚房器物	0.04	0.28	0.04	0.17	0.09	0.06	0.02	0.13	0.07	0.06	0.06	0.00
性質	0.04	0.18	0.04	0.17	0.08	0.06	0.02	0.13	0.14	0.06	0.08	0.01
植物	0.02	0.48	0.03	0.16	0.08	0.04	0.01	0.09	0.04	0.03	0.03	0.00
武器	0.02	0.13	0.01	0.16	0.13	0.17	0.01	0.18	0.07	0.06	0.06	0.00
無生命	0.02	0.26	0.04	0.12	0.13	0.06	0.06	0.13	0.10	0.03	0.04	0.00
生命性質	0.04	0.07	0.03	0.11	0.12	0.14	0.03	0.14	0.08	0.04	0.20	0.00
生活器物	0.03	0.24	0.00	0.11	0.09	0.05	0.02	0.31	0.05	0.05	0.04	0.00
禮樂	0.01	0.06	0.01	0.15	0.19	0.09	0.05	0.23	0.11	0.06	0.04	0.00
穿著器物	0.02	0.30	0.02	0.16	0.12	0.07	0.01	0.15	0.07	0.04	0.05	0.00
野獸	0.00	0.59	0.01	0.13	0.09	0.02	0.01	0.06	0.04	0.02	0.03	0.00
顏色	0.02	0.17	0.02	0.22	0.28	0.05	0.02	0.10	0.08	0.02	0.03	0.00

Figure 4.2: Heatmap of $p(Word\ Type|Radical\ Type)$, conditional probabilities of a word semantic type given a radical semantic type.

ately pops up is the column of the radical semantic type *Inanimate* (無生命). This reveals the fact that the radical semantic category *Inanimate* participates in various kinds of word categories and makes up a nontrivial proportion for each of them, except for the category *Human* (人), which makes sense as humans are animate entities. Another revealing radical semantic type is the *Human Limbs* (人體四肢). Radicals belonging to this semantic type make up the greatest proportion (32%) of the word category *Actions* (動作). This matches our intuition. A glance at the composite of the word category reveals why—many action words such as 拭, 蹦, 踩, 扛, 跺, 推, and 捏 are related to actions done with hands (拭, 扛, 推, and 捏) and actions done with feet (蹦, 踩, and 跺), and there is an accurate correspondence between the radicals of the single-character words (手 'hand' and 足 'foot') and the meaning of the words ('done with hands' and 'done with feet'). This also explains why the radical semantic type *Human Limbs* makes up a significant proportion of the word categories *Physical activities* (活動), *Phenomena and States* (現象與狀態), and *Relations* (關聯). Since many members of these categories are verbs, and verbs are often denoted by Chinese characters with limbs-related radicals (e.g., 手, 足, 肉, and 辵), the radical semantic type *Human Limbs* thus constitutes a non-negligible proportion of these word categories.

We now turn our focus to the radical-type conditioned matrix in Figure 4.2. In this matrix, the row indicated by the word category *Objects* (物) has numerous cells with high probabilities. The pattern becomes more sensible when we look down a level in the composition of the word category *Objects*. For instance, the subcategory *Body* (全身, Bk in Table 4.1) under *Objects* explains the high conditional probability of *Objects* given the radical semantic type *Human Organs* (人體內部). Most of

doi:10.6342/NTU202200369

the characters with the radical type *Human Organs* also fall under the word category *Body.* These characters with human-organ radicals, however, only comprise a tiny proportion of the word category *Objects*, making up only 0.5% (9/1651) of the single-character words in *Objects*, and thus result in the low word-type conditioned probability p(Human Organs|Objects). Hence, there is a large difference in probability between the corresponding cells of the two matrices. The radical semantic types *Animal Body Parts* (動物軀體), *Livestock* (家畜), *Kitchenware* (廚房器物), and *Wild Animal* 野獸 pattern similarly. These radical semantic types all correspond neatly to particular subcategories of *Objects.* Characters with radicals of *Animal Body Parts*[4] and *Kitchenware* fall under the word category *Daily Supplies* (用品), and Characters with radicals of *Livestock* and *Wild Animal* fall under the word category *Animal* (動物).

Four particular interesting word-radical type pairs are (1) the pair of *Mental Activities* (心理活動) and *Human Mind* (人體精神), (2) the pair of *Human* (人) and *Human Relations* (人倫關係), (3) the pair of *Auxiliaries* (助詞) and *Entities related to Human Head* (人體頭部), and (4) the pair of *Auxiliaries* and *Properties of Living Entities* (生命性質). These pairs are interesting since they all rank high in both conditional probabilities. Taking the pair (*Mental Activities*, *Human Mind*) for instance, p(Mental Activities|Human Mind) is the highest among all conditional probabilities given the radical type *Human Mind,* $p(Word\ Type|$Human Mind), and p(Human Mind|Mental Activities) is the highest among all conditional probabilities given the word type *Mental Activities*, p(Radical Type|Mental Activities). The high

[4]The characters falling under *Daily Supplies* are 觴, 毯, 毫, 觥, 角, and 觚. The characters 觴, 觥, and 觚, which have the radical 角, are related to tablewares used for wine drinking. The characters 毫 and 毯 have the radical 毛 and denotes Chinese brushes (毛筆) and blankets, respectively. The radicals of these characters, 角 'horn' and 毛 'fur', are said to denote the materials of these tools, which are taken from animals, in ancient China.

doi:10.6342/NTU202200369

ranks in both conditional probabilities imply that there is a strong one-to-one correspondence between the word and radical semantic type. That is, given the radical semantic type of a word, we can be highly certain about its word semantic type, and vice versa.

In the explorations above, we focus on the association, in terms of conditional probabilities, between word and radical semantic types. However, simple associations of word-radical pairs may not be enough. Comparisons between the pairs may be needed to arrive at new insights, as illustrated in the previous paragraph, in which the two matrices are eyeballed to locate important cells. This can be done automatically by taking another perspective to the problem. Instead of working on conditional probabilities directly, we can consider the process as a problem of classification. A classification model can be trained to predict the semantic type of a group of words from the words' radical semantic types. Algorithms of feature importance can then be applied to locate radical semantic types that are important for predicting the word semantic types. We describe these in detail in the next section.

4.3 Radical Semantic Type Importance

Similar to Section 4.2, we focus here only on single-character words in Cilin. To create groups of words that could be assigned class labels by a classification model, the word groups at the third level of Cilin are adopted, resulting in a dataset of 1428 word groups that would be used for training and testing later. The classification task is to assign one of the twelve word semantic types (i.e., first-level class labels in Cilin) to these word groups based on the radical semantic information within

doi:10.6342/NTU202200369

Table 4.3: The number of third-level word groups under each word semantic type.

A 人	B 物	C 時空	D 抽象事物
120	240	61	174
E 特徵	F 動作	G 心理活動	H 活動
180	58	44	296
I 現象與狀態	J 關聯	K 助語	L 敬語
129	43	77	6

the word group. To encode radical semantic information for the word groups, each word is assigned a radical semantic type of Ma (2016), and each word group is represented as a vector of counts of the radical semantic types in the group. This results in a 1428-by-23 count matrix with rows corresponding to word groups and columns corresponding to the radical semantic types. We can conceive this matrix as a document-term matrix by treating the word groups as documents and the radical semantic types as terms (or features). The matrix is then transformed to TF-IDF counts.

We adopt the Multinomial Naive Bayes classifier[5] for the classification task. Other tested algorithms (Random Forest and Support Vector Machine) achieved comparable performance on this dataset (with accuracy scores falling between 0.3 and 0.4). The Multinomial Naive Bayes is chosen for its computational efficiency. Fifty percent of the data is used for training and fifty percent for testing. Stratified sampling is used to pick out 50% samples for each of the twelve word semantic types from the full dataset. Table 4.3 shows the number of third-level word groups under each word semantic type. The results of the classification model on the testing dataset are summarized in Table 4.4. Table 4.5 shows the average performance of 50 random baseline models trained on shuffled data. In total, the model achieves 0.34 in accuracy and 0.26 in weighted F1 score on the test set. Varied performances

[5]The Lidstone smoothing parameter is set to 0.001.

are observed between classes within the test set. The best-predicted classes are *Human* (A), *Objects* (B), *Mental Activities* (G), and *Auxiliaries* (K), achieving a 0.2 or more increase in F1 score compared to the random baseline. This shows that these classes, compared to other classes, are best distinguished by the radical semantic types. The classes *Abstract Entities* (D), *Traits* (E), and *Physical Activities* (H) are less predictable and achieve about 0.1 or less increase in F1 score compared to the random baseline. The remaining classes *Time and Space* (C), *Actions* (F), *Phenomena and States* (I), *Relations* (J), and *Honorifics* (L) are not predicted by the model, showing that radical semantic types of single-character words cannot reliably distinguish these classes from others.

Table 4.4: Prediction performance of the classification model on the testing dataset.

	Precision	**Recall**	**F1**	**N$_{\textbf{test samples}}$**
A 人	0.4848	0.2667	0.3441	60
B 物	0.4300	0.7167	0.5375	120
C 時空	NULL	NULL	NULL	30
D 抽象事物	0.1842	0.0805	0.1120	87
E 特徵	0.2917	0.0778	0.1228	90
F 動作	NULL	NULL	NULL	29
G 心理活動	0.3750	0.1364	0.2000	22
H 活動	0.3005	0.8041	0.4375	148
I 現象與狀態	NULL	NULL	NULL	64
J 關聯	NULL	NULL	NULL	22
K 助語	0.5000	0.1842	0.2692	38
L 敬語	NULL	NULL	NULL	3
Weighted Avg.	0.2731	0.3436	0.2599	
Accuracy	0.3436			

We now turn our focus to the features, the radical semantic types, used for predicting the class labels. We are interested here in the contribution of each radical semantic type in predicting the word classes. The contribution of a feature can be retrieved from a classification model by dropping out the information of the feature during training. If the resulting model performs worse after dropping out the

doi:10.6342/NTU202200369

Table 4.5: Average performance of 50 random baseline models corresponding to the model in Table 4.5. The values in the parentheses are the standard deviations of the performance.

	Precision (std)	**Recall (std)**	**F1 (std)**
A 人	0.108 (0.27)	0.006 (0.03)	0.011 (0.04)
B 物	0.190 (0.07)	0.198 (0.08)	0.191 (0.07)
C 時空	0.003 (0.02)	0.001 (0.00)	0.001 (0.01)
D 抽象事物	0.152 (0.17)	0.021 (0.02)	0.034 (0.03)
E 特徵	0.125 (0.12)	0.024 (0.02)	0.038 (0.03)
F 動作	0.000 (0.00)	0.000 (0.00)	0.000 (0.00)
G 心理活動	0.000 (0.00)	0.000 (0.00)	0.000 (0.00)
H 活動	0.212 (0.01)	0.787 (0.06)	0.334 (0.02)
I 現象與狀態	0.022 (0.09)	0.001 (0.01)	0.002 (0.01)
J 關聯	0.000 (0.00)	0.000 (0.00)	0.000 (0.00)
K 助語	0.000 (0.00)	0.000 (0.00)	0.000 (0.00)
L 敬語	0.000 (0.00)	0.000 (0.00)	0.000 (0.00)
Weighted Avg.	0.1214 (0.0409)	0.2029 (0.0184)	0.1116 (0.0162)
Accuracy	0.2029 (0.0184)		

information, it is inferred that the feature contributes positively to the classification and is thus important for the classification model. Conversely, if the performance decreases or does not change without the information of the feature, the feature is considered to be unimportant. This can be operationalized by the algorithm of permutation feature importance, in which the classification model of interest is re-fitted multiple times by systematically removing the information of one of the features. The removal of a feature's information is done by randomly permuting the cells of that feature's column in the training data to scramble potential correlations between the feature and other parts of the data.

The algorithm of permutation feature importance is applied to the classification model. The importance of each radical semantic type to the model is summarized in Table 4.6. Measures of importance are quantified in terms of four different performance scores of the model—accuracy, precision, recall, and F1. The value in a cell indicates the *decrease* in a performance score after randomly permuting the fea-

doi:10.6342/NTU202200369

Table 4.6: Importance of each radical semantic type (feature) to model performance. Only the features in the first six rows have positive feature importance values in all four measures.

	Accuracy	**Precision**	**Recall**	**F1**
人體內部	0.0026	0.0015	0.0026	0.0020
生命性質	0.0026	0.0060	0.0026	0.0021
人倫關係	0.0025	0.0073	0.0025	0.0007
家畜	0.0021	0.0071	0.0021	0.0025
城鄉	0.0007	0.0121	0.0007	0.0008
人體頭部	0.0002	0.0073	0.0002	0.0010
生活器物	-0.0004	0.0003	-0.0004	0.0002
住宿	-0.0005	0.0006	-0.0005	0.0005
人體性質	-0.0011	-0.0065	-0.0011	-0.0016
性質	-0.0016	-0.0073	-0.0016	-0.0034
顏色	-0.0016	-0.0046	-0.0016	-0.0015
廚房器物	-0.0018	0.0045	-0.0018	-0.0012
植物	-0.0018	0.0000	-0.0018	-0.0005
穿著器物	-0.0021	0.0064	-0.0021	0.0026
無生命	-0.0023	0.0085	-0.0023	0.0016
人	-0.0035	-0.0050	-0.0035	-0.0033
NULL	-0.0035	-0.0051	-0.0035	-0.0020
人體精神	-0.0040	-0.0046	-0.0040	-0.0017
動物軀體	-0.0046	-0.0062	-0.0046	-0.0035
野獸	-0.0054	0.0004	-0.0054	-0.0033
武器	-0.0077	-0.0006	-0.0077	-0.0034
禮樂	-0.0077	-0.0047	-0.0077	-0.0045
人體四肢	-0.0096	0.0012	-0.0096	-0.0041

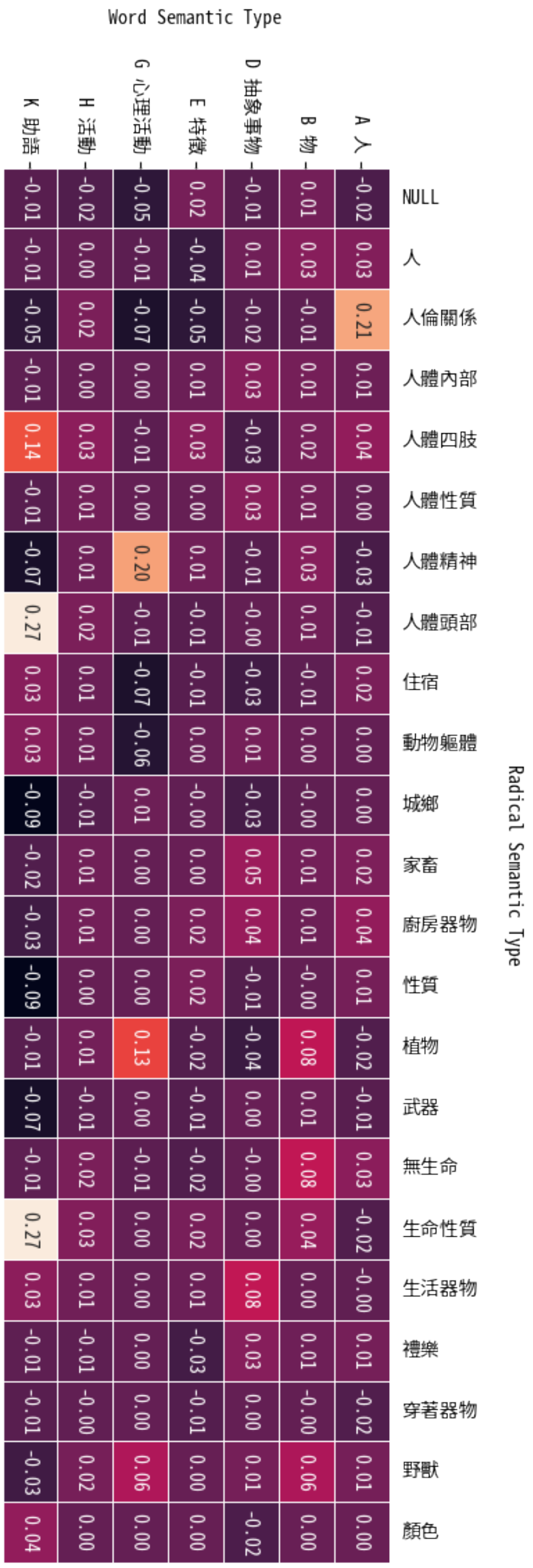
ture. Hence, a positive value indicates that the feature contributes positively to the classification, in terms of a particular measure. It is shown in Table 4.6 that only the features in the first six rows, *Human Organs* (人體內部), *Properties of Living Entities* (生命性質), *Human Relations* (人倫關係), *Livestock* (家畜), *Administrative Division* (城鄉), and *Entities related to Human Head* (人體頭部), have positive feature importance values in all four measures. These values of feature importance, however, are not informative enough to serve our purposes since they are measured against the full data. That is, we can learn that particular features are important to the classification as a whole, but we do not know the fine-grained contribution

doi:10.6342/NTU202200369

of a particular feature in predicting a class. This is simply solved by calculating feature importance in terms of the subsetted data for each class. The heatmap in Figure 4.3 lays out these fine-grained feature importance in terms of each class. F1 is used as the measure of feature importance here.

Compared to the heatmaps of the conditional probabilities given word and radical semantic types (Figure 4.1 and Figure 4.2), the heatmap in Figure 4.3 provides a much more compact view of the relationships between the semantic types of words and radicals. Similar to the conclusions we arrived at Section 4.2, the close relationships between the word-radical type pairs (*Human*, *Human Relations*), and (*Mental Activities*, *Human Mind*) are also highlighted in the heatmap here. This time, however, the relationships are summarized automatically without the need to manually compare two large heatmaps. The three highlighted cells in the row of the word semantic type *Auxiliaries* require further explanations. By manually inspecting the data, it is found that the pair (*Auxiliaries*, *Human Limbs*) resulted from the radicals 手 and 辵, which together accounts for 6.65% of all single-character words in *Auxiliaries*. The pairs (*Auxiliaries*, *Entities related to Human Head*) and (*Auxiliaries*, *Properties of Living Entities*) resulted from the radical 口, which is assigned both the radical semantic types *Entities related to Human Head* and *Properties of Living Entities*. The radical 口 accounts for 18.12% of the single-character words in *Auxiliaries*. A relationship not noticed in the comparison of the heatmaps of conditional probabilities is the pair (*Mental Activities*, *Plants*). Figure 4.3 shows that the radical semantic type *Plants* contributes moderately to the prediction of the word class *Mental Activities*. This relationship is neglected in the comparison of the heatmaps in Section 4.2 because it is a negative one. Both conditional probabilities,

doi:10.6342/NTU202200369



	K 助語	H 活動	G 心理活動	E 特徵	D 抽象事物	B 物	A 人
NULL	-0.01	-0.02	-0.05	0.02	-0.01	0.01	-0.02
人	-0.01	0.00	-0.01	-0.04	0.01	0.03	0.03
人倫關係	-0.05	0.02	-0.07	-0.05	-0.02	-0.01	0.21
人體內部	-0.01	0.00	0.00	0.01	0.03	0.01	0.01
人體四肢	0.14	0.03	-0.01	0.03	-0.03	0.02	0.04
人體性質	-0.01	0.01	0.00	0.00	0.03	0.01	0.00
人體精神	-0.07	0.01	0.20	0.01	-0.01	0.03	-0.03
人體頭部	0.27	0.02	-0.01	-0.01	-0.00	0.01	-0.01
住宿	0.03	0.01	-0.07	-0.01	-0.03	-0.01	0.02
動物軀體	0.03	0.01	-0.06	0.00	0.01	0.00	0.00
城鄉	-0.09	-0.01	0.01	-0.00	-0.03	-0.00	0.00
家畜	-0.02	0.01	0.00	0.00	0.05	0.01	0.02
廚房器物	-0.03	0.01	0.00	0.02	0.04	0.01	0.04
性質	-0.09	0.00	0.00	0.02	-0.01	-0.00	0.01
植物	-0.01	0.01	0.13	-0.02	-0.04	0.08	-0.02
武器	-0.07	-0.01	0.00	-0.01	0.00	0.01	-0.01
無生命	-0.01	0.02	-0.01	-0.02	-0.00	0.08	0.03
生命性質	0.27	0.03	0.00	0.02	0.00	0.04	-0.02
生活器物	0.03	0.01	0.00	0.01	0.08	0.00	-0.00
禮樂	-0.01	-0.01	0.00	-0.03	0.03	0.01	0.01
穿著器物	-0.01	-0.00	0.00	-0.01	0.00	-0.00	-0.02
野獸	-0.03	0.02	0.06	0.00	0.01	0.06	0.01
顏色	0.04	0.00	0.00	0.00	-0.02	0.00	0.00

Figure 4.3: Importance of each radical semantic type to predicting a specific word semantic type.

doi:10.6342/NTU202200369

p(Mental Activities|$Plants$) and p($Plants$|Mental Activities), are small and ranked particularly low in the conditional probabilities given *Plants* and the conditional probabilities given *Mental Activities*, respectively. This implies that the presence of *Plants* predicts the absence of *Mental Activities* and vice versa. The semantic types *Plants* and *Mental Activities* do predict each other but in a negative fashion. This kind of relationship is easily masked in the matrices of conditional probabilities since the entries are dominated by probabilities close to zeros. The relationship is automatically revealed in a classification model like the one here since positive and negative relationships are both automatically exploited by the model to improve predictions.

4.4 Interim Summary

In this chapter, a case study is conducted to explore the relationship of semantic information encoded at different levels of the Chinese writing system. In particular, we examined the semantic information at the word level, which was operationalized by exploiting an existing thesaurus with predefined semantic types of words (Mei, 1983), and the semantic information at the radical level, which we operationalized as radical semantic types by leveraging Ma (2016)'s system. The results showed that word and radical semantic types are related and that radical semantic types could reliably predict word semantic types to a certain degree. Finer-grained examinations of the relationships between pairs of word semantic type and radical semantic type revealed a few of which with significant correlations. For instance, the presence of the radical semantic type *Human Relations* predicts a group of words to belong to the semantic type *Human*, and the presence of the radical semantic type *Human*

doi:10.6342/NTU202200369

Mind predicts the word class *Mental Activities.* Negative relations likewise exist. Given the presence of the radical semantic type *Plants* in a group of words, it is predicted that the type *Mental Activities* is unlikely to be its word semantic type.

Although relationships exist between radical and word semantic types, the relationships seem to be limited to particular types. Some word semantic types cannot be reliably predicted. For instance, the types *Time and Space*, *Action*, *Phenomena and States*, and *Honorifics* are never predicted by the classification model. A possible explanation is that since only single-character words were included in the classification model, information may not be enough to distinguish more abstract word classes. These abstract word classes may be better represented by two-character words. To deal with this possibility, a classification model taking pairs of radical semantic types, which was derived from two-character words, as features were trained and tested. The new model did correctly predict the word semantic types *Time and Space* and *Phenomena and States*, though the performance is quite poor. And still, word types such as *Actions*, *Relations*, and *Honorifics* are not predicted by the model at all.

Focusing on the values in the heatmap in Figure 4.3, it should be apparent that most of the values are insignificant. This suggests that the one-to-one correspondence between most radical and word semantic types is vague, at least at the first level of Cilin word categories. In hindsight, this is reasonable if we consider the extremely wide range of semantic space covered in Cilin—from concrete categories such as *Humans* and *Objects* to abstract concepts and terms such as *Relations* and *Auxiliaries.* Given such a wide range of meanings, it would not be a surprise that the radicals poorly predict the semantics of words as the meanings carried in radicals

doi:10.6342/NTU202200369

seem to be particularly concrete and basic (Chou & Huang, 2010; C.-R. Huang & Hsieh, 2015). Instead of predicting the full range of word categories in Cilin, future research might examine particular word semantic types such as *Humans*, *Objects*, or *Actions* and subcategories within these types to see if radical semantics yield fruitful results in these restricted semantic domains.

doi:10.6342/NTU202200369

doi:10.6342/NTU202200369

Chapter 5

Conclusion

5.1 Summary

This thesis starts out from the rich information encoded inside Chinese characters. This information, however, is blocked from access in modern computers due to the way characters are represented in computers. This fact arguably has hindered computational and computer-assisted research of or based on Chinese characters. For instance, in the fields of computational linguistics and corpus linguistics, this fact forces researchers to abandon information encoded below the character level.

Facing these challenges, we developed a software toolkit that is tailored to analyze Chinese corpus data at and below the level of characters, making available the rich internal information of Chinese characters to researchers and users. A suite of functions are included in this corpus toolkit, and selected functionalities have been demonstrated, hoping that this would open up new research directions and spawn new ideas on Chinese text analysis.

Finally, we conducted a case study as a preliminary investigation on the usefulness of sub-character information in the setting of computational research. Relationships between information encoded at the sub-character level and information

doi:10.6342/NTU202200369

encoded at the word level were explored and gauged. The results indicated that there indeed exist reliable relationships between these two levels, suggesting that the rich information inside a Chinese character is not simply intuitions of Chinese users but rather robust relationships that could be shown empirically.

5.2 Limitations and Future Work

In this work, we opened up new possibilities to look into character structures in Chinese corpora. Limitations, however, also came along.

A serious issue is raised by the non-uniqueness of the IDSs for a given character. Of the 88,937 characters in the IDS dataset, 3503 of them are assigned more than one IDS. Scanning through these 3503 characters reveals two causes of the non-uniqueness in IDS. The first and the predominant cause is character variants. Since Chinese characters have been used in different areas outside where they had originated, variants of the same character are bound to emerge. For instance, the character variant 毎 is used in Japan whereas the variant 每 is used in Taiwan, Hong Kong, and Mainland China. This caused the character 侮 to be decomposed differently as ⿰亻毎 (Japan) and ⿰亻每 (other places). Other similar examples are 綠 (⿰糸彔 / ⿰糹彔), 倩 (⿰亻青 / ⿰亻靑), and 笧 (⿱竹册 / ⿱竹冊). Future versions of *hgct* should add support for specifying regions in the search function. The second cause of the non-uniqueness of the IDSs arises from the structural ambiguity in decomposing a character. For instance, 叕 could be decomposed as either ⿱双双 or ⿰叒叒, 並 as ⿱䒑业 or ⿱丷亚, and 串 as ⿻吕丨 or ⿻中口. There are several potential solutions to these structural ambiguities. The most straightforward one is to trace the etymological origins of the characters. This approach, however, requires special-

doi:10.6342/NTU202200369

ties in ancient Chinese scripts, and substantial manual effort is needed. Another solution is suggested by Morioka (2020). He proposes that a better decomposition schema could be arrived at by looking at recurrent structures across characters. For instance, the characters 條, 修, 絛, and 倏 in the current IDS dataset are all decomposed by splitting apart 亻丨 from the rest of the character as in 修: ⿰ (⿰亻丨, ⿱夂彡). Morioka (2020) suggests, however, 亻, 丨, and 夂 should together form a functional component, and characters of this paradigm should be decomposed as ⿸攸 O, such as 條: ⿸攸木, 修: ⿸攸彡, and 絛: ⿸攸糸.

Another limitation in the current work is the inability of the search function to locate components nested deeply inside a character. As noted in Chapter 3, the query of character components in a corpus with *hgct* is limited to first-level components. Future work may improve upon this limitation by implementing a search algorithm that recursively searches through a character. A possible implementation is described in the following by using the example characters/components in Figure 5.1. Suppose that we would like to locate all characters with the component 木, regardless of how deeply the component 木 is nested inside, as in characters such as 琳, 霖, and 瀮. This could be achieved with a recursive algorithm that looks for characters "up a level" until there are no more characters at the upper level. In Figure 5.1, the starting point of the search is indicated by the rounded node. In the first recursive call, the square nodes (林, 柔, and 松) are reached. Each square node is subsequently used as a component to query characters up a level. This results in the pentagonal nodes, which are again used as the components for query at the next level. The process is repeated until no more characters are reached.

Although the recursive search algorithm provides a potential fix to the problem

doi:10.6342/NTU202200369

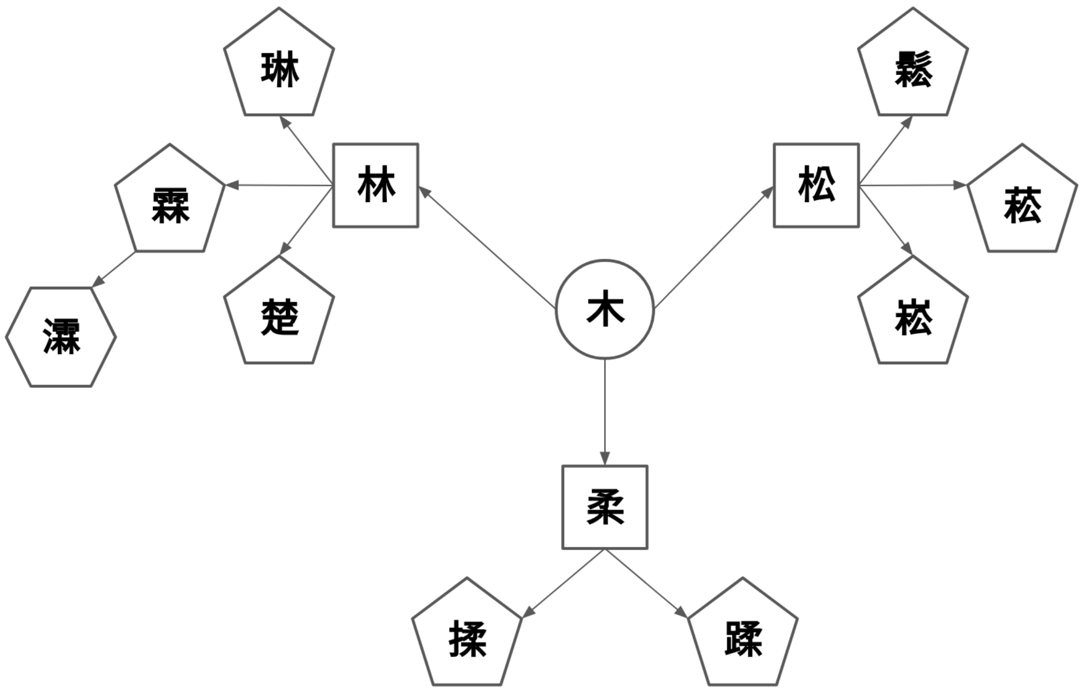


Figure 5.1: A (partial) graph representation of characters with the component 木.

of locating deeply nested components inside characters, it does not provide answers to an interesting question: are there any natural boundaries at which it makes sense to stop further decomposing characters into smaller components? For instance, the character 熙 can be decomposed into ⿱ (巸, 灬) and ⿱ (⿰ (臣[1], 巳), 灬). Which of these two decompositions makes more sense? To the author, it is the latter that makes more sense, as the components, 臣 and 巳, seem to be more *familiar* than the component 巸. This is reminiscent of the proposal by Morioka (2020), which is related to the issues of type frequency and productivity in usage-based linguistics, in which high type frequencies of a pattern is known to strengthen its representational schema such that the schema becomes more likely to be used with new items (Gries & Ellis, 2015). To confirm this intuition, the productivity scores are calculated for 巸, 臣, and 巳 respectively. As shown in Table 5.1, this aligns well with the author's

[1] 臣 is used here as a placeholder for the rare character U+268DE since the glyph for this character is missing on the platform in which this thesis was compiled. The correct character form could be found at https://glyphwiki.org/wiki/u268de.

doi:10.6342/NTU202200369

Table 5.1: Productivity measures for the three components of 熙. Both of the components 𦣝 and 巳 are realized in much more character types than the component 𦣞. The corpus data used for calculating the measures are the data used and described in Appendix A.

	V1C	NC	$P_{realized}$	$P_{expanding}$	$P_{potential}$	Example
𦣞	0	239	2	0.0000	0.0000	熙, [illegible]
𦣝	3	1464	10	0.0010	0.0020	姬, 頤, 賾
巳	1	9880	12	0.0003	0.0001	起, 祀, 包

intuition as 𦣞 has the lowest productivity among the three components.

In addition to future works inspired by current limitations of the corpus toolkit, other directions for future research are also possible, and probably even more interesting. We anticipate the development of this toolkit would bring about new ideas for research, and hopefully, these ideas in turn become the driving forces for the improvement of the current work.

doi:10.6342/NTU202200369

doi:10.6342/NTU202200369

References

Baayen, H. (1993). On frequency, transparency and productivity. In G. Booij & J. van Marle (Eds.), *Yearbook of Morphology 1992* (pp. 181–208). Springer Netherlands. https://doi.org/10.1007/978-94-017-3710-4_7

Baayen, H. (2001). *Word frequency distributions*. Kluwer Academic Publishers.

Baayen, H. (2009). Corpus linguistics in morphology: Morphological productivity. In A. Lüdeling & M. Kytö (Eds.), *Corpus Linguistics: An International Handbook* (Vol. 2, pp. 899–919). De Gruyter Mouton. https://doi.org/10.1515/9783110213881.2.899

Chou, Y.-M., & Huang, C.-R. (2006). Hantology-a linguistic resource for Chinese language processing and studying. *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*. http://www.lrec-conf.org/proceedings/lrec2006/pdf/532_pdf.pdf

Chou, Y.-M., & Huang, C.-R. (2010). Hantology: conceptual system discovery based on orthographic convention. In C.-R. Huang, N. Calzolari, A. Gangemi, A. Lenci, A. Oltramari, & A. Prevot (Eds.), *Ontology and the lexicon* (pp. 122–143). Cambridge University Press.

Christ, O. (1994). A Modular and Flexible Architecture for an Integrated Corpus Query System. *Proceedings of COMPLEX'94*, 23–32.

Chuang, D. M. [莊德明]., & Hsieh, C. C. [謝清俊]. (2005). 漢字構形資料庫的

doi:10.6342/NTU202200369

建置與應用 [Construction and application of Chinese characters information database]. In C.-R. Huang (Ed.), *漢字與全球化國際學術研討會論文集 [Proceedings of the international conference on Chinese characters and Globalization].*

Daniels, P. T. (1996). The study of writing systems. In P. T. Daniels & W. Bright (Eds.), *The study of writing systems* (pp. 3–17). Oxford University Press.

Evert, S., & Hardie, A. (2011). Twenty-first century Corpus Workbench: Updating a query architecture for the new millennium. *Proceedings of the Corpus Linguistics 2011 Conference.*

Gries, S. Th. (2020). Analyzing dispersion. In M. Paquot & S. Th. Gries (Eds.), *A practical handbook of corpus linguistics* (pp. 99–118). Springer International Publishing. https://doi.org/10.1007/978-3-030-46216-1$_5$

Gries, S. Th., & Ellis, N. C. (2015). Statistical measures for usage-based linguistics. *Language Learning*, *65*(S1), 228–255. https://doi.org/https://doi.org/10.1111/lang.12119

Handel, Z. (2019). *Sinography: The Borrowing and Adaptation of the Chinese Script.* Brill.

Huang, C.-R., & Hsieh, S.-K. (2015). Chinese lexical semantics. In *The oxford handbook of chinese linguistics* (pp. 290–305). Oxford University Press.

Huang, D. K. [黄德宽]. (2003). 汉字构形方式的动态分析 [Dynamic analysis of the formation of Chinese characters]. *安徽大学学报 (哲学社会科学版) [Journal of Anhui University (Philosophy and Social Sciences)]*, *27*(4), 1–8.

Ma, S.-L. [馬叔禮]. (2016). *方塊字的靈魂 [The Soul of Chinese Character].* Gallop Service Co., Ltd. [策馬入林文化].

doi:10.6342/NTU202200369

Mei, J.-J. [梅家駒]. (1983). *同義詞詞林 [Tongyici Cilin].* 上海辭書出版社 [Shanghai Lexicographical Publishing House].

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). *Efficient estimation of word representations in vector space.*

Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., & Joulin, A. (2018). Advances in pre-training distributed word representations. *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018).*

Morioka, T. (2008). CHISE: Character processing based on character ontology. *Proceedings of the 3rd International Conference on Large-Scale Knowledge Resources: Construction and Application*, 148–162.

Morioka, T. (2020). Viewpoints on the structural description of chinese characters. In Y. Haralambous (Ed.), *Proceedings of Grapholinguistics in the 21st Century, 2020* (Vol. 5, pp. 683–712). Fluxus Editions. https://doi.org/https://doi.org/10.36824/2020-graf-mori

Myers, J. (2019). *The Grammar of Chinese Characters: Productive Knowledge of Formal Patterns in an Orthographic System* (1st ed.). Routledge.

Niles, I., & Pease, A. (2001). Towards a standard upper ontology. *Proceedings of the International Conference on Formal Ontology in Information Systems - Volume 2001*, 2–9. https://doi.org/10.1145/505168.505170

Pustejovsky, J. (1995). *The generative lexicon.* MIT press.

Rogers, H. (2004). *Writing Systems: A Linguistic Approach.* Blackwell Publishing.

Sampson, G. (1985). *Writing systems.* Hutchinson.

Sproat, R. (2000). *A computational theory of writing systems.* Cambridge University Press.

doi:10.6342/NTU202200369

Sproat, R., & Gutkin, A. (2021). The taxonomy of writing systems: How to measure how logographic a system is. *Computational Linguistics*, *47*(3), 477–528. https://doi.org/10.1162/coli_a_00409

The Unicode Consortium. (2003). *The Unicode Standard, Version 4.0*. Addison-Wesley Professional. https://www.unicode.org/versions/Unicode4.0.0/

Tseng, Y.-H. (2021). *CompoTree: An Unicode IDS Component Tree Representation* [Python 3 Library]. https://github.com/seantyh/CompoTree

Unger, J. M., & DeFrancis, J. (1995). Logographic and Semasiographic Writing Systems: A Critique of Sampson's Classification. In I. Taylor & D. R. Olson (Eds.), *Scripts and Literacy: Reading and Learning to Read Alphabets, Syllabaries and Characters* (pp. 45–58). Springer Netherlands. https://doi.org/10.1007/978-94-011-1162-1_4

Woods, C. (Ed.). (2010). *Writing in the Ancient Middle East and Beyond*. Oriental Institute of the University of Chicago. https://oi.uchicago.edu/research/publications/oimp/oimp-32-visible-language-inventions-writing-ancient-middle-east-and

Zhou, Y. G. [周有光]. (1978). 现代汉字中声旁的表音功能问题 *[To what degree are the "phonetics" of present-day Chinese characters still phonetic?]*. *146*(3), 172–177.

doi:10.6342/NTU202200369

Appendix A — Search API in hgct

In the following tutorials (Appendix A and B), we will use a small collection of texts as the example corpus. The text data is available on GitHub at https://github.com/liao961120/hgct/raw/main/test/data.zip. After extracting `data.zip` to the directory `data`, it should have the following structure:

```
data
├── 01
│   ├── 儀禮_公食大夫禮.txt
│   ├── ...
│   └── 黃帝內經_靈樞經.txt
├── 02
│   ├── ...
│   └── 鹽鐵論_卷四.txt
├── 03
│   ├── 三國志_吳書一.txt
│   ├── ...
│   └── 魯勝墨辯注敘_魯勝墨辯注敘.txt
├── 08
│   ├── asbc1.txt
│   └── asbc2.txt
└── 10
    ├── dispersion1.txt
    ├── ...
    └── dispersion5.txt
```

The directory `data` corresponds to the corpus in *hgct*'s corpus representation. It contains five directories, each of which corresponds to a subcorpus. Directory `01`,

doi:10.6342/NTU202200369

02, and 03 consists of small samples of Literary Chinese texts collected from the Chinese Text Project (https://ctext.org). Directory 08 holds modern Chinese texts sampled from ASBC. The directory 10 is a toy corpus in Gries (2020, p. 102) used for illustrating calculations of dispersion measures.

In this tutorial, we demonstrate the supported functionalities in *hgct* for searching the corpus.

A.1 Loading Corpus Data into Concordancer

Provided that the input corpus follows the required directory structure mentioned in Section 3.1, users could convert the input corpus to the internal corpus representation with `PlainTextReader()` as in the following code block. Since we are now demonstrating the search functions, we immediately pass the corpus to `Concordancer()`, which is the object used in *hgct* for searching the corpus.

```
from hgct import PlainTextReader, Concordancer
c = Concordancer(PlainTextReader("data/").corpus)
```

The `Concordancer` object could be used to retrieve results matching the search pattern as a sequence[2] of concordance lines. Since many of the search patterns would return plenty of results, we define a wrapper function `get_first_n()` here for the purpose of demonstration.

```
def get_first_n(cql, n=10, left=5, right=5):
    out = []
    for i, r in enumerate(c.cql_search(cql, left=left, right=right)):
        if i == n: break
        out.append(r)
    return out
```

[2]More precisely, a *generator* of concordance lines.

doi:10.6342/NTU202200369

A.2 Search by Character

In our first example, we define the search pattern as `[char=" 龜"] [char="[一-龜]"]`, which roughly means

> a sequence of two characters starting with “龜” and ending with any Chinese characters (not, e.g., punctuations)

Passing this pattern to `get_first_n()` (or `Concordancer.cql_search()`) gives us a sequence of `Concord` objects. A `Concord` object is used to represent a matched result returned from the corpus in *hgct*.

```
cql = '''
[char=" 龜"] [char="[一-龜]"]
'''
# left/right: left/right context size around the keyword
results = get_first_n(cql, n=5, left=6, right=3)
results
```

```
[<Concord 還有無，貨自{龜貝}，至此>,
 <Concord 山在西北。有{龜山}。有龍>,
 <Concord ，故獸不狘；{龜以}為畜，>,
 <Concord 江郡常歲時生{龜長}尺二寸>,
 <Concord 無為頓復卜三{龜知}。聖人>]
```

To get more information about a particular matching result, we can look at the `data` attribute in a `Concord` object, which is a dictionary holding the relevant information of the matching result.

```
result_1 = results[0]
result_1.data
```

```
{'captureGroups': {},
 'keyword': '龜貝',
 'left': '還有無，貨自',
 'meta': {'id': '02/漢書_傳.txt',
  'text': {'book': '漢書', 'sec': '傳'},
```

doi:10.6342/NTU202200369

```
  'time': {'label': '漢', 'ord': 2, 'time_range': [-205, 220]}},
 'position': (1, 6, 3482, 42),
 'right': '，至此'}
```

Note the `position` key in `Concord.data`. It holds the position of the matched keyword in the corpus. The elements in the 4-tuple `(1, 6, 3482, 32)` correspond respectively to the indices of `(subcorpus, text, sentence, character)`.

We did not mention above how the index of a subcorpus is determined. The index of a subcorpus is automatically determined according to the **character order of the directory names**. Remember that there are four directories (subcorpora) in our input corpus---`01`, `02`, `03`, `08`, and `10`. So by character order, `01` appears before `02`, `02` before `03`, `03` before `08`, and so on. Hence, the first directory `01` is given the index of 0, the second is given the index of 1, and so on. These indices of the subcorpora, as seen later in Appendix B, could be used for limiting the scope of the functions in *hgct* in computing corpus statistical measures.

A.3 Search by Character Components

In addition to character forms, we can also describe search patterns in terms of character compositions, such as the Kangxi Radical or Ideographic Descriptions of a character.

A.3.1 Kangxi Radicals

To take a look at all the present Kangxi radicals in the characters of the corpus, the attribute `Concordancer.chr_radicals` could be used:

```
print(c.chr_radicals)
```

```
{'火', '豆', '鳥', '鹿', '辵', '風', '龱', '手', '欠', '瓦', '见',
 '卜', '网', '彐', '冫', '夕', '鬥', '子', '勹', '饣', '鬼', ...}
```

doi:10.6342/NTU202200369

To search the corpus with Kangxi radicals, simply use the attribute `radical` in the description of the search pattern.

```
cql = '''
[radical=" 立"]
'''
get_first_n(cql, 5)
```

```
[<Concord 》有竘匠。{竵}：不正也。>,
 <Concord ，遠塗也，{竫}立安坐而至>,
 <Concord ？惟諓諓善{竫}言。俾君子>,
 <Concord 自申束也。{竫}：亭安也。>,
 <Concord 聲。靖：立{竫}也。从立青>]
```

A.3.2 Ideographic Description Characters (IDCs)

Character components defined according to the Unicode's Ideographic Description Characters (IDCs) could also be used for searching. The IDCs and their names in *hgct* are found in `Concordancer.chr_idcs`:

```
c.chr_idcs
```

```
{'curC': '⿷', 'encl': '⿴', 'horz2': '⿰', 'horz3': '⿲',
 'over': '⿻', 'sur7': '⿹', 'surL': '⿺', 'surN': '⿵',
 'surT': '⿸', 'surU': '⿶', 'vert2': '⿱', 'vert3': '⿳'}
```

To search according to Ideographic Descriptions, use the attributes `compo` and/or `idc`.

```
cql = '''
[compo=" 木" & idc="vert2" & pos="0"]
'''
get_first_n(cql, 5)
```

```
[<Concord 以中牟叛，{桼}雕刑殘，莫>,
 <Concord 銅錮其內，{桼}塗其外，被>,
 <Concord 行，堅如膠{桼}，昆弟不能>,
 <Concord 陳、夏千畝{桼}；齊、魯千>,
 <Concord 千兩；木器{桼}者千枚，銅>]
```

doi:10.6342/NTU202200369

```
cql = '''
[compo=" 木" & idc="vert2" & pos="1"]
'''
get_first_n(cql, 5)
```

```
[<Concord 之也。从手{罙}聲。撢：探>,
 <Concord 營道。从水{罙}聲。潭：水>,
 <Concord 吉臺原姑與{柒}里，使海於>,
 <Concord 吹咀，以水{柒}升，微火煮>,
 <Concord ，綿裹。右{柒}味，吹咀。>]
```

```
cql = '''
[compo=" 木" & idc="vert2"]
'''
get_first_n(cql, 5)
```

```
[<Concord 之也。从手{罙}聲。撢：探>,
 <Concord 營道。从水{罙}聲。潭：水>,
 <Concord 城，積木為{寨}，匈奴不敢>,
 <Concord 入侍。以邊{寨}無寇。減戍>,
 <Concord 親王（柬埔{寨}）等針撥白>]
```

Either `compo` or `idc` could be left out if a more abstract search pattern is preferred. For instance, if the shape (`idc`) and the position (`pos`) are not of interest, these attributes could be left out.

```
cql = '''
[compo=" 木"]
'''
get_first_n(cql, 5)
```

```
[<Concord 梅。楥，柜{柳}。栩，杼。>,
 <Concord 。讀若過。{柳}：馬柱。从>,
 <Concord 繫其頸著馬{柳}，五葬反。>,
 <Concord 其甲胄、干{楯}也；钁鍤、>,
 <Concord 句踐也以甲{楯}三千，棲於>]
```

If one is interested only in the shape of the character, `idc` could be specified while all other attributes could be left out.

doi:10.6342/NTU202200369

```
cql = '''
[idc="encl"] [idc="encl"]
'''
get_first_n(cql, 5)
```

```
[<Concord 岸崩。始置{圃囿}署，以宦者>,
 <Concord 日：「請以{國因}。」故日可>,
 <Concord 天子東出其{國四}十六里而壇>,
 <Concord 入{國四}旬，五行九>,
 <Concord 君約，破趙{國因}封二子者各>]
```

A.3.3 Radical Semantic Type

Ma's (2016) semantic type classification of Kangxi Radicals is also incorporated in *hgct*'s search function. Use the attribute `semtag` to specify a radical semantic type. Refer to Table 3.2 for the 22 available semantic types.

```
cql = '''
[semtag=" 植物"] [semtag=" 植物"]
'''
get_first_n(cql, 5)
```

```
[<Concord 。且夫山不{槎蘖}，澤不伐夭>,
 <Concord 龜有艽蒚，{槎櫛}堀虛，連比>,
 <Concord 則從行獵，{槎桎}拔，失鹿，>,
 <Concord 。冒甯柘，{槎棘}枳，窮浚谷>,
 <Concord 嶽之山，多{枳棘}剛木。有獸>]
```

A.4 Search by Phonetic Properties

hgct also provides searching the corpus with sound properties. The sound properties are defined according to the data from two system—Guanyun 廣韻 (Middle Chinese) and Chinese Dictionary compiled by the Ministry of Education (MOE) in Taiwan (Mandarin).

```
c.cql_attrs['CharPhonetic']
```

```
{'moe': ['phon', 'tone', 'tp', 'sys="moe"'],
 '廣韻': ['攝', '聲調', '韻母', '聲母', '開合', '等第',
```

doi:10.6342/NTU202200369

```
         '反切', '拼音', 'IPA', 'sys="廣韻"']
}
```

A.4.1 Mandarin (based on 萌典)

```
cql = '''
[phon="ㄨㄥ" & tone="1" & sys="moe"]
'''
get_first_n(cql, 5)
```

```
[<Concord 」耳邊不斷{嗡}嗡的縈繞著>,
 <Concord 耳邊不斷嗡{嗡}的縈繞著類>,
 <Concord 哭泣不秩聲{翁}，線經垂涕>,
 <Concord ，黑文而赤{翁}，名曰櫟，>,
 <Concord 發猛，塤篪{翁}博，瑟易良>]
```

```
cql = '''
[phon="^p" & tp="ipa" & sys="moe"] [phon="^p" & tp="ipa" & sys="moe"]
'''
get_first_n(cql, 5)
```

```
[<Concord 大禍或遭流{炮波}及。我們步>,
 <Concord 牀版也。从{片扁}聲。讀若邊>,
 <Concord 如看推理名{片般}，由姐妹的>,
 <Concord 了進來，一{片片}綠油油的田>,
 <Concord 好高哇！一{片片}的竹葉，好>]
```

A.4.2 Middle Chinese (based on 廣韻)

```
cql = '''
[韻母 =" 東" & 聲調 =" 平" & sys=" 廣韻"]
'''
get_first_n(cql, 5)
```

```
[<Concord 从雨相聲。{霿}：地气發，>,
 <Concord 山，其上多{銅}，其下多玉>,
 <Concord 無草木，多{銅}玉。蠠水出>,
 <Concord 玉，其下多{銅}，其獸多閭>,
 <Concord 山，其上多{銅}玉，其下多>]
```

doi:10.6342/NTU202200369

Appendix B — Corpus Analysis API in hgct

In this second tutorial, we demonstrate functions for quantitative analysis of the corpus in *hgct*. To get started, we need two additional objects `CompoAnalysis` and `Dispersion` in addition to the `Concordancer` object introduced in the previous tutorial. The corpus used is identical to the one in Appendix A.

Note that when initializing with `CompoAnalysis()` and `PlainTextReader()`, the argument `auto_load=False` needs to be given to `PlainTextReader()`. This prevents the full corpus to be loaded into the memory, such that functionalities provided by `CompoAnalysis` could be used to analyze large data that do not fit into the computer's memory. For more information, refer to the source code on GitHub[3].

```
import pandas as pd
from hgct import PlainTextReader, Concordancer
from hgct import CompoAnalysis, Dispersion

CC = Concordancer(PlainTextReader("data/").corpus)
CA = CompoAnalysis(PlainTextReader("data/", auto_load=False))
DP = Dispersion(PlainTextReader("data/").corpus)
```

B.1 Frequency List (Distribution)

Frequency lists are provided by the function `CompoAnalysis.freq_distr()`. Based on the arguments passed, this function computes and returns the frequency distribution of either the characters, IDCs, Kangxi radicals, or characters with a

[3]https://github.com/liao961120/hgct/blob/main/hgct/compoAnalysis.py

doi:10.6342/NTU202200369

given radical/component. Below, we demonstrate each of these types of frequency distributions.

B.1.1 Character

To return the frequency distribution of the characters in the corpus, set the argument `tp` to `"chr"`. `CompoAnalysis.freq_distr()` by default returns a `Counter`[4], which has the convenient method `most_common()` that could be used to retrieve the terms with the highest frequencies.

```
CA.freq_distr(tp="chr").most_common(4)
```

```
[('之', 210608), ('不', 129212), ('也', 107639), ('以', 104578)]
```

As mentioned in Section A.2, we could limit the scope of calculation to a particular subcorpus by specifying its index. To do this, pass the argument `subcorp_idx` to the function. The example below sets the subcorpus to `3`, which is the subcorpus of modern Chinese (ASBC).

```
CA.freq_distr(tp="chr", subcorp_idx=3).most_common(4)
```

```
[('的', 15826), ('一', 5537), ('是', 5130), ('不', 4469)]
```

B.1.2 IDC

Frequency distributions of the Ideographic Description Characters (IDCs) could similarly be retrieved by setting `tp` to `"idc"`. Note that there is an argument `use_chr_types` that applies when `tp="idc"` (IDC) or `tp="rad"` (radical). `use_chr_types` is used to determine how to compute the frequencies. If it is set to `False`, character frequencies are considered. If it is `True`, character frequencies are discarded. In other words, when `use_chr_types=False`, an IDC or a radical

[4]https://docs.python.org/3/library/collections.html#collections.Counter

doi:10.6342/NTU202200369

would only be counted once for each type of character. See Section 3.4.1 for a toy example.

```
CA.freq_distr(tp="idc", subcorp_idx=3)
```

```
Counter({'': 48725,
         '⿰': 167681,
         '⿱': 120035,
         '⿲': 1965,
         '⿳': 4068,
         '⿴': 5744,
         '⿵': 7834,
         '⿶': 1637,
         '⿷': 537,
         '⿸': 18511,
         '⿹': 4412,
         '⿺': 13451,
         '⿻': 10324})
```

```
CA.freq_distr(tp="idc", use_chr_types=True, subcorp_idx=3)
```

```
Counter({'': 119,
         '⿰': 2454,
         '⿱': 1019,
         '⿲': 26,
         '⿳': 39,
         '⿴': 18,
         '⿵': 45,
         '⿶': 6,
         '⿷': 12,
         '⿸': 176,
         '⿹': 41,
         '⿺': 123,
         '⿻': 32})
```

B.1.3 Radical

To retrieve frequency distributions for radicals, set `tp="rad"`. The usage of `use_chr_types` here is similar to the IDC described above.

doi:10.6342/NTU202200369

```
CA.freq_distr(tp="rad", subcorp_idx=3).most_common(4)
```

```
[('人', 28149), ('白', 16640), ('一', 15567), ('口', 15443)]
```

```
CA.freq_distr(tp="rad", use_chr_types=True, subcorp_idx=3).most_common(4)
```

```
[('水', 233), ('口', 207), ('手', 201), ('人', 172)]
```

B.1.4 Characters with a given radical

It is also possible to look into characters of a specific type. By setting `tp=None`, one could then pass in a radical to the argument `radical` to look at the frequency distribution of the characters with this particular radical.

```
CA.freq_distr(tp=None, radical=" 广").most_common(4)
```

```
[('度', 4757), ('廣', 4050), ('廟', 3067), ('府', 3064)]
```

B.1.5 Characters with a given IDC component

Similarly, a frequency distribution of characters of a specific type defined according to a component and an optional IDC describing the the shape could also be retrieved by specifying `tp=None` and the arguments `compo` and `idc` (optional).

```
CA.freq_distr(tp=None, compo=" 水", idc="vert2")
```

```
Counter({'氶': 1,
         '汞': 15,
         '沓': 89,
         '泉': 1349,
         '泵': 3,
         '淼': 4,
         '滎': 344,
         '漀': 1,
         '漐': 9,
         '漿': 153,
         '澩': 3,
         '灓': 5})
```

doi:10.6342/NTU202200369

B.2 Dispersion

Measures of dispersion could be calculated based on a character or a search pattern.

B.2.1 Dispersion Measures for Characters

`Dispersion.char_dispersion()` is used for calculating dispersion measures for a character. The examples below—using the toy corpus in Gries (2020)—demonstrate the validity of the returned measure. The values should be identical to those in Table 1 of Gries (2020).

```
# Gries (2020, Table 1)
DP.char_dispersion(char='a', subcorp_idx=4)
```

```
{'DP': 0.18,
 'DPnorm': 0.2195121951219512,
 'JuillandD': 0.7851504534504508,
 'KLdivergence': 0.13697172936522078,
 'Range': 5,
 'RosengrenS': 0.9498163423042408}
```

```
# return_raw=True to get the raw data for dispersion calculation
DP.char_dispersion(char='a', return_raw=True, subcorp_idx=4)
```

```
({'DP': 0.18,
  'DPnorm': 0.2195121951219512,
  'JuillandD': 0.7851504534504508,
  'KLdivergence': 0.13697172936522078,
  'Range': 5,
  'RosengrenS': 0.9498163423042408},
 {'corpus_size': 50,
  'f': 15,
  'n': 5,
  'p': [0.1111111111111111, 0.45454545454545453, 0.3, 0.2, 0.4],
  's': [0.18, 0.22, 0.2, 0.2, 0.2],
  'v': [1, 5, 3, 2, 4]})
```

doi:10.6342/NTU202200369

To see how dispersion measures behave on real data, we calculate dispersion measures for four characters (之, 也, 草, and 巾) in a corpus of Literary Chinese texts. The first two characters 之 and 也 are often used as function words and the last two as content words in Literary Chinese. Hence, we would expect the first two to be distributed evenly, and the latter two unevenly in the corpus.

```
subcorp_idx = 0
df_disp = []
for ch in '之也草巾':
    stats, raw = DP.char_dispersion(
        char=ch, subcorp_idx=subcorp_idx, return_raw=True
    )
    d = {
        'char': ch,
        'Range(%)': '{:.2f}'.format(100 * stats['Range'] / raw['n']),
        **stats
    }
    df_disp.append(d)
df_disp = pd.DataFrame(df_disp)
df_disp
```

char	Range(%)	Range	DP	DPnorm	KLdivergence	JuillandD	RosengrenS
之	90.98	666	0.13	0.13	0.10	0.98	0.96
也	77.05	564	0.25	0.25	0.40	0.96	0.82
草	22.40	164	0.65	0.65	2.33	0.86	0.32
巾	3.69	27	0.84	0.84	4.08	0.54	0.10

B.2.2 Dispersion Measures of Complex Forms (defined by CQL)

Dispersion measures for abstract units could also be calculated with the returned concordance lines provided by `Concordancer.cql_search()`. The function `DP.pattern_dispersion()` is designed to take the queried results from `Concordancer.cql_search()` to calculate dispersion measures.

```
cql = """
[semtag=" 人體精神"] [semtag=" 人體精神"]
"""
results = list(CC.cql_search(cql, left=3, right=3))
```

doi:10.6342/NTU202200369

```
print('Num of results:', len(results))
for r in results[:3]: print(r)
```

```
Num of results: 8459
<Concord 俟之心{惕惕}焉。」>
<Concord 突盜，{惕悍}憍暴，>
<Concord 皆有怵{惕惻}隱之心>
```

```
DP.pattern_dispersion(data=results, subcorp_idx=2)
```

```
{'DP': 0.1504848557289626,
 'DPnorm': 0.15050344195568013,
 'JuillandD': 0.9387038720245429,
 'KLdivergence': 0.135483902941753,
 'Range': 134,
 'RosengrenS': 0.9428568965311757}
```

The example below calculates dispersion measures for **each subcorpus 0, 1, and 2**. This is useful when the user is interested in contrasting dispersion measures in different corpora (e.g., genre/diachronic comparison).

```
# Compute separate dispersion measures for each subcorpus
df_pat_disp = []
for i in range(3):
    stats, raw = DP.pattern_dispersion(
        data=results, subcorp_idx=i, return_raw=True
    )
    d = {
        'Range(%)': '{:.2f}'.format(100 * stats['Range'] / raw['n']),
        **stats,
        'freq': raw['f'],
        'corp_size': raw['corpus_size']
    }
    df_pat_disp.append(d)
df_pat_disp = pd.DataFrame(df_pat_disp)
df_pat_disp
```

Range(%)	Range	DP	DPnorm	...	JuillandD	RosengrenS	freq	corp_size
44.40	325	0.40	0.40	...	0.91	0.63	1689	1858228
53.38	560	0.33	0.33	...	0.95	0.75	3500	3938310
85.90	134	0.15	0.15	...	0.94	0.94	2489	2097273

doi:10.6342/NTU202200369

B.3 Ngram Frequency

We now turn to the relationships across characters. To compute character n-grams, one can use `Concordancer.freq_distr_ngrams()`.

```
CC.freq_distr_ngrams(n=2, subcorp_idx=0).most_common(4)
```

```
[('而不', 3913), ('天下', 3661), ('不可', 2985), ('之所', 2723)]
```

```
CC.freq_distr_ngrams(n=3, subcorp_idx=0).most_common(4)
```

```
[('天下之', 946), ('歧伯曰', 766), ('之所以', 605), ('不可以', 580)]
```

B.4 Collocation

Association measures could be used to quantify the strengths of attraction between a pair of characters. Pairs with strong attractions could be considered as collocations. *hgct* implements two types of collocation extraction functions. The first (`Concordancer.bigram_associations()`) is based on bigrams, which simply computes association scores for all bigrams. With the second implementation (`Concordancer.collocates()`), users could specify a node and a window size, and characters falling within this window around the node would be treated as a node-collocate pair. Each pair is then computed for an association score.

B.4.1 Bigram Association

```
bi_asso = CC.bigram_associations(subcorp_idx=3, sort_by="Gsq")
bi_asso[0]
```

```
('自己',
 {'DeltaP12': 0.9778668701918644,
  'DeltaP21': 0.36342714003090937,
  'Dice': 0.5303392259913999,
  'FisherExact': 0.0,
  'Gsq': 6188.677676112116,
```

doi:10.6342/NTU202200369

```
  'MI': 7.855905225817536,
  'RawCount': 555,
  'Xsq': 128210.23324106314})
```

```
d = pd.DataFrame([{'bigram': x[0], **x[1]} for x in bi_asso][:5])
d
```

bigram	MI	Xsq	Gsq	Dice	DeltaP21	DeltaP12	FisherExact	RawCount
自己	7.86	128210.23	6188.68	0.53	0.36	0.98	0.00	555
什麼	9.15	192859.82	4474.73	0.70	0.98	0.55	0.00	339
我們	6.18	42280.22	4389.15	0.33	0.26	0.45	0.00	592
台灣	8.13	111740.17	4314.05	0.54	0.45	0.69	0.00	401
沒有	6.39	43012.13	4186.92	0.27	0.73	0.16	0.00	518

B.4.2 Node-Collocate Association

The example below use the character sequence 我們 as the node and looks for collocates occurring on the immediate right (`left=0` and `right=1`) on the node. After computing association scores for each node-collocate pair, these pairs are sorted based on the MI measure. The data frame below shows the top-5 collocates with the highest MI scores (a minimum frequency threshold of 6 is applied) of the node 我們.

```
cql = """
[char=" 我"] [char=" 們"]
"""
collo = CC.collocates(cql, left=0, right=1, subcorp_idx=3,
                      sort_by="MI", alpha=0)
collo[0]
```

```
('釘',
 {'DeltaP12': 0.0016848237685590844,
  'DeltaP21': 0.33204500782950214,
  'Dice': 0.0033613445378151263,
  'FisherExact': 0.003866505328061448,
  'Gsq': 9.493215334772461,
  'MI': 8.012895027477056,
  'RawCount': 1,
  'Xsq': 256.6351579547297})
```

doi:10.6342/NTU202200369

```
d = pd.DataFrame([{'char': x[0], **x[1]} for x in collo
                  if x[1]['RawCount'] > 5][:5])
d
```

char	MI	Xsq	Gsq	Dice	DeltaP21	DeltaP12	FisherExact	RawCount
認	3.98	124.86	33.08	0.02	0.02	0.01	0.00	9
還	3.39	77.32	26.21	0.01	0.01	0.01	0.00	9
都	3.33	122.65	42.67	0.02	0.01	0.02	0.00	15
就	3.21	125.44	45.83	0.02	0.01	0.03	0.00	17
所	3.05	76.93	29.88	0.01	0.01	0.02	0.00	12

B.5 Productivity

Finally, we demonstrate the usage of the tentative applications of Productivity measures (Baayen, 1993, 2009) to character components. This is implemented in `CompoAnalysis.productivity()`. The categories for computing measures of productivity are defined based on the arguments passed.

```
# Productivity of a radical
CA.productivity(radical=" 广", subcorp_idx=0)
```

```
{'N': 1505967,
 'NC': 5889,
 'V1': 1896,
 'V1C': 7,
 'productivity': {'expanding': 0.003691983122362869,
  'potential': 0.0011886568177958906,
  'realized': 58}}
```

```
# Productivity of a component
CA.productivity(compo=" 虫", idc="horz2", pos=0, subcorp_idx=0)
```

```
{'N': 1505967,
 'NC': 1027,
 'V1': 1896,
 'V1C': 72,
 'productivity': {'expanding': 0.0379746835443038,
  'potential': 0.07010710808179163,
  'realized': 178}}
```

doi:10.6342/NTU202200369

```python
# Productivity of Hanzi shapes (IDCs)
df_prod = []
for idc_nm, idc_val in CC.chr_idcs.items():
    p = CA.productivity(idc=idc_nm, subcorp_idx=0)
    df_prod.append({
        'name': idc_nm,
        'shape': idc_val,
        **p['productivity'],
        'V1C': p['V1C'],
        'V1': p['V1'],
        'NC': p['NC'],
        'N': p['N'],
    })

df_prod = pd.DataFrame(df_prod)
df_prod
```

name	shape	realized	expanding	potential	V1C	V1	NC	N
horz2	⿰	5436	0.7194	0.0031	1364	1896	437854	1505967
vert2	⿱	2045	0.2194	0.0007	416	1896	561357	1505967
horz3	⿲	35	0.0016	0.0005	3	1896	6240	1505967
vert3	⿳	80	0.0058	0.0008	11	1896	14371	1505967
encl	⿴	27	0.0016	0.0002	3	1896	14409	1505967
surN	⿵	84	0.0047	0.0004	9	1896	25231	1505967
surU	⿶	6	0.0000	0.0000	0	1896	7275	1505967
curC	⿷	20	0.0021	0.0024	4	1896	1641	1505967
surT	⿸	332	0.0264	0.0005	50	1896	91208	1505967
sur7	⿹	48	0.0026	0.0002	5	1896	25379	1505967
surL	⿺	178	0.0132	0.0009	25	1896	26844	1505967
over	⿻	43	0.0005	0.0000	1	1896	37846	1505967

doi:10.6342/NTU202200369