9

What can corpus software do?

Laurence Anthony

1 Comparing online tools, offline tools and DIY tools

What is the difference between online, offline and DIY tools?

Corpus software comes in primarily three different forms: online tools, offline tools and DIY tools. Online tools usually exist in the "cloud", meaning that they reside on one or more remote servers and are developed, managed and updated remotely. To use these tools, you will need to have an internet connection, and you connect to them through a standard web browser, such as Chrome, Firefox, Edge or Safari (on a Macintosh computer). Some of these tools are made freely available, while others require a one-time payment or a subscription fee. Some commonly used and frequently cited online corpus tools include *CQPweb,*[1] *English-Corpora.org,*[2] and *Sketch Engine.*[3]

In contrast, offline tools need to be purchased as a standalone package or, more commonly, downloaded from a provider (for free, with a one-time payment or with a subscription fee) and then manually installed onto your local computer's hard drive. Although these tools are created by external programmers, as a user of these tools, you are generally responsible for managing where they are stored and when they are updated. To use these tools, you need to have a computer with an operating system, hard disk space, and random access memory (RAM) memory that matches the requirements of the software. Some commonly used and frequently cited offline corpus tools include *AntConc,*[4] *#LancsBox,*[5] *Wmatrix,*[6] and *WordSmith Tools.*[7]

DIY tools usually come in the form of "scripts". These are text files containing a list of commands that need to be executed by a scripting programming language, such as Python or R. These scripts may be given as a listing in a research paper or personal blog, supplied as an addendum to a book or book chapter, appear in a forum post (such as on *Stack Overflow*[8]) or sent as a personal communication through email. They may also be provided on an individual researcher's website or in a project repository hosted on sites such as *GitHub.*[9] DIY tools are often in a state of constant flux, with tools frequently created, updated, renamed, moved and deleted. Two examples of DIY corpus tool repositories that are relatively stable but sit at the opposite ends of the scale in terms of size

DOI: 10.4324/9780367076399-9

and scope are the huge library of Python scripts that form the *Natural Language Toolkit*[10] and the R scripts for the book *Quantitative Corpus Linguistics with R* (Gries 2016).[11] Of course, a researcher may also choose to write their own scripts using a knowledge of programming.

How are online, offline and DIY tools used in corpus linguistics?

Online, offline and DIY corpus software tools can play an important role at every stage of a corpus linguistics project. At the corpus design and development stage, software tools can assist in the collection of raw data that goes into the corpus. For example, there are numerous general-purpose online and offline tools as well as DIY tools that can help researchers convert raw data in the form of audio recordings, webpages, scanned books, PDF/Word files and other data formats into a form more suitable for corpus analysis. For example, many online PDF-to-text converters can be easily found through a simple Internet search. Care should be taken when using these, however, as it can be unclear if the site will store and use the data for its own purposes. They may also only convert a single file at a time, which is inappropriate for many corpus building tasks that require huge numbers of files to be converted. One alternative is *AntFileConverter,*[12] which is an offline tool that can batch-convert PDF and Word files into a text format without any restriction on the number of files being processed.

There are also online, offline and DIY tools that can automatically visit a website and automatically "scrape" the text data from each page and store the results locally. One such tool is the online *Web Scraper*[13] tool that is available for both Chrome and Firefox browsers, and a popular offline web scraping tool is *BootCat.*[14] A more advanced data collection tool for the automatic download and storage of academic research articles is *AntCorGen.*[15] This tool provides an interface to the *PLOS ONE*[16] open-access interdisciplinary journal and allows you to select not only the area of specialisation but also the sections of the article (e.g. title, abstract, introduction) that you are interested in analysing. Another advanced data collection tool was used as part of the *CorCenCC* project.[17] This was a custom-built online mobile app that recorded spoken data from participants in the project and sent the data to a server for upload into the project database. There are even tools such as *FireAnt*[18] that can access the application programming interfaces (APIs) of social media platforms and collect data in the form of posts, "tweets" and so on, as well as the associated metadata of the sender (e.g. name, location, date of post) and that of their target audience members.

Online, offline and DIY corpus software tools can also help corpus designers to clean, tag and annotate their corpora. For part-of-speech (POS) tagging (i.e. the explicit labeling of words in a corpus as nouns, verbs, adjectives and so on), a commonly used offline tool is *TagAnt.*[19] This tool provides a one-click method to POS-tag a batch of text files using the *TreeTagger* engine[20] and works with multiple Western languages, including Dutch, English, French, German, Italian and Spanish. A similar offline tool called *SegmentAnt*[21] provides one-click POS tagging of Chinese and Japanese texts. Semantic tagging of texts (i.e. the tagging or word senses, such as *emotion*, *time*, *education* and *science and technology*), can be achieved using the online *UCREL Semantic Analysis System* (*USAS)* tool,[22] which is also incorporated into the general-purpose offline corpus analysis toolkit *Wmatrix*.[23] This tool also works with multiple languages, including Chinese, Dutch, English, Italian, Portuguese and Spanish. There are also tools to assist with higher-level annotation of corpus data. For sentence parsing of texts to

annotate structural features such as the subject and predicate, tools like the *Stanford Parser*[23] can be used. There are also tools such as the offline *UAM tool*[24] that can be used to apply a custom annotation scheme to texts at multiple levels (e.g. phrase, clause, sentence, paragraph, document) and also output the results in different formats for further processing and analysis.

Corpus software tools are most commonly associated with the task of analysing large amounts of corpus data. Indeed, numerous online, offline and DIY tools have been specially designed for this purpose, providing researchers with qualitative and quantitative information on a wide range of features and patterns in text-based and multimodal corpora. Tools that assist in the bottom-up analysis of corpora can provide information on the frequency of occurrence and dispersion of single words and also multiword units (MWUs), such as clusters (sometimes referred to as n-grams or lexical bundles) (see later and Chapter 15, this volume). They can also be used to identify which words commonly co-occur with other words in the same corpus (i.e. collocations, see later and Chapter 15, this volume) and which words appear unusually frequently compared to their frequency in a reference corpus (i.e. keywords, see Chapter 10, this volume). They can also be used to identify common patterns of word and phrase usage in context (i.e. concordances, see later and Chapter 10, this volume) and visualise where these word and phrase patterns occur in individual tests and/or the corpus as a whole (i.e. dispersion plots, see later). Although these same tools can be used more indirectly to identify top-down patterns of usage, there are other tools designed specifically to analyse corpus data at the top-down level. For example, there are tools that can identify prototypical texts in a corpus (e.g. *ProtAnt*[25]), as well as tools that can analyse discourse structure and rhetorical purpose (e.g. *AntMover*[26] and *AWSum*[27]). The use of corpus tools to assist in the analysis of corpora at the bottom-up and top-down level will be the focus of Sections 2 and 3 of this chapter.

One other important area that online, offline and DIY corpus software tools play an important role in is the visualisation of results from a corpus linguistics project. Most online and offline tools will produce results in a tabular form, such as concordance outputs and word frequency lists. However, some can produce more complex visualizations. For example, *AntConc*[4] and *WordSmith Tools*[7] can produce dispersion plots of word and phrase usage in the form of stacked "barcode" charts that immediately reveal where the results appear in the corpus. *#LancsBox*[5] can produce collocates in a tabular form or as a network of connected nodes, where clicking on one node (either the target word or one of its collocates) will expand the network to show further connections between that word and its related collocates. Online corpus tools and DIY tools offer even more possibilities for advanced visualisation of results using browser-based *JavaScript*[28] libraries that can generate and even animate 2D and 3D visualisations of massive amounts of data. Some of these features can be seen in the visualisations provided by the online *Michigan Corpus of Upper-Level Student Papers* (MICUSP) interface,[29] which shows results as dynamically updated bar charts and pie charts with user-initiated color highlighting of relevant columns and values. *Kaleidographic*[30] is an online tool that allows users to upload their own data and generate animated multi-layered visualisations. Even more impressive visualisations can be generated using DIY tools. These tools can utilise dedicated graphics packages and JavaScript libraries to produce complex and often striking visualisations. To see what can be achieved with these tools, one only needs to look at the visualisations produced by Jack Grieve[31,32] and Stefan

Gries,[33] among many other researchers who use them to explain a wide range of linguistic phenomena.

Which are better: online, offline or DIY tools?

It is difficult to say whether an online, offline or DIY tool is the right choice for a particular corpus linguistics project, but each has certain strengths and weaknesses that are worth considering.

Online tools have largely become the tool of choice for many projects, both inside and outside of corpus linguistics. There are numerous reasons why this is the case. First, the growth of the internet, the advance of server technology and the fall in prices and increased efficiency and capacity of memory devices has meant that online tools can be cheaply stored and quickly accessed at remote locations. Server systems can also distribute tasks across multiple computers, hugely increasing the overall speed and performance of a system. From a user's perspective, the complexity of an online tool can also be largely ignored. All a user needs to do is launch a standard internet browser, type or search for the address of the hosting server and start using the tool. Many software developers also prefer creating online tools, because they only need to develop the software to run on standard browsers instead of developing the software to run natively on multiple computer systems running different operating systems and different versions of the same operating system. Updating software is also simple when it is hosted online. The developer only needs to send the new code to the server, and it will be immediately accessible to all the users of the software around the world.

In the field of corpus linguistics, online tools offer some unique advantages over offline and DIY tools. First, they can be designed to access massive corpora that would be too large to fit on anybody's personal computer. They can also be designed to limit a user's access to the corpus data (e.g. only producing wordlists and concordances of a fixed context size), which might be important if the corpus data are copyright protected or contain sensitive information. Also, custom online corpus tools can be designed and tailored to work with a specialised corpus that might use a unique POS-tagging system or rich annotation layer. They can also be specially tailored to facilitate the types of searches and analyses that the corpus was designed for and visualise the results accordingly. There are numerous examples of such online corpus interfaces. For example, online tools designed to provide a general interface to various large-scale corpora include *CQPweb,*[1] *English-Corpora.org*[2] and *Sketch Engine,*[3] mentioned earlier. There are also tools designed specifically for a single corpus, including the *MICUSP* interface[29] mentioned earlier, the *Scottish Corpus of Texts and Speech* (SCOTS) interface,[34] *OpenSourceShakespeare*[35] and many others. To get an idea of the breadth of availability of online tools that interface to custom corpora, you can visit the *Clarin* portal, which lists a vast number of corpora resources.[36]

Although online tools have many strengths, these same strengths can also be viewed as the tools' greatest weaknesses. For example, the fact that online tools often hide the raw data of a corpus behind a web browser interface means that the researcher is essentially dealing with a "black box" technology. They may not know what data are in the corpus or how the data were collected, cleaned or processed. They may not even know which parts of the corpus are being displayed by the tool. *English-Corpora.org,*[2] for example, restricts users to only viewing a random sample of concordance lines in its output. While this may be sufficient for viewing general patterns of word usage, it is far

from sufficient for discourse-level analyses. A related problem is that most online tools (with *Sketch Engine*[3] being a notable exception) do not allow the import of custom data. For researchers investigating a niche language variety or even a major language variety that happens to not have an online corpus tool available, it means that online tools can no longer be an option. Developing specialised online interfaces to different corpora also means that we get an explosion of different tools, each with its own idiosyncratic interface and results display. It should also be mentioned that the most widely used online corpus tools require registration and a subscription fee.

In today's world of the internet, you might wonder why anybody would still use an offline tool for corpus analysis. This is especially true when it is noted that the distinction between online and offline tools is becoming increasingly blurred. Through the development of technologies such as JavaScript and HTML5, online tools can now replicate many of the features of traditional offline tools, including menus, scrolling windows and animated displays. However, offline tools still have their place, especially when it comes to corpus research. First, offline tools provide the user with direct access to the data. It can be viewed in its raw form, loaded into the tool and probed in various ways. In this way, offline tools are more transparent than online tools. They can also offer users a more familiar and intuitive interface, reminiscent of other tools on their system, such as word processors. Importantly, the same interface can be used irrespective of the corpus data loaded into the tool. Offline tools also have more scope for expansion, especially if they provide third-party developers with the ability to create add-ons for the tool. The limited memory and processing power of personal computers may also not be a major concern for researchers if they are only dealing with moderately sized corpora that do not require distributed systems or advanced hardware to process. It is also important to note that offline tools are increasingly gaining features traditionally only associated with online tools, including direct access to cloud resources.

Currently, the most widely used general-purpose offline corpus analysis toolkits are *AntConc,*[4] which is a freeware tool for Windows, Macintosh and Linux computers, and *Wordsmith Tools,*[7] which is a commercial tool available for Windows (and which can also be run on other operating systems if Windows is installed as a virtual machine). Of course, there are numerous other offline tools, some designed to provide a broad range of corpus analysis functions (as discussed in Sections 2 and 3 of this chapter), and others designed for quite specific purposes, such as collecting, cleaning and tagging corpus data, or counting unique features in a corpora, as exemplified by *KfNgram*[37] and *AntGram,*[38] that are designed to identify open-slot phrases (p-frames) in a corpus.

DIY tools have a unique place in the history of corpus linguistics. In the early 1960s, the development of mainframe computers led to the birth of modern corpus linguistics. These mainframe computers are quite reminiscent of modern server computers, in the sense that they were large, powerful (for their time), externally managed machines that served entire departments or organisations. However, to use these machines, corpus researchers had to write (or use) specialised software programmes, load them into these mainframe computers (usually with punch cards) and wait sometimes days or weeks for the results to be returned. These DIY tools in many ways served as templates for the online and offline tools used today. Examples of these early programmes include *Concordance* (Dearing 1966), *Concordance Generator* (Smith 1966), *Discon* (Clark 1966) and *Drexel Concordance Program* (Price 1966). One particularly interesting example is *CLOC* (Reed 1978), which was a tool developed for the highly influential COBUILD project at the University of Birmingham, headed by John Sinclair. To create the CLOC

tool, the researchers in Sinclair's team invented a new programming language called ATOL (Reed 1978; Sinclair *et al.* 2004; Moon 2007). Today, DIY tools are again gaining prominence in corpus linguistics as researchers adopt increasingly advanced statistical measures and visualisation techniques to understand their data. Many of these measures and techniques are only applicable in narrowly scoped contexts, and so are outside the realm of more general-purpose online and offline tools. Therefore, DIY tools offer a researcher the possibility to be truly innovative in their work. Of course, the downside to this approach is that they need at least some knowledge of programming to confidently use DIY tools developed by others and a much greater knowledge of programming if they hope to develop such tools on their own. To illustrate this point, a bug in the code of a well-established online or offline tool is likely to be noticed by its many users, leading to corrections and updates, especially if the code is made open source.[39] On the other hand, a bug in the code of a DIY tool that has been posted on a forum or created by the researcher themselves may go unnoticed for a long period, potentially invalidating all the research produced by the tool.

2 Finding bottom-up language patterns

Online, offline and DIY tools are primarily used in corpus linguistics research studies to identify bottom-up language patterns. To illustrate the ways in which corpus software can assist in such tasks, I will use *AntConc*.[4] However, it should be noted that alternative tools can be used for this purpose. Many operate in similar ways, with the biggest differences being the interface design and simplicity or complexity of the searches made possible. The corpus data I will use with the software is the written component of the *International Corpus Network of Asian Learners of English (ICNALE)* corpus,[40] which comprises 1.3 million words of learner English from ten countries in Asia, collected in the form of 200- to 300-word essays (Ishikawa 2013).

What are the "important" words in a corpus?: word frequency and keyword analysis

Word frequencies

Corpus software tools are extremely accurate and fast at counting and ranking the occurrences of strings of characters in text files. If these strings of characters are defined to represent word tokens, then we can use the tools to count and rank the frequencies of words in and across the texts of a corpus. This gives us a rough measure of word "importance" based on frequency. By extension, we can also obtain a measure of word "importance" based on their "range" (dispersion) values through a corpus. Figure 9.1 shows a screenshot of *AntConc* displaying frequency values for all the words in the ICNALE corpus after the files are loaded into the tool via the file menu. The figure shows that the corpus comprises a total of 1,316,265 *tokens* (individual words) from 15,355 unique word *types*. What immediately stands out is the predominance of function words (*the*, *to*, *a*, *and*, *in*, *of*), which certainly play an important role in language. You may also notice two nouns in the list that stand out for their content value: *time* (rank 8) and *smoking* (rank 9). The ICNALE corpus essays are controlled for content with only the following two prompts being used: (a) *It is important for college students to have a part-time job* and (b) *Smoking should be completely banned at all the restaurants in the*

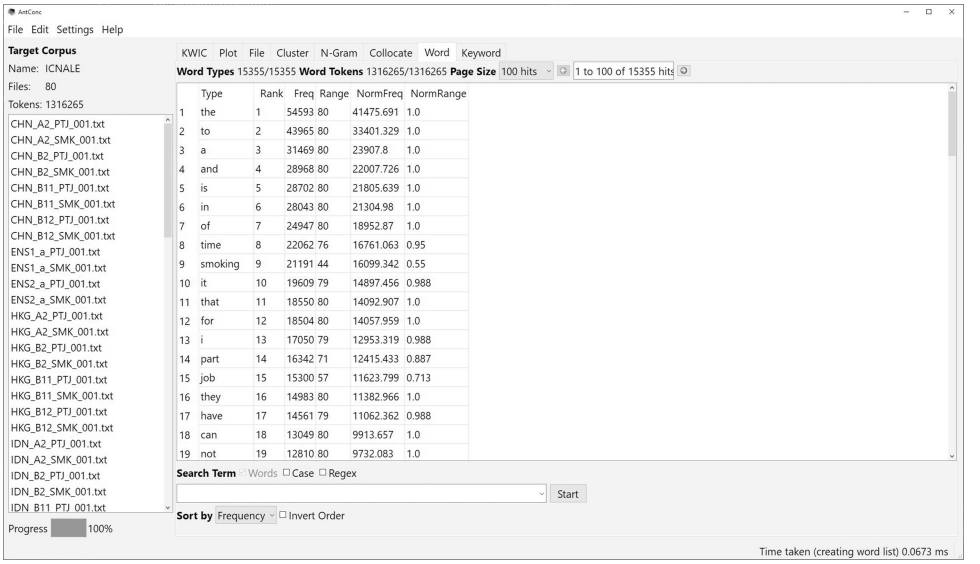


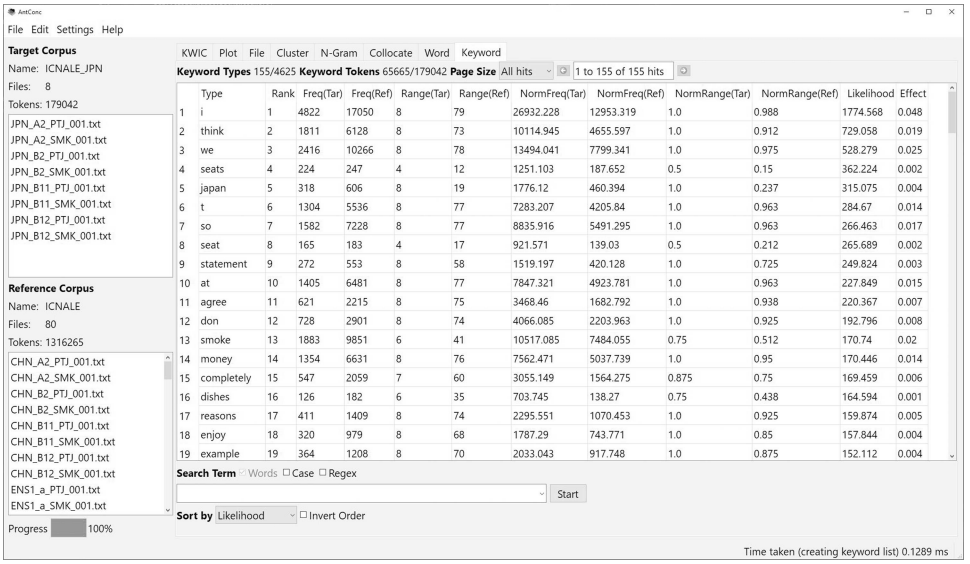
Figure 9.1 Screenshot of *AntConc* displaying frequency values for all the words in the ICNALE corpus

country. It is not surprising, therefore, to see these important content words ranked so high. It should be noted that the words *part* and *job* from *part-time-job* are ranked fifteenth and fourteenth in the list.

Keywords

We can go one step further and compare the frequencies of words in the *target* corpus (i.e. the corpus which is being examined) with those in another corpus that represents some kind of language standard (i.e. a *reference* corpus). By determining which words appear unusually frequently in the target corpus compared to the reference corpus using a statistical measure such as log likelihood, we can identify so-called *keywords* (see Pojanapunya and Watson Todd 2018). We can also do something similar but using dispersion as the counting measure (see Egbert and Biber 2019). For more on the statistical underpinnings of dispersion, see Chapter 13, this volume.

Figure 9.2 shows a screenshot of *AntConc* showing a standard frequency-based keyword list for the ICNALE written corpus. In this example, only the Japanese learner essays have been loaded into the tool through the file menu and serve as the *target* corpus, whereas the complete corpus (of 1.3 million words of learner English from ten countries in Asia) has been loaded into the tool via the keyword tool preference menu to serve as the *reference* corpus. In this way, the list of keywords represents words that appear unusually frequently in Japanese learner writing compared with Asian learner writing in general. Note that the keywords here have been ranked by the log likelihood keyness measure with a cutoff at $p < 0.05$ with a Bonferroni correction (Armstrong 2014), leaving 155 keywords remaining. This is the default setting, but it can be changed in the keyword tool preferences window.



	Type	Rank	Freq(Tar)	Freq(Ref)	Range(Tar)	Range(Ref)	NormFreq(Tar)	NormFreq(Ref)	NormRange(Tar)	NormRange(Ref)	Likelihood	Effect
1	i	1	4822	17050	8	79	26932.228	12953.319	1.0	0.988	1774.568	0.048
2	think	2	1811	6128	8	73	10114.945	4655.597	1.0	0.912	729.058	0.019
3	we	3	2416	10266	8	78	13494.041	7799.341	1.0	0.975	528.279	0.025
4	seats	4	224	247	4	12	1251.103	187.652	0.5	0.15	362.224	0.002
5	japan	5	318	606	8	19	1776.12	460.394	1.0	0.237	315.075	0.004
6	t	6	1304	5536	8	77	7283.207	4205.84	1.0	0.963	284.67	0.014
7	so	7	1582	7228	8	77	8835.916	5491.295	1.0	0.963	266.463	0.017
8	seat	8	165	183	4	17	921.571	139.03	0.5	0.212	265.689	0.002
9	statement	9	272	553	8	58	1519.197	420.128	1.0	0.725	249.824	0.003
10	at	10	1405	6481	8	77	7847.321	4923.781	1.0	0.963	227.849	0.015
11	agree	11	621	2215	8	75	3468.46	1682.792	1.0	0.938	220.367	0.007
12	don	12	728	2901	8	74	4066.085	2203.963	1.0	0.925	192.796	0.008
13	smoke	13	1883	9851	6	41	10517.085	7484.055	0.75	0.512	170.74	0.02
14	money	14	1354	6631	8	76	7562.471	5037.739	1.0	0.95	170.446	0.014
15	completely	15	547	2059	7	60	3055.149	1564.275	0.875	0.75	169.459	0.006
16	dishes	16	126	182	6	35	703.745	138.27	0.75	0.438	164.594	0.001
17	reasons	17	411	1409	8	74	2295.551	1070.453	1.0	0.925	159.874	0.005
18	enjoy	18	320	979	8	68	1787.29	743.771	1.0	0.85	157.844	0.004
19	example	19	364	1208	8	70	2033.043	917.748	1.0	0.875	152.112	0.004

Figure 9.2 Screenshot of *AntConc* showing a standard frequency-based keyword list for the Japanese component of the ICNALE written corpus

One thing that you might immediately notice in the figure is the disappearance of function words, which dominated a standard word frequency list. This is because function words generally appear with the same frequency distribution across all corpora. You might also notice that the words in the list are presented in lowercase, which is a common practice in the field, as it conflates the same word types together, although this setting can be changed in the tool preferences. The highest ranked keywords in the list are *I* and *think*, suggesting that the Japanese learners use the expression *I think* more frequently in their writing than other Asian students. The words *seat* and *seats* are also ranked high in the list, suggesting that seating in restaurants and other places might be an important theme of the essays. Two words in the list might be confusing on first viewing: *don* and *t*. With the default settings in *AntConc*, "words" are defined as strings of letter characters based on the Unicode letter character set.[41] This means that words like *don't* will be split into two parts. Similarly, words like *I'm* will be split into *I* and *m* and words like *20th* will lose the number and appear as just *th*. Although this setting reflects the standard practice in the field, it can be changed so that such words are not split. One final note of interest in the list is the fact that not only do *don* and *t* appear with a high ranking but also *I*, *agree* and *statement*. This result suggests that a key phrase in the Japanese writing might be *I don't agree with the statement that…* This point will be explored later.

One question that many novice researchers in corpus linguistics have is where to cut off the list of keywords before the rankings become meaningless. Strictly speaking, all 155 words in the list are keywords as defined by the statistical measure. However, they are not all "important" to the same degree. Log likelihood captures the size (or effect) of difference in frequency so it can be used not only as a cutoff measure but also a ranking measure. So, you may decide to look at only the first 50 or 100 keywords, as these are the

"most important". However, some researchers have argued that keyness measures should be complemented by a more robust effect size measure by which the words are ranked. Unfortunately, there is currently no established effect size measure that researchers agree on. In Figure 9.2, effect sizes for the keywords are shown based on the dice coefficient (the same measure used to rank keywords in *Sketch Engine*[3]). Although the keywords are ranked by log likelihood keyness values, this setting can also be changed in *AntConc* to order the keywords by effect size.

Word frequency, keyness and effect size measures are not the only ways to identify *important* words in a corpus. In the field of vocabulary research, one common practice is to divide up frequency and/or dispersion lists (or even lists based on other criteria such as curriculum plans or knowledge of words, see Schmitt *et al.* 2021) to create *level lists*. Then, going beyond *AntConc*, online tools such as *MultilingProfiler*[42] and *Compleat Lexical LexTutor*[43] and offline tools such as *AntWordProfiler*[44] can be used to *profile* corpus texts, highlighting words at these different levels and showing how many words the levels cover. This information can provide researchers (as well as teachers and learners) with a measure of the suitability or difficulty of texts for a particular purpose.

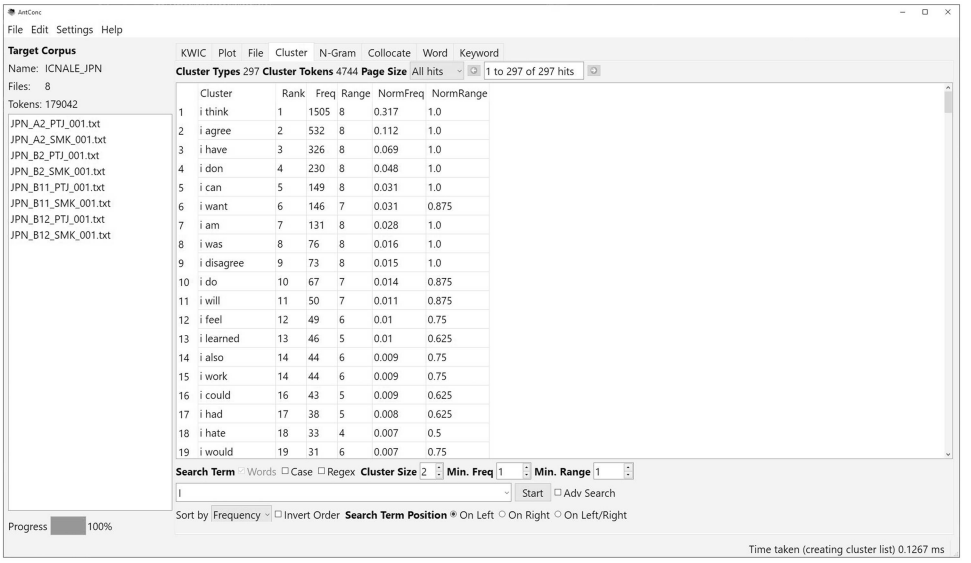
What are the "important" multi-word units in a corpus?: cluster, n-grams and lexical bundle analysis

The results of the keyness analysis shown in Figure 9.2 suggests that Japanese learners of English use *I think* and *I don't agree with the statement that…* in their writing relatively more frequently than other Asian students. Clearly, however, the two phrases are very different in nature. The first expression perhaps reflects a feature of the learners' writing style, whereas the second expression may reflect a particular worldview. Both aspects of language use can be explored using corpus tools.

To identify phrases or MWUs in a corpus, you can use a "cluster" tool (sometimes referred to as an "n-gram" tool or "lexical-bundle" tool depending on the researcher/developer). *AntConc*'s cluster tool allows you to search for a particular word or phrase and then displays all occurrences of that word/phrase together with a set number of words immediately preceding or following it. These results can immediately reveal frequent multi-word patterns that may have gone unnoticed using just a frequency or keyword list. Figure 9.3a shows a screenshot of *AntConc* displaying two-word clusters for the search word *I* ranked by frequency, where *I* appears on the left. (The settings can be changed to show clusters with the word appearing on the right.) You will notice, first, that the clusters appear in lowercase, as you found with the frequency and keyword tools. The highest-ranked cluster confirms the hypothesis that *I think* is prevalent in Japanese learner writing (the expression is used 1505 times across all 8 files in the corpus). You might also notice that *i agree*, *i don[t]*, and *i disagree* are all high-ranked clusters, suggesting that the worldview of the learners is more complex and diverse than that hinted at from just the keyword list results. When the tool settings are set to four-word clusters, the phraseology used by the students becomes more pronounced, with the highest-ranked clusters being *i think it is*, *i agree with the*, *i think that it* and *i agree with this* (see Figure 9.3b).

Corpus software tools can also be used to find the occurrence of every cluster in a corpus regardless of the search word, be it two words, three words, or four words long. Without additional filtering, these lists can sometimes look quite noisy, with two-word n-grams like *in the*, *it is* and *of the* ranking highly. However, by setting a minimum

(a)



(b)

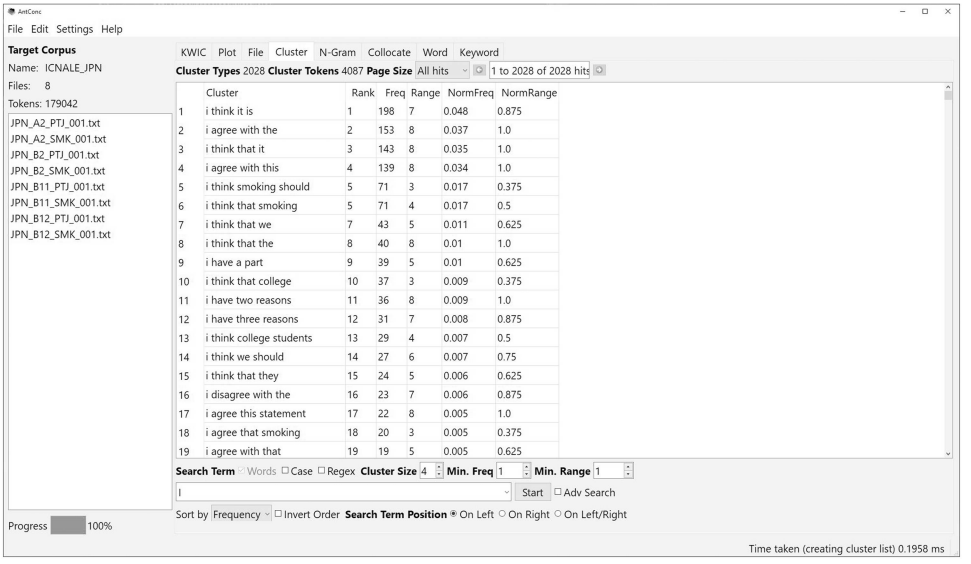


Figure 9.3 (a and b) Screenshots of *AntConc* displaying clusters in the ICNALE corpus

length, frequency and/or range for the n-gram (criteria that are traditionally used to define "lexical bundles"), more salient or "important" n-grams can be identified. Figure 9.4a and b, respectively, show screenshots of *AntConc* displaying all the two-word and four-word n-grams in the ICNALE corpus. As the length of the n-grams increases, you will notice phrases such as *a part time job*, *restaurants in the country* and *should be completely banned* stand out. In this way, n-gram analysis can be useful in exploratory research, revealing features of a corpus that might easily go unnoticed through a close reading of individual texts or a casual inspection of the corpus as a whole.

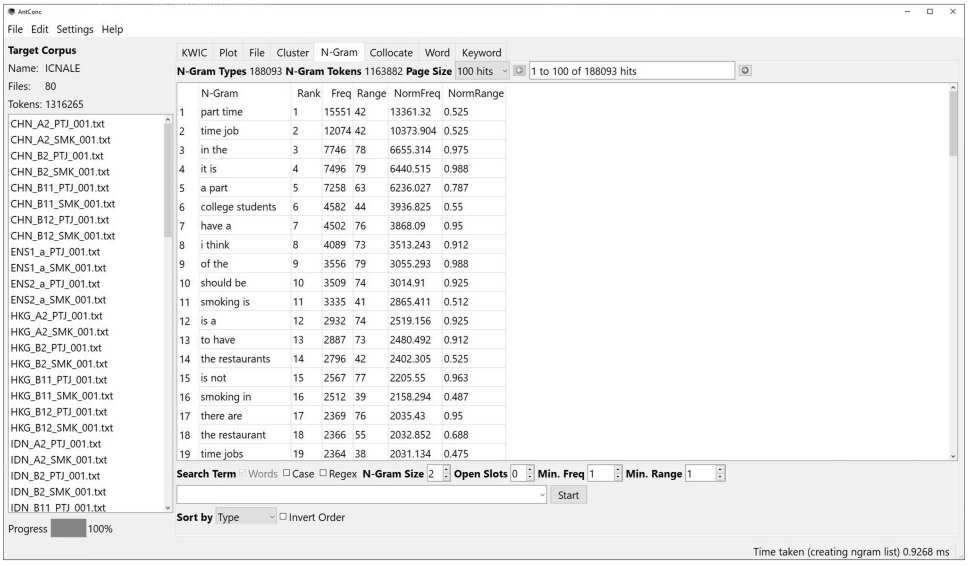
How are words connected to other words?: collocate analysis

Cluster tools are useful for revealing phrase and MWU patterns when the words are contiguous (appearing next to each other). However, we also know that words are connected to each other at further distances. For example, in a learner's essay on the topic of smoking, we can imagine that the word *smoking* may be linked strongly with the word *banned*, as in the phrase *smoking should be banned*, even though the words are not contiguous. Corpus software tools can quicky and accurately calculate the strength of connection between a search word and every other word in the corpus using a statistical measure, with the Mutual Information and T-Score measures being common choices (for more information see Gries 2010 and Chapter 13, this volume). The result of such an analysis is a list of so-called "collocates" of a target word, with the strength of connection (or collocation) used to rank the words.

To prevent the tool from generating spurious collocates (e.g. words with very low frequencies that just happen to appear near the target word), the tools include options and filters. For example, *AntConc* has an option to only include collocates that reach a statistical threshold level based on log likelihood. It also has options to restrict collocates to only those that appear above a minimum frequency. Figure 9.5 shows a screenshot of *AntConc* displaying the collocates of *smoking* in the ICNALE corpus that appear statistically significantly together (log likelihood; $p < 0.05$) and have a minimum frequency of ten in the corpus. The strongest collocate is *cessation*, which appears 3 times to the left and 12 times to the right of *smoking* in the corpus in phrases such as *smoking cessation leads to reduced stress*. The next highest collocate is *section*, which appears far more frequently in the corpus, but not quite at the same strength of association. One phrase in the corpus where this collocate appears is *divided in smoking and non smoking section*, revealing that perhaps this collocate reflects a common language learning error (*section* being treated as a plural noun). As expected, the word *banning* (rank 10) is also a strong collocate of *smoking*, appearing in phrases such as *banning smoking*, *banning of smoking* and *banning all smoking areas*.

Collocate analysis is an extremely powerful way to identify patterns of language usage. It is not surprising, therefore, that many online tools offer this function in different forms, including *FLAX,*[45] *Hyper Collocation,*[46] *Just* the *Word,*[47] *SkELL,*[48] *StringNet*[49] and *WriteAway.*[50] As mentioned earlier, there are also offline tools such as *#LancsBox*[5] that can present collocates of a target word in the form of a network of connected nodes, where clicking on one node will expand the network to show further connections between the collocate and its related collocates.

(a)



(b)

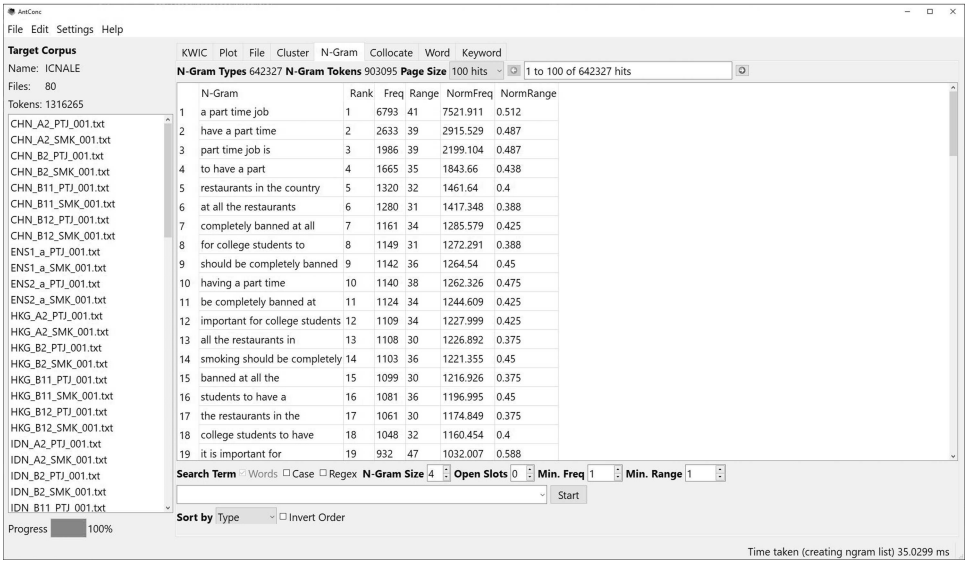
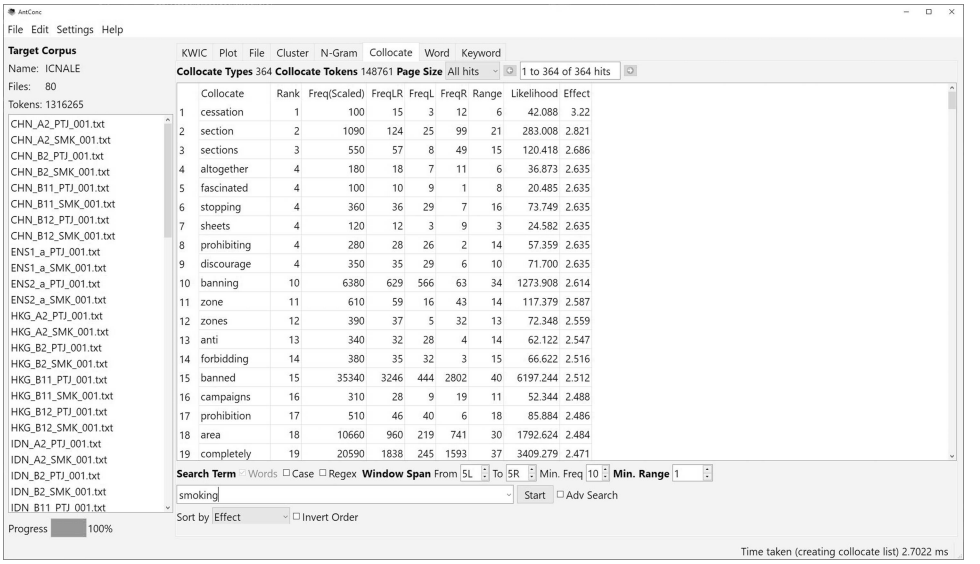


Figure 9.4 (a and b) Screenshots of *AntConc* displaying n-grams in the ICNALE corpus



	Collocate	Rank	Freq(Scaled)	FreqLR	FreqL	FreqR	Range	Likelihood	Effect
1	cessation	1	100	15	3	12	6	42.088	3.22
2	section	2	1090	124	25	99	21	283.008	2.821
3	sections	3	550	57	8	49	15	120.418	2.686
4	altogether	4	180	18	7	11	6	36.873	2.635
5	fascinated	4	100	10	9	1	8	20.485	2.635
6	stopping	4	360	36	29	7	16	73.749	2.635
7	sheets	4	120	12	3	9	3	24.582	2.635
8	prohibiting	4	280	28	26	2	14	57.359	2.635
9	discourage	4	350	35	29	6	10	71.700	2.635
10	banning	10	6380	629	566	63	34	1273.908	2.614
11	zone	11	610	59	16	43	14	117.379	2.587
12	zones	12	390	37	5	32	13	72.348	2.559
13	anti	13	340	32	28	4	14	62.122	2.547
14	forbidding	14	380	35	32	3	15	66.622	2.516
15	banned	15	35340	3246	444	2802	40	6197.244	2.512
16	campaigns	16	310	28	9	19	11	52.344	2.488
17	prohibition	17	510	46	40	6	18	85.884	2.486
18	area	18	10660	960	219	741	30	1792.624	2.484
19	completely	19	20590	1838	245	1593	37	3409.279	2.471

Figure 9.5 Screenshot of *AntConc* displaying the collocates of *smoking* in the ICNALE corpus

How are words used in context?: concordance analysis

Understanding why words, keywords, clusters, n-grams, lexical bundles and collocates appear at the ranks they do requires analysing the context in which they appear in a corpus. One way to do this is simply by searching for the item in an individual text and then closely reading that text to understand the context. However, when there are many tens, hundreds or thousands of texts in the corpus, this becomes a serious challenge. Also, it is virtually impossible to identify frequently occurring patterns of usage this way. Corpus linguists understood this problem from the very earliest days and developed software tools to address the problem. The result is a concordancer (sometimes referred to as Key-Word-In-Context (KWIC) concordancer).

Concordancers are designed to find all the occurrences of a search term in a corpus and display these in an ordered fashion together with the words that surround them. The display is usually referred to as a "concordance", and the analysis of concordances is often referred to as "concordancing". Figure 9.6 shows a screenshot of *AntConc* displaying the concordance for the search term (or node) *smoking* in the ICNALE corpus. You will notice many interesting features from the figure. First, the concordance lines do not correspond to sentences. Rather, they are designed to position the target word in the center of the screen surrounded by a fixed number of characters (or words) to the left and right. (Note, however, that some concordancers give the option to display concordance lines as complete sentences.) Second, the lines are ordered by the frequency of occurring patterns in the results according to the sort parameters. In Figure 9.6, these are the phrases that result from combining the first, second, and then third word to the right of the search word as indicated by the 1R, 2R, and 3R settings at the bottom of the screen. In AntConc, these are referred to as KWIC patterns. This ordering allows salient patterns to be noticed as you scroll through the concordance. For example, the ordering shown in Figure 9.6 immediately reveals that *smoking is a bad...* is the most salient smoking pattern used by the

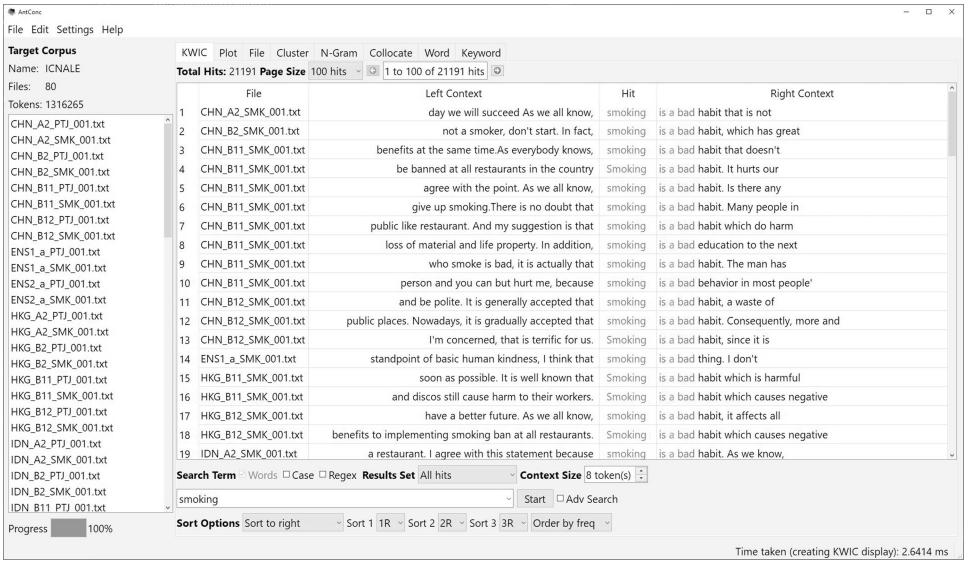


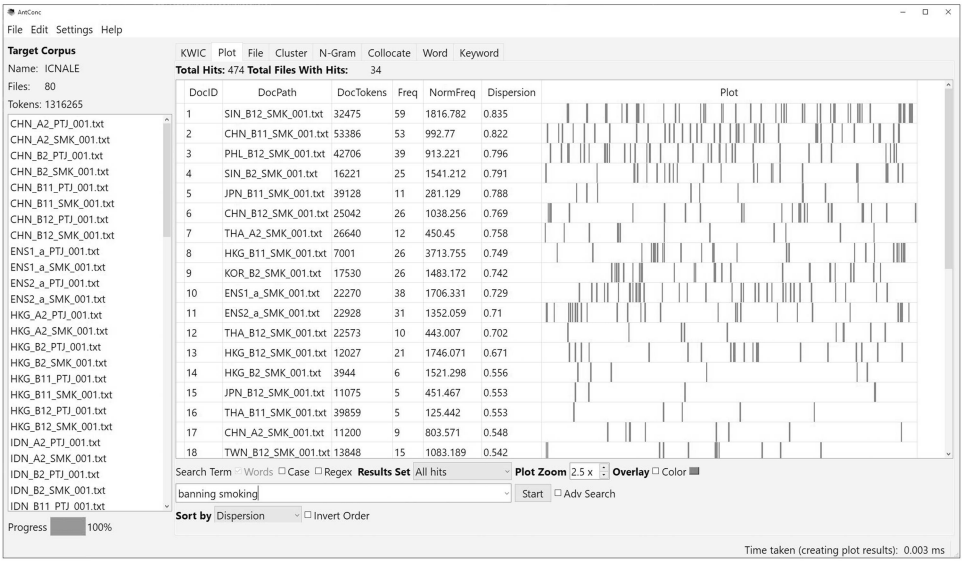
Figure 9.6 Screenshot of *AntConc* displaying the concordance for the search term *smoking* in the ICNALE corpus

learners in the corpus. Note that the ordering presented in *AntConc* is a first in the field, with other tools opting to order concordance lines alphabetically according to the sort parameters (an option also available in *AntConc*). Newcomers to the field may question why an alphabetical ordering of the lines would be used in so many tools, despite its obvious limitations. For a discussion of this intriguing, question, see Anthony (2018).

Another feature of concordancers is that they usually provide a way for the user to see the target word in a concordance line within a wider context. For example, in *AntConc*, if you click on the target word in any concordance line, the software will jump to the *File* tool and show the target word within the context of the original file. Finally, you may notice a Results Set option in the concordance display of *AntConc*. This allows a so-called "thinning" of the results to reduce the overall number of lines displayed in the concordance whilst still revealing common patterns of usage. This feature (and the equivalent "random sample" option in other tools) is particularly useful when dealing with very large corpora that might generate thousands or millions of results and thus far too many to scroll through to find all the salient patterns.

Concordancers are not the only tools that can reveal the contexts in which a word is used. One alternative tool is called a concordance plot (or dispersion plot) tool, which shows the positions of a search word or phrase in all the corpus texts, positioned relative to the start and end points of the files. Figure 9.7a and b show screenshots of *AntConc* displaying the concordance plot for the phrase *banning smoking* in the ICNALE corpus. In Figure 9.7a, the lengths of the texts are normalised, which makes it easier to compare the relative positions of the search term across texts. In Figure 9.7b, the relative lengths of the texts are maintained, highlighting the fact that some essays are a great deal longer or shorter than others. In both figures, it is clear that the phrase *banning smoking* is not dispersed through texts in the same way, with some learners using the term frequently,

(a)



(b)

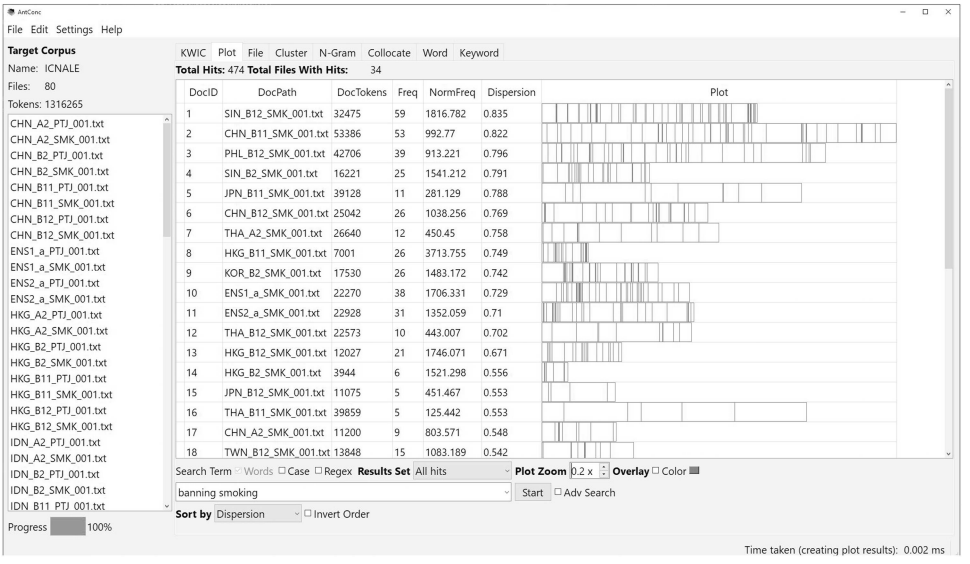


Figure 9.7 (a and b) Screenshots of *AntConc* displaying concordance plots for the phrase *banning smoking* in the ICNALE corpus

presumably arguing for or against the banning of smoking, and others rarely using the phrase at all, perhaps because they express an alternative opinion. To investigate this further, you can click on any of these plot lines, and *AntConc* will take you directly to the *File* tool and show the word in context as it does with the concordance tool.

There are also concordancers specifically designed to work with aligned or parallel corpora. Online tools such as *SketchEngine*[3] and dedicated offline tools such as *AntPConc,*[51] for example, allow you to generate concordances in a corpus of works in one language and link these results to matching lines in a corpus of translations. These tools can be invaluable for translators and others working on multi-lingual projects (see Chapter 34, this volume).

3 Finding top-down language patterns

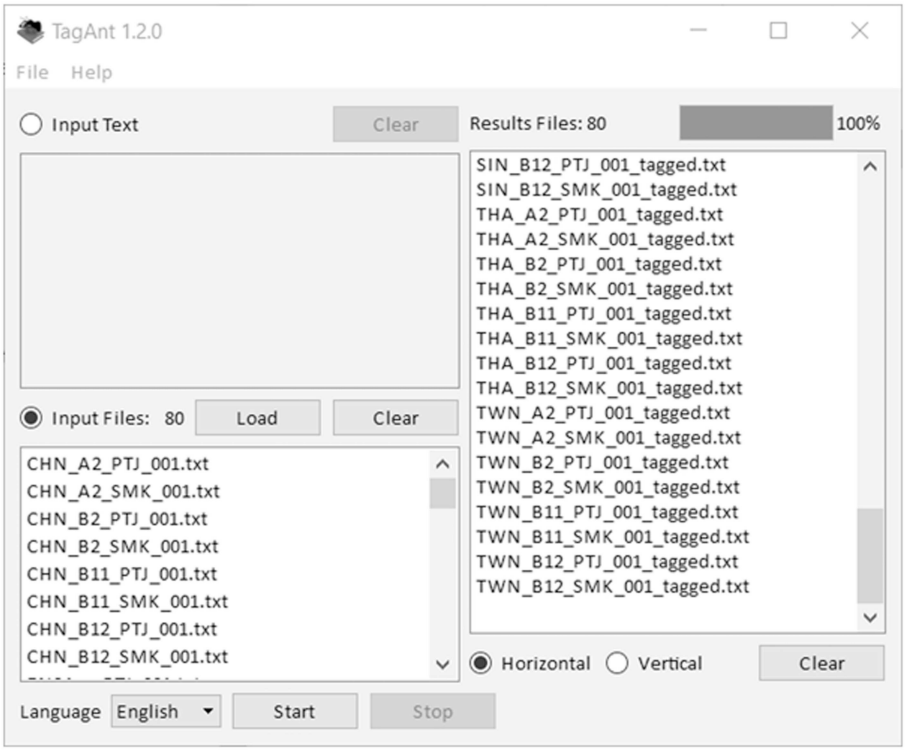
How can corpora be searched for top-down features of languages?

In the previous section, I described some of the most common corpus tools for finding bottom-up language patterns. Through the study of words, keywords, clusters, n-grams, lexical bundles, collocates and concordances, researchers can also begin to form hypotheses and make predictions about top-down language features, such as text cohesion, register, discourse structure and pragmatic phenomenon. The first step towards this goal is often to tag the corpus texts with POS information and/or annotate the texts with higher-level markup that signals features of interest. It is then a relatively trivial task to count these features or search for them in a corpus tool as you would count or search for words and phrases. Fortunately, many existing corpora are released with such information. *The Spoken BNC2014*[52] (Love *et al.* 2017) is a recent example of such a corpus, containing a rich set of tags and annotations marking token-level features, such as POS, lemma forms, utterances and overlaps; sub-text-level features, such as speaker demographics; and text-level features, such as the situational context and number of speakers (see Chapter 4, this volume). Another recent example of such a corpus is *CorCenCC*[17] (Knight *et al.* 2020), which provides a similar depth of tags and annotations for spoken and written Welsh language (see Chapter 3, this volume).

When a corpus is not released with POS tags or you are making your own corpus, it is also possible to use corpus tools to tag the corpus directly. Figure 9.8a shows a screenshot of the *TagAnt*[19] tool in the process of tagging texts from the ICNALE corpus. After dropping the files into the interface (or loading them via the file menu), with one click of the start button, the tool can rapidly and surprisingly accurately tag the corpus with POS tags using the *TreeTagger*[20] engine. Figure 9.8b shows a screenshot of *AntConc* displaying a wordlist for the ICNALE corpus using this tagged data. The most frequent "words" are now a sentence period, a comma, and the word *the*, which is tagged as a determiner (DT). It is important to note that to use this POS data, it had to be loaded into AntConc via one of the corpus manager settings in the file menu. Other settings in the manager allow AntConc to read data with even more complex formats.

Although POS-tagging corpus data is a fast and relatively simple task, manually annotating corpora for top-down-level features of language can be a slow and error-prone activity. Therefore, several corpus software developers have strived to develop tools that can automatically annotate corpus texts with top-down features, although this is still a surprisingly underdeveloped area (see Chapter 8, this volume). One very early example is *AntMover,*[53] which aims to automatically annotate the "move" structure of

(a)



(b)

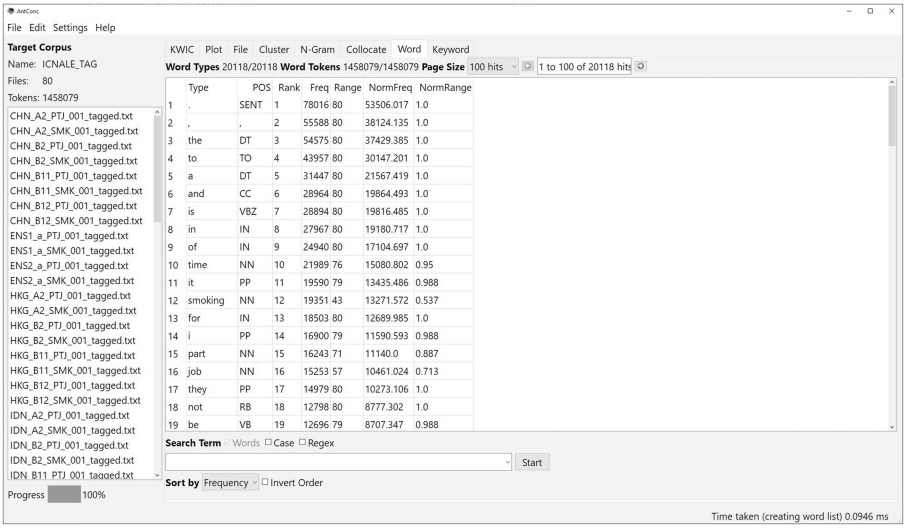


Figure 9.8 (a and b) *TagAnt* POS tagging texts and *AntConc* displaying a wordlist for the ICNALE corpus tagged by part of speech

texts in a way that is reminiscent of the work of Swales (1990) and many others. A more recent example is *AntCorGen,*[15] mentioned earlier, which collects research articles and assigns them to section categories such as *title*, *abstract* and *introduction*, allowing researchers to easily compare and contrast language features as they appear in these higher-level structural units.

For researchers interested in evaluating texts against a range of top-down measures, such as readability, sentiment, lexical cohesion and lexical sophistication, *the Suite of Automatic Linguistic Analysis Tools (SALAT)* tools can be invaluable.[54] To use these freely available tools, you generally need to just download and launch the relevant tool, load in your target texts, set the various options and click start. Also, if you are interested in identifying prototypical or atypical texts within a larger corpus, perhaps to confirm a hypothesis or identify outliers, the *ProtAnt*[25] can be useful. Again, to use this freeware tool, you just need to just download and launch the software, load in your target texts, set the various options and click start, and it will rank the texts accordingly. Figure 9.9 shows a screenshot of *ProtAnt* after it has ranked all the texts of the ICNALE corpus in terms of their use of academic words[55] (Coxhead 2000). The results reveal that the file *SIN_B2_PTJ_001* is the most prototypically academic, with 178.7 academic words per 1,000 words, compared with file *THA_B2_SMK_001*, which only uses 33.8 academic words per 1,000 words.

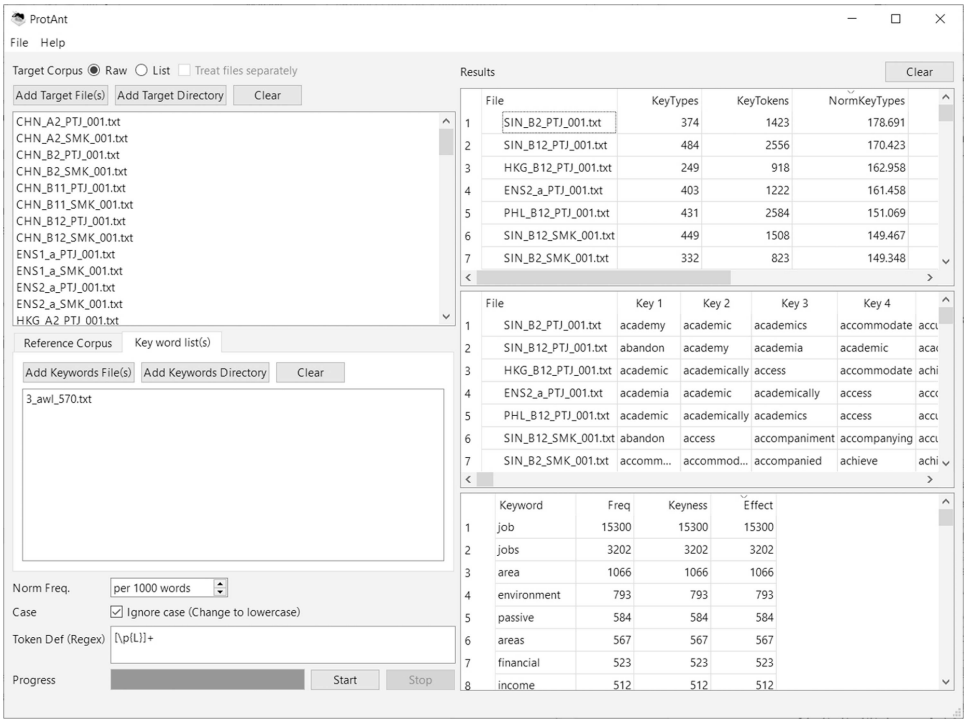


Figure 9.9 Screenshot of *ProtAnt* after ranking texts of the ICNALE corpus by their use of academic words

4 Managing data in and between corpus tools

Online, offline and DIY software tools provide corpus linguists with a huge range of options for collecting, analysing and visualizing data. However, no single tool is likely to provide the full range of features and functions necessary to complete a corpus study. For example, most data collection tools will offer few, if any, ways to analyse the data collected. General-purpose corpus toolkits like *AntConc*[4] provide many of the basic features for analysing corpora, but they do not provide advanced statistical measures and visualisation techniques that are only available in cutting-edge DIY tools. Conversely, DIY tools are unlikely to come with any data and can usually only perform a very narrow range of analyses. So, they not only need to be supplied with data, but they also need to output data in a form that can be used with other tools. These limitations of all online, offline and DIY tools highlight the importance of *data interoperability*, a term used in computer science to describe the ability of systems to share data and resources with other systems.

Anthony and Evert (2019) consider four levels of interoperability that can be adopted by different systems. One level is the adoption of a common corpus format, which allows corpus creators to build corpora with the knowledge that the data can be loaded smoothly into an existing tool for analysis. One commonly adopted format at this level is the UTF-8 encoded plain text format. Corpora saved in this format are stripped of all tags and mark-up, and the UTF-8 encoding ensures that all the word strings in the corpus will be interpreted and rendered correctly by the corpus analysis tool. The ICNALE corpus used in the previous section is released in such a format. Another commonly adopted format is a simple tagged corpus format, where each word in a UTF-8 encoded plain text corpus is tagged with an underscore followed by the relevant tag (e.g. *the_dt*).

A second level of interoperability is the adoption of a standardised plug-in API architecture by the host system, which allows third-party developers to add features to existing systems. While such plug-ins are commonly found in many computer applications (notably text editors such as *Notepad++*[56] and programming environments such as *Visual Studio Code*[57]), they are rarely found in current corpus analysis tools, perhaps due to the size of the community and the limited number of people in the community who could contribute such plug-ins. The third level of interoperability is a reverse of the second level, with the host system adopting a generalised platform that feeds data and results to specialised plug-in modules. Again, though, this is rarely seen in existing corpus tools.

The fourth and perhaps most promising form of interoperability described by Anthony and Evert (2019) is the adoption of a standard *data* format (as compared to a *corpus* format). Already, this form of interoperability is used to some extent by many tools. For example, tools will often output results in a UTF-8 encoded text-based, tabular format (e.g. tab-separated values [TSV]) that can be loaded directly into spreadsheet programs like Microsoft Excel for further analysis and visualisation. The tabular format can also be read by most DIY tools, offering more opportunities for analysis and visualisation. Anthony and Evert extend this idea and propose a standardised tabular data model composed of one or more tables stored as a collection of text files in a TSV format or collectively in a single *SQLite*[58] database file. The structure can capture the output of corpus tools in a standardised way and greatly simplify the sending and receiving of data between tools. *AntConc*[4] fully adopts this *multiple TSV (MTSV)* format for import and export of all data, but it has yet to be incorporated into other

mainstream tools. However, the authors have received interest from many of the major corpus tool developers in the field and so there is hope that it will become more widely adopted.

5 Programming your own tools

Much of this chapter has been focused on the possibilities afforded by online, offline and DIY corpus software tools. This does not mean, however, that you should ignore the option of programming your own custom tools in a truly DIY fashion. Minimally, a basic understanding of programming will provide you with an understanding of how existing corpus tools work, raising your awareness of their limitations and the existence of possible bugs in the system. A basic knowledge of programming is also vital if you want to safely use DIY scripts provided by others. Simple mistakes in the format of input data or the commands to run these tools can often render the output completely meaningless. However, a knowledge of programming also offers far greater possibilities. During the corpus data collection stage, simple file (re)naming and cleaning scripts that run in fractions of a second can save a researcher days, weeks or even months of manual work. During the analysis stage, existing scripts can be modified or new scripts created that circumvent the need to manually edit data to suit a particular tool or constantly switch between different tools before arriving at a desired answer. In short, programming opens the possibility to create new tools that advance your research is ways that no existing tools can. In the words of Stefan Gries (2009: 12), programming puts you 'in the driver's seat'.

Fortunately, learning to program is becoming far easier than it was in the past. Firstly, modern scripting languages such as Python are designed to be easier to understand and more "human-readable" than older languages, like C. Secondly, there are now many freely available and excellently designed online courses such as those at *w3schools.com,*[59] as well as numerous online video tutorials on *YouTube* and other content creator platforms. Thirdly, languages such as Python and R have a huge community of users, who are willing to answer questions and offer advice on programming issues at sites such as *StackOverflow.*[8] This is especially true in the case of the R community within corpus linguistics, which has a strong and growing presence in the field. As an example, the *Language Technology and Data Analysis Laboratory (LADAL)*[60] run by Martin Schweinberger provides information and practical, hands-on tutorials to help researchers develop their data, text and statistical analysis skills. In short, there has never been a better time to start learning to code.

Notes

1 *CQPWeb*. Accessed at: https://cqpweb.lancs.ac.uk/
2 *English-Corpora.org*. Accessed at: https://www.english-corpora.org/corpora.asp
3 *Sketch Engine.* Accessed at: https://www.sketchengine.eu/
4 *AntConc*. Accessed at: https://www.laurenceanthony.net/software/AntConc/
5 *#LancsBox*. Accessed at: http://corpora.lancs.ac.uk/lancsbox/
6 *Wmatrix*. Accessed at: http://ucrel.lancs.ac.uk/wmatrix/
7 *WordSmith Tools*. Accessed at: https://www.lexically.net/wordsmith/
8 *Stack Overflow*. Accessed at: https://stackoverflow.com/
9 *GitHub.* Accessed at: https://github.com/
10 *NLTK*. Accessed at: https://www.nltk.org/

11 *Stefan Th. Gries companion website for QCLWR*: Accessed at: http://www.stgries.info/research/qclwr/qclwr.html
12 *AntFileConverter*. Accessed at: https://www.laurenceanthony.net/software/antfileconverter/
13 *Web Scraper*. Accessed at: https://webscraper.io/
14 *BootCaT*. Accessed at: https://bootcat.dipintra.it/
15 *AntCorGen*. Accessed at: https://www.laurenceanthony.net/software/antcorgen/
16 *PLOS ONE*. Accessed at: https://journals.plos.org/plosone/
17 *CorCenCC*. Accessed at: https://www.corcencc.org/
18 *FireAnt*. Accessed at: https://www.laurenceanthony.net/software/fireant/
19 *TagAnt*. Accessed at: https://www.laurenceanthony.net/software/TagAnt/
20 *TreeTagger*. Accessed at: https://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/
21 *SegmentAnt*. Accessed at: https://www.laurenceanthony.net/software/segmentant/
22 *UCREL Semantic Analysis System (USAS)*. Accessed at: http://ucrel.lancs.ac.uk/usas/
23 *Stanford Parser*. Accessed at: https://nlp.stanford.edu/software/lex-parser.shtml
24 *UAM CorpusTool*. Accessed at: http://www.corpustool.com/
25 *ProtAnt*. Accessed at: https://www.laurenceanthony.net/software/protant/
26 *AntMover*. Accessed at: https://www.laurenceanthony.net/software/antmover/
27 *AWSuM*. Accessed at: http://langtest.jp/awsum/
28 *What is JavaScript*? Accessed at: https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript
29 *MICUSP*. Accessed at: https://elicorpora.info/
30 *Kaleidographic*. Accessed at: http://www.kaleidographic.org/
31 *Jack Grieve's homepage*. Accessed at: https://sites.google.com/view/grievejw
32 *Jack Grieve's resources*. Accessed at: https://sites.google.com/view/grievejw/resources?authuser=
33 *Stefan Th. Gries homepage*. Accessed at: http://www.stgries.info/index.html
34 *Scottish Corpus of Texts and Speech (SCOTS) corpus*. Accessed at: https://www.scottishcorpus.ac.uk/
35 *OpenSourceShakespeare*. Accessed at: https://www.opensourceshakespeare.org/
36 *CLARIN Resource Families*. Accessed at: https://www.clarin.eu/portal
37 *KfNgram*. Accessed at: http://www.kwicfinder.com/kfNgram/kfNgramHelp.html
38 *AntGram*. Accessed at: https://www.laurenceanthony.net/software/antgram/
39 *What is open source?* Accessed at: https://opensource.com/resources/what-open-source
40 *ICNALE: The International Corpus Network of Asian Learners of English*. Accessed at: http://language.sakura.ne.jp/icnale/
41 *Programming with Unicode*. Accessed at: https://unicodebook.readthedocs.io/unicode.html
42 *MultiLingProfiler*. Accessed at: http://www.multilingprofiler.net/
43 *Compleat Lexical LexTutor*. Accessed at: https://www.lextutor.ca/
44 *AntWordProfiler*. Accessed at: https://www.laurenceanthony.net/software/antwordprofiler/
45 *FLAX*. Accessed at: http://flax.nzdl.org/
46 *Hyper Collocation*. Accessed at: https://hypcol.marutank.net/?q=highlightandd=f
47 *Just the Word*. Accessed at: http://www.just-the-word.com/
48 *SkELL*. Accessed at: https://skell.sketchengine.co.uk/
49 *String Net*. Accessed at: http://nav4.stringnet.org/ret.php
50 *WriteAway*. Accessed at: http://writeaway.nlpweb.org/
51 *AntPConc*. Accessed at: https://www.laurenceanthony.net/software/antpconc/
52 *British National Corpus 2014*. Accessed at: http://corpora.lancs.ac.uk/bnc2014/
53 *AntMover*. Accessed at: https://www.laurenceanthony.net/software/antmover/
54 Suite of Automatic Linguistic Analysis Tools (SALAT). Accessed at: https://www.linguisticanalysistools.org/
55 *The Academic Word List*. Accessed at: https://www.wgtn.ac.nz/lals/resources/academicwordlist
56 *Notepad++*. Accessed at: https://notepad-plus-plus.org/
57 *Visual Studio Code*. Accessed at: https://code.visualstudio.com/
58 *SQLite*. Accessed at: https://www.sqlite.org/
59 w3schools.com. Accessed at: https://www.w3schools.com/
60 *Language Technology and Data Analysis Laboratory*. Accessed at: https://slcladal.github.io/

Further reading

Anthony, L. (2013) 'A Critical Look at Software Tools in Corpus Linguistics', *Linguistic Research* 30(2): 141–61. (A look at the history of corpus tools development and a proposal for corpus tools development as part of a team.)

Anthony, L. (2021) 'Programming for Corpus Linguistics', in M. Paquot and St. Th. Gries (eds) *Practical Handbook of Corpus Linguistics*, Berlin and New York: Springer, pp. 181–207. (An introduction to the basic concepts of programming with step-by-step examples for writing simple corpus tools.)

McEnery, T. and Hardie, A. (2012) *Corpus Linguistics: Method, Theory and Practice*, Cambridge: Cambridge University Press. (A comprehensive review of corpus linguistics charting its history from the 1960s and earlier and explaining the most common tools, methods and practices.)

Viana, V., Zyngier, S. and Barnbrook, G. (eds) (2011) *Perspectives on Corpus Linguistics*, Amsterdam: John Benjamins Publishing. (A unique book that records interviews with leading corpus linguists of the field who express their opinions on a wide range of topics, including the importance of tools and the value of programming knowledge.)

References

Anthony, L. (2018) 'Visualization in Corpus-Based Discourse Studies', in C. Taylor and A. Marchi (eds) *Corpus Approaches to Discourse: A Critical Review*, London: Routledge, pp. 197–224.

Anthony, L. and Evert, S. (2019) 'Embracing the Concept of Data Interoperability in Corpus Tools Development', *paper presented at Corpus Linguistics 2019 (CL2019)*, 1st–3rd July 2019, Cardiff, UK.

Armstrong, R. A. (2014) 'When to Use the Bonferroni Correction', *Ophthalmic and Physiological Optics* 34(5): 502–508.

Clark, R. (1966) *Computers and the Humanities* 1(3): 39.

Coxhead, A. (2000) 'A New Academic Word List', *TESOL Quarterly* 34(2): 213–38.

Dearing, V. A. (1966) *Computers and the Humanities* 1(3): 39–40.

Egbert, J. and Biber, D. (2019) 'Incorporating Text Dispersion into Key Word Analyses', *Corpora* 14(1): 77–104.

Gries, St. Th. (2009) 'What is Corpus Linguistics?', *Language and Linguistics Compass* 3: 1–17.

Gries, St. Th. (2010) 'Useful Statistics for Corpus Linguistics', in A. Sánchez Pérez and M. Almela Sánchez (eds) *A Mosaic of Corpus Linguistics: Selected Approaches*, Frankfurt am Main: Peter Lang, pp. 269–91.

Gries, St. Th. (2016) *Quantitative Corpus Linguistics with R: A Practical Introduction*, London: Routledge.

Ishikawa, S. (2013) 'The ICNALE and Sophisticated Contrastive Interlanguage Analysis of Asian Learners of English', in S. Ishikawa (ed.) *Learner Corpus Studies in Asia and the World*, Kobe, Japan: Kobe University, pp. 91–118.

Knight, D., Morris, S., Fitzpatrick, T., Rayson, P., Spasić, I., Thomas, E-M., Lovell, A., Morris, J., Evas, J., Stonelake, M., Arman, L., Davies, J., Ezeani, I., Neale, S., Needs, J., Piao, S., Rees, M., Watkins, G., Williams, L., Muralidaran, V., Tovey-Walsh, B., Anthony, L., Cobb, T., Deuchar, M., Donnelly, K., McCarthy, M. and Scannell, K. (2020) *CorCenCC: Corpws Cenedlaethol Cymraeg Cyfoes - The National Corpus of Contemporary Welsh*, Cardiff: Cardiff University. 10.17035/d.2020.0119878310

Love, R., Dembry, C., Hardie, A., Brezina, V. and McEnery, T. (2017) 'The Spoken BNC2014: Designing and Building a Spoken Corpus of Everyday Conversations', *International Journal of Corpus Linguistics* 22(3): 319–44.

Moon, R. (2007) 'Sinclair, Lexicography, and the COBUILD Project: The Application of Theory', *International Journal of Corpus Linguistics* 12(2): 159–81.

Pojanapunya, P. and Watson Todd, R. (2018) 'Log-likelihood and Odds Ratio: Keyness Statistics for Different Purposes of Keyword Analysis', *Corpus Linguistics and Linguistic Theory* 14(1): 133–67.

Price, K. (1966) *Computers and the Humanities* 1(3): 39.

Reed, A. (1978) *CLOC [Computer Software]*, Birmingham: University of Birmingham.
Schmitt, N., Dunn, K., O'Sullivan, B., Anthony, L. and Kremmel, B. (2021) *Knowledge-Based Vocabulary Lists*, Sheffield: Equinox Publishing Ltd.
Sinclair, J., Jones, S. and Daley, R. (2004) *English Collocation Studies: The OSTI Report*, London: Continuum.
Smith, P. H. (1966) *Computers and the Humanities* 1(3): 39.
Swales, J. (1990) *Genre Analysis: English in Academic and Research Settings*, Cambridge: Cambridge University Press.