
# TACL 图形用户界面（GUI）用户手册

[GUI 版本 1.0.0]

何书群(Michael Radich)
纪子茵(Sharon Chi)
（海德堡大学跨文化研究中心）

翻译：林乾 王诗玮
（青岛大学历史学院）

# 目录

我们建议用户在使用TACL GUI之前通读本手册的内容。

使用TACL GUI需要掌握一定的背景知识，包括该工具如何工作，以及如何在概念上最好地构思其在中国佛教问题研究中的使用。请用户务必耐心地学习相关背景知识。

本手册中使用的TACL专用术语在本文件末尾的词汇表中列出。

## 鸣谢

TACL GUI图形用户界面由海德堡大学博士研究生Roland Borsos（roland.borsos@hcts.uni-heidelberg.de）自告奋勇为我们编写。TACL的源代码则是由Jamie Norrish（https://github.com/ajenhl/tacl）创建。我们非常感谢他们在时间和专业知识上慷慨的贡献。

TACL和TACL GUI的相关工作得到了亚历山大·冯·洪堡基金会、惠灵顿维多利亚大学、海德堡大学和德国科学基金会(DFG, 项目编号 RA 3202/1-1)的支持。我们对这些支持表示感谢。

## 技术要求

TACL是用Python编程语言编写的，但用户无须单独安装Python——GUI中已捆绑了Python和所有必要组件。

TACL GUI适用于Windows、Linux和Mac操作系统（感谢Merlin Beutlberger为Mac版本提供的帮助）。

Windows用户需要Windows10或更高版本（低于Windows 10的版本和所用的Python版本不兼容）。

用户所需的磁盘空间取决于语料库的大小。对某些研究问题，用户可以使用相对较小语料库来运行GUI。然而，像完整的《大正藏》这样的大型语料库，TACL的数据库可能相当大（可达170GB)。因此，使用完整数据库的用户可能需要高达200GB的可用磁盘空间。

创建大的数据库时需要较高的计算机硬件配置。内存越大越好，且处理器和磁盘读写的速度越快越有利于计算。如果你的系统配置不高，计算所需时间可能更长。即使计算机的硬件配置足够高，生成一个大型的数据库通常也需要几个小时的时间；如若硬件配置较低，所需时间则可能翻倍。因此，用户可以选择下载已生成的完整的数据库（见下文）。无论以哪种方式获得数据库，在有数据库的情况下，标准的TACL运算过程（如交叉运算和差异运算，见下文）可能需要几分钟或几小时，这取决于所分析的语料库的大小。

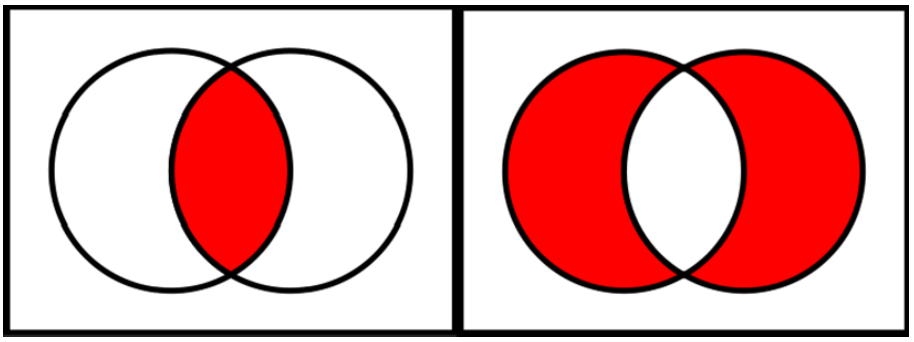
如下所述，我们已生成基于《大正藏》语料库的完整的TACL数据库供用户下载。然而，即使是压缩格式，下载文件也很大(约47GB)。根据互联网连接速度，用户首次下载这个数据库也可能需要较长时间，对某些用户这也可能成为一个技术上的障碍。

如下文所述，我们已将TACL GUI（Windows 版本）所需的所有内容（包括本手册中的所有练习）捆绑，命名为“TACL GUI入门工具包”（TACL GUI Starter Kit）。然而，在我们深入讨论这些细节之前，我们将首先介绍一些基本的背景。

# 背景简介

顾名思义，TACL GUI是为TACL而设计的GUI（Graphic User Interface 图形用户界面，即大多数计算机程序中常见的那种用鼠标点击来操作软件的界面）“TACL”是一个用于计算机辅助分析数字化的汉传佛教典籍文本的工具。“TACL”是一个利用计算机来辅助分析数字化汉语佛教藏经的工具。“TACL”这个名字是一个逆向首字母缩略词（ backronym ），主要是出于对双关语的喜爱，至于它代表什么并不重要。若用户有其他更有趣的解释，可与我们联系。希望得到答案的用户可以将其理解为“文本和语料库的实验室”（Text and Corpus Lab）。

Norrish和Radich于2012年底开始研发TACL。它最初被设想为一个可对汉语佛教文本或语料库之间进行简单且大规模比对的工具。更具体地说，该工具的概念核心是两种比较操作：⑴找到被分析文本/语料库共有的所有材料；或⑵找到一个文本/语料库区别于另一个文本/语料库的所有的不同材料。换言之，TACL的功能是发现文本之间的交叉（intersect）或者差异（difference）。其原理如下图所示：

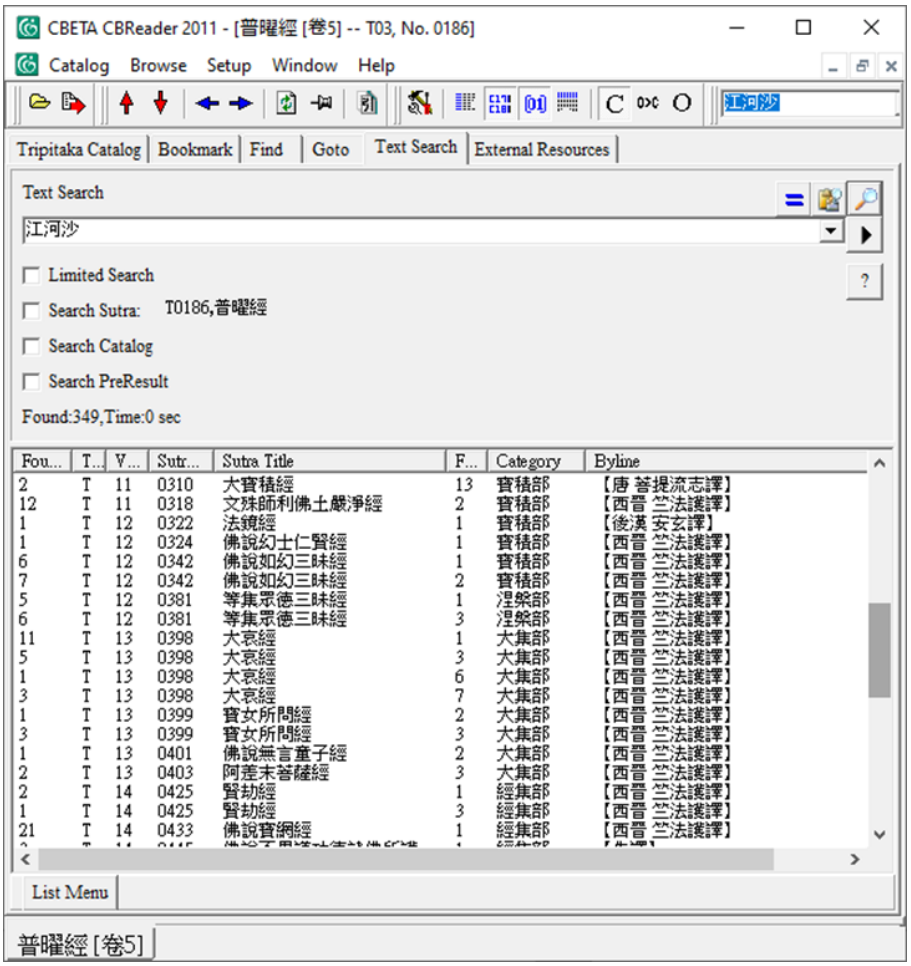


(这真的很简单。)

具体而言，“材料（material）”（共有的或独有的）指n-gram，即“字符串”（一系列连续的字符，即汉字）。此处，n代表字符串的长度。例如，“如來”是一个长度为2的字符串（2-gram）；“摩訶般若波羅蜜”是一个长度为7的字符串（7-gram）。

用户应谨记关于TACL的一个基本事实：它是针对n-gram的分析工具，这对于理解它所得出的结果的意义相当重要。这个事实意味着我们应该注意：TACL只能找到连续的、字面完全相同的、与所选择的测试运算类型所要求的分布模式在输入的文本/语料库中相匹配的字符串。“连续的”（contiguous）意味着TACL不能找到中间由不同的文本或字符隔开的字词组合，无论这种组合对人类多么明显。例如，它不能找到“與...具”或“以...故”这样的组合。“字面完全相同”（literal strings）意味着只有当两个字符串的书写（即编码）完全一致时，TACL才认为它们是相同的。例如，它并不知道“燕坐”在某些情况下与“晏坐”或“燕座”可以看作是相同的。

因此，TACL可以被视为一种强化了的搜索工具。然而，在这两个核心操作，即交叉运算（intersect）和差异运算（difference）中，TACL和CBETA阅读器（CBReader）这类搜索工具有一关键不同之处。使用CBETA阅读器这类搜索工具时，用户输入一个字符串，它就输出该字符串在文本中的分布模式。例如，用户想知道哪里出现了“江河沙”，可以得到如下图所示的结果（显示该字符串集中出现在竺法护（Dharmarakṣa）的作品中）：




相比之下，在TACL中，用户可以自行决定要寻找怎样的分布模式，TACL工具会找到并输出所有符合该条件的字符串。“TACL型问题”就像这些例子：“哪些字符串同时出现于竺法护和鸠摩罗什的作品中?”或者“哪些字符串只出现在竺法护的作品中，而从未在佛典的其他文本中出现?”

系统地发现与这种模式相匹配的字符串有助于我们找到判断各种文本间相互关系的研究问题的证据。最典型的是，交叉运算测试（intersect）可用于研究某一特定文本所依据的早期来源或其后续影响；差异运算测试（difference）则可以用来发现某一语料库的独特的文体特征。TACL GUI以这两种功能的应用为其主要设计目的。TACL应用于实际研究问题的例子可参考本手册后边的文献目录。下面的练习中我们会介绍几个简单的应用实例。

TACL的核心操作交叉运算（intersect）和差异运算（difference）在概念上看似很简单，但它的功能却是很强大的。其强大功能主要来自于以下几个方面：

（1）首先，个别测试可以串联或组合成一系列的操作，这种组合原则上有无限种。例如，我们可以问：哪些字符串只竺法护的作品中有，而在其他任何有可靠的归属的翻译作品中无？（这可以通过一个差异运算实现）。我们可以在此基础上进一步问：“哪些竺法护字符串也出现在归属不详或阙

疑的文本Q中？”（针对这个问题可以运行一个Q与第一个差异运算测试的结果之间的交叉运算测试）。与任何单独的测试相比，这种组合测试可以为我们提供更强大的分析能力来解决研究问题。（测试的串联可通过名为“结果交叉运算”[supplied intersect]的功能实现，详见下文）。

（2）其次，计算机可以相对较快的速度处理大量的文本，而一个人类读者穷其一生也难以分析完体量如此大的文本。一些TACL测试虽然需要几个小时，但在这段时间内可以分析完整个《大正藏》。此外，虽然TACL只能进行字符串的比较测试，但结果总是完全精确。

（3）最后，颇为讽刺的是，恰恰是机器的盲目性赋予了TACL强大的功能，尤其是当其与具有相关知识且有创造性的人类用户的洞察力相结合的时候。如前文所提，用户可以指定一种字符串在文本中的分布模式，而机器则可以发现所有符合该模式的字符串。相比之下，人类在猜测什么样的字符串可能符合有关的分布模式时，更有可能仅限于那些已经符合现有知识的东西，或者是由各种条件所制约的对文本的有限了解。我们使用TACL的经验表明，电脑程序的盲目性反而有助于我们看到原本被忽略的东西。以往对写作风格和作品归属问题的经典研究往往侧重于以佛教概念为证据分析文本中的措辞和表述方式。而TACL让我们更易注意到佛教文本中“非描述性”语料的证据意义——可以类比为发现措辞上的“背景图案”——而人类在无任何辅助的情况下几乎不可能注意到这些材料。《中阿含》T26和《增一阿含》T125中有个让人大开眼界的例子，“村邑”和“村落”这两个词的分布完全互补。（见Radich and Anālayo 2017：229）

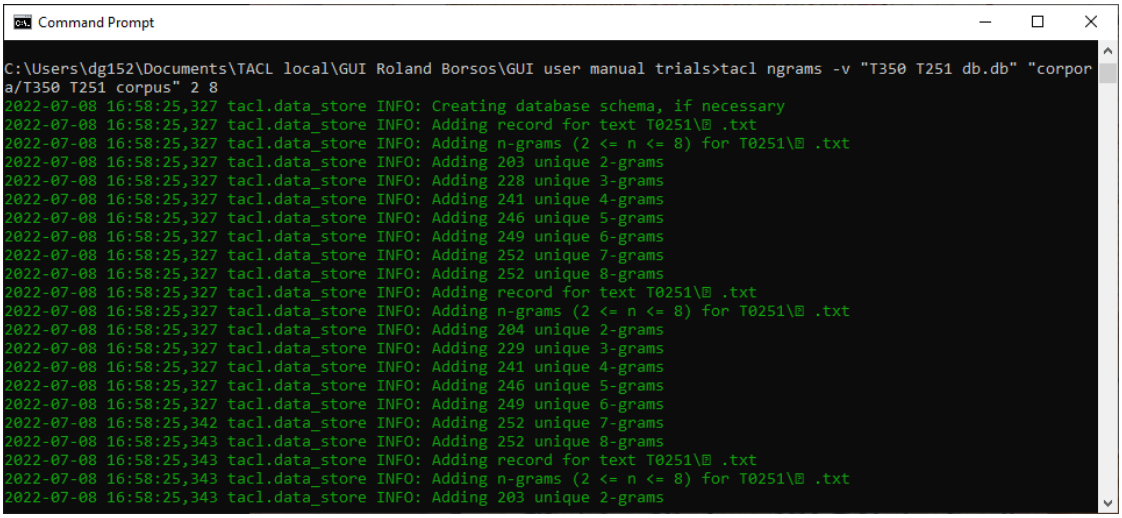
TACL的这两个特点结合起来使它就像显微镜和望远镜的强大组合，前者能发现肉眼看不见的细微差别，后者能让我们看到比普通人的视力所及更远的地方。

同时，应谨记TACL的力量只来自于机器和训练有素、知识渊博、认真思考的人类用户的结合。也就是说，TACL方法可以看作是“半机械人”（cyborg）方法——人类和机器的合作。机器本身不能做任何研究，也不能找到答案。它只能为我们的研究问题找到潜在的证据，然后必须由一个有语文学知识的人类头脑来对其加以权衡判断。在这个意义上，我们可以把这个工具比作一个金属探测器，它可以帮助考古学家在面对一个大遗址时决定先在哪里挖掘。这个工具提示我们，有趣的证据可能在某些地方，但不是所有我们挖掘出来的东西都是有用的（我们也可能得到一些垃圾）。同样，也不是所有有用的东西都会被工具发现（可能有金属以外的材料的证据，我们需要其他工具来发现）。在金属探测器找到每件可能的证据后，人类研究者还要对其进行分析，以确定其证据意义。但是使用金属探测器仍然比费力地用手挖掘整个遗址要省力得多。

关于TACL测试对特定研究问题的概念设计，以及使用的佛教学-语文学方法的更多信息，用户可参考

《TACL方法指南》（TACL Methods Guide，https://dazangthings.nz/tacl-methods-guide/）。

TACL最初用Python编写，在命令行环境下使用。整套TACL工具会一直以这种形式在Github（https://github.com/ajenhl/tacl）上免费提供。你可以通过TACL文档（https://pythonhosted.org/tacl/）来学习如何使用命令行版本的TACL。在那个世界里，TACL的操作界面看起来如下图：




然而，经验表明，只有基础计算机知识和技能水平的普通用户（比如有TACL之前的我们）使用TACL的命令行版本会比较困难。因此我们设计了TACL的图形用户界面（GUI）使之更易使用。

为了方便用户使用，TACL GUI图形用户界面不仅允许用户以更接近于普通计算机使用模式的方式与TACL对接，而且还精心选择了一个TACL的功能子集。TACL的很多功能是不能通过GUI使用，其他的一些功能则被纳入GUI简化的工作流程中，并作为默认选项无需用户干预自动实现。如上文所提及，TACL是为两种类型的研究问题而设计——按我们估计，可能是TACL在实际研究中最常见和最基本的两种用途：（1）在其他文本中寻找给定文本的来源，尤其是可以证明文本是在中国编撰而不是翻译的来源（参见Radich 2014）；（2）寻找特定作者、译者或翻译团队的独特文体特征，并将之应用于著译者的判定和年代推定（例如，Radich and Anālayo 2017，Radich 2017a）。最终，对图形用户界面的有限功能感到不满的用户或许会愿意花些时间来学习“升级”到使用命令行版本。

在讨论使用TACL GUI的具体细节之前，有必要简要概述TACL的工作方式（包括其GUI版本）和使用TACL时典型的工作流程。在这一节中，我们将对该工具的工作方式做一个初步的概括。我们将在下文全面详细地介绍工作流程中的每一个步骤。每一步骤都配有实践练习，以训练用户自己动手操作每个步骤。TACL GUI的核心工作流程可简单分为八个步骤。

1）选择或构建一个语料库（corpus）

2）生成或导入一个数据库（database）

3）编辑或选择一个目录文件（catalogue）

4）运行核心测试——交叉运算（intersect）或差异运算（difference）

5）（可选）将两个或更多的测试串联——结果交叉运算（supplied intersect）

6）对原始结果进行后处理——筛选（filter）/合理化（rationalize）

7）将筛选后的结果导入Excel等工具中，并进行排序

8）在上下文中检查实际文本中的n-gram

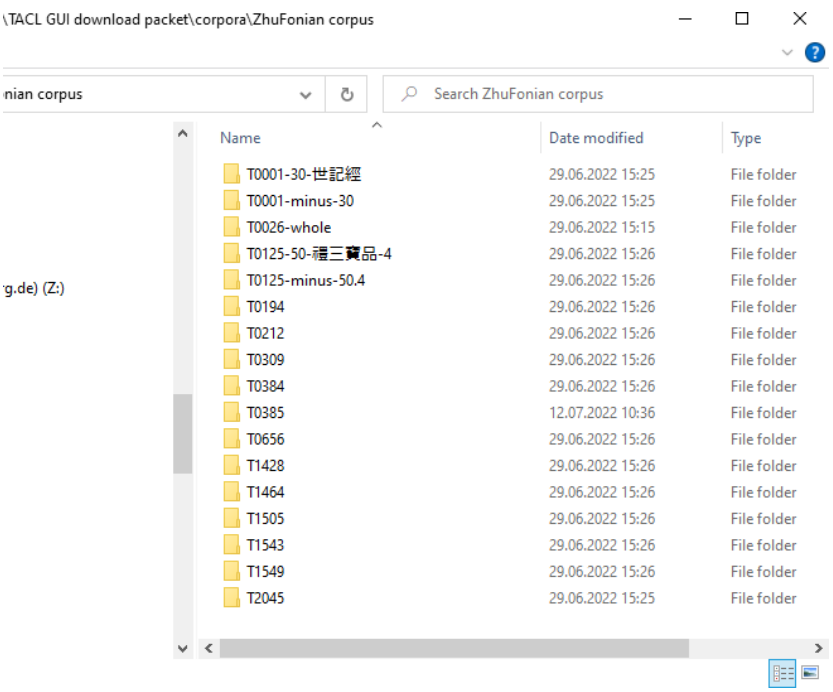
1）语料库（corpus）：TACL使用的是CBETA（https://cbeta.org）数字化的《大正藏》（以及其他版本的藏经，如《续藏经》等）的文本，主要是（在撰写本文时）2019年的版本，可在CBETA的github（https://github.com/cbeta-org/xml-p5-2018）上下载。TACL最终需要纯文本格式的文本。为此，TACL需要两个步骤来预处理XML文件：第一个步骤是创建一个过渡的文本形式，从而使程序更容易处理这些文本；第二个步骤是清除文件中的XML标签，生成纯文本。然而，TACL GUI用户无需为预处理步骤费心。为了降低TACL GUI的使用难度，我们已经通过这些步骤预先处理了两个完整的语料库供用户下载（见下文）。

用户需注意，在这一步，TACL使用《大正藏》的XML版本中的校勘注释来重建所有的《大正藏》所参考的底本，也包括CBETA编辑以及电子化过程中所参考的、《大正藏》中没有用到的底本。也就是说，对于《大正藏》中的某个文本，如《大智度論》T1509，TACL语料库收录的不只是该文本的某个版本，而是包括了《大正藏》所录的各个版本，每一版本都以一个独立的文本文件存储，以“大”（代表《大正藏》）“宋”“元”“明”“宫”“石”等命名。这意味着，TACL实际上是在处理所有这些不同的底本中所包含的信息——这一情形可以形象地表示为“搜索《大正藏》和CBETA的校勘注释”。（用户应注意，这整个过程的可靠性取决于《大正藏》和CBETA的编辑校勘工作的可靠性，因为TACL无法直接接触到这些底本的原始刊本或写本。因此，《大正藏》或CBETA的校勘错误将导致TACL的结果错误）。

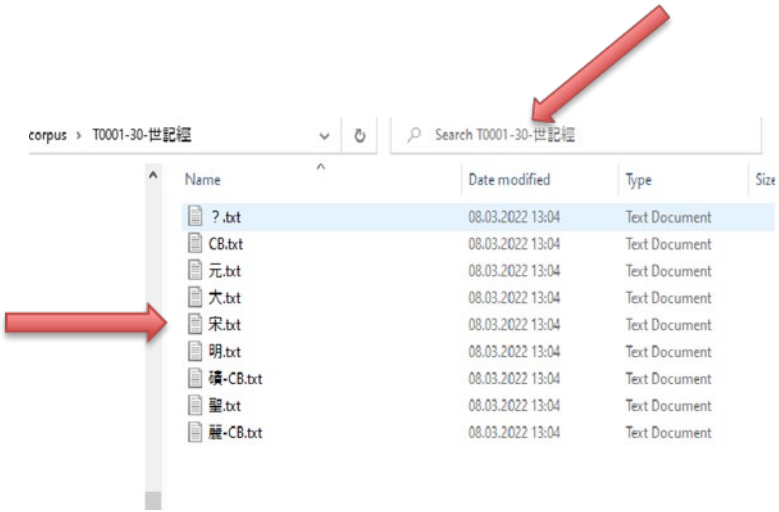
一旦数字化的文件被转换为TACL所要求的格式，用户就可以选择或构建一个特定的语料库（corpus）用以分析。语料库有可能是包含整个《大正藏》的大型语料库，但计算机难以处理这样的语料库（见下文）。也可以使用针对特定研究问题的小型语料库。我们在下面的“语料库”部分会介绍如何创建这种定制语料库（第20页）。

你拿到的“TACL GUI入门套件”（TACL GUI Starter Kit）中可以找到这样一个较小的语料库的例子，我们可以用它来快速展示一个典型语料库的样子。这个语料库包含竺佛念的作品，以及其他几个与

之相关的作品，我们可以用来做下面的练习。见下载的套件中的文件夹“corpora/ZhuFonian corpus”。当我们打开该语料库时，可以看到该语料库包含多个子文件夹，通常语料库中的每部作品都有一个子文件夹。《長阿含經》T1，《增一阿含經》T125，《出曜經》T212，等等。有些作品实际上有两个子文件夹，因为我们在这里使用的是经何书群（Radich）处理过的《大正藏》语料库中的文本，这些文本被分割成若干部分，以便对这些部分进行单独研究——详情可点击这个链接（https://dazangthings.nz/tacl-gui-one-stop-shop/Radich-taish%C5%8D-corpus/）。

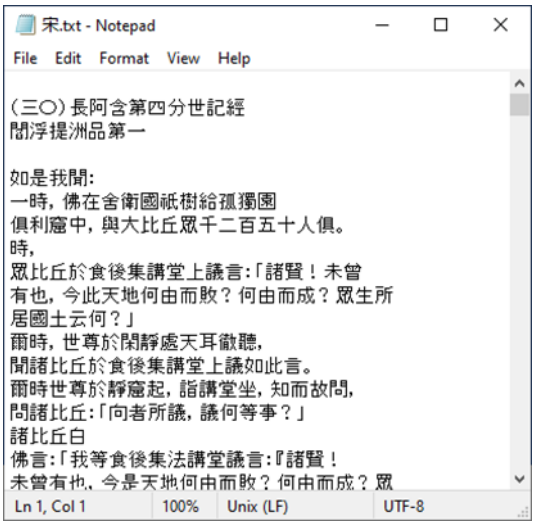



当我们查看这些子文件夹中的一个作品时——比如以第一个文件夹“T0001-30-世記經”（《長阿含經》中的第30号经《世記經》）为例——我们看到它包含多个文本文件，每个文本文件就是一个底本（witness）：




例如，名为“宋.txt”的文件是T1(30)《大正藏》中称作“宋”本的底本文本，是根据《大正藏》的校勘注释（由CBETA数字化）重建的。如果我们在文本编辑器中打开这

样的文件，我们会看到纯文本的内容，这是上述TACL预处理步骤的最终结果。进一步的TACL分析将在这些文本的基础上进行。




2）数据库（database）：以上就是程序所处理的语料库的样子。对于任何用户定义的语料库，我们需要使用TACL生成一个数据库（database），其中包含该语料库中的所有n-gram，以及该语料库中每个作品的每个底本的n-gram的计数。

在为任何给定的语料库建立数据库时，用户须定义自己感兴趣的n-gram的长度范围。我们通常建议2-8个字符的范围，这是基于我们试图在两个关注点之间取得平衡的经验：一方面，我们想抓住大多数会重复出现大量次数的字符串，从而确保我们感兴趣的大部分内容会出现在数据库中；另一方面，我们想把数据库保持在一个合理的大小。如果数据库包括非常长的字符串（例如长达100-gram），那么它所记录的绝大部分内容都是只出现一次的长字符串，对于多数研究目的来说，其证据价值相对较小；但是这样数据库会膨胀到巨大的、无法使用的规模，而且还要花费非常长的时间来生成。

再强调一次：数据库包含了语料库中所有的n-gram，以及语料库中每个作品的每个底本中的n-gram的计数。这意味着，数据库中的典型的一行将包含这样的信息（这里以简化的示意形式呈现）：

佛言,T0001-30-世記經,宋,5

意味着“佛言”在T1(30)的“宋”本中出现了5次。

请注意，这就是数据库存储的全部信息。尤其需要注意的是，通过数据库我们可以知道一个特定的n-gram出现在一个特定作品的特定底本中，以及出现了多少次，但不能知道这个n-gram出现在文本的什么地方（数据库没有存储关于文本内部结构的任何信息）。

关于字符串证据在上下文中的位置或作用的问题需要人工来处理，我们将在下面看到。

所有随后的TACL测试——最典型的是交叉运算（intersect）和差异运算（difference）测试——事实上就是在这个数据库的基础上工作的，而不是直接检验语料库本身。

3）目录文件（catalogue）：测试前的准备工作的最后一步是准备一个目录（catalogue）文件。当我们有了一个特定的语料库和相应的数据库时，我们不一定希望每一个测试对语料库中的每一个文本都进行操作；但另一方面，为每一个测试建立一个新的数据库又是没有效率的；针对这个问题，使用目录文件可以让我们指定整个语料库中的一个文本子集作为一个给定文本的比对目标。我们还需要告诉计算机我们要如何对语料库中的文本进行分组：任何交叉运算或差异运算测试都是在两个（或更多）对象之间进行比较，但如果没有目录文件中给出的分类信息，任何文本所属于的唯一类别就是语料库的整体。TACL目录是一个文本文件，我们用它来（选择性地）选择语料库中所有文本的一个子集，并将被比较的文本分组。基本上，它是一个文本的列表，标明了每个文本所属的文本组（见下文）。

4）核心功能：现在我们有了一个包含我们想要分析的文本的语料库，一个包含每个文本的每个底本的n-gram计数信息的数据库，以及一个确定我们想要比较的文本组的目录。然后我们运行TACL的核心操作差异运算（difference）或交叉运算（intersect）。通过这两种运算，TACL可以找到：同时出现于比较双方的文本中的n-gram及其计数（交叉运算intersect）；或者只出现于比较双方中的一方的文本中所有的n-gram及其计数（差异运算difference）。此类测试的原始结果以.csv（“comma separated values”逗号分隔值）格式输出；关于这种存储格式见下文的讨论。

5）串联组合操作（可选）：本文前面提到，我们可以选择使用“结果交叉运算”（supplied intersect）将测试操作“串联”成一个序列。例如，这使我们能够将交叉运算（intersect）和差异运算（difference）测试的结果结合起来，找到同时出现于文本组A和文本组B中，但在文本组C中则没有的n-gram。

6）对原始结果的后处理：在通常的工作流程中，下一步是对这些测试产生的原始结果进行后处理，以便对其进行整理，有时通过筛选处理只保留我们最感兴趣的那一部分结果。用TACL GUI中使用的术语，我们把这个步骤称为“筛选/合理化”

（Filter/Rationalize）。这个后处理步骤的目的是消除结果中的冗余，使人类分析员更容易处理原始结果，并能够有效地找到与研究问题相关的证据。

我们可以通过一个简单的例子来说明这种后处理辅助人类分析的一些优势。正如我们刚刚看到的，每个数据库都有一个用户定义的最大n-gram大小（字符串长度）——通常是8个字符。但是，难免有一些情况下，我们最终会对长于8个字符的字符串感兴趣。例如，下面的67-gram只由两个文本共享——T150A和T735——并构成一个证据模式的一部分，表明这两个文本有密切的特殊关系（见Nattier 2008: 51, 131；下文中我们还会讨论到这个例子，展示如何自己生成这个结果）。

> ...盡為生死盡識何等為生死欲盡受行識為是八行識諦見至諦定為八如是為生死欲滅受行識何等為生死味識所為生死因緣生樂喜意如是為生死味識何等為生死...
> T150A (II) 876b13-17; T735 (XVII) 537c2-6.

如果我们坚持使用最大字符串长度为8的结果，那么T150A和T735之间的匹配将在结果文件中由60行表示，显示在这个单一的67-gram中发现的60个不同的8-gram：

> 盡為生死盡識何等
> 為生死盡識何等為
> 生死盡識何等為生
> 死盡識何等為生死
> 盡識何等為生死欲
> ... (等等)

对于人类分析员来说，最终的分析结果应该只是一个单一的信息条目，但这里TACL提供的信息则是零散和重复的。

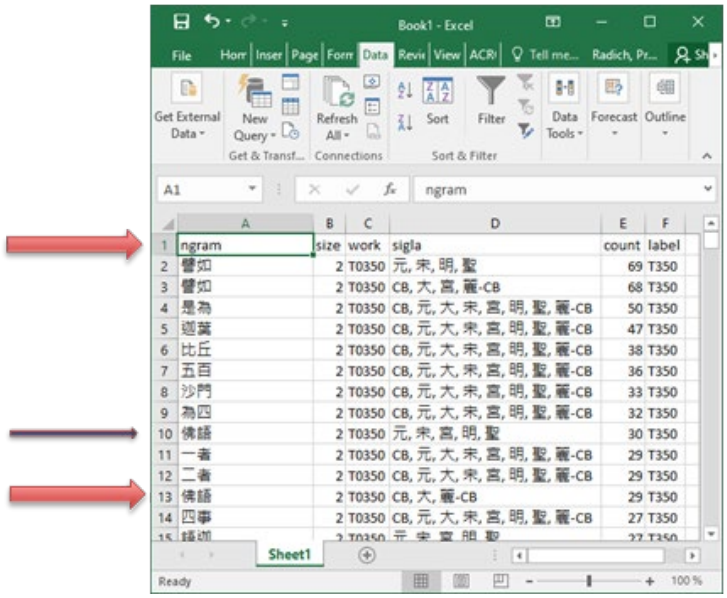
幸运的是，TACL事实上可以给出包含这种较长字符串的结果，尽管它们没有直接以完整的形式包含在长度为2-8-grams的数据库中。这项工作由一个我们称之为“扩展”（extend）的功能实现。在TACL GUI中，“扩展”（extend）功能默认包含在TACL交叉运算测试的所有结果的“筛选/合理化”（Filter/Rationalize）后处理中。这就意味着人类分析员在后处理结果中会自动得到类似上述67-gram的长字符串。

除了寻找这种我们感兴趣的较长的、最初被数据库的参数排除在原始结果之外的字符串，TACL GUI的“筛选/合理化”后处理还允许我们根据字符串的长度、出现次数或包含此

字符串作品（文本）的数量等因素来筛选结果。这些工具使用户能够对数量过于庞大的结果进行筛选，以便更有效地锁定可能对所研究的问题具有证据价值的字符串。（深入讨论请参见《TACL方法指南》TACL Methods Guide）。

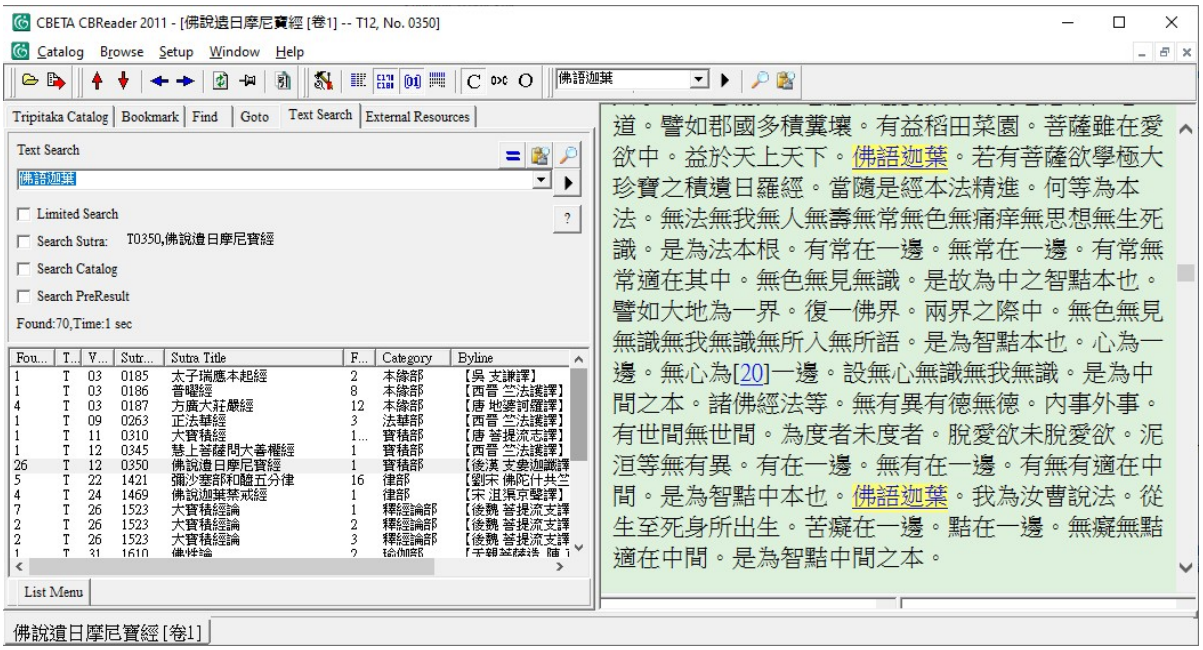
小结：简而言之，一个典型的TACL测试包括以下步骤：（1)用户构建或选择一个语料库（corpus），（2)在此基础上，生成一个数据库（database）。接下来，（3）用户写一个“目录”（catalogue）文件，定义两组或多组要比较的文本（或子语料库）。（4）然后，用户通常在这些文本/语料库上进行差异运算difference或或交叉运算intersect进行比较。（5）用户可以选择运行进一步的测试，将两个或多个先前测试的原始结果串联起来。最后（6）用户对测试的原始结果进行后处理（“筛选/合理化”Filter/Rationalizes），以找到感兴趣的较长字符串（如果与研究的问题相关），或通过字符串计数、在文本中的分布等条件对结果进行筛选。此时，机器的工作已经基本完成，现在轮到人-机结合的“机械人”（cyborg）中的人类伙伴来接管，开始对原始结果进行定性的语文学分析。

7）在Excel（或其他类似的软件）中检视和分析：所有基本的TACL操作——主要是指差异运算（difference）、交叉运算（intersect）和后处理（“筛选/合理化”Filter/Rationalizing）——以一种叫做.csv（“逗号分隔值”）的文件格式输出原始结果。然后，我们通常将原始结果导入像Excel这样工具，在其中，我们还可以根据适合每个测试类型的规则对结果进行排序（我们会在下面详细介绍）。从某种意义上说，人类研究者的工作在这一点上才真正开始。人类研究者需要仔细检视这些结果，评估哪些是有证据价值的，就像考古学家评估用金属探测器挖出的金属物件一样。将结果导入Excel并排序后得到的是这样的视图：




最上面一行（第1行）是列标签，告诉用户每一列的内容是什么。A列是n-gram——以第13行为例，“佛語”。在B列中是n-gram的大小（字符串长度）。在C列中是包含结果字符串的作品（work）；在D列中，“底本标记”（sigla），显示作品的底本，用《大正藏》/CBETA校勘注释中的缩写表示。在E列中是有关作品（C列）的所有底本（D栏）中的n-gram的计数。F列是在TACL目录（catalogue）文件中使用的“标签”（label），以将文本归入一个此标签所标记的文本组进行比较。例如，第13行显示，“佛語”，它是一个2-gram，在T0350（支娄迦谶的《佛說遺日摩尼寶經》）的“CB”、“大”以及“Liang-CB”这几个底本中出现了29次。注意在第10行，由较小的箭头标记，我们看到相同的n-gram在同一文本的其他几个底本中出现了30次）。

8）在上下文中检查结果：下一步是人工对原始结果进行定性的语文学分析，通常需要从原始结果中提取看起来很有趣的n-gram，然后在其所出现的文本的上下文中对其进行检查，以弄清它们对所研究的问题可能意味着什么。请再次记住，TACL并不“知道”n-gram在文本中出现的位置，也不知道它们意味着什么。对TACL发现的n-gram的这些方面进行分析是人类的工作。我们通常通过使用CBReader这样的工具来完成这项工作，例如：




然而，这种人工分析的工作超出了TACL GUI图形用户界面本身的操作，因此，我们在《TACL方法指南》（TACL Methods Guide，见下文）而不是在本用户手册中更详细地讨论了这个问题。

总之，在使用TACL GUI时，你的工作流程的典型步骤是这样的：

1）选择或构建一个语料库（corpus）（第20页）

2）生成（generate）或导入（import）一个数据库（database）（第23页）

3）编辑或选择一个目录文件（catalogue）（第34页）

4）运行核心测试——交叉运算（intersect）（第40页）或差异运算（difference）（第48页）

5）（可选）将两个或更多的测试串联——结果交叉运算（supplied intersect）（第57页）

6）对原始结果进行后处理——筛选（filter）/合理化（rationalize）（第64页）

7）将筛选后的结果导入Excel等工具中，并进行排序（第73页）

8）在实际文本的上下文中检查n-gram（第82页）

下面，我们将按这个顺序详细介绍TACL GUI使用中的每一个步骤，并附有实例练习。通过这些练习，你可以学会自己操作。但在此之前，还要做几个初步的准备工作。

## 与TACL相关的工作方法

为了将TACL用于中国佛教文本的研究，用户须掌握两种技能：(1)能够通过GUI或命令行

来操作TACL；(2)清晰地掌握严格分析TACL所发现的数据的概念设计，以及从结果中筛选并应用证据检验假设和构建论点的工作流程。

本用户手册只介绍（1）如何操作和使用TACL GUI和与TACL有关的研究方法（2）请参考《TACL方法指南》（TACL Methods Guide）（https://dazangthings.nz/tacl-methods-guide/）。阅读使用TACL的已发表研究成果或也有助益（见本手册末尾的参考文献）。“TACL入门套件”[TACL Starter Kit]囊括所有这些参考文献的PDF文件。）

如果用户使用TACL时对研究项目的设计有疑问，欢迎通过电子邮件联系我们。（michael.Radich@hcts.uni-heidelberg.de）

## TACL GUI图形用户界面

如上所述，TACL GUI是TACL的简化版本（它只具有命令行版本全部功能的一个子集）。开发GUI是为了让TACL处理一些常见问题时更易上手使用。

用户若在使用TACL GUI时遇到技术问题，或者发现错误，可以在TACL GUI的github（https://github.com/rolait/tacl-gui/）上报告，或者通过电子邮件（michael.Radich@hcts.uni-heidelberg.de）与我们联系。

## GUI 下载

TACL GUI可以从TACL GUI GitHub（https://github.com/rolait/tacl-gui/）下载。此文件的格式为可执行文件，无须安装，只须把TACL-GUI.exe放到你打算进行TACL工作的文件夹中。（如果要学习使用本用户手册中的练习，你可以暂时把它留在解压缩后的“入门套件”TACL Starter Kit文件夹中。）双击就可以启动该程序。然而，在你第一次启动之前，请先阅读以下说明。

《TACL GUI入门套件》（TACL GUI starter kit）中包含了TACL GUI的可执行程序，可在此下载（https://dazangthings.nz/tacl-gui-one-stop-shop/tacl-gui-starter-kit/）。该《入门套件》还包括我们在本《用户手册》中介绍的几个实例的材料，你可以在阅读的

同时用它们进行实际操作，以练习使用GUI。注意，入门套件中的GUI只适用于Windows。Mac和Linux用户需要直接从TACL GUI的GitHub下载相应平台的GUI。手册中的所有截图和说明都基于Windows版本。除了从操作系统继承的不同默认窗口布局外，不同操作系统上的版本之间应该没有大的区别。

有了这些前期准备，我们现在回到我们在上面概要介绍过的八步工作流程。在本手册的其余部分中，我们将详细介绍这个工作流程的各个步骤，其顺序与上面的概要描述相同。工作流程中的步骤和GUI的操作还会通过实例练习来说明。这些练习将使用以下真实的例子，尽管是以某种简化的形式（这些例子在下面有更详细的描述；在这里按出现的顺序列出）：

例1：竺佛念语料库，重点关注《增一阿含經》T125的译者问题。

例2：《七處三觀經》T150A和《四願經》T735共有的文本内容表明二者之间的密切关系。

例3：《增一阿含經》48.3和《彌勒下生經》T453共有的文本内容表明二者之间的密切关系。

例4：从安玄、严佛调和竺法护翻译的《郁伽長者問經》中的不同用词区分各自的翻译风格。

# 工作步骤一：语料库（Corpus）

如上所述，TACL需要一个语料库才能工作。TACL使用的是CBETA数字化的佛教藏经文本（https://github.com/cbeta-org/xml-p5-2018）。然而，如上文所提，TACL只能处理经过几个步骤预处理后的CBETA的XML文件。为了简化GUI用户的操作，我们把这些准备工作从GUI的功能中剔除，而将已准备好的CBETA语料库供用户下载（https://zenodo.org/record/6795219）。

用户可选择两个大型语料库的任何一个（通过点击各自的链接）：(a)“T-X语料库”是《大正藏》和《续藏经》合集，其内容结构与《大正藏》/CBETA编辑后的电子版完全一致（https://zenodo.org/record/6798320 DOI: 10.5281/zenodo.6798320）。(b)修改后的“Radich《大正藏》语料库”（https://zenodo.org/record/6798305 DOI:

10.5281/zenodo.6798305），其中一些文本或文集（collection）被Radich分割成较小的组成部分，以便更准确地反映文本/文集结构的某些方面或其历史。我们在此称第二个语料库为“Radich语料库”，文件（https://dazangthings.nz/documents/49/Radich_Taisho_corpus_description.pdf）对此有说明，并给出了为何有些地方它与《大正藏》/CBETA不同的理由。

用户有了“T-X语料库”或“Radich语料库”后，至于如何使用其中之一作为他们自己工作的基础，特别是选择或生成TACL数据库（见下文），他们还有两个选项：

(1)用户可以直接用其中之一工作。

(2)然而，许多情况下用户考察的目标范围有限，只想使用《大正藏》（或其他CBETA语料库）的子集。在这种情况下，用户事先创建自己的小型语料库的会让后续的研究效率高得多。

这类较小的定制语料库应通过将“T-X语料库”或“Radich语料库”中的文件移到为新的子语料库创建的独立文件夹中来创建。然后用户可以在定制语料库的基础上生成一个定制的数据库（见下文）。《入门套件》中的“语料库/ZhuFonian语料库”（corpora/ZhuFonian corpus）子文件夹就是这样一个为特定研究题目定制的语料库。

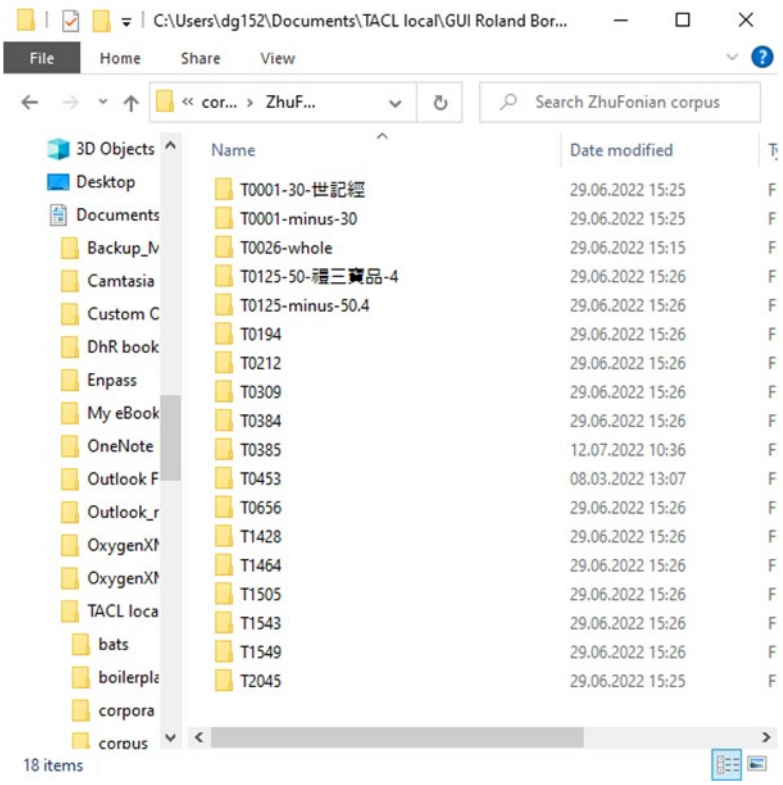
二择一时用户应考虑两个主要因素。首先，用户须考虑自己的研究目标，以及实现这些目标所需的语料库（和数据库）。其次，用户还须考虑导入（import）或生成（generate）一个大型数据库时会遇到的实际问题，有时困难可能相当大（见下文）。

无论怎样，TACL GUI至少需要一个语料库才能运行。因此，当用户第一次启动GUI时，应有已准备好的语料库，可以是自己筛选和制作而成语料库，或是下载而来的“标准”TACL语料库之一。

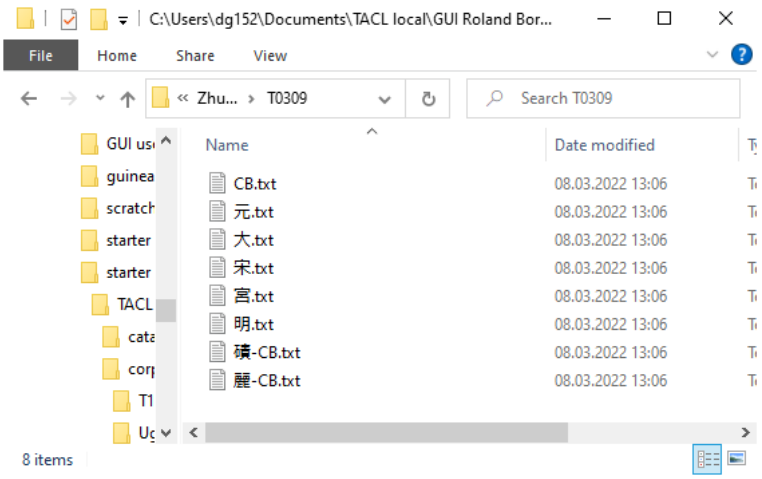
需要注意的是，TACL和GUI都要求语料库精确的结构，上文已有简述。这一信息和想添加源于CBETA以外的文本或语料的用户尤为相关。

语料库作为一个整体必须是一个单一的文件夹。语料库文件夹包含子文件夹，每一个子文件夹内含语料库中的一个作品（work）。每个文件夹的文件名必须与TACL目录文件

（catalogue）中的作品的标识符（identifier）完全一致。每个作品的子文件夹包含一个或多个文本文件（扩展名为.txt），每个文件代表该作品的一个底本（witness）。这些语料库的组成部分和结构的实例可参见“入门套件”（Starter Kit）中的“语料库”（corpora）文件夹。例如，“竺佛念语料库”（ZhuFonian corpus）文件夹看起来如下图：




每个子文件夹代表一部作品。例如，“T0309”是《十住斷結經》T309的子文件夹，而“T0656”是《菩薩瓔珞經》T656的子文件夹。子文件夹“T0309”看起来如下图：



在这里，“元.txt”是《大正藏》编者在校勘注释中称为“元”的T309底本的文本文件，“大.txt”则是以《大正藏》作为底本的文本文件，其他文件的内容以此类推。(可供下载的大型语料库也是这样的结构，用户可以通过直接查看文件来确认。)

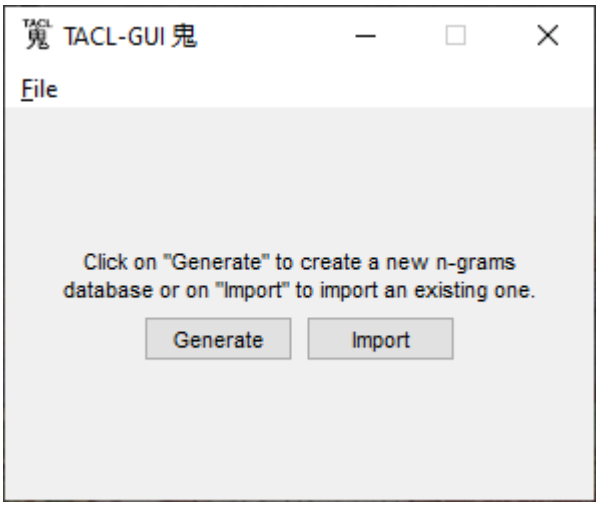
用户还须注意，每个作品的名称（也是其子文件夹名称）和底本的.txt文件的文件名不应包含空格或特殊字符。使用空格和特殊字符会在的接下来的步骤中造成麻烦。

TACL GUI只能处理以这种结构存储文本文件的子文件夹中的语料。如果用户想手动添加CBETA数据库中没有的作品到语料库中，请谨记这点。用户必须为每个作品创建一个子文件夹，并在该子文件夹中为每个底本（即使你只有一个底本）创建一个文本文件。

原则上，除了CBETA数字化的文件外，TACL也可以处理其他来源的语料库（例如，用户可能有意处理《道藏》或《四库全书》中的文本，或者把只有手稿的文本数字化）。然而，往TACL语料库添加任何材料都必须以上文所述的结构格式组织内容。另一个条件是，必须以TACL可辨识的形式将文本进行数字化和标记化处理。请有意扩展语料库的用户与我们联系。

## 工作步骤二：数据库（Database）

如概要中所说，TACL需要一个有字符串（n-gram）和其计数的数据库来进行操作。这是其进行分析的真正基础。因此，TACL（和GUI）需要一个包含你所要研究的语料库中所有n-gram的数据库。当用户第一次启动TACL GUI时，首先映入眼帘的是一个要求生成（Generate）或导入（Import）数据库的对话框：




用户可以点击“生成”（Generate）在GUI中生成自己的数据库，或者点击“导入”（Import）从其他来源导入一个数据库。

我们之所以给用户这样的选项是因为对于大型语料库而言，如“T-X语料库”或“Radich语料库”，完整的数据库非常庞大。例如，作为一个2-8gram的数据库，“Radich语料库”大约是170GB。这意味着无论选择“生成”还是“导入”，建立一个在GUI种使用的数据库都非常耗时。

GUI在一台配置相对较高的计算机上（64GB内存，i7处理器，SSD）生成“Radich语料库”的完整数据库耗时约6个小时。

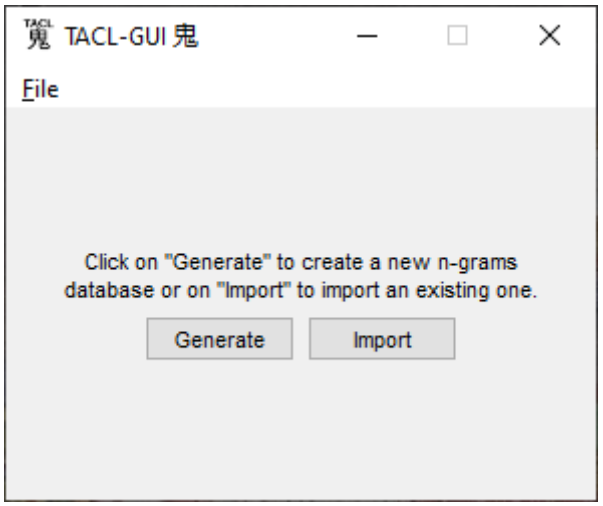
用户的另一选择是，从Zenodo下载现成的完整“Radich《大正藏》语料库”数据库（https://zenodo.org/record/6795219 DOI：10.5281/zenodo.6795219）。然而，压缩后的下载文件也要大约47GB，因此，若用户的互联网连接速度较慢，下载耗时可能相当长。文件下载后，使用前还必须先解压文件，即使是一台高配置的计算机，这一步这也需要相当长的时间。但此后将其导入GUI则无需多长时间，对电脑的要求也不高。出于这些原因，对于计算机配置有限的用户而言，下载这个完整的数据库比在本地从头开始生成或是更好的选择。下文有对下载更全面的描述。

对于想要使用完整语料库和数据库的用户来说，这些准备工作意味着需要投入大量的时间

进行设置。然而，一般而言，这种设置只需操作一次，之后使用TACL GUI会更简单，可谓一劳永逸。因此，想要使用完整语料库和数据库的用户应该考虑计算机的配置、可用的存储空间和互联网连接速度等因素，从而选择是在本地生成数据库，还是下载并导入数据库。我们将在下文详细描述这两个操作过程。

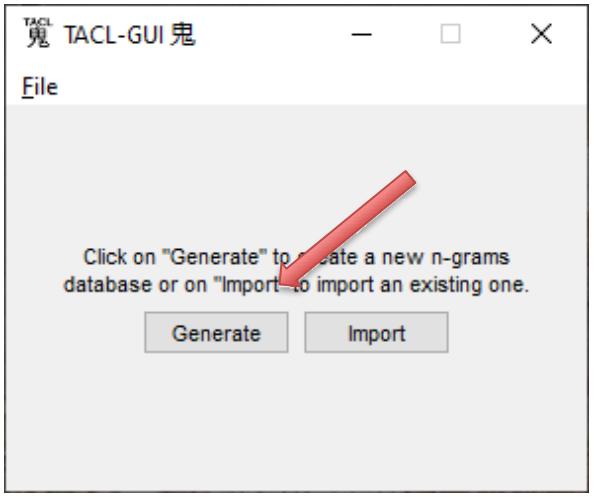
## 在GUI中创建或导入数据库

首次启动TACL GUI时，用户首先会看到一个要求生成或导入数据库的对话框：

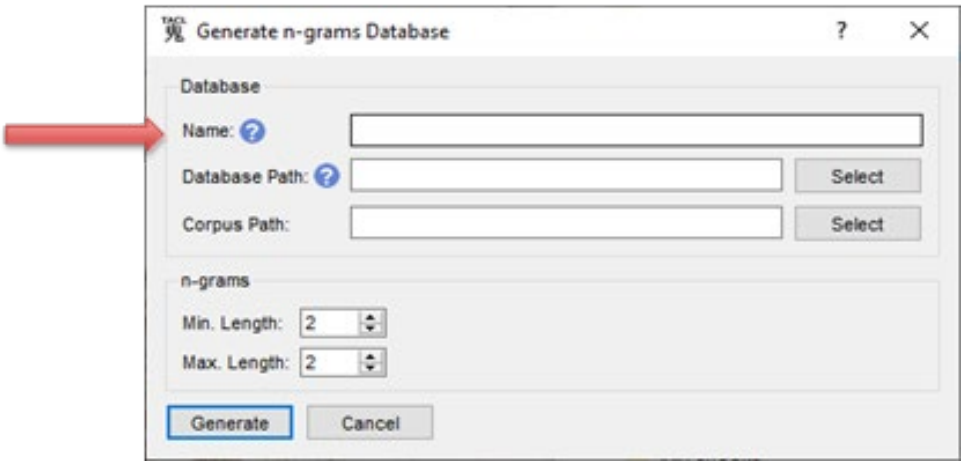



（1）生成（Generate）一个数据库

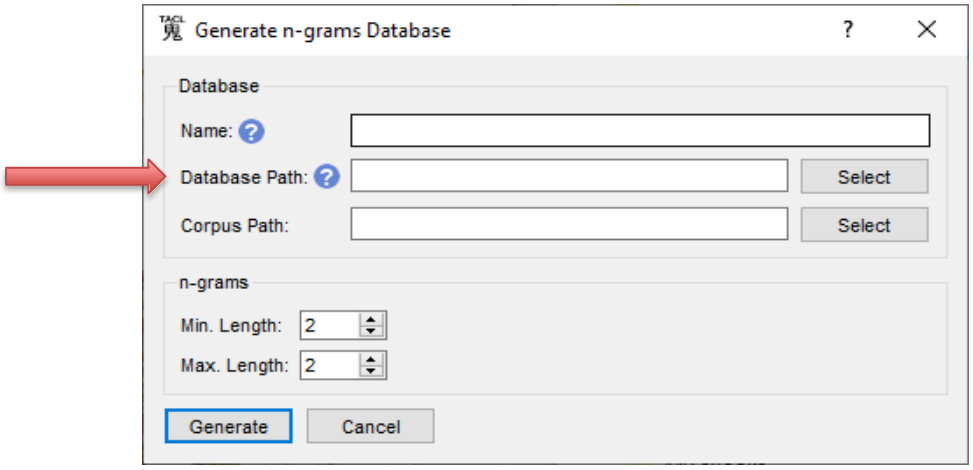
如果用户使用的是自己挑选和制作的语料库，现在需要在该语料库的基础上生成一个数据库。在最初的这个界面中：



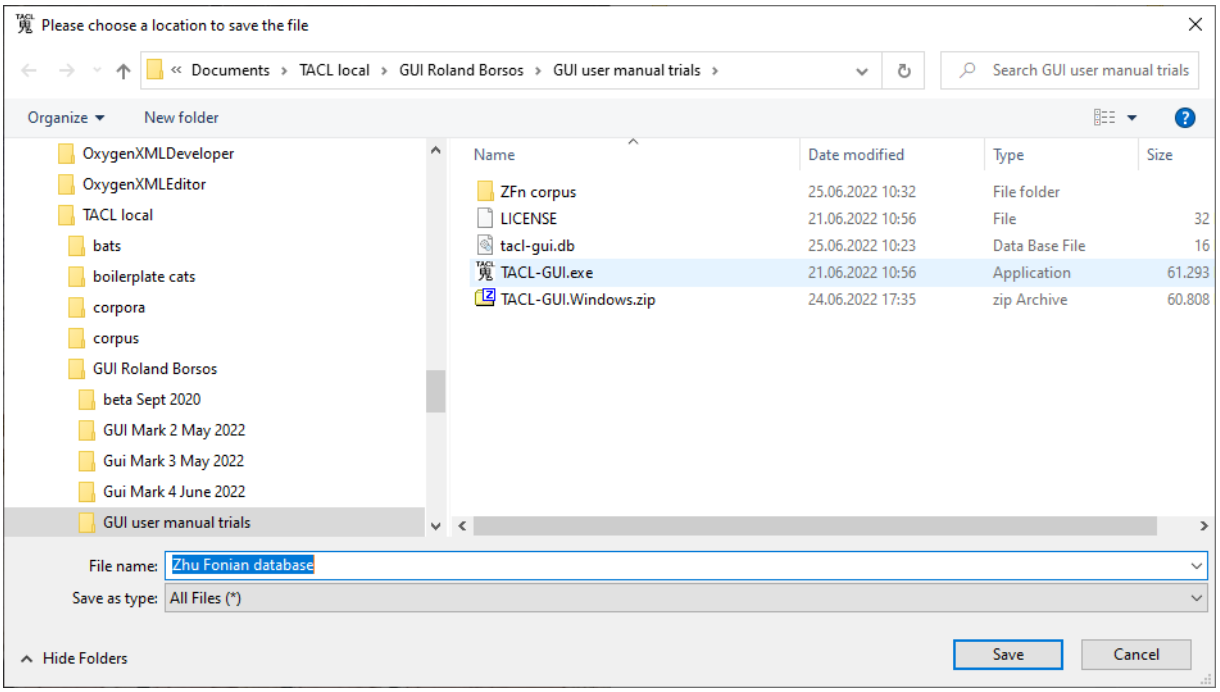

点击“生成”（Generate）。你会看到这个对话框：



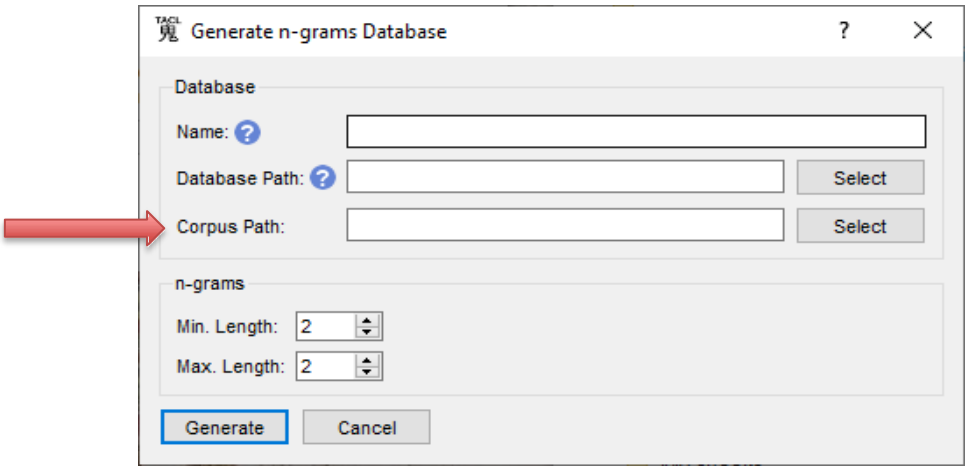

在“名称”（name）栏中，为你的数据库指定一个名称。在这里，我们将以竺佛念的文本的一些小规模的测试为例（前面提到的“例1”）。为此，你的数据库可以命名为“ZhuFonian database”。



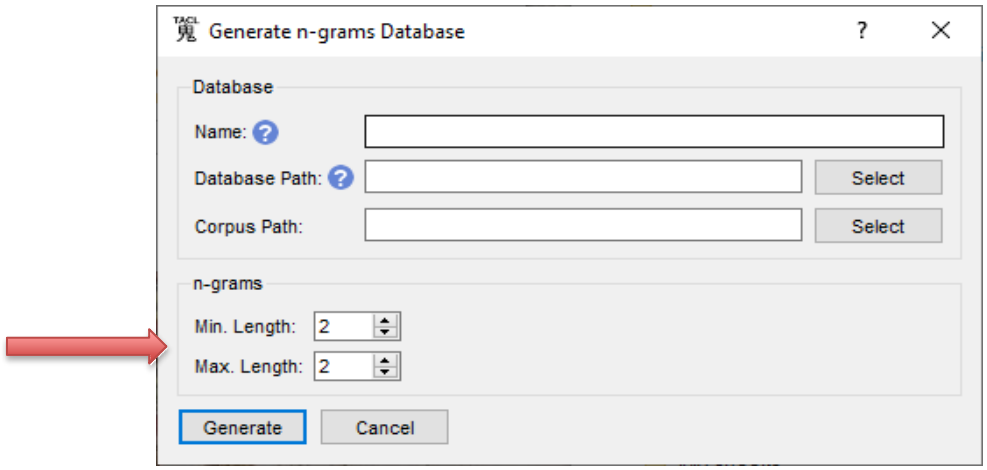

在“数据库路径”（Database Path）字段中，输入一个指向你要进行TACL分析的文件夹的路径。最简单的方法是点击“选择”（Select）按钮并导航到相关的文件夹。你也可以直接用键盘输入一个路径。注意，“数据库路径”必须在末尾包括一个文件名。这里最简单的选择就是在“文件名”（File name）字段中再次输入你刚刚选择的数据库名称，如这样：



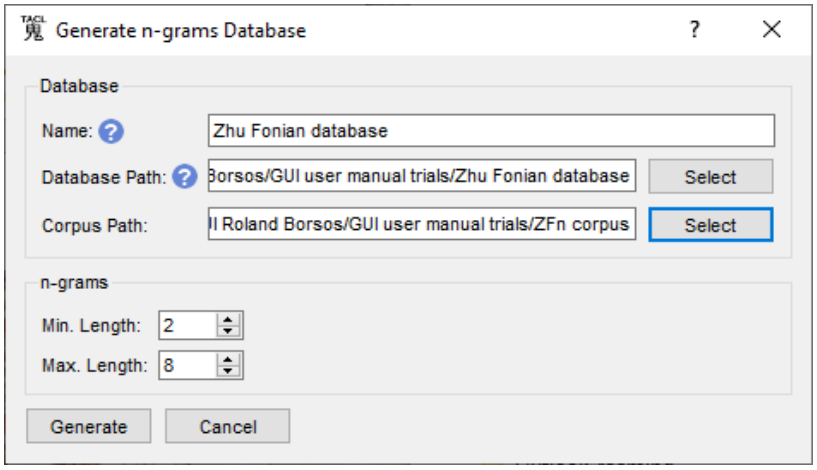

回到“生成n-grams数据库”（Generate n-gram database）界面（截图见上），接下来要选择你想用作数据库基础的语料库（“语料库路径”Database Path）。同样，点击“选择”(Select)按钮，然后导航进入相应的文件夹。



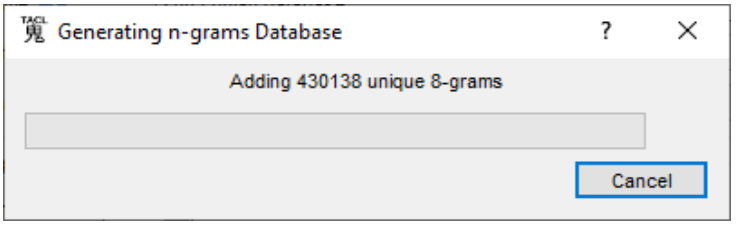

最后，你需要规定将被纳入数据库的n-gram的最小（Min. Length）和最大长度(Max. Length)。如上所述，我们建议最小：2，最大：8。




这时你的界面应该是这样：

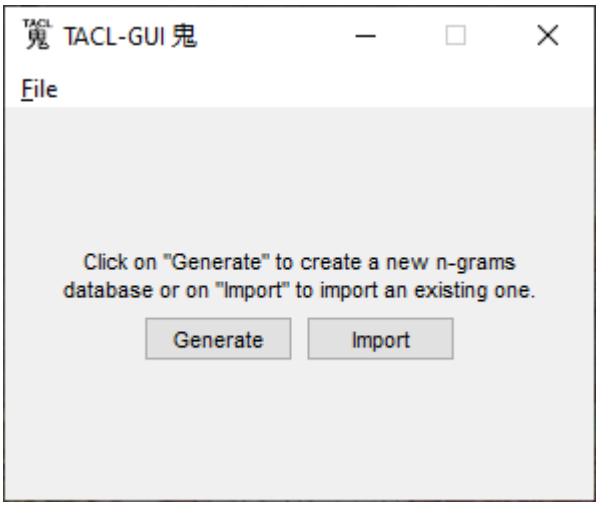



点击“生成”（Generate）。你会看到一个正在生成数据库的进度条：




取决于你的语料库的大小，数据库的生成可能需要从一两分钟到许多小时甚至几天的时间。

正如上面已经提到的，GUI的第一个窗口给了我们两个选项。我们不仅可以自己生成一个数据库，还可以导入（Import）：




因此，你可能想导入一个已经准备好的数据库，比如上面提到的可以下载的“Radich《大正藏》语料库”的数据库。为此，我们将首先概述一下下载过程，然后描述如何从外部来源将数据库导入GUI中。

(2a)下载“Radich《大正藏》语料库”的完整数据库

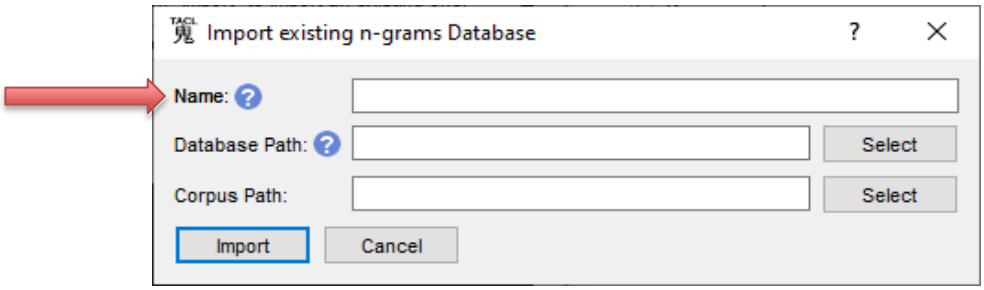
对于某些计算机来说，生成一个完整的《大正藏》语料库可能是费时费力的工作。为此，我们在Zenodo网站上提供了“Radich《大正藏》语料库”的完整数据库（压缩版）（https://zenodo.org/record/6795219 DOI：10.5281/zenodo.6795219）。点击链接可以下载。下载文件很大（约47GB），因此可能需要相当长的时间。

下载完成后，请保存或移动这个压缩文件到你用来做TACL工作的文件夹，并使用适当的工具如7zip（https://7-zip.org/）来解压数据库。请注意，由于该文件非常大，解压缩也可能需要相当长的时间（可能长达几个小时）

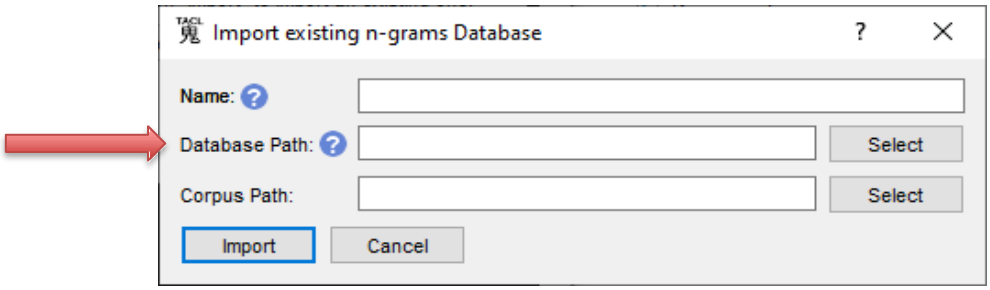
解压后，这个“完整的《大正藏》数据库”约为170GB。在尝试下载和解压缩之前，你应该确保你有足够的可用磁盘空间。

(2b)导入数据库

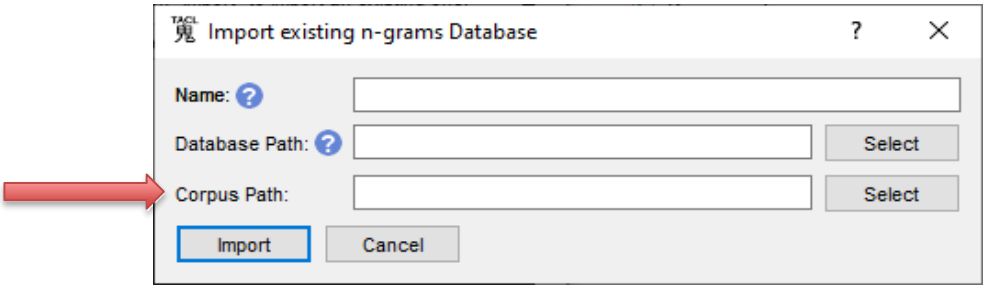
再次打开GUI，在起始界面中，你可以选择生成（Generate）或导入（Import）一个数据库（见上文）。如果你点击“导入”（Import），会看到这样一个对话框：




在“名称”（Name）栏中,为你的数据库命名(例如“full Taishō database”)。

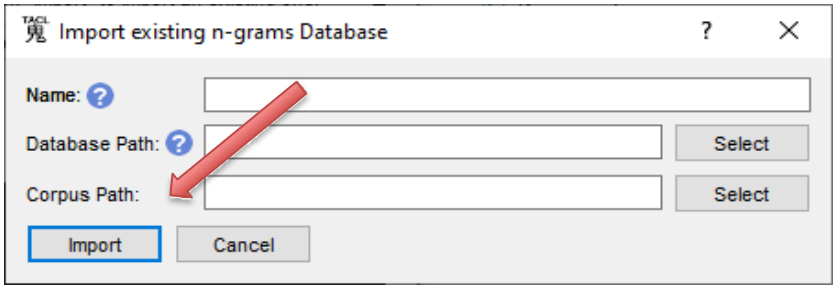



在“数据库路径”（Database Path）栏，输入一个通往你要使用的数据库的路路径。最简单的方法是点击“选择”（Select）按钮并导航到你要导入的数据库文件。你也可以直接键盘输入一个路径。




在“语料库路径”（Corpus Path）栏，输入通往包含与你要导入的数据库相关的语料库的文件夹的路径。

点击“导入”（Import）。

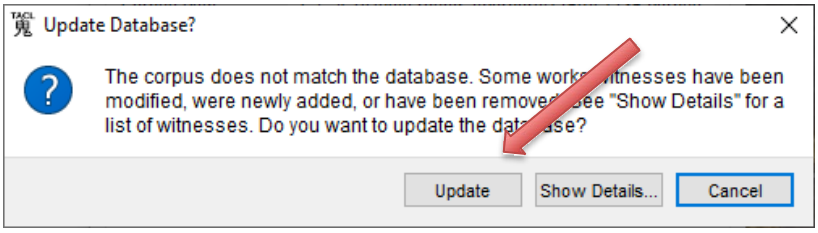



TACL GUI在此时会进行检查，以确保你的数据库和语料库相匹配。（它们必须相匹配，才能运行测试。）如果你要导入的数据库很大，核查数据库和语料库的过程会需要一段时间。

你应该看到一个类似“进度条”的东西。由于技术原因，这里不可能显示一个真正的进度条——你所看到的实际上只是一个占位图形，告诉你程序正在运行，请耐心等待。

如果数据库和语料库不匹配（例如，如果语料库在数据库生成后被修改过），你会看到一个错误信息。

> “语料库与数据库不匹配。一些作品/底本已经被修改，或被新添加，或已被移除。点击‘Show Details’会给出底本列表）。您要更新数据库吗？”（The corpus does not match the database. Some works/witnesses have been modified, were newly added, or have been removed. See “Show Details” for a list of witnesses. Do you want to update the database?）

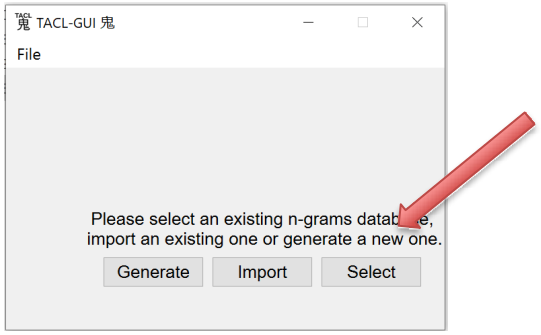



一般来说，如果你看到这个信息，你应该选择“更新”（Update）。更新过程是生成数据库的修改版（见下文）——程序会略过语料库中没有变化的部分，只更新其中与原始数据不一致的部分。如果你的语料库很大，这个过程可能需要相当长的时间。

（3）选择一个数据库

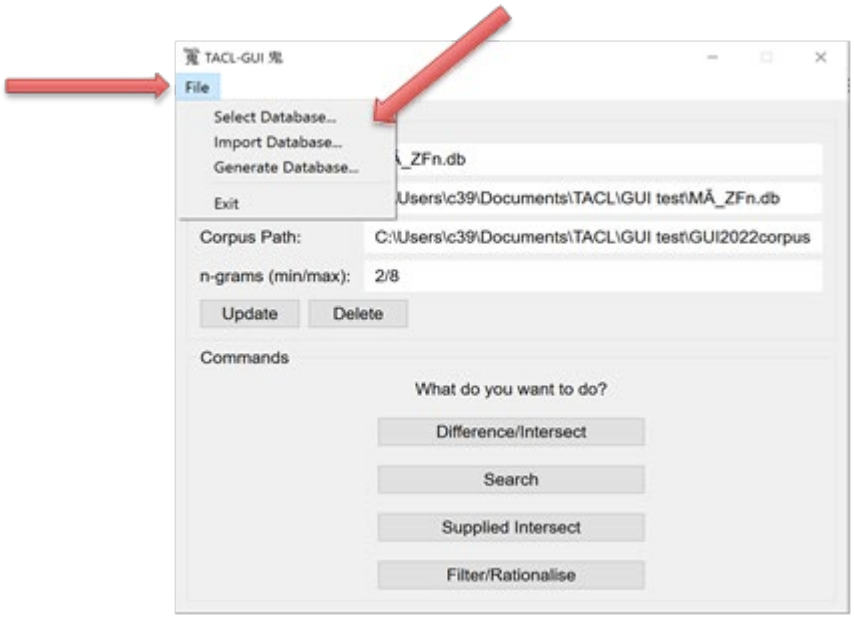
一旦你生成或导入了至少两个数据库，在进行TACL GUI测试操作之前，你可以简单地选择你需要用到的数据库。你可以通过以下两种方式之一进行选择：

在启动时（即初始界面），点击“选择”（Select）按钮：



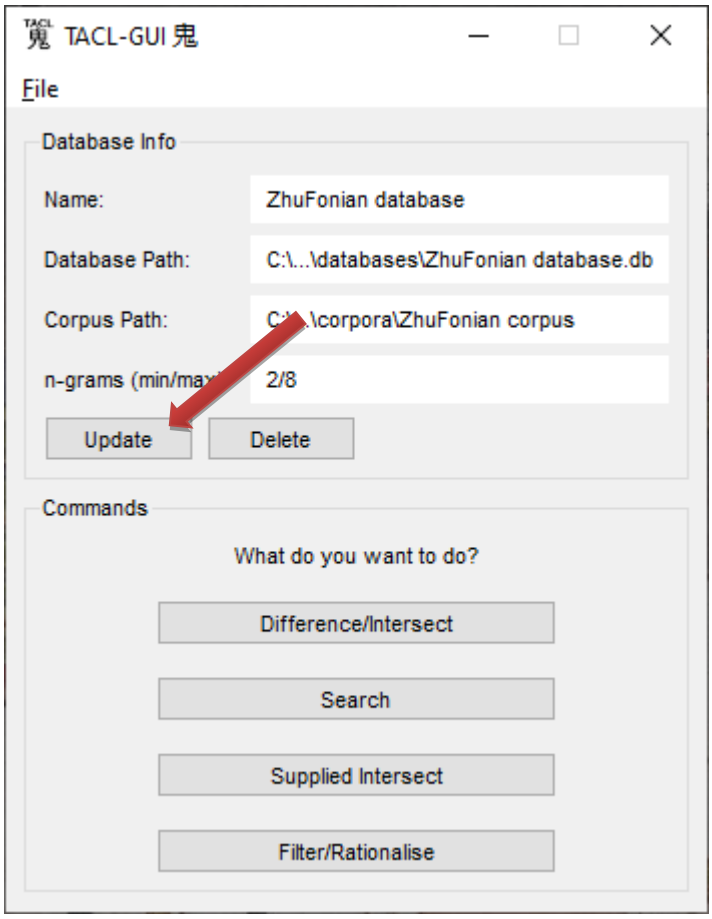

或者，当你已经在GUI界面内工作时，你可以在任何时候通过“文件”（File）选单切换数

据库而不需要重新启动。在这里，你还可以选择生成或导入一个新的数据库：

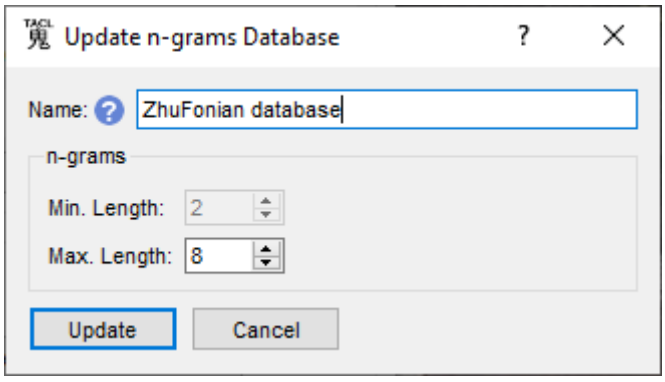



（4）更新一个数据库

如果你对语料库进行了修改，你也可以自己发起更新，而不必等待GUI的提示。要启动更新，请进入主选单窗口，并点击“更新”（Update）按钮（在“数据库信息”Database Info部分）。




然后你会看到下面的“更新n-grams数据库”（Update n-grams Database）窗口。在这里，如果你愿意，你也可以重新调整n-gram的长度（但原则上，我们建议你不要改变这些参数）。但是请注意，如果你把数据库移到不同的位置，这里没有任何选项可以通知GUI——在这个对话窗口中，通往语料库的路径与你第一次生成/导入数据库时给出的路径相同（关于改变语料库和数据库路径的更多信息，见下文）。

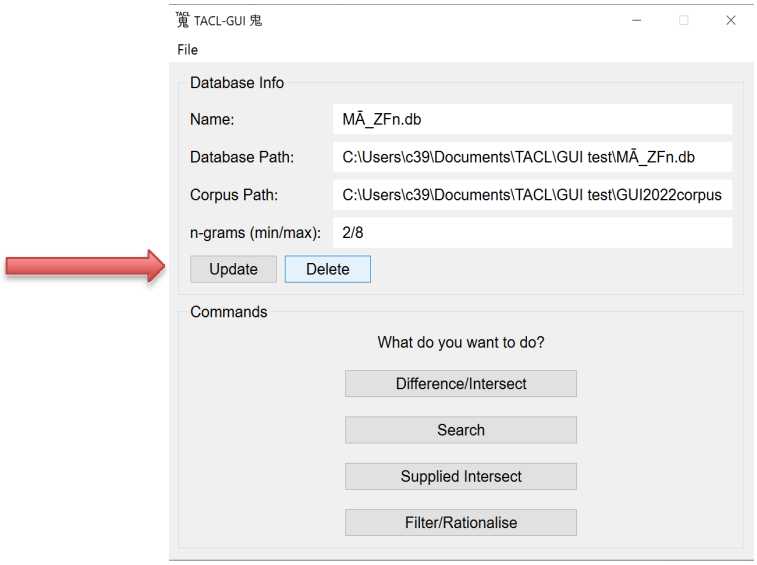



只需点击“更新”（Update）按钮即可实现对数据库的更新。

（5）删除一个数据库

在你选择了一个数据库之后，你始终可以选择更新它（例如，如果你的语料库发生了变

化），或者删除它。见TACL GUI主选单屏幕上的“更新”（Update）和“删除”（Delete）按钮。




用户必须注意，当一个数据库在GUI中被删除，它也在磁盘上被完全删除，永远消失。如果你想找回被删除的数据库，你将不得不重新生成它，或者重新导入它。

“一次只用一个数据库”。请注意，TACL在同一时间只能从一个给定的数据库运行一个操作。然而，有了这个约束，你也应该可以同时运行多个不同的GUI进程——只要每个进程是在不同的数据库上工作的。

## 工作步骤三：目录文件（Catalogue）

要想使用TACL的核心功能（一般情况下，即差异运算difference和交叉运算intersect）用户需要输入一个“目录”（catalogue）文件。TACL目录由两类元素组成：

（1）作品（work）列表，也就是这些作品在语料库中的“标识符”（identifier）（如T0002）（也即语料库中每个作品的子文件夹的名称，例如“T0002”，此作品各底本的txt文件便储存在这些子文件夹中）；

（2）“标签”（label），TACL用它来识别文本所属的分组。用户可以直接在他们创建的目录文件中定义标签。标签名由用户自己定义（建议使用意义明确的标签）。唯一需要注意的是，标签不能包括空格（所以“ZhuFonian”是正确的，但“Zhu Fonian”就不行）。

以下是一个简单的目录（catalogue）文件的例子：

```
T0002 CORPUS1
T0003 CORPUS2
```

在这里，作品（work），即要研究的文本，是“T0002”和“T0003”。“标签”（label）则是“CORPUS1”和“CORPUS2”。这个TACL“目录”告诉软件程序，为了进一步分析，我们要把这些文本分为两个不同的组，即CORPUS1和CORPUS2，每个组正好包含一个文本。使用这样的目录，我们就可以在这两个组之间进行比较——运行交叉（intersect）或差异（difference）运算。

对于TACL的核心测试，即“差异运算”（difference）与“交叉运算”（intersect），目录中必须至少包含两个标签（label）（简单起见，所有例子都设定有两个标签）。这两个标签代表你打算比较的两个子语料库；你希望分析的每个文本都需要携带这两个标签中的一个。如果一个文本在目录中没有标签，它就会被程序忽略。

下面是一个稍微复杂一点的真实例子（此目录亦见于练习中的“例1”）：

```
T0001-30-世記經 ZhuFonian
T0001-minus-30 ZhuFonian
T0125-50-禮三寶品-4 ZhuFonian
T0125-minus-50.4 ZhuFonian
T0194 ZhuFonian
T0212 ZhuFonian
T0309 ZhuFonian
T0384 ZhuFonian
T0385 ZhuFonian
T0656 ZhuFonian
T1428 ZhuFonian
T1464 ZhuFonian
T1505 ZhuFonian
T1543 ZhuFonian
T1549 ZhuFonian
T2045 ZhuFonian

T0026-whole Saṅghadeva
```

这里的作品（文本）有T0001-30-世記經，T0001-minus-30，T0125-50-禮三寶品-4，T0125-minus-50.4，T0194等等。(这些多样的作品标识符是“Radich语料库”对文本结构重新调整后的产物。)它们的“标签”（label）则是“ZhuFonian”（竺佛念）和“Saṅghadeva”（僧伽提婆）。由此，这个目录把列出的第一组文本分成单独的一组（即竺佛念的作品,用标签

“ZhuFonian”表示）；同时，它将另一个单一的文本，即《中阿含經》T26，设置为第二组，用于与第一组进行比较（实际上，此作品就是我们现在能确认属于僧伽提婆之作品的唯一文本）。有了这个目录，我们就可以通过TACL进行比较，找出僧伽提婆（Saṅghadeva）的作品，也就是《中阿含經》，和竺佛念（ZhuFonian）的作品之间可能的相似或不同之处。

TACL要求目录文件的格式为纯文本文件（以.txt文件扩展名保存）。用户可以使用纯文本编辑器来编辑和操作目录文件。我们推荐使用免费软件Notepad++（https://notepad-plus-plus.org/）。

请注意，如果你正在处理一个大的语料库，目录文件会变得相当长。以我们在编写本手册时使用的《大正藏》语料库的版本之一为例，其目录长达3,880行。这意味着，通过手动输入每一行来构建这样大的目录文件是非常费力的。因此，我们提供了我们所提供的两个大型语料库（基本的CBETA T-X语料库和Radich修改后的《大正藏》语料库）的“模板”基础目录文件供下载：

“catalogues/T-X base cat.txt”
“catalogues/Radich Taisho base cat.txt”

这两个文件可以从你的TACL GUI “入门工具包”（Starter Kit）中的“catalogue”文件夹中找到；也可从这两个网址下载：https://dazangthings.nz/tacl-gui-one-stop-shop/boilerplate-catalogues/）。我们还强烈建议用户熟悉正则表达式（regular expression（缩写为regex，https://en.wikipedia.org/wiki/Regular_expression）[1]在查找和替换方面的简单应用，以尽可能高效地处理此类大型目录，特别是为作品添加标签。

具体来说，在有了语料库之后，你就可以按如下步骤来建立一个目录（catalogue）文件，用来分析其中的部分或全部作品。在一个.txt文件中，你首先需要为语料库中用于测试的所有作品构建一个标识符（identifier，即作品子文件夹的名称）列表。这可以通过两种方法实现：(a)手工输入子文件夹名称/标识符列表（这只有在语料库很小的情况下才真正可行），或者(b)编辑一个现成的（“模板”）目录文件（当你的语料库很大时是更好的选择）。接下来，在每个作品名称后边添加一个标签（label），确保一个目录文件中至少有两个标签。需要注意的是，TACL将同样忽略任何没有标签的作品标识符。这意味着，通过使用上面提到的大型“模板”基础目录之一，你通常可以更有效地选择较小的文本组，只需给这些作品添加

[1] 另一个有用的资源：https://regexr.com/，此网站可以实时创建并测试regex搜索及替换操作。

标签，撇开所有你想忽略的文本。相比之下，删除所有你不感兴趣的作品标识符可能要费力得多，也更浪费时间。

## 修改语料库、数据库、和/或目录

在使用TACL的过程中，我们可能会在分析特定的语料库和数据库时，发现需要对语料库进行相应的修改（例如增加或删除其中的作品）。例如，我们可能有一个小的语料库和数据库，用于研究某个翻译团队的作品，却发现有另外一部作品没有被包含在初始语料库之中，但它又与研究对象相关，因此，我们需要把这个作品添加到语料库和数据库中。

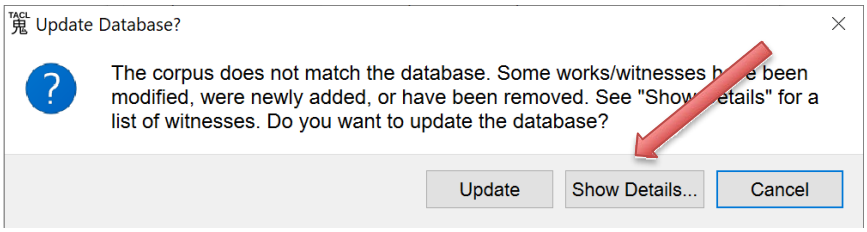
然而，用户在修改语料库之前，应该首先考虑使用目录（catalogue）来管理TACL的操作范围（见上文），而不是直接修改语料库或数据库。当某部作品一直在分析范围之内，而后来我们却希望将其排除在外时，让数据库保持原样，并简单地将这部作品从目录中排除，往往更有效率。

例如，假设在处理上面介绍的“竺佛念语料库”（ZhuFonian corpus）时，我们认为有一件作品实际上不是竺佛念的作品——在这个例子中是T2045，因此我们想相应地调整我们的测试。与其将T2045从语料库中删除并更新数据库，不如在目录文件中去掉T2045的标签（label），如下所示：

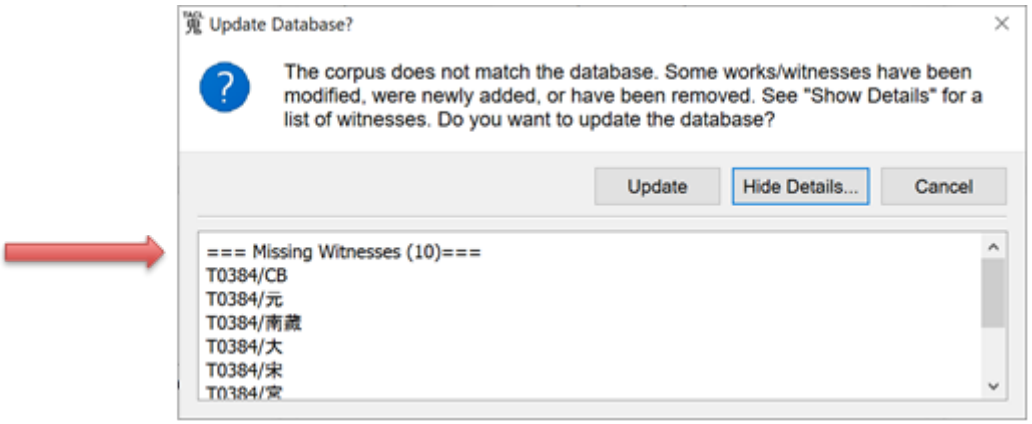
```
T0001-30-世記經 ZhuFonian
T0001-minus-30 ZhuFonian
T0125-50-禮三寶品-4 ZhuFonian
T0125-minus-50.4 ZhuFonian
T0194 ZhuFonian
T0212 ZhuFonian
T0309 ZhuFonian
T0384 ZhuFonian
T0385 ZhuFonian
T0656 ZhuFonian
T1428 ZhuFonian
T1464 ZhuFonian
T1505 ZhuFonian
T1543 ZhuFonian
T1549 ZhuFonian
T2045

T0026-whole Saṅghadeva
```

没有标签的作品会被TACL忽略。或者，你可以从目录文件中删除此作品所在的整行。在这两种情况下，T2045都将被排除在使用该目录文件进行的所有测试之外。这种做法可能比重新生成数据库更简单、更省时，特别是在数据库很大的情况下。

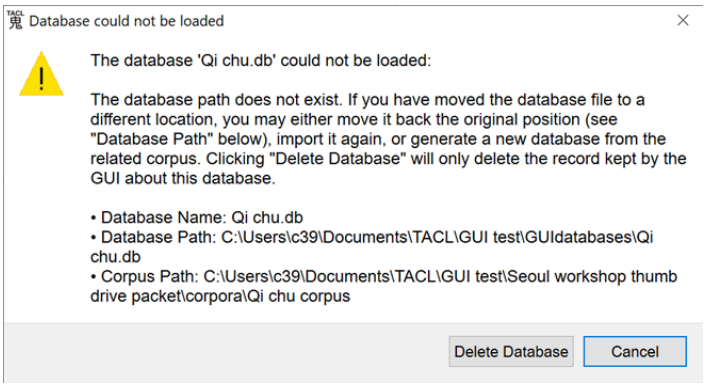
然而，如果我们意识到语料库和数据库中缺少某一作品，那就别无选择，只能修改语料库并相应地更新数据库。在这种情况下，我们可能会需要把新作品添加到语料库中。在直接修改了语料库后，下次启动GUI或尝试TACL操作时，你会看到这个数据库更新的提示：




如果你不确定是什么原因导致不匹配，点击“显示细节”（Show Details），你会看到导致语料库和数据库不匹配的作品列表。




还要注意的是，如果你把数据库或语料库移到一个不同的位置，或者改动了数据库或语料库的名称，GUI将提示一个错误，表示此数据库路径不存在：



这种情况下你有如下这些选项：

1) “删除数据库”（Delete Database）（这实际上意味着只删除GUI所知道的旧数据库路径）。
2) 将数据库移回原来的位置或恢复到原来的文件名。
3) 从新的位置/用新的文件名重新导入数据库。
4) 用相应的语料库再次生成数据库。

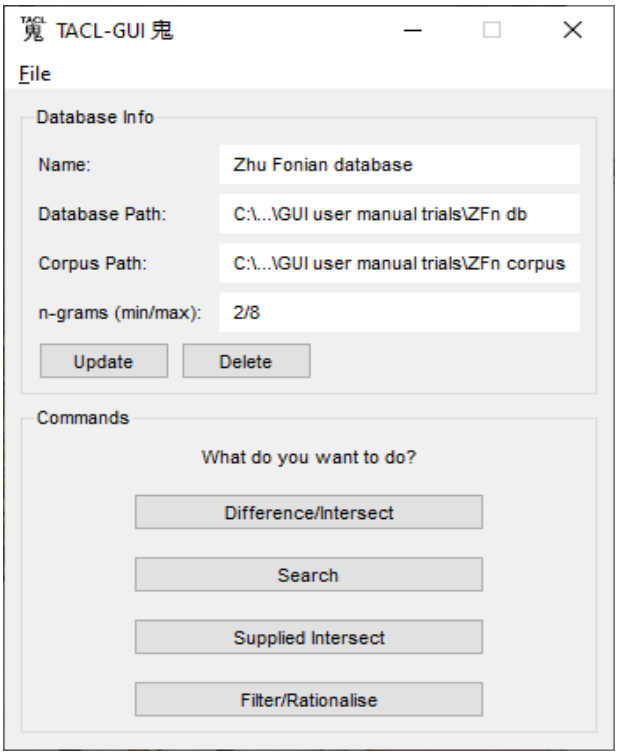
## 工作步骤四：运行TACL核心测试

当你已经生成或导入了至少一个数据库，并且知道什么是目录（catalogue）文件，就可以准备运行实际的TACL测试。

注意：GUI中所有下列功能的界面中，你都会看到这个问号图标:

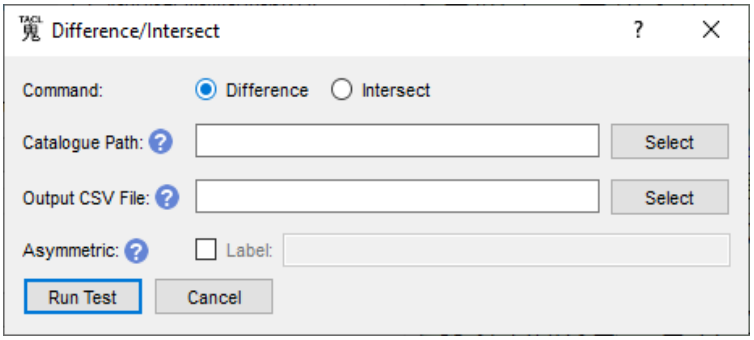
用户可以将鼠标悬停在这个图标上，以获得有关该功能的更多信息（建议尝试）。

在生成/导入数据库过程结束后，你将立即看到TACL GUI的主界面：



在第二次和之后再使用TACL GUI时，如果你已经生成或导入了至少一个数据库，重新启动时，你会立即看到这个界面（除非GUI检测到你的数据库和语料库之间不匹配——参看上面的说明）。

最基本的TACL测试是“差异运算”（difference）和“交叉运算”（intersect）。我们先解释这两种运算。点击“差异运算/交叉运算”（Difference/Intersect）按钮，将出现下面这个界面：




交叉运算（Intersect）

交叉运算（Intersect）可以找出在所有子语料库（由目录catalogue文件中的标签label所定义的作品分组）中都出现的字符串n-gram。这在寻找后期文本对早期文本内容的借用时特别有用——例如，在中国创作的佛经借用了早期真正译自印度文本的佛经（如著名的T156：https://dazangthings.nz/cbc/text/1553/）；或当我们希望找到有哪些后来的作者（如论师）引用了一个文本时，也可以使用交叉运算测试。我们将通过一个具体的例子解释整个操作过程。你可以在阅读过程中使用你的“入门套件”（Starter Kit）中的材料，一步一步跟着做。

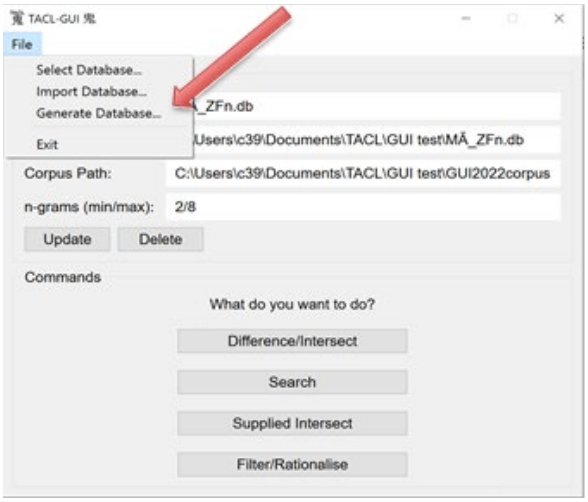
让我们以一个已知的案例作为简单的例子，你可以按照下面的指导一步一步自己操作（这个练习的材料在“入门套件”Starter Kit中），找出《七處三觀經》T150A和《四願經》T735（Nattier称之为“T735C”的部分）中内容重叠的部分。这就是前面介绍过的“例2”（稍后我们会回头再讨论关于竺佛念的“例1”）。这个例子的背景信息可以参考那体慧的论文（Nattier2008: 131-132），或https://dazangthings.nz/cbc/text/274/。请注意，这个新的例子将使用与上一个例子不同的语料库和数据库。

首先设想我们正在寻找T150A的可能来源。在这个练习中，T735代表了一个包含了更大范围

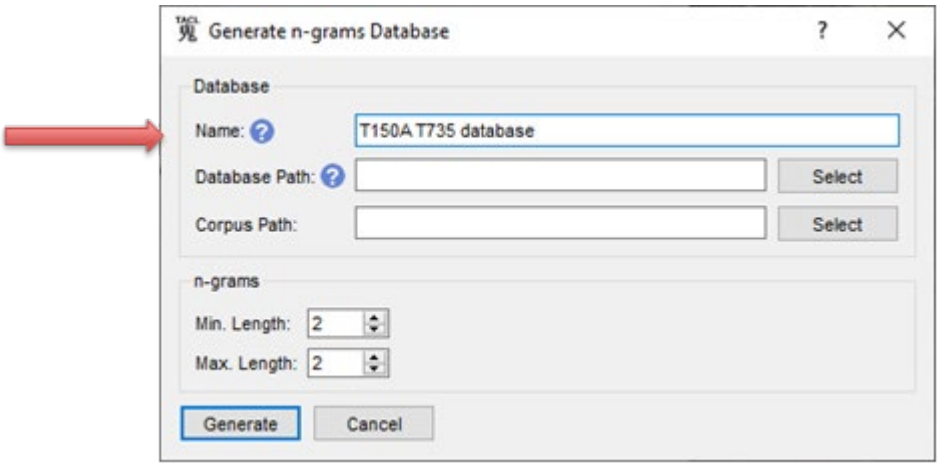
的可能来源的语料库。（为了让测试更为简单、快速地进行，我们在这里仅采用此单一文本与T150A进行比较。）

我们首先需要一个包括T150和T735的语料库（下载包中的“corpora”[语料库]子文件夹中提供了这个语料库，名为“T150A T735 corpus”）。

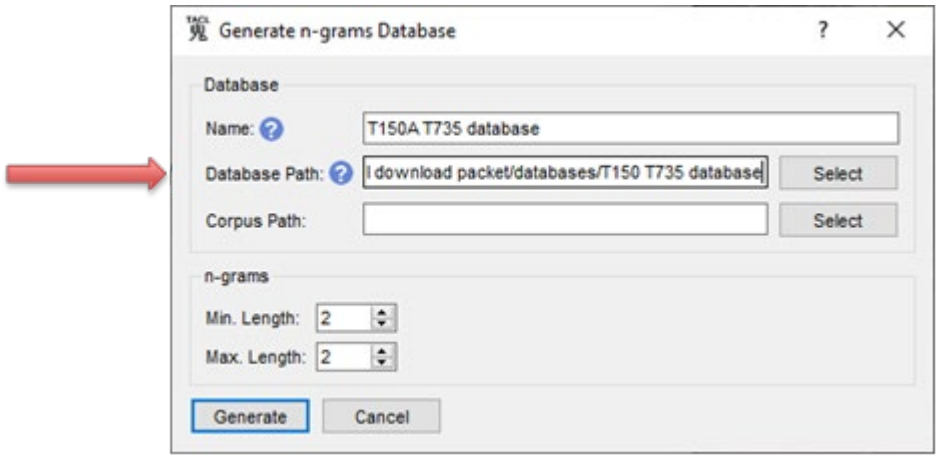
接下来，我们为这个语料库建立一个小型的自定义数据库（如果你已经下载并导入了一个完整的数据库，可以跳过这一步）。然后，选择“生成数据库”（Generate Database）：



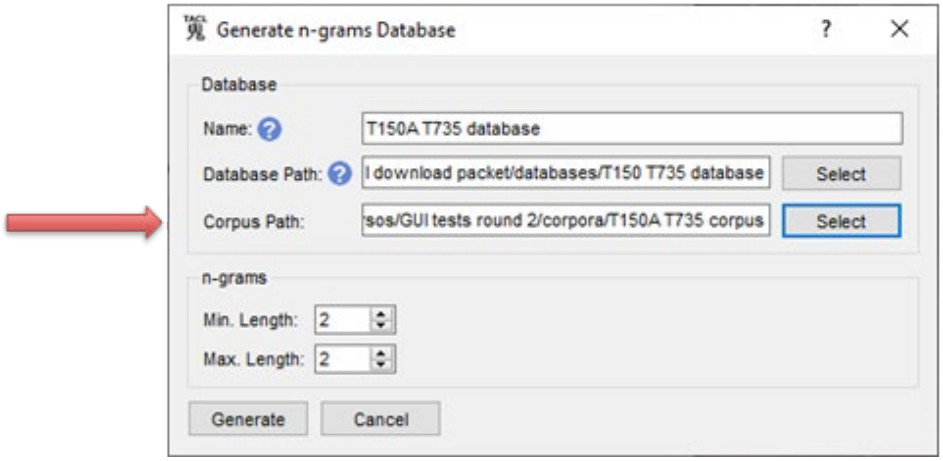

然后在“名称”（Name）一栏中将数据库命名：



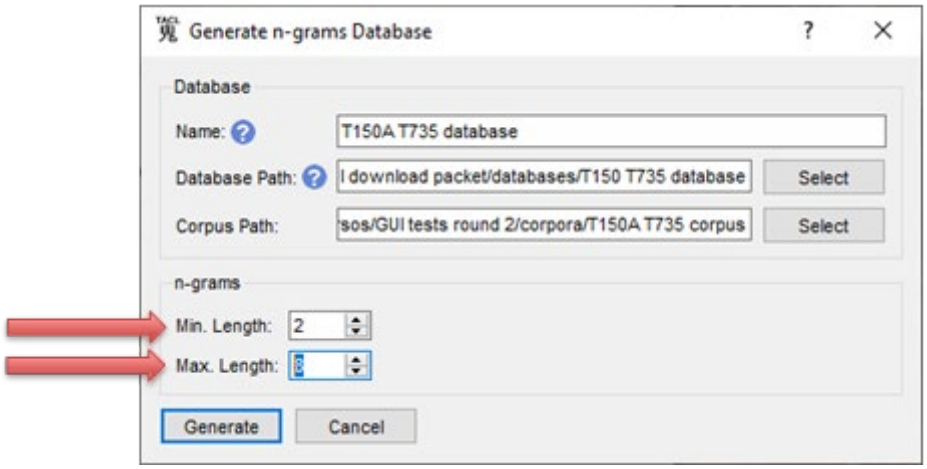

通过选择（select）或键盘输入数据库所在文件夹的路径，并再次键入数据库名称：



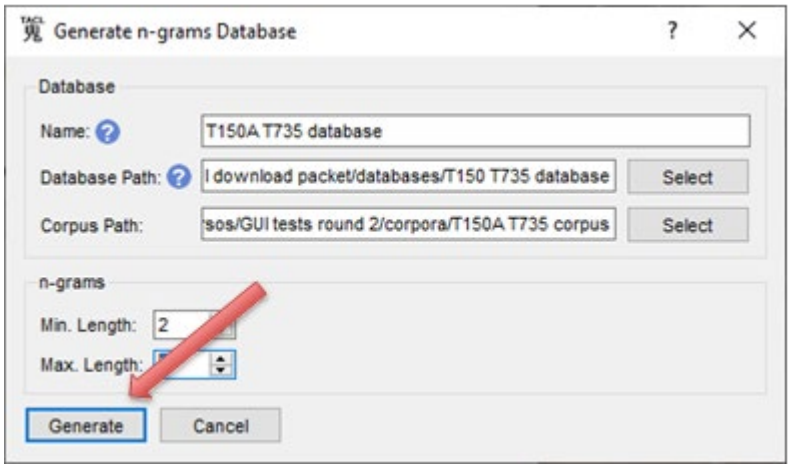

下一步，通过选择（select）或键盘输入，告诉GUI需要生成数据库的语料库在哪里：




接下来，我们设置最小和最大的n-gram字符串长度，这里用的是2-gram到8-gram：




最后，点击“生成”（Generate），并等待TACL完成它的工作：

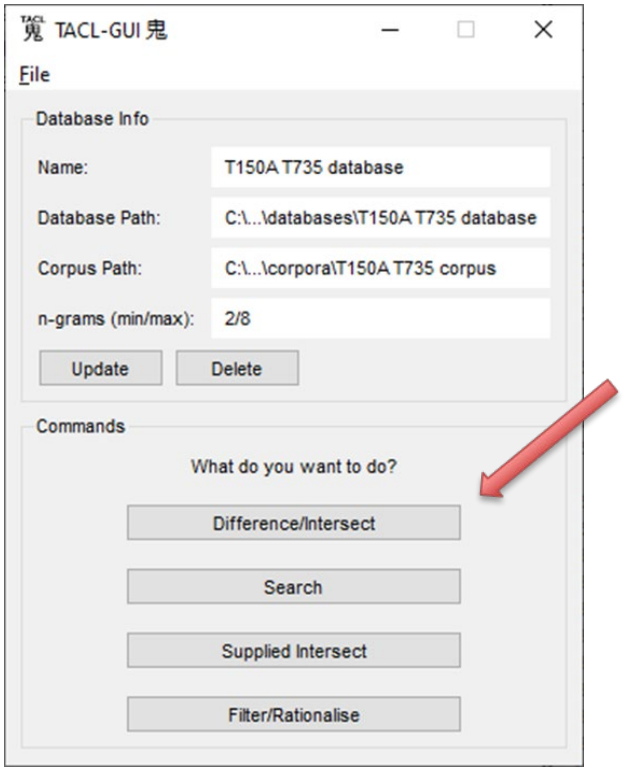



建立好数据库之后，我们需要一个目录（catalogue）文件，其内容像这样：

```
T0150A Qichu
T0735 Siyuan
```

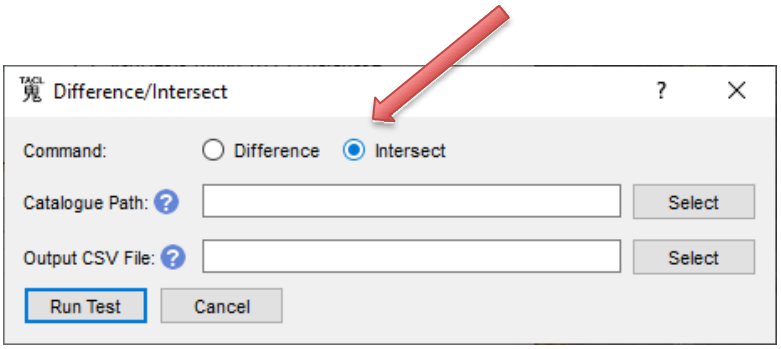
将这个文件保存，拟名为“T150A vs T735 cat.txt”（这个文件已经放在下载包的“catalogues”子文件夹中）。

现在我们可以运行交叉运算（intersect）测试了。在主选单窗口中，点击“差异运算/交叉运算”（Difference/Intersect）按钮：

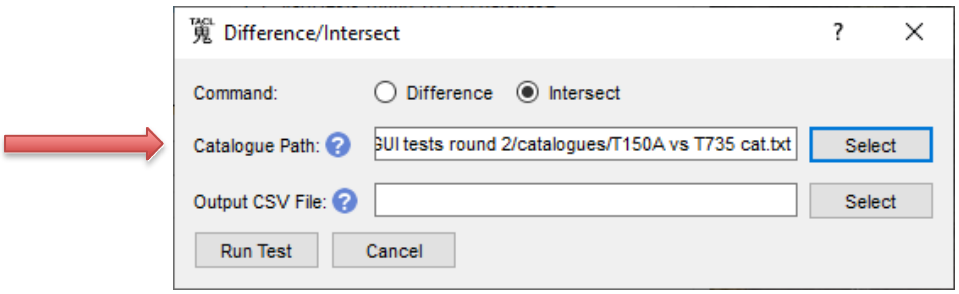


如果前面你都按照本手册描述的步骤一步一步操作，那么现在你应该已经选择了正确的数据库和语料库（“T150A T735 database”，“T150A T735 corpus”），因为我们刚刚从该语料库生成了这个数据库。然而，以后当你运行其他测试时，你可能会直接来到这个窗口。如果这之前你曾经使用了别的数据库，那么当前要运行的测试就会提示你选择了错误的数据库和语料库。因此，你应该随时检查所选择的数据库和语料库是否正确（如上面截图中的窗口上方所示）。如果没有，请按照上面“（3）选择数据库”（第31页）一节中的说明进行操作。

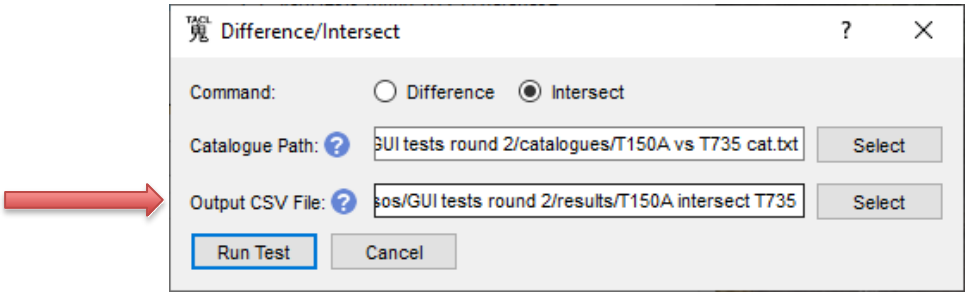
接下来，要运行交叉运算测试，首先要选择“交叉运算”（Intersect）选项：




通过点击“选择”（Select）按钮导航到目录（catalogue）文件所在的文件夹路径，或手工输入路径：

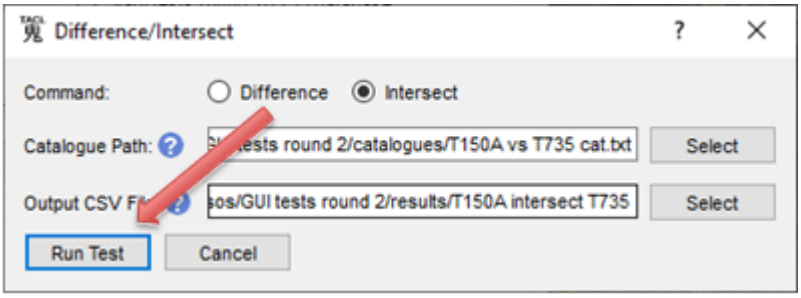



再次提醒，目录中列出的作品需要至少两个标签（label）来分组。没有标签的作品将被忽略。



接下来，在“输出CSV文件”（Output CSV File）一栏中，填入要输出的.csv文件的路径。你可以点击“选择”（Select）按钮导航到一个文件夹，或者手工输入路径，并且为结果输入一个文件名。

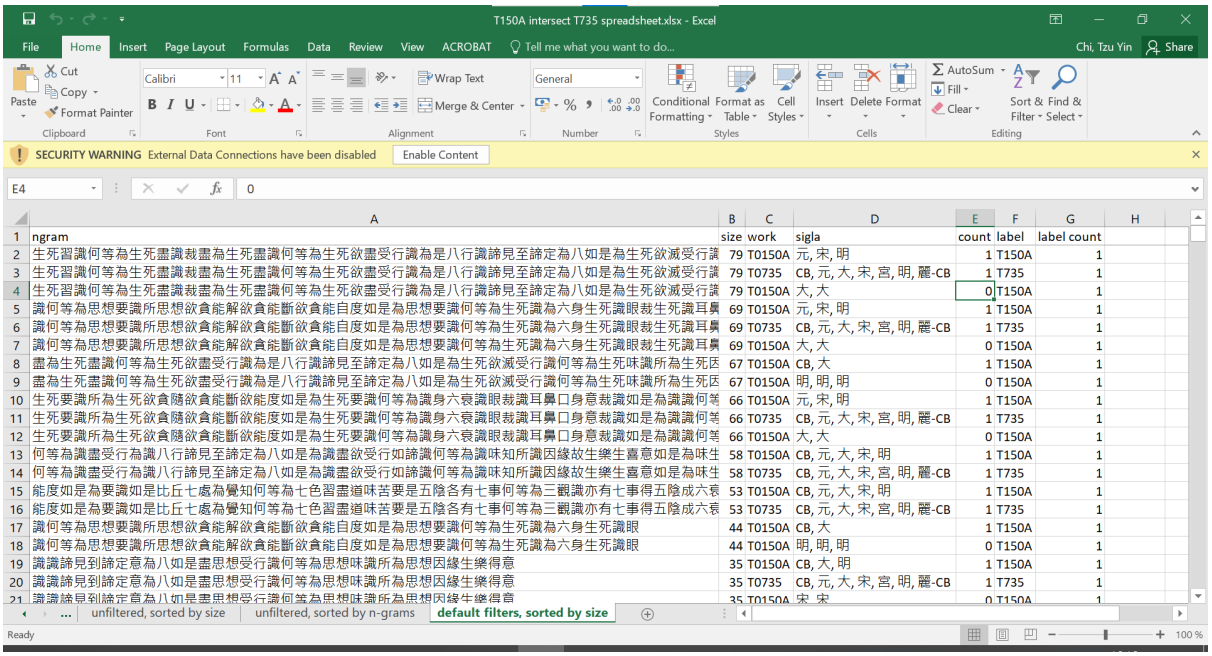
然后点击“运行测试”（Run Test）：




在TACL工作时，会弹出一个带有类似“进度条”的窗口（它同样不会真正显示工作进度，只是一个代表程序正在工作的示意图形）。测试完成后，弹出的窗口将消失，你将在你指定的位置找到初步结果的.csv文件。

接下来，在对初步结果进行人工语文学分析之前，建议你通过一个基本的“筛选/合理化”（Filter/Rationalize）操作来对其进行处理。此步骤将在下文详细解释（见“筛选/合理化：” Filter/Rationalize，第64页）。就目前而言，在这个练习中，我们假定你已经完成了这一步骤，并且已经有了一个筛选后的结果文件，例如“T150A intersect T735 filtered.csv”。

你的TACL结果现在可以用来进行人工语文学分析了。此时，建议你使用Excel等工具来查看和分析结果。我们也会在下文中详细说明（见下文第73页“处理结果”Working with the results）。（注意!在Windows中，如果Excel被设置为处理.csv文件的默认应用程序——它通常是默认的——你不能通过简单地通过双击csv文件来打开结果；如果这样它们显示的都是乱码，请按照后边描述的步骤操作。）在此，我们再次假定你已经学会了这一步骤；这里我们的目的只是向你展示本练习的结果是什么样子的。你可以将结果导入Excel，然后进行排序（使用我们在下面推荐的交叉测试的排序方法来寻找文本的来源）。在Excel中你将会看到这样的界面：




| | ngram | size | work | sigla | count | label | label count |
|---|---|---|---|---|---|---|---|
| 2 | 生死習識何等為生死盡識裁盡為生死盡識何等為生死欲盡受行識為是八行識諦見至諦定為八如是為生死欲滅受行識 | 79 | T0150A | 元, 宋, 明 | 1 | T150A | 1 |
| 3 | 生死習識何等為生死盡識裁盡為生死盡識何等為生死欲盡受行識為是八行識諦見至諦定為八如是為生死欲滅受行識 | 79 | T0735 | CB, 元, 大, 宋, 宮, 明, 麗-CB | 1 | T735 | 1 |
| 4 | 生死習識何等為生死盡識裁盡為生死盡識何等為生死欲盡受行識為是八行識諦見至諦定為八如是為生死欲滅受行識 | 79 | T0150A | 大, 大 | 0 | T150A | 1 |
| 5 | 識何等為思想要識所思想欲貪能解欲貪能斷欲貪能自度如是為思想要識何等為生死識為六身生死識眼裁生死識耳鼻 | 69 | T0150A | 元, 宋, 明 | 1 | T150A | 1 |
| 6 | 識何等為思想要識所思想欲貪能解欲貪能斷欲貪能自度如是為思想要識何等為生死識為六身生死識眼裁生死識耳鼻 | 69 | T0735 | CB, 元, 大, 宋, 宮, 明, 麗-CB | 1 | T735 | 1 |
| 7 | 識何等為思想要識所思想欲貪能解欲貪能斷欲貪能自度如是為思想要識何等為生死識為六身生死識眼裁生死識耳鼻 | 69 | T0150A | 大, 大 | 0 | T150A | 1 |
| 8 | 盡為生死盡識何等為生死欲盡受行識為是八行識諦見至諦定為八如是為生死欲滅受行識何等為生死味識所為生死因 | 67 | T0150A | CB, 大 | 1 | T150A | 1 |
| 9 | 盡為生死盡識何等為生死欲盡受行識為是八行識諦見至諦定為八如是為生死欲滅受行識何等為生死味識所為生死因 | 67 | T0150A | 明, 明, 明 | 0 | T150A | 1 |
| 10 | 生死要識所為生死欲貪隨欲貪能斷欲能度如是為生死要識何等為識身六衰識眼裁識耳鼻口身意裁識如是為識識何等 | 66 | T0150A | 元, 宋, 明 | 1 | T150A | 1 |
| 11 | 生死要識所為生死欲貪隨欲貪能斷欲能度如是為生死要識何等為識身六衰識眼裁識耳鼻口身意裁識如是為識識何等 | 66 | T0735 | CB, 元, 大, 宋, 宮, 明, 麗-CB | 1 | T735 | 1 |
| 12 | 生死要識所為生死欲貪隨欲貪能斷欲能度如是為生死要識何等為識身六衰識眼裁識耳鼻口身意裁識如是為識識何等 | 66 | T0150A | 大, 大 | 0 | T150A | 1 |
| 13 | 何等為識盡受行為識八行諦見至諦定為八如是為識盡欲受行如諦識何等為識味知所識因緣故生樂生喜意如是為味生 | 58 | T0150A | CB, 元, 大, 宋, 明 | 1 | T150A | 1 |
| 14 | 何等為識盡受行為識八行諦見至諦定為八如是為識盡欲受行如諦識何等為識味知所識因緣故生樂生喜意如是為味生 | 58 | T0735 | CB, 元, 大, 宋, 宮, 明, 麗-CB | 1 | T735 | 1 |
| 15 | 能度如是為要識如是比丘七處為覺知何等為七色習盡道味苦要是五陰各有七事何等為三觀識亦有七事得五陰成六衰 | 53 | T0150A | CB, 元, 大, 宋, 明 | 1 | T150A | 1 |
| 16 | 能度如是為要識如是比丘七處為覺知何等為七色習盡道味苦要是五陰各有七事何等為三觀識亦有七事得五陰成六衰 | 53 | T0735 | CB, 元, 大, 宋, 宮, 明, 麗-CB | 1 | T735 | 1 |
| 17 | 識何等為思想要識所思想欲貪能解欲貪能斷欲貪能自度如是為思想要識何等為生死識為六身生死識眼 | 44 | T0150A | CB, 大 | 1 | T150A | 1 |
| 18 | 識何等為思想要識所思想欲貪能解欲貪能斷欲貪能自度如是為思想要識何等為生死識為六身生死識眼 | 44 | T0150A | 明, 明, 明 | 0 | T150A | 1 |
| 19 | 識識諦見到諦定意為八如是盡思想受行識何等為思想味識所為思想因緣生樂得意 | 35 | T0150A | CB, 大, 明 | 1 | T150A | 1 |
| 20 | 識識諦見到諦定意為八如是盡思想受行識何等為思想味識所為思想因緣生樂得意 | 35 | T0735 | CB, 元, 大, 宋, 宮, 明, 麗-CB | 1 | T735 | 1 |
| 21 | 識識諦見到諦定意為八如是盡思想受行識何等為思想味識所為思想因緣生樂得意 | 35 | T0150A | 宋, 宋 | 0 | T150A | 1 |

(我们也把这个例子的排好序的excel格式的结果放在“results”子文件夹中，文件名为“T150A intersect T735 spreadsheet sorted.xlsx”。请看一下。)

在这里的Excel表格的第一行中，我们可以看到TACL找到了一个n-gram，长度为79字符（即79-gram，一个79个字符的字符串），由所分析的两个文本/组所共享（如目录catalogue中所定义）。

也就是说，这个79字的字符串在T150A和T735中都出现了。情况确实如此——但有一点复杂的是，在T150A中，正如我们在“底本标记”（sigla）一栏中看到的，这个字符串只出现在“宋”“元”“明”版的底本中，但在《大正藏》版的底本“大”中则没有。它没有出现在《大正藏》版的底本中是因为有一个字的写法不同（红色标出部分），从而让TACL在这里给出不同的搜索结果（记住，TACL只能找到精确的字面匹配）。

> ...生死習識。何等為生死盡識？裁[v.l.裁,SYM]盡為生死盡識。何等為生死欲盡受行識？為是八行識，諦見至諦定為八，如是為生死欲滅受行識。何等為生死味識？所為生死因緣生樂喜意，如是為生死味識。何等為生死..., T150A (II) 876b12-17.
>
> ...生死習識？何等為生死盡識？裁盡為生死盡識。何等為生死欲盡受行識？為是八行識，諦見至諦定為八，如是為生死欲滅受行識。何等為生死味識？所為生死因緣生樂喜意，如是為生死味識。何等為生死..., T735 (XVII) 537c2-6.

这个长字符串在整个《大正藏》中是唯一的（用CBETA搜索看看，用一个通配符——星号“*”——来搜索不同的写法。也就是说，你应该搜索“生死習識何等為生死盡識*盡為生死盡識何等

為生死欲盡受行識為是八行識諦見至諦定為八如是為生死欲滅受行識何等為生死味識所為生死因緣生樂喜意如是為生死味識何等為生死）。

显然，如此长的精确匹配不可能是巧合，而是证据之一，揭示了两个文本之间的关系，对此，上文所引论文已有论述。假设此案例尚不为人所知，那我们就发现了T150A和T735在文本上的密切关系。

下面我们再看交叉运算（Intersect）的第二个例子，即前面提到的“例3”。我们再援引一个已知的案例：《增一阿含經》T125（48.3）和《彌勒下生經》T453的文本几乎完全匹配(见 https://dazangthings.nz/cbc/text/616/)。我们有理由怀疑，《增一阿含經》T125的部分内容借用了早期文本。同样，我们用T453代表一个更大的语料库，来快速运行测试，以达到演示的目的。我们对这个练习的描述比较简单，没有屏幕截图和逐步说明。如果你遇到困难，可以参考上文中更详尽的T150A的例子。

测试的步骤如下：

（1）建立一个包括T125和T453的语料库（这也是为什么T453也包含在你的下载包里，在竺佛念语料库中“corpora/ZhuFonian corpus”——建议你查看一下语料库中的内容来了解它是如何工作的，但作为练习，我们也建议你尝试自己建立语料库）。

（2）为语料库生成或导入一个数据库。建议你用竺佛念语料库“ZhuFonian corpus”练习一下，因为我们将在接下来的练习（“例1”）中使用相同的语料库和数据库。

（3）构建一个目录（catalogue）文件，将T125和T453定义为两个不同的语料库（见“catalogues/Ekottarikagama vs T453 cat.txt”——建议你用我们的目录内容作为例子，但你也可以自己写一个，作为练习）[2]。

（4）（4）通过这个语料库（corpus）、数据库（database）以及目录（catalogue）来进行交叉运算（intersect）测试。

（5）对结果进行“筛选/合理化”（Filter/Rationalize）。

（6）将结果导入Excel，并对结果进行排序（我们提供了一个样本文件“results/Ekottarikagama intersect T453 spreadsheet sorted.xlsx”，但我们也建议你最好独立完成这些练习步骤）。

[2] 请注意：在我们的竺佛念语料库“ZhuFonian corpus”中,《增一阿含經》T125分成了两个文本：
T0125-50-禮三寶品-4
T0125-minus-50.4
这是一个“Radich”《大正藏》语料库针对TACL应用对《大正藏》文本进行调整的典型例子。之所以这样调整的原因，见https://dazangthings.nz/cbc/text/4175/。

如果正确地完成了所有这些步骤，第一个结果应该是这个224字的字符串（224-gram）：

> …訓之所致也亦由四事因緣惠施仁愛利人等利爾時阿難彌勒如來當取迦葉僧伽梨著之是時迦葉身體奄然星散是時彌勒復取種種華香供養迦葉所以然者諸佛世尊有敬心於正法故彌勒亦由我所受正法化得成無上正真之道阿難當知彌勒佛第二會時有九十四億人皆是阿羅漢亦復是我遺教弟子行四事供養之所致也又彌勒第三之會九十二億人皆是阿羅漢亦復是我遺教弟子爾時比丘姓號皆名慈氏弟子如我今日諸聲聞皆稱釋迦弟子爾時彌勒與諸弟子說法汝等比丘當思惟無常之想樂有苦想計我無我想實有空想色變之想青瘀之想…

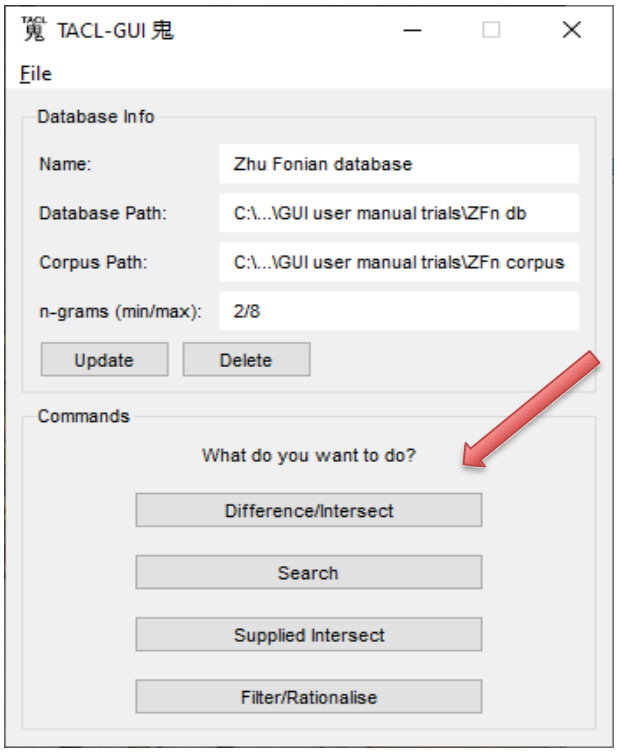
和上一个例子类似，其中一个结果（这次是在《增一阿含經》T125中）因为有两个字的不同（红色部分）而没有被从《大正藏》版的底本中找出。但正如我们在“筛选/合理化”（Filter/Rationalized）后的“底本标记”（sigla）栏中所看到的，同样的字符串在《高丽藏》中也出现了（CBETA标记为“麗”）。同样地，在全部藏经中，这一文本匹配也是独一无二的（鉴于它是如此之长，在其他文本中几乎不可能原文复现），这样长的文本匹配当然不可能是巧合。如果我们没有从前人的研究得知此事，从这个结果中，我们可以推断《增一阿含經》48.3一定是借用了T453的文本内容（或者反之——T453借用了T125的文本）。

差异运算（Difference）

“差异运算”(difference)测试可以找出参与比较的每个子语料库（sub-corpus）中各自所特有的字符串n-gram，子语料库是由目录（catalogue）文件中的标签（label）所定义的；这个测试也可以只针对比较中的一方（见下面的“非对称差异运算”asymmetric difference）。差异运算测试尤其适用于把一组文本与另一组进行比较，从而找出其中重复出现的特征，例如写作风格的特征（详见《TACL方法指南》*TACL Method Guide*）。

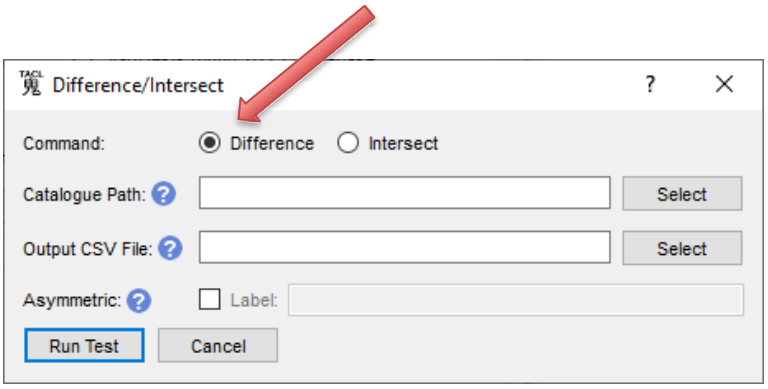
这里我们将首先引导你用GUI一步一步完成差异运算测试的主要的步骤，然后给出一个例子作为练习，你可以尝试自己完成。

要运行一个差异运算测试，请在主选单界面上点击“差异运算/交叉运算”（Difference/Intersect）按钮：



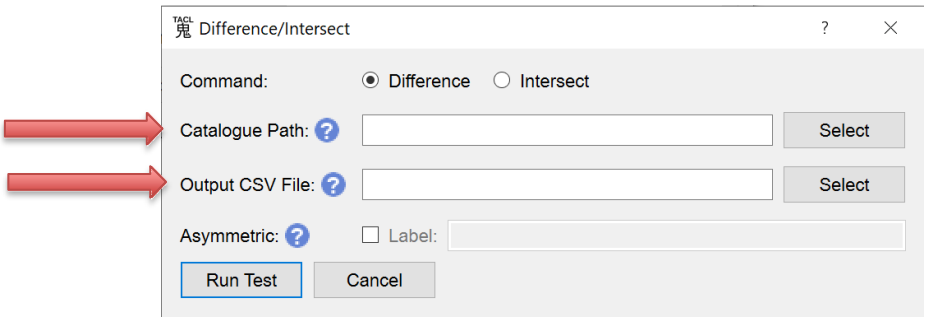

同样地，如前面交叉运算（intersect）测试所描述的那样，此时，你必须检查是否选择了正确的数据库（database）和语料库（corpus），如窗口截图的上半部分所示。如果没有，请按照上面“（3）选择数据库”一节（第31页）中的说明进行操作。

选择“差异运算”（Difference）（通常默认选择此选项）：




除了数据库之外，该操作还需要一个目录文件（catalogue，见上文）。与“交叉运算”（intersect，见上文）一样，通过按“选择”（Select）按钮导航或手工输入路径来填上目录文件的路径。

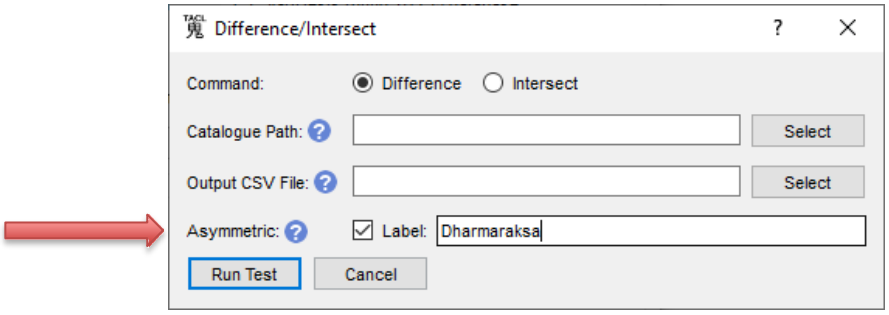
同样地，填上要输出的.csv文件的路径和文件名。




非对称差异运算（Asymmetric Difference）

TACL差异运算（difference）中有一个“非对称”（Asymmetric）选项，选中它后，差异运算只找出相比较的两个子语料库之一所特有的字符串（n-gram）。这个选项的好处是，我们经常在一个小的语料库（如单个文本）和一个大得多的语料库（如整部大藏经）之间进行不平衡的比较，将结果限制在差异的一边（通常是较小的那一边）可以大大减少输出结果的数量，否则结果可能会多到让人无从下手。通常，我们也只对比较的一方感兴趣（例如，与所有其他翻译文本相比，支谦的文本有什么特点？）这意味着我们不需要产生大量的、实际上用不到的结果。

要进行非对称差异运算，请在对话窗口中勾选“非对称”（Asymmetric）选项，并填入你要关注的子语料库的标签（label）。




我们再一次从一个简化的例子开始——首先，是一个对称的差异运算（我们稍后再尝试“不对称”的选项）。我们鼓励你在阅读过程中按步骤完成这个例子（“例4”）。

在这个例子中，设想我们试图发现（a）安玄和严佛调，以及竺法护（Dharmarakṣa）之间翻译风格的差异。幸运的是，这两个翻译团队都翻译了同一作品：《郁伽長者問經》（Ugra-paripṛcchā）（安玄和严佛调译本T322，竺法护Dharmarakṣa译本T323）。

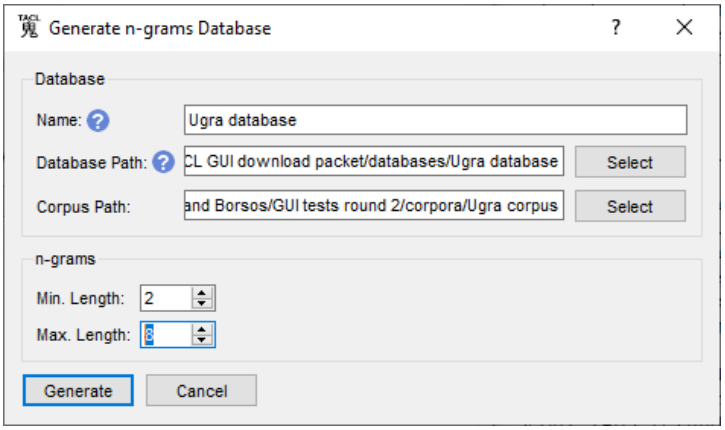
此外，（从我们未发表的基于TACL的研究中）也可以看出，竺法护的团队在工作时，面前

一定有安玄和严佛调的作品。由于这两个文本在内容上基本重合，它们之间的差异更可能透露出这两个团队各自的独特翻译风格。此外，因为竺法护了解当时已有的译本，那么就竺法护而言，译文之间的差异甚至可能是有意为之的。

同样地，我们在下载包中提供了这个练习的语料库、目录和样本结果文件，但我们建议你尝试独立完成所有步骤，自己运行测试。在这个例子中，我们将在一定程度上简化屏幕截图，前提是你已经熟悉了GUI。

首先，我们建立一个语料库，包含T322和T323（在“入门套件”Starter Kit中的“corpora/Ugra corpus”文件夹中）。

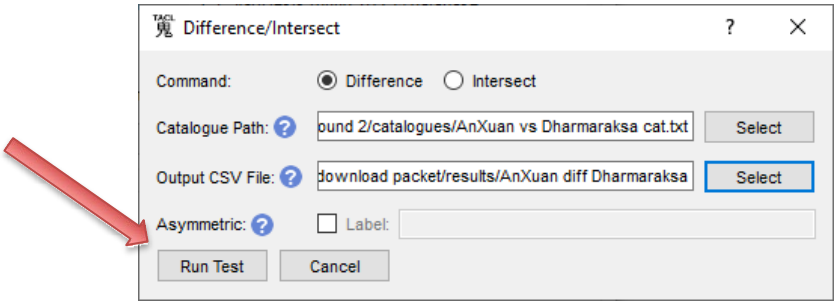
除非你已经有一个包括这些文本的完整数据库，否则，要为该语料库建立一个自定义的数据库。




接下来，建立一个目录文件（catalogue），定义并区分两个语料库（目录文件见“catalogues/AnXuan vs Dharmaraksa cat.txt”）：

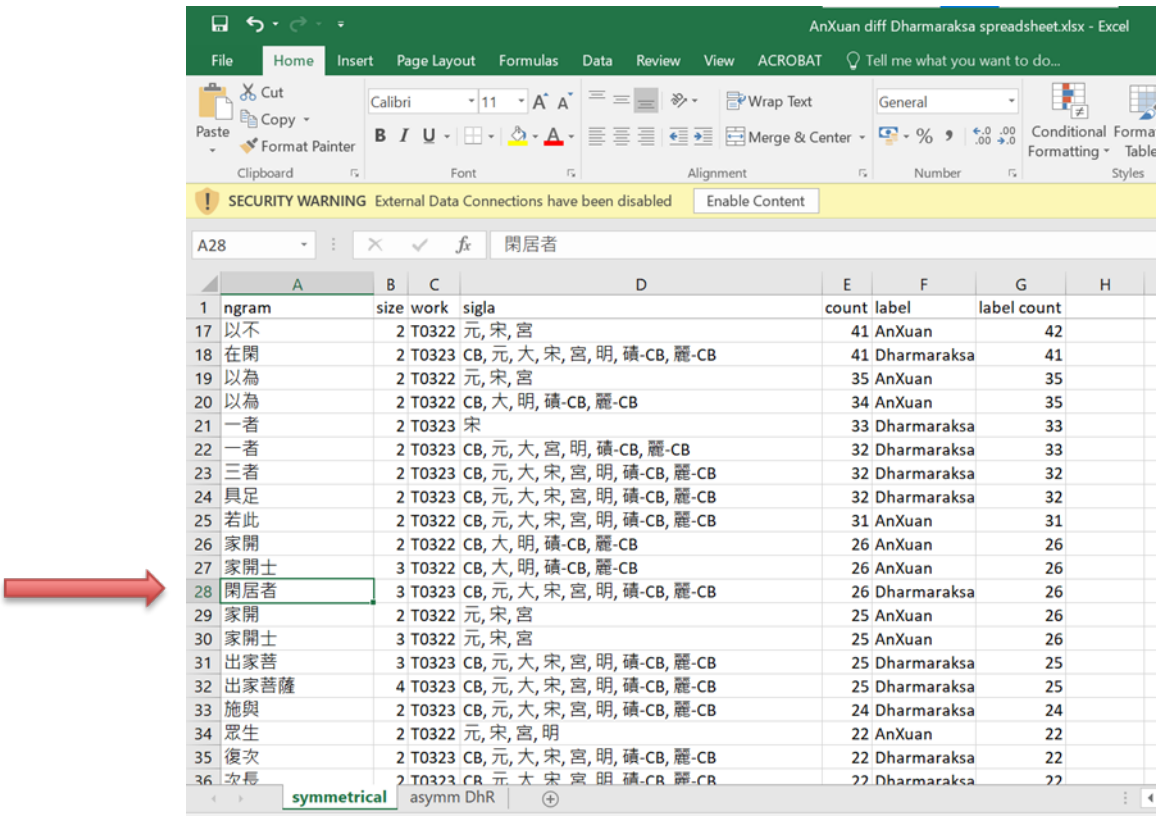
```
T0322 AnXuan
T0323 Dharmaraksa
```

进入“差异运算/交叉运算”（Difference/Intersect）界面，告诉GUI在哪里找到目录文件（catalogue），在哪里保存结果文件，然后运行测试。



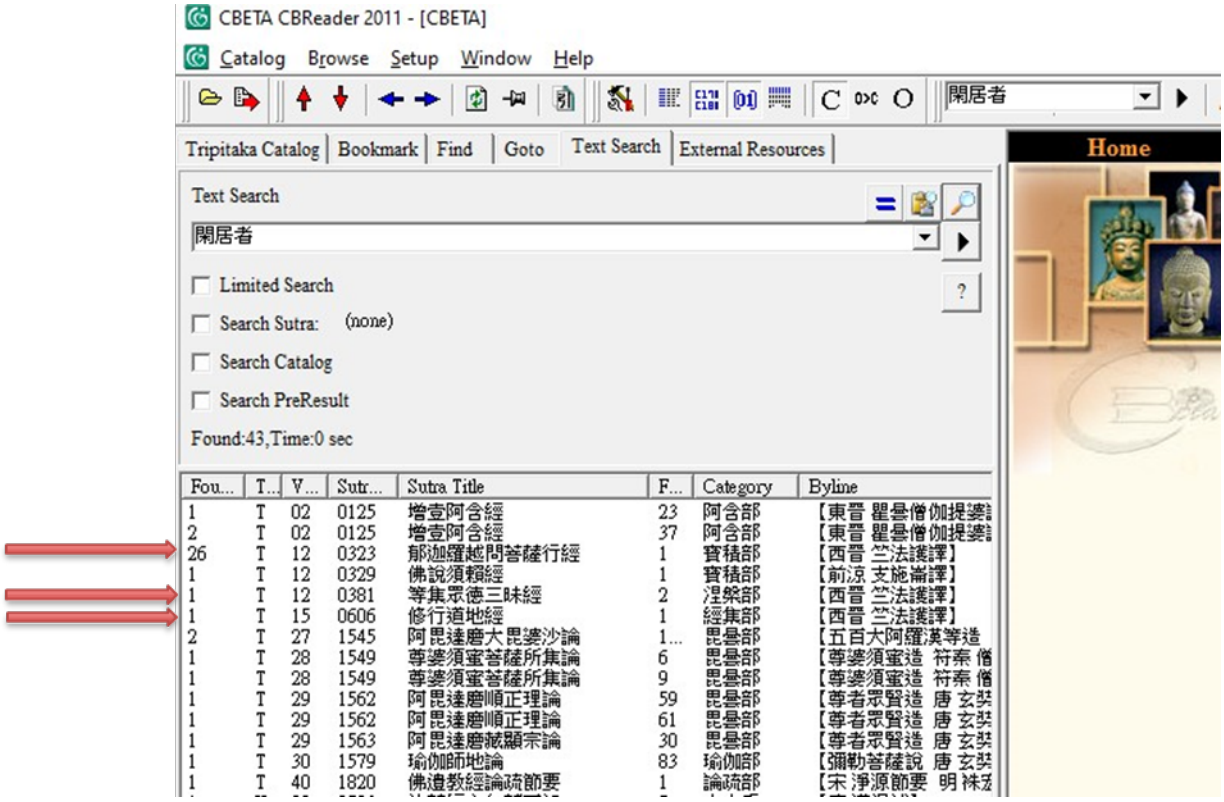

在这里，我们没有选择不对称差异运算（Asymmetric）的选项。测试运行结束后可以通过基本的“筛选/合理化”（Filter/Rationalize）操作来处理结果，将结果导入Excel，并对其进行排序（可使用这样的排序规则来分析翻译风格：先按n-gram计数以降序排列；然后按长度以升序排列。这样得到的结果可在“results/AnXuan diff Dharmaraksa spreadsheet.xlsx”中检视，注意是名为“symmetrical”的工作表）。

如果前面的步骤都正确完成，在排序后的Excel结果文件中看到的结果之一是“閑居者”这个词，它出现在第28行，在竺法护的T323中出现了26次（根据定义，因为这是一个差异运算测试，所以应该从未在T322中出现）。




| | A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|
| 1 | ngram | size | work | sigla | count | label | label count | |
| 17 | 以不 | 2 | T0322 | 元, 宋, 宮 | 41 | AnXuan | 42 | |
| 18 | 在閑 | 2 | T0323 | CB, 元, 大, 宋, 宮, 明, 磧-CB, 麗-CB | 41 | Dharmaraksa | 41 | |
| 19 | 以為 | 2 | T0322 | 元, 宋, 宮 | 35 | AnXuan | 35 | |
| 20 | 以為 | 2 | T0322 | CB, 大, 明, 磧-CB, 麗-CB | 34 | AnXuan | 35 | |
| 21 | 一者 | 2 | T0323 | 宋 | 33 | Dharmaraksa | 33 | |
| 22 | 一者 | 2 | T0323 | CB, 元, 大, 宮, 明, 磧-CB, 麗-CB | 32 | Dharmaraksa | 33 | |
| 23 | 三者 | 2 | T0323 | CB, 元, 大, 宋, 宮, 明, 磧-CB, 麗-CB | 32 | Dharmaraksa | 32 | |
| 24 | 具足 | 2 | T0323 | CB, 元, 大, 宋, 宮, 明, 磧-CB, 麗-CB | 32 | Dharmaraksa | 32 | |
| 25 | 若此 | 2 | T0322 | CB, 元, 大, 宋, 宮, 明, 磧-CB, 麗-CB | 31 | AnXuan | 31 | |
| 26 | 家開 | 2 | T0322 | CB, 大, 明, 磧-CB, 麗-CB | 26 | AnXuan | 26 | |
| 27 | 家開士 | 3 | T0322 | CB, 大, 明, 磧-CB, 麗-CB | 26 | AnXuan | 26 | |
| 28 | 閑居者 | 3 | T0323 | CB, 元, 大, 宋, 宮, 明, 磧-CB, 麗-CB | 26 | Dharmaraksa | 26 | |
| 29 | 家開 | 2 | T0322 | 元, 宋, 宮 | 25 | AnXuan | 26 | |
| 30 | 家開士 | 3 | T0322 | 元, 宋, 宮 | 25 | AnXuan | 26 | |
| 31 | 出家菩 | 3 | T0323 | CB, 元, 大, 宋, 宮, 明, 磧-CB, 麗-CB | 25 | Dharmaraksa | 25 | |
| 32 | 出家菩薩 | 4 | T0323 | CB, 元, 大, 宋, 宮, 明, 磧-CB, 麗-CB | 25 | Dharmaraksa | 25 | |
| 33 | 施與 | 2 | T0323 | CB, 元, 大, 宋, 宮, 明, 磧-CB, 麗-CB | 24 | Dharmaraksa | 24 | |
| 34 | 眾生 | 2 | T0322 | 元, 宋, 宮, 明 | 22 | AnXuan | 22 | |
| 35 | 復次 | 2 | T0323 | CB, 元, 大, 宋, 宮, 明, 磧-CB, 麗-CB | 22 | Dharmaraksa | 22 | |

事实上，这个长度为3的字符串3-gram也出现在竺法护的其他作品中（还有另外几个地方），尽管次数非常少。




虽然与TACL有能力发现的许多事情相比，这个结果不能算什么重大发现，但即使是这个经过简化的小例子也可以表明，我们确实可以发现能将竺法护与其前辈区分开来的，翻译风格上的真正差异。

接下来我们介绍第二个例子：使用差异运算（difference）测试。在这里，我们可以尝试解决更接近真实研究的问题。为此，让我们回到“例1”，竺佛念和《增一阿含經》（*Ekottarikāgama）的问题。

TACL“差异运算”的一个关键用途是，对照一些有意义的比较点，找到一个译者（或翻译团队，或其他一些“作者”）可能的特征性文体标记。下面我们可以通过测试的实例来说明如何探讨现存的《增一阿含經》T125是由竺佛念（或以其为核心的翻译团队）还是僧伽提婆（Saṅghadeva）翻译的这一长期存在的问题（前人对此问题的研究，见 https://dazangthings.nz/cbc/text/2237/）。然而，这一次，我们将把“差异运算”的结果

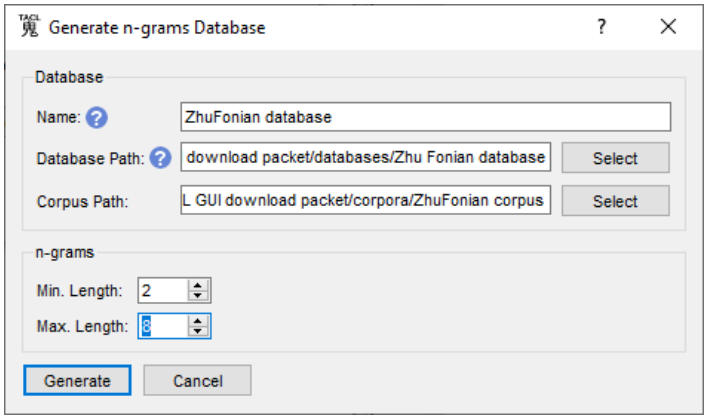
与更多的测试相结合，以获得更强大的分析能力。

首先，对僧伽提婆（Saṅghadeva）和竺佛念这两个子语料库进行差异运算。我们使用了一个确定属于这两个译者的，可靠的作品列表的语料库TACL目录（catalogue）文件。在这个阶段，我们先把《增一阿含經》T125本身从竺佛念的语料库中剔除，因为这个不确定的文本是我们要考察的对象。我们通过在目录中保留T125，但不给它任何标签（这样TACL就会忽略它）来实现这一目的。

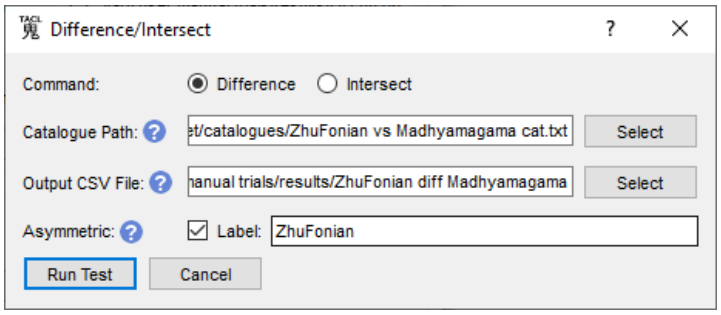
```
T0001-30-世記經 ZhuFonian
T0001-minus-30 ZhuFonian
T0125-50-禮三寶品-4
T0125-minus-50.4
T0194 ZhuFonian
T0212 ZhuFonian
T0309 ZhuFonian
T0384 ZhuFonian
T0385 ZhuFonian
T0656 ZhuFonian
T1428 ZhuFonian
T1464 ZhuFonian
T1505 ZhuFonian
T1543 ZhuFonian
T1549 ZhuFonian
T2045 ZhuFonian

T0026-whole Saṅghadeva
```

简单起见，设想我们一开始只对比较运算结果中的竺佛念一方感兴趣。（实际上，在还不知道答案的情况下，要想对这个问题进行更全面的调查，就需要对比较的两边都进行系统的测试；另见Radich 2017a, Radich and Anālayo 2017）。因此，我们需要把“竺佛念语料库”（ZhuFonian corpus）与僧伽提婆语料库进行非对称差异运算测试，以找出竺佛念作品中所特有的，重复出现的n-gram，而这些n-gram在僧伽提婆的《中阿含經》中从未出现。在这些n-gram中，我们很有可能找到区分竺佛念和僧伽提婆的翻译风格的特征性字符串。

为了进行这项测试，我们必须首先有一个基于竺佛念作品语料库“ZhuFonian corpus”的数据库（见下载包中“corpora”文件夹中的“Zhu Fonian corpus”子文件夹），或一个包括该语料库中所有作品的更大的语料库（如“Radich Taisho corpus”）。如果你在前面我们提到竺佛念数据库时没有生成它，建议你现在就生成这个数据库。

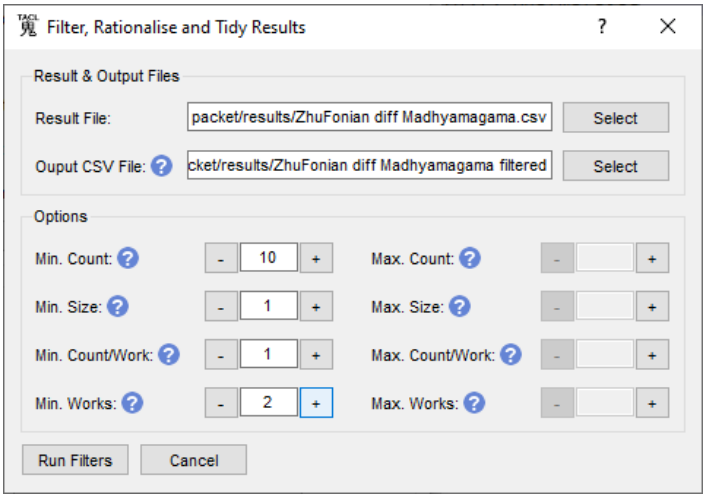



然后，我们在竺佛念和《中阿含經》/僧伽提婆（Saṅghadeva）之间进行一个差异运算（Difference）——这次是一个不对称（Asymmetric）的差异运算。(请注意：这个测试所需的运行时间比我们之前所做的其他测试要长一些）。



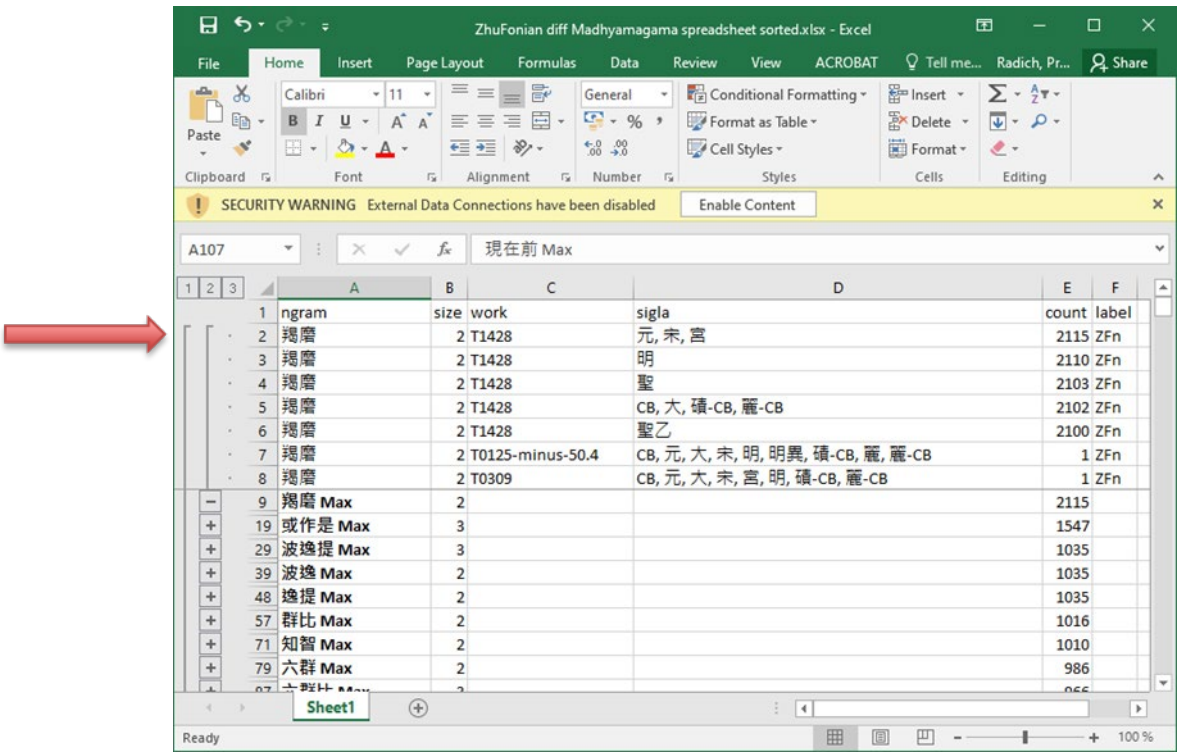

下面，我们将把这个差异操作的结果与其他TACL操作“串联”起来（见“结果交叉运算” Supplied Difference，第57页）。这个串联的测试需要使用我们刚生成的原始结果（即尚未经过“筛选/合理化”Filter/Rationalize的结果）。

然而，与此同时，为了练习，先假设我们直接对这个测试的当前结果感兴趣。像往常一样，在我们进行人工语文学分析之前，要先用基本的“筛选/合理化”（Filter/Rationalize）操作来处理结果。然而，这一次，因为结果太多，无论是Excel还是人工都无法完全处理，所以我们还指定了最小计数（minimum count）为10，最小作品数（minimum works）为2（详见下文“筛选/合理化”Filter/Rationalize，第64页），以便过滤掉在竺佛念语料库中相对较少出现的字符串。




然后，我们将其导入Excel，并进行排序（这次推荐你采取用于写作风格测试的排序规则：先按计数count以降序排列，然后按长度size以升序排列；详见“处理结果”一节，第73页）。同样，在下载包里我们提供了这样一个已经排好序的电子表格的例子，但建议你最好还是尝试自己动手得到这些结果。

如果做得正确，你将看到这样的结果（“ZhuFonian diff Madhyamagama spreadsheet sorted.xlsx”）：




不出所料，结果中的第一个字符串“羯磨”出现于竺佛念的三部作品中——在T1428中尤其多，

但在T125和T309中各只出现一次，而在《中阿含經》中则从未出现过。（这可以从上面的截图中看出，只需要点击n-gram旁边的加号，展开“羯磨”的结果；你也可以通过在CBETA中搜索来自己确认。）下一个字符串“或作是”（在上下文中看到的则是“或作是說”“或作是語”“或作是論”等等）出现于六部竺佛念的作品中（绝大部分在T1549中），但从未在《中阿含經》中出现过。从这区区二个字符串，我们已经开始发现竺佛念与僧伽提婆之间可能的风格差异了。

## 工作步骤五：结果交叉运算（Supplied Intersect）

结果交叉运算（supplied Intersect）可以列出之前的TACL测试（已经运行过的差异运算或交叉运算测试）给出的所有结果文件所共有的n-gram。

换句话说，你必须至少进行两个先前的TACL操作之后，才能对结果进行交叉运算。请注意，所有的结果文件都必须采用原始文件，也就是说，必须未经任何筛选/合理化（Filter/Rationalize）操作。

在这里，我们继续使用上面差异运算（Difference）的第二个例子，即“例1”，竺佛念与《增一阿含經》。

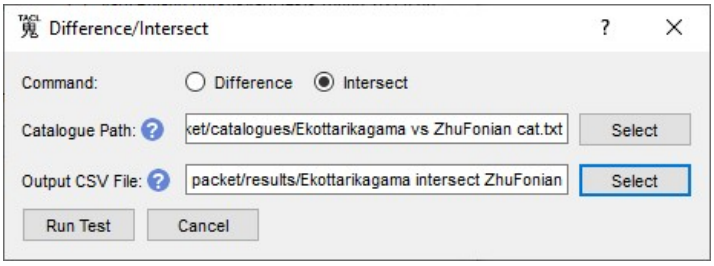
首先，竺佛念语料库和《中阿含經》之间的不对称差异运算（asymmetric difference）只是我们调查《增一阿含經》是出自竺佛念还是僧伽提婆之手的两个准备步骤之一。差异运算为我们提供了只出现在竺佛念作品中、而没有出现于《中阿含經》中的n-gram。

我们还需要一个测试来从另一个侧面进行考察：把《增一阿含經》与除《增一阿含經》以外的其他竺佛念作品（ZhuFonian-minus-Ekottarikāgama）语料库进行交叉运算（intersect）取交集，从而得到在《增一阿含經》和竺佛念作品中都可以找到的n-gram。该测试与上述测试使用相同的竺佛念语料库（ZhuFonian corpus）和数据库，但使用不同的目录（catalogue）文件（“catalogues/Ekottarikagama vs ZhuFonian cat.txt”）：

```
T0125-50-禮三寶品-4 Ekottarikagama
T0125-minus-50.4 Ekottarikagama

T0001-30-世記經 ZhuFonian-other
T0001-minus-30 ZhuFonian-other
T0194 ZhuFonian-other
T0212 ZhuFonian-other
```

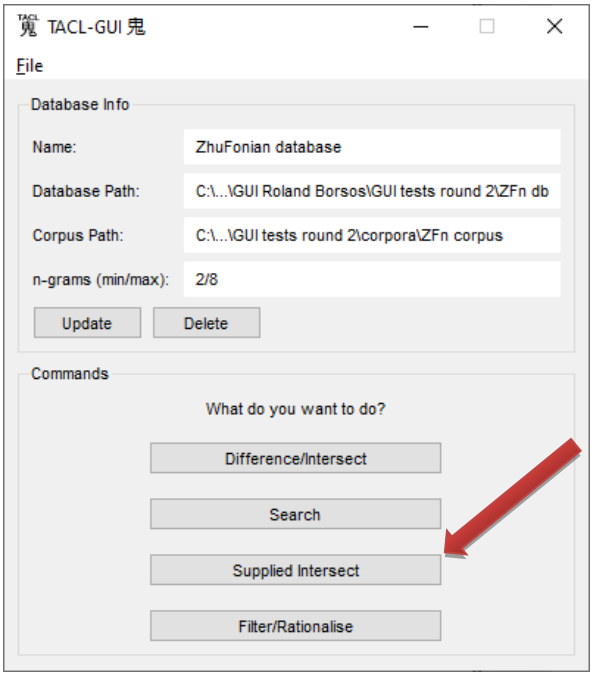
```
T0309 ZhuFonian-other
T0384 ZhuFonian-other
T0385 ZhuFonian-other
T0656 ZhuFonian-other
T1428 ZhuFonian-other
T1464 ZhuFonian-other
T1505 ZhuFonian-other
T1543 ZhuFonian-other
T1549 ZhuFonian-other
T2045 ZhuFonian-other
```

有了这些材料，我们就可以运行交叉运算（intersect）测试：

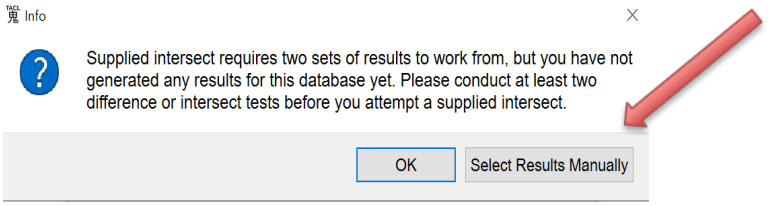



最后，我们使用“结果交叉运算”（Supplied Intersect）来比较前两次测试得出的两个原始结果文件。这一步可以结合前两个操作的结果，找到（a）同时存在于《增一阿含經》和竺佛念作品中；但（b）在《中阿含經》中没有的n-gram。

要运行这个测试，你需要先进入主选单界面，点击“结果交叉运算”（Supplied Intersect）：



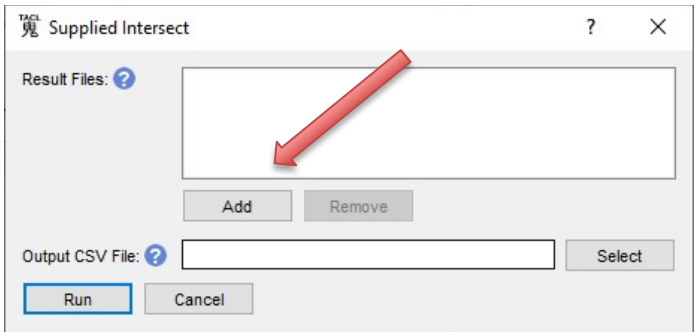

点击此按钮后，如果你之前没有从你选择的数据库中生成至少两个结果文件，你可能会看到以下提示信息：



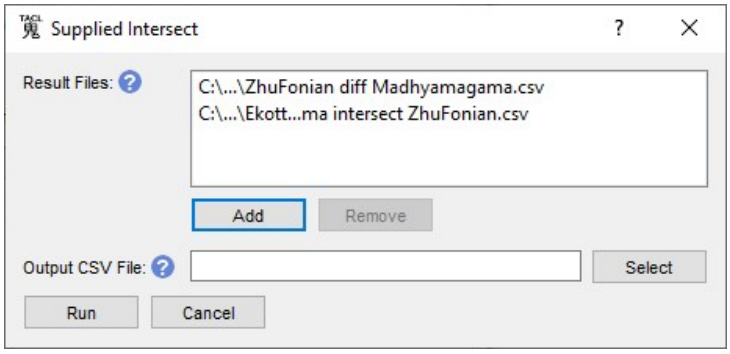

然而，如果你已经有了至少两套来自其他数据库的结果，就可以继续下一步，只需点击“手动选择结果”（Select Results Manually）。

或者，如果你确实事先没有准备任何结果文件，则点击“OK”按钮，GUI将引导你直接进入“差异运算/交叉运算”（Difference/Intersect）窗口，然后你可以先进行差异运算/交叉运算测试，以产生原始结果，然后提供给“结果交叉运算”（Supplied Intersect）操作。

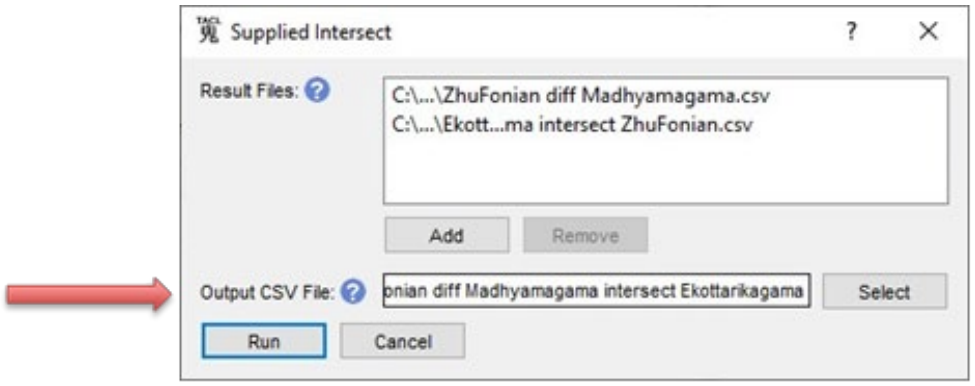
假设你已经通过上述任何一种方法准备好了原始结果文件，请进入“结果交叉运算”（Supplied Intersect）窗口中，点击“添加”（Add），以选择你打算进行交叉运算的结果文件。




然后导航到存储结果文件的文件夹，点击原始的.csv文件，或者键入路径和文件名：




然后点击“输出CSV文件”（Output CSV File）栏旁的“选择”（Select）按钮，输入结果文件的路径和文件名（通过导航或键入）。




最后，点击“运行”（Run）按钮。

总结一下此例中“结果交叉运算”（Supplied Intersect）的操作顺序，有以下三个步骤：

1. 对竺佛念语料库与《中阿含經》做不对称差异运算（asymmetric difference）
2. 对《增一阿含經》与竺佛念语料库做交叉运算（intersect）
3. 对（1）和（2）得到的两个结果进行“结果交叉运算”（Supplied Intersect）

与其他测试的结果一样，从“结果交叉运算”中得到的原始结果最后也应该用“筛选/合理化”（Filter/Rationalize）操作来处理，然后导入Excel进行人工语文学分析（见下文“筛选/合理化”[Filter/Rationalize]第64页，和“处理结果”第73页）。

这个例子（“例1”）其实是一个实际的佛教文献研究问题。《增一阿含經》和《中阿含經》之间的差异运算测试是Radich and Anālayo (2017)的研究工作的基础，它表明《增一阿含經》和《中阿含經》在翻译各自所基于的印度原本时，系统性地使用不同的方式来呈现相同的重复文本元素，因此，《增一阿含經》和《中阿含經》出自同一译者——僧伽提婆（Saṅghadeva）——的可能性极小。Radich随后发表了第二篇文章(2017a)，其中的研究结果与我们前面的“结果交叉运算”（Supplied intersect）所产生的结果相似，表明许多在《增一阿含經》中反复出现的文体特征与其他一些竺佛念所译核心文本之特征相同，而在《中阿含經》/僧伽提婆中从未出现。这些发现再结合外部证据，几乎可以肯定，现存的《增一阿含經》T125是竺佛念（和他的合作者）的作品，而非僧伽提婆（Saṅghadeva）所译。

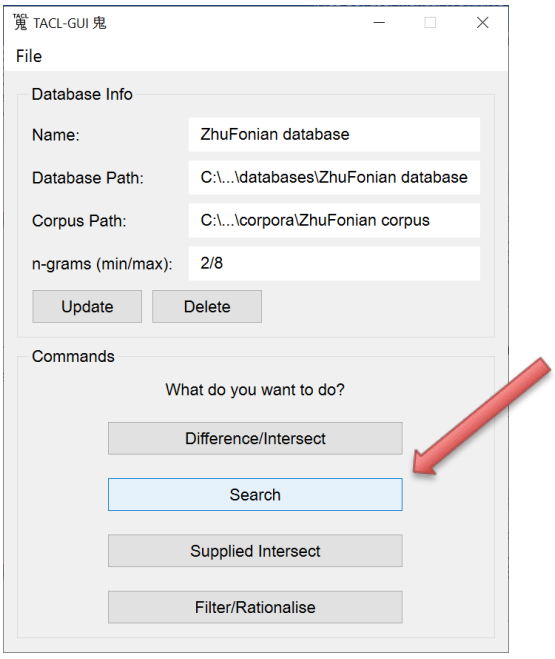
# 搜索（Search）

TACL的搜索（Search）功能需要用户提供一个n-gram列表和一个目录（catalogue）文件，目录（catalogue）文件中用“标签”（label）把作品分为不同的文本组，搜索（Search）功能可以找到并输出每个n-gram在每个作品中出现的次数。

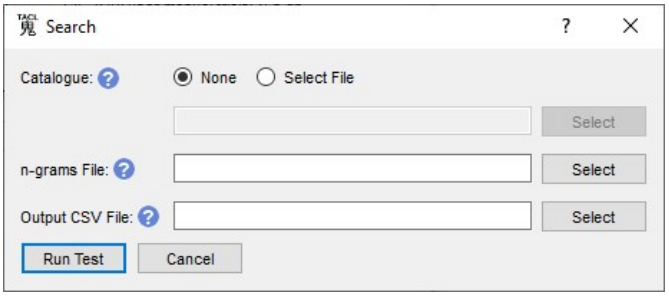
搜索功能要求用户输入一个n-gram列表文件。因此，你需要首先准备好这个文件，其中列出你要搜索的n-gram。该文件的格式为每行一个n-gram，以纯文本（.txt）格式保存。作为例子，我们将使用一些特征性的n-gram来区分竺佛念的风格(包括《增一阿含經》)和由《中阿含經》代表的僧伽提婆风格。下边是一个n-gram样本文件（缩减版）（见下载的“入门套件”Starter Kit中的“ZhuFonian 10 ngrams.txt”）：

一時佛在
在一面坐
村落
閻浮提
釋迦文
苦出要諦
四意止
阿闍世
典兵寶
在閑靜

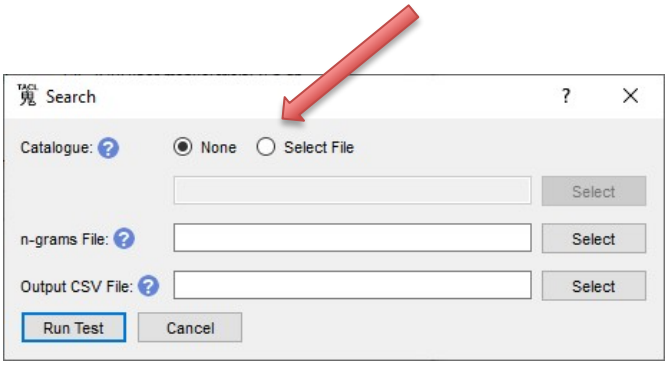
只要有一个语料库和相应的数据库，就可以运行TACL搜索了。在TACL GUI主选单窗口中，单击“搜索”（Search）按钮：



然后你会看到这个窗口：

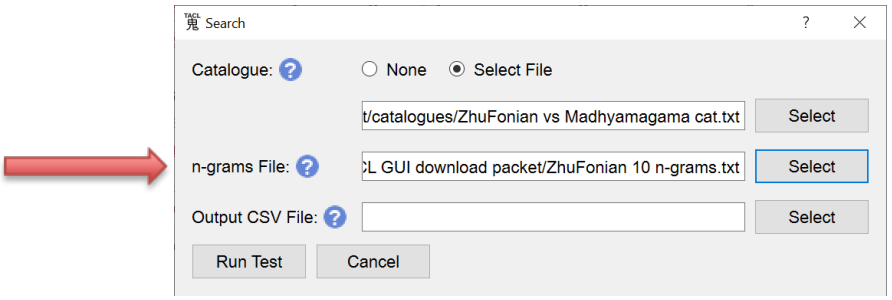



目录（catalogue）文件是可选的。这意味着用户可以提供一个目录文件，将感兴趣的作品分类到不同的标签（组）中。然后，TACL搜索将根据这些标签对找到的n-gram进行相应的分组。如果您想使用目录文件，请单击“选择文件”（Select File）选项。

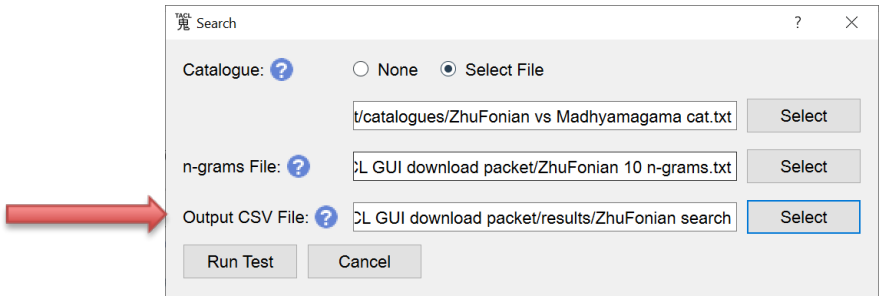



然后按“选择”（Select）按钮导航到文件夹或直接键入，给出目录文件的路径。这个练习中，我们将使用“ZhuFonian vs Madhyamagama cat.txt”（在“入门套件”Starter Kit的“catalogue”文件夹中），它把文本分为两组：（a）整个竺佛念语料库，包括T125；和（b）《中阿含經》。

如前所述，TACL搜索需要一个n-gram的列表。因此，用户必须选择一个n-gram列表文件。这意味着，在“n-grams File”栏中，你需要将GUI指向你已经准备好的n-gram列表文件“ZhuFonian 10 n-grams.txt”。



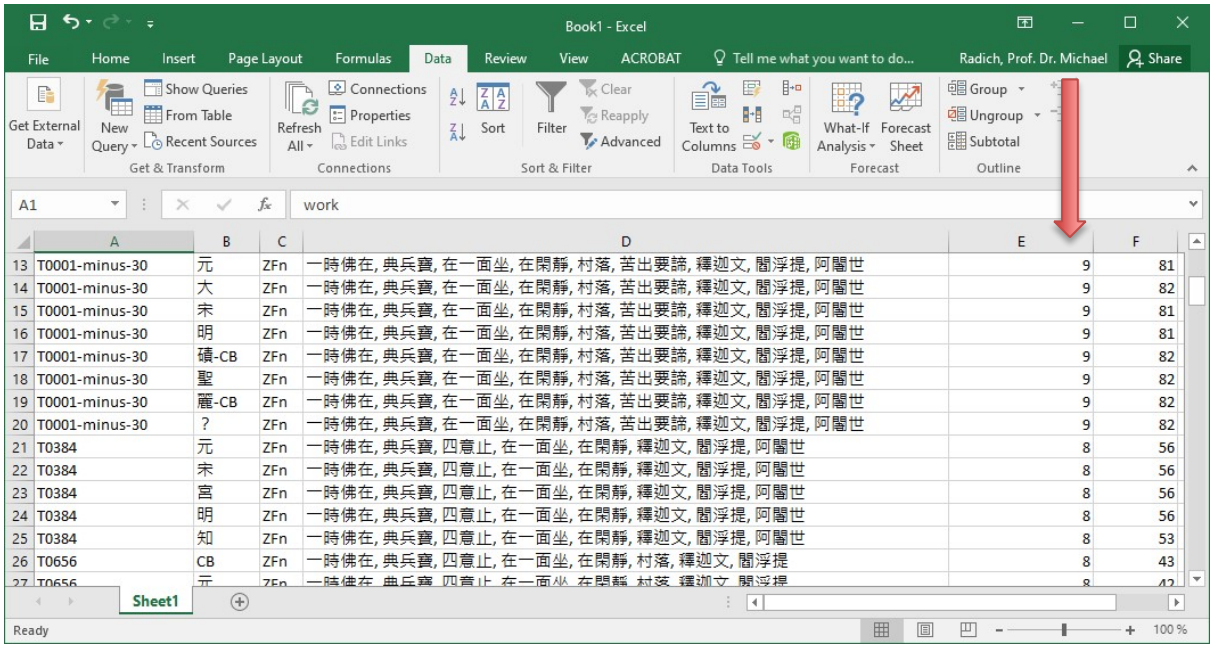

最后，在“输出CSV文件”（Output CSV File）一栏，通过导航和/或键入要输出的CSV文件的路径和文件名。




然后点击“运行测试”（Run Test）。

请注意，这里我们不用“筛选/合理化”（Filter/Rationalize）来处理TACL搜索的结果。TACL搜索产生的结果文件与其他结果文件的格式不同，不可以通过“筛选/合理化”来处理。如果你试图这样做，TACL GUI会报错。这一点可以从另一个角度理解——后台的默认程序已经用与“筛选/合理化”（Filter/Rationalize）类似的方式对TACL搜索的结果进行了整理。

如果正确完成了所有步骤，就可以将结果导入Excel，并根据n-gram的数量按降序排列，最后得到如此的结果：




| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 13 | T0001-minus-30 | 元 | ZFn | 一時佛在,典兵寶,在一面坐,在閑靜,村落,苦出要諦,釋迦文,閻浮提,阿闍世 | 9 | 81 |
| 14 | T0001-minus-30 | 大 | ZFn | 一時佛在,典兵寶,在一面坐,在閑靜,村落,苦出要諦,釋迦文,閻浮提,阿闍世 | 9 | 82 |
| 15 | T0001-minus-30 | 宋 | ZFn | 一時佛在,典兵寶,在一面坐,在閑靜,村落,苦出要諦,釋迦文,閻浮提,阿闍世 | 9 | 81 |
| 16 | T0001-minus-30 | 明 | ZFn | 一時佛在,典兵寶,在一面坐,在閑靜,村落,苦出要諦,釋迦文,閻浮提,阿闍世 | 9 | 81 |
| 17 | T0001-minus-30 | 磧-CB | ZFn | 一時佛在,典兵寶,在一面坐,在閑靜,村落,苦出要諦,釋迦文,閻浮提,阿闍世 | 9 | 82 |
| 18 | T0001-minus-30 | 聖 | ZFn | 一時佛在,典兵寶,在一面坐,在閑靜,村落,苦出要諦,釋迦文,閻浮提,阿闍世 | 9 | 82 |
| 19 | T0001-minus-30 | 麗-CB | ZFn | 一時佛在,典兵寶,在一面坐,在閑靜,村落,苦出要諦,釋迦文,閻浮提,阿闍世 | 9 | 82 |
| 20 | T0001-minus-30 | ? | ZFn | 一時佛在,典兵寶,在一面坐,在閑靜,村落,苦出要諦,釋迦文,閻浮提,阿闍世 | 9 | 82 |
| 21 | T0384 | 元 | ZFn | 一時佛在,典兵寶,四意止,在一面坐,在閑靜,釋迦文,閻浮提,阿闍世 | 8 | 56 |
| 22 | T0384 | 宋 | ZFn | 一時佛在,典兵寶,四意止,在一面坐,在閑靜,釋迦文,閻浮提,阿闍世 | 8 | 56 |
| 23 | T0384 | 宮 | ZFn | 一時佛在,典兵寶,四意止,在一面坐,在閑靜,釋迦文,閻浮提,阿闍世 | 8 | 56 |
| 24 | T0384 | 明 | ZFn | 一時佛在,典兵寶,四意止,在一面坐,在閑靜,釋迦文,閻浮提,阿闍世 | 8 | 56 |
| 25 | T0384 | 知 | ZFn | 一時佛在,典兵寶,四意止,在一面坐,在閑靜,釋迦文,閻浮提,阿闍世 | 8 | 53 |
| 26 | T0656 | CB | ZFn | 一時佛在,典兵寶,四意止,在一面坐,在閑靜,村落,釋迦文,閻浮提 | 8 | 43 |

这表明我们要找的n-gram中有9个在《長阿含經》T1中找到；8个在T384中找到；以此类推。相比之下，我们清单上的n-gram没有一个在《中阿含經》中出现，因此就没有出现在搜索结果中。读者若是对怎样使用这样的结果作为文本归属问题的证据感兴趣，可以将这些结果与这篇论文中的表格相比较：Radich and Anālayo (2017): 228-229。

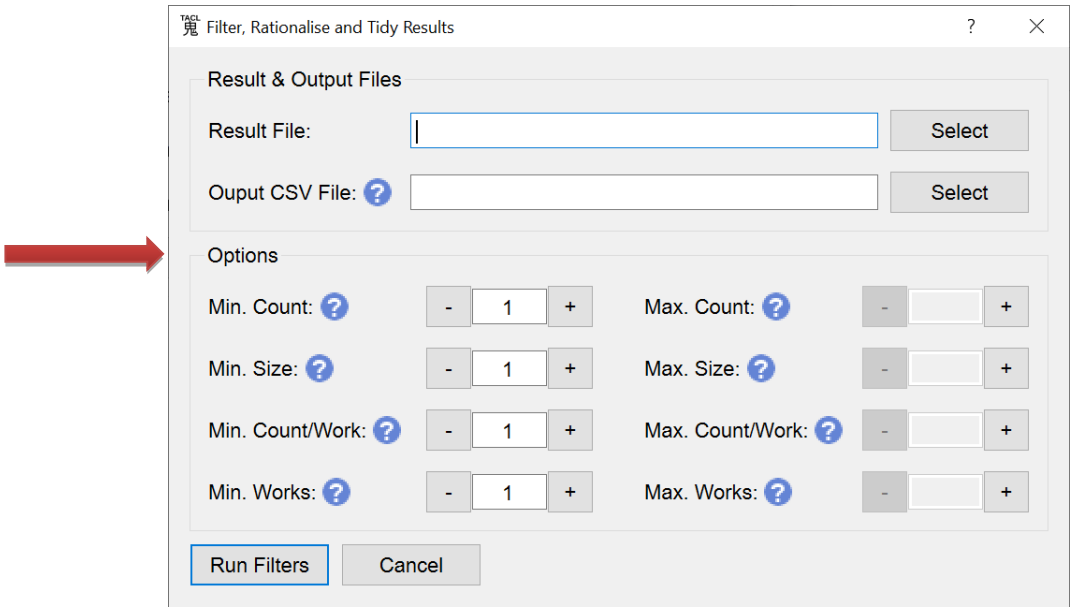
正如《TACL方法指南》（TACL Methods Guide）中进一步描述的那样，当我们有了一个可以反映某个译者或翻译团队的风格特征的n-gram列表，就可以用它在已知的该译者/团队的语料库之外进行尽可能大范围的广泛搜索，来找到这些特征性n-gram集中的文本——也就是说，当用来寻找可能是已知人物的未知作品的文本时，TACL的搜索功能是一个特别强大的工具。

## 工作步骤六：筛选/合理化（Filter/Rationalize）

前面已经多次提到，对于除TACL搜索（search）操作的结果之外的所有结果，我们建议用户将原始结果通过“筛选/合理化”（Filter/Rationalize）进行处理，作为人工语文学分析之前的最后一个步骤。这一步必须留到最后，因为筛选后的结果的格式已经改变，如果再进行结果交叉运算（supplied intersect）等测试会产生错误的结果。

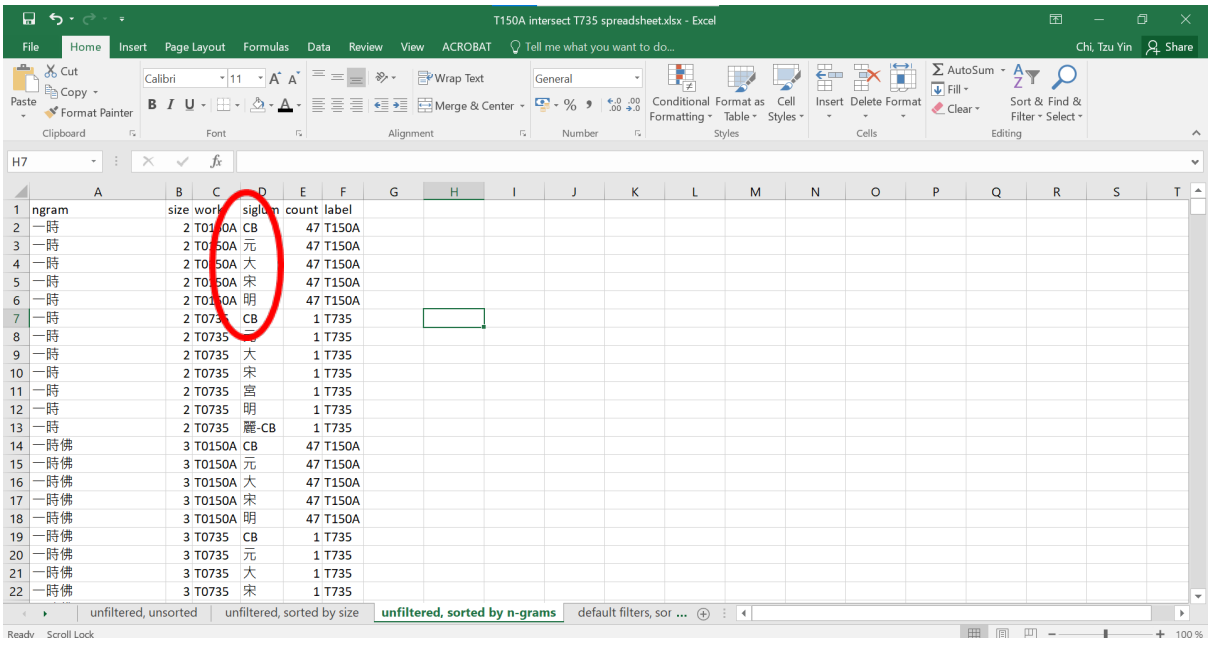
从用户的角度看，可以把TACL的筛选/合理化（Filter/Rationalize）的功能分为两组。首先，有一组“整理”操作已经被内置到图形用户界面GUI中，并在后台自动实现，用户无需做任何事

情。第二，还有一组参数，用户可以在筛选/合理化窗口看到并加以调整。

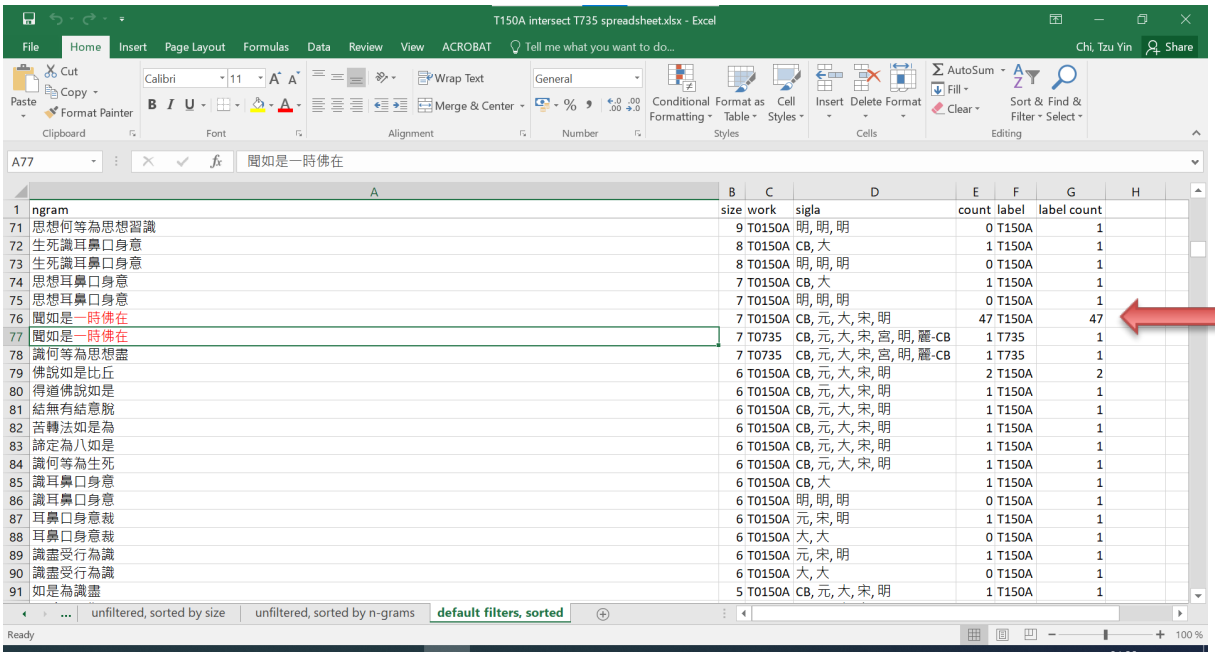



首先我们描述一下后台的“整理”功能，这些功能是内置于GUI的。因为差异运算（Difference）和交叉运算（Intersect）二者的目的不同，所以后台的“整理”操作对它们的作用也不同。交叉运算的结果要经过四种整理操作，而差异运算的结果只需经过这四种操作中的两种。为了演示一下所有这四种操作，我们将使用一个交叉运算的测试作为例子：T150A和T735的交叉运算结果（“T150A intersect T735 spreadsheet.xlsx”在“入门套件”的“results”文件夹中；这就是我们前面提到过的“例2”的数据），包括未筛选和筛选后的结果。为了显示这些不同的操作所带来的差异，我们在电子表格的不同数据表上展示了经过和未经“筛选/合理化”处理的结果，分别是：没有排序的未经筛选的结果；未经筛选的结果按n-gram长度排序；未经筛选的结果按n-gram排序；最后是经过筛选的结果。下面，我们将引导你对这些结果进行比较。

无论是差异运算和交叉运算的结果，筛选/合理化操作（Filter/Rationalize）都会将所有具有相同n-gram和相同计数（count）的底本分组在表格的同一行中。（否则，原始结果中每个底本都会占有单独的一行，这意味着人类用户将看到同一个结果——给定作品中n-gram的出现次数——每个底本都单独出现一次，这样会造成结果表格比未处理过的长数倍。）在“T150A intersect T735 spreadsheet.xlsx”中，我们可以通过比较“未筛选，按n-gram排序“”的结果（在unfiltered, sorted by n-gram名称的数据表中）和筛选后的结果（在“default filters, sorted by size”数据表中）来看到这种整理操作的效果（见下面的截图）。我们的例子是“一时”这个词。在未经筛选的结果中，我们看到，对于测试中的作品（T150A），所有五个底本（CB=CBETA版本，元=元本，宋=宋本，明=明本，大=《大正藏》本）中，“一时”这个词都出现相同次数（47次）。但在未经筛选的结果中，同样的信息重复出现在五个不同的行中，每个底本一行。

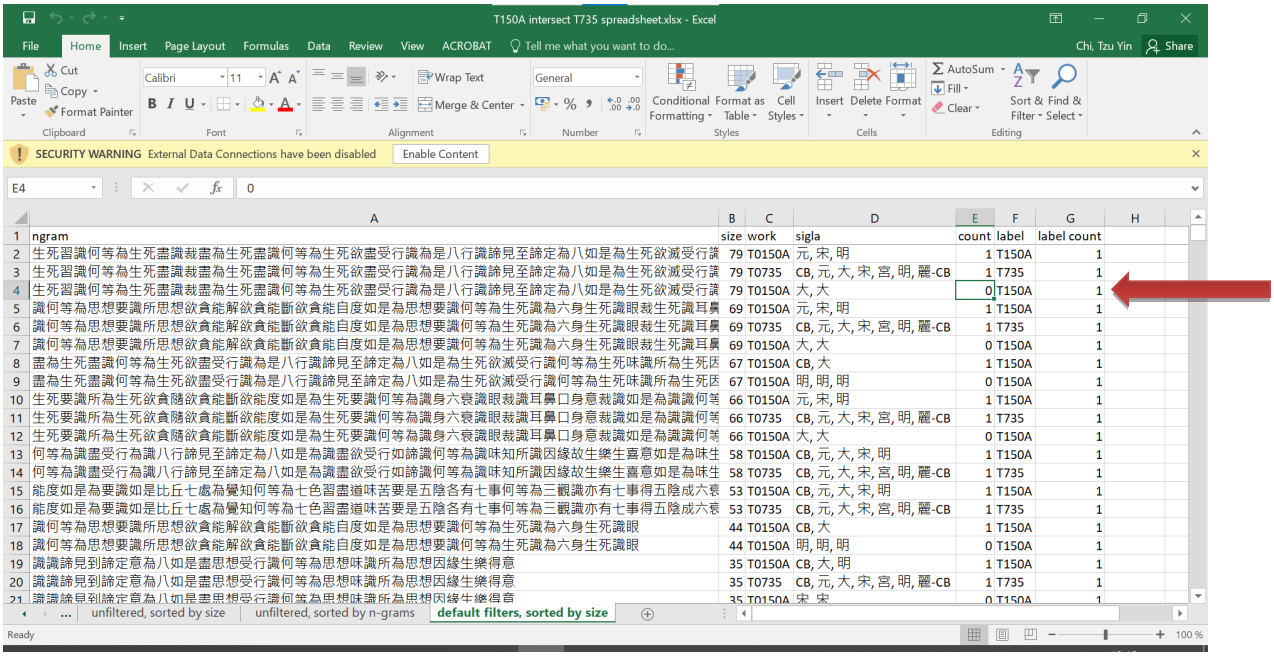



相比之下，在经过筛选的结果中（数据表为“default filters, sorted...”），总共只有两行包含“一时”，一个是T150A，一个是T735。每个文本的不同底本都被集中在一起，因为对于所有的底本来说，同一个n-gram有相同的计数。




类似地，对于所有的差异运算（Difference）和交叉运算（Intersect）结果，如果一个n-gram在某些底本没有出现，而在同一作品的其他底本中出现，筛选/合理化就会为那些根本没有出现的底本添加一个零计数。之所以这样处理，原因在于一个TACL数据库只包括那些在文本中出现的n-grams的计数。如果对原始结果不添加零计数，用户将不得不从结果数据中未出现的底本来推断出n-gram的缺失，这可能需要超人的观察力和注意力。在我们的例子中，如上文“交叉运

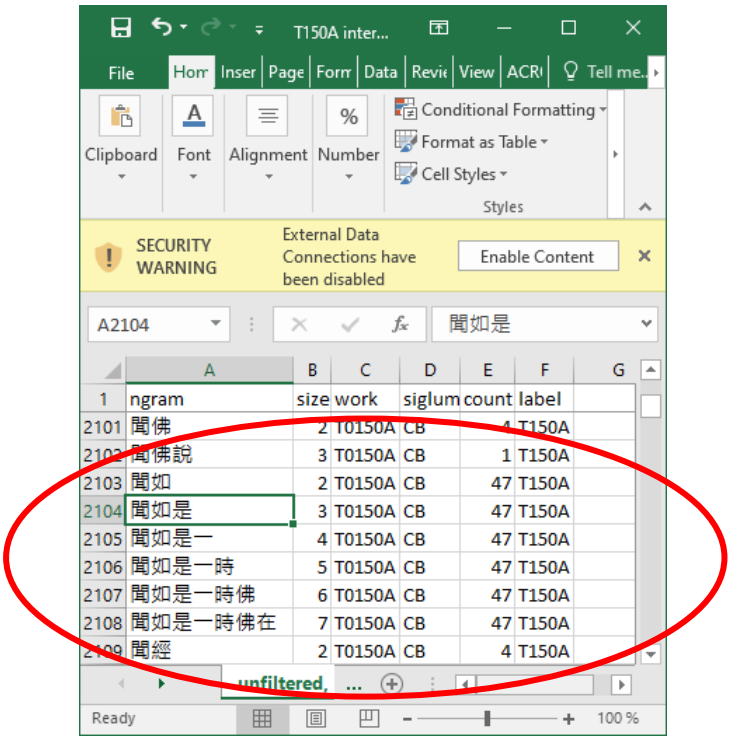
算”部分所述，T150A的《大正藏》底本中存在一个与其他底本不同之处，从而打断了T150A和T735之间的交叉运算测试所发现的长度为79的共同字符串。然而，由于增加了零计数，我们可以更容易地注意到《大正藏》本中缺少了这个79-gram，从而意识到其中文本的不同之处：




| | ngram | size | work | sigla | count | label | label count |
|---|---|---|---|---|---|---|---|
| 2 | 生死習識何等為生死盡識裁盡為生死盡識何等為生死欲盡受行識為是八行識諦見至諦定為八如是為生死欲滅受行識 | 79 | T0150A | 元, 宋, 明 | 1 | T150A | 1 |
| 3 | 生死習識何等為生死盡識裁盡為生死盡識何等為生死欲盡受行識為是八行識諦見至諦定為八如是為生死欲滅受行識 | 79 | T0735 | CB, 元, 大, 宋, 宮, 明, 麗-CB | 1 | T735 | 1 |
| 4 | 生死習識何等為生死盡識裁盡為生死盡識何等為生死欲盡受行識為是八行識諦見至諦定為八如是為生死欲滅受行識 | 79 | T0150A | 大, 大 | 0 | T150A | 1 |
| 5 | 識何等為思想要識所思想欲貪能解欲貪能斷欲貪能自度如是為思想要識何等為生死識為六身生死識眼裁生死識耳鼻 | 69 | T0150A | 元, 宋, 明 | 1 | T150A | 1 |
| 6 | 識何等為思想要識所思想欲貪能解欲貪能斷欲貪能自度如是為思想要識何等為生死識為六身生死識眼裁生死識耳鼻 | 69 | T0735 | CB, 元, 大, 宋, 宮, 明, 麗-CB | 1 | T735 | 1 |
| 7 | 識何等為思想要識所思想欲貪能解欲貪能斷欲貪能自度如是為思想要識何等為生死識為六身生死識眼裁生死識耳鼻 | 69 | T0150A | 大, 大 | 0 | T150A | 1 |
| 8 | 盡為生死盡識何等為生死欲盡受行識為是八行識諦見至諦定為八如是為生死欲滅受行識何等為生死味識所為生死因 | 67 | T0150A | CB, 大 | 1 | T150A | 1 |
| 9 | 盡為生死盡識何等為生死欲盡受行識為是八行識諦見至諦定為八如是為生死欲滅受行識何等為生死味識所為生死因 | 67 | T0150A | 明, 明, 明 | 0 | T150A | 1 |
| 10 | 生死要識所為生死欲貪隨欲貪能斷欲能度如是為生死要識何等為識身六衰識眼裁識耳鼻口身意裁識如是為識識何等 | 66 | T0150A | 元, 宋, 明 | 1 | T150A | 1 |
| 11 | 生死要識所為生死欲貪隨欲貪能斷欲能度如是為生死要識何等為識身六衰識眼裁識耳鼻口身意裁識如是為識識何等 | 66 | T0735 | CB, 元, 大, 宋, 宮, 明, 麗-CB | 1 | T735 | 1 |
| 12 | 生死要識所為生死欲貪隨欲貪能斷欲能度如是為生死要識何等為識身六衰識眼裁識耳鼻口身意裁識如是為識識何等 | 66 | T0150A | 大, 大 | 0 | T150A | 1 |
| 13 | 何等為識盡受行為識八行諦見至諦定為八如是為識盡欲受行如諦識何等為識味知所識因緣故生樂生喜意如是為味生 | 58 | T0150A | CB, 元, 大, 宋, 明 | 1 | T150A | 1 |
| 14 | 何等為識盡受行為識八行諦見至諦定為八如是為識盡欲受行如諦識何等為識味知所識因緣故生樂生喜意如是為味生 | 58 | T0735 | CB, 元, 大, 宋, 宮, 明, 麗-CB | 1 | T735 | 1 |
| 15 | 能度如是為要識如是比丘七處為覺知何等為七色習盡道味苦要是五陰各有七事何等為三觀識亦有七事得五陰成六衰 | 53 | T0150A | CB, 元, 大, 宋, 明 | 1 | T150A | 1 |
| 16 | 能度如是為要識如是比丘七處為覺知何等為七色習盡道味苦要是五陰各有七事何等為三觀識亦有七事得五陰成六衰 | 53 | T0735 | CB, 元, 大, 宋, 宮, 明, 麗-CB | 1 | T735 | 1 |
| 17 | 識何等為思想要識所思想欲貪能解欲貪能斷欲貪能自度如是為思想要識何等為生死識為六身生死識眼 | 44 | T0150A | CB, 大 | 1 | T150A | 1 |
| 18 | 識何等為思想要識所思想欲貪能解欲貪能斷欲貪能自度如是為思想要識何等為生死識為六身生死識眼 | 44 | T0150A | 明, 明, 明 | 0 | T150A | 1 |
| 19 | 識識諦見到諦定意為八如是盡思想受行識何等為思想味識所為思想因緣生樂得意 | 35 | T0150A | CB, 大, 明 | 1 | T150A | 1 |
| 20 | 識識諦見到諦定意為八如是盡思想受行識何等為思想味識所為思想因緣生樂得意 | 35 | T0735 | CB, 元, 大, 宋, 宮, 明, 麗-CB | 1 | T735 | 1 |
| 21 | 識識諦見到諦定意為八如是盡思想受行識何等為思想味識所為思想因緣生樂得意 | 35 | T0150A | 宋, 宋 | 0 | T150A | 1 |

前面这两个“整理”操作——将所有具有相同计数（count）的底本分组，并在底本中缺少某一n-gram时增加明确的指示——对于差异运算（Difference）和交叉运算（Intersect）测试都是在后台实现的。另外，交叉运算的结果还需要进行两个额外的操作：首先，筛选/合理化会在后台将结果中发现的所有字符串“扩展”（extend）到最大长度。这里我们复习一下前面提到的知识——生成数据库时的一个参数是设定最大n-gram长度，我们推荐8-gram。“扩展”（extend）是这样一个功能，尽管数据库有这个n-gram长度限制，我们还是可以用它在交叉运算测试中找到非常长的匹配字符串，比如上面所举的长度为79的例子（见上面的截图）。由于差异运算（Difference）的主要功能是发现代表翻译风格的特征性字符串，根据定义，这些特征必须重复出现，所以通常对发现作品特有的长字符串没有帮助。出于这个原因，“扩展”（extend）操作只应用于交叉运算（Intersect），而不被应用于差异运算。

在进行“筛选/合理化”（Filter/Rationalize）时，交叉运算（Intersect）的结果还要经过另外一个整理操作。不可避免的是，刚才描述的“扩展”（extend）操作所产生的较长的字符串会包括许多较短的字符串，这些较短的字符串也会单独出现在原始结果文件中，因为它们也符合交叉运算（Intersect）结果的条件（同时出现于被比较的两个文本中）。我们可以从下面这个未经筛选的结果的小片段中清楚地看到这一点：



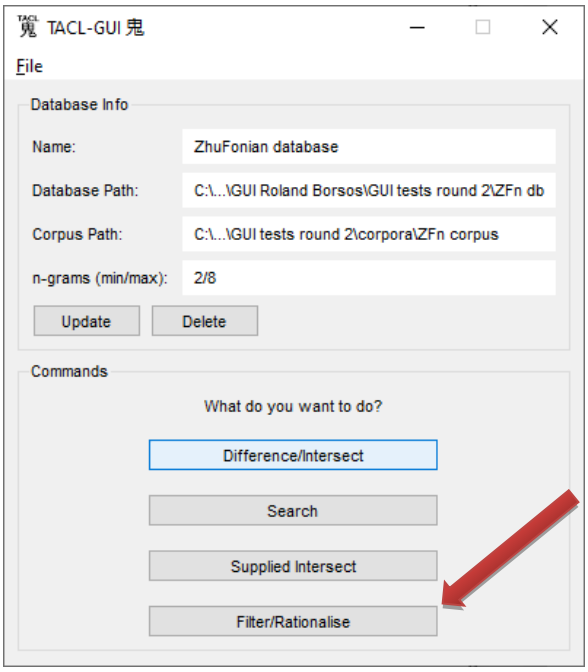

| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 1 | ngram | size | work | siglum | count | label | |
| 2101 | 聞佛 | 2 | T0150A | CB | 4 | T150A | |
| 2102 | 聞佛說 | 3 | T0150A | CB | 1 | T150A | |
| 2103 | 聞如 | 2 | T0150A | CB | 47 | T150A | |
| 2104 | 聞如是 | 3 | T0150A | CB | 47 | T150A | |
| 2105 | 聞如是一 | 4 | T0150A | CB | 47 | T150A | |
| 2106 | 聞如是一時 | 5 | T0150A | CB | 47 | T150A | |
| 2107 | 聞如是一時佛 | 6 | T0150A | CB | 47 | T150A | |
| 2108 | 聞如是一時佛在 | 7 | T0150A | CB | 47 | T150A | |
| 2109 | 聞經 | 2 | T0150A | CB | 4 | T150A | |

然而，所有这些包含在长字符串中的“子字符串”的计数都与长字符串的计数相同（这个例子中，“闻如”“闻如是”“闻如是一”“闻如是一时”“闻如是一时佛”“闻如是一时佛在”均出现47次）。因此，相比较的文本间关系的信息是在主要的、较长的字符串的结果中揭示的，而那些较短的子字符串并没有告诉人类用户任何有用的更多信息。因此，只要它们有相同的计数，“筛选/合理化”就会删除所有已经包含在长字符串中的较短字符串。由于这个原因，在“筛选”的结果中，不会再有单独的关于“一时”的结果（你可以搜索电子表格看看）。经过“扩展”（extend）操作后，所有冗余的字符串都被删除了，因此，所有包含“一時”的原始条目都被扩展并浓缩为“闻如是一時佛在的”两个条目（T150A和T735各一个）。

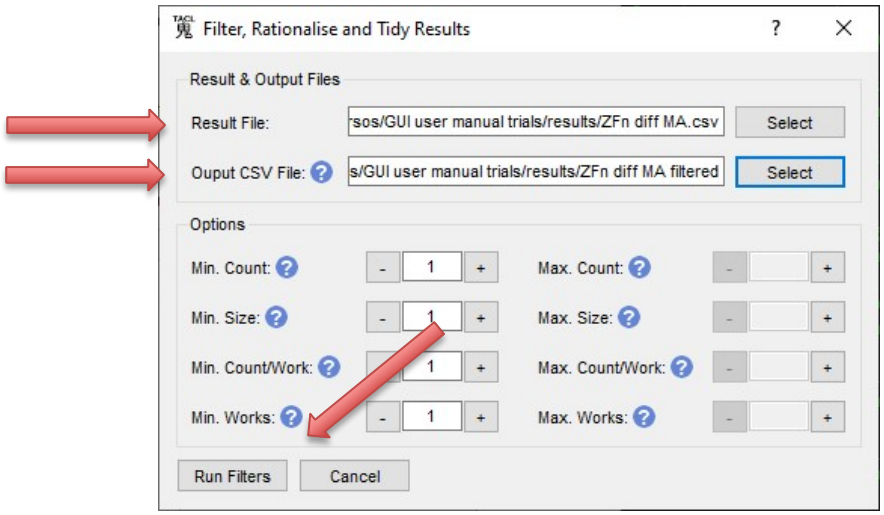
我们建议用户在导入结果和用Excel中查看之前，一定要把“差异运算”和“交叉运算”的结果在最后一步用“筛选/合理化”来处理一下，以发挥这些“整理”功能的优势。不然的话，结果会非常混乱，包含大量的冗余信息，需要更长的时间来处理。

然而，还要再提醒一次，把结果“筛选/合理化”（Filter/Rationalize）一定是人工分析之前对原始结果进行的最后一步处理。

掌握了这些基本概念后，实施最基本的“筛选/合理化”操作来“整理”结果其实很简单。在TACL GUI的主选单窗口，点击“筛选/合理化”（Filter/Rationalize）按钮：

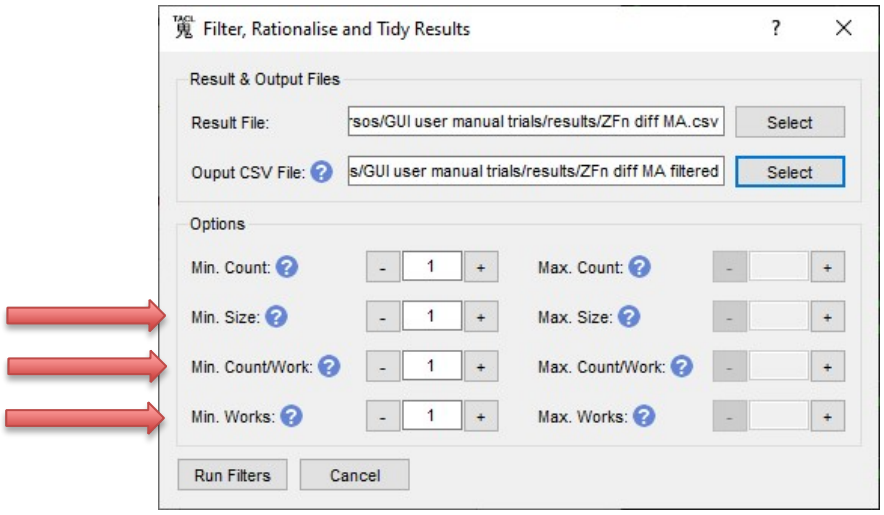



然后会看到这个窗口：




在这个例子中，如截图所示，在“结果文件”（Result File）框中，选择一个要处理的原始结果文件。在“输出CSV文件”（Output CSV File）框中，设定一个输出文件的位置和文件名。然后点击“运行筛选”（Run Filters）按钮。最后会得到一组结果，其中的冗余已被删除，具有相同结果的底本均已集中在表格的同一行中，n-gram的计数为零的底本信息也已加入，并且（如果您的结果是交叉运算结果），结果已被“扩展”（extend）以找出与搜索条件匹配的非常长的n-gram。

不过，除了这些使用内置的缺省参数的“整理”操作之外，从上面的截图已经可以猜到，“筛选/合理化”功能还允许用户将各种参数应用于筛选结果。




— 最低计数/最高计数 Min(imum) Count/Max(imum) Count：限定最低或最高计数，作为将n-gram包含在筛选结果中的条件。这里“计数”（count）指的是一个n-gram在一组由目录文件（catalogue）中的“标签”（label）所规定的文本中的出现次数。例如，选择Min. Count: 3和Max. Count: 7仅保留在结果中总共出现3到7次的n-gram（无论其在各个作品之间的分布如何）。

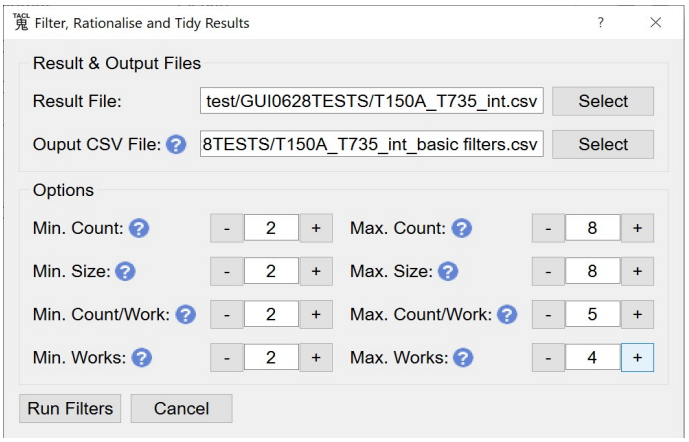
— 最小长度/最大长度Min. Size/Max. Size：限定保留在结果中的n-gram的长度范围。这里的“长度”（Size）指的是字符串的长度，也就是它包含的字符的个数。例如，选择Min. Size: 4，只保留4-gram和更长的字符串；选择Max. Size：6，只保留6-gram和更短的字符串。

— 最少计数/作品和最多计数/作品 Min. Count/Work and Max. Count/Work：限定n-gram在每个作品（work）中出现次数的范围。例如，选择Min Count/Work: 2，就仅保留在每个作品中至少出现2次的n-gram。

— 最少作品/最多作品Min. Works/Max. Works：限定n-gram在由一组由目录（catalogue）中的标签（label）标记的作品（work）中出现的作品个数，与每部作品中出现的数量无关。例如，选择Min. Works: 3，只保留至少出现于3部作品中的n-gram。

对大型文本或语料库的TACL测试经常会产生数万甚至百万行计的大量的原始结果，令人类分析者不知该从何下手，这时这些“筛选/合理化”选项就会很有用。事实上，这么大量的原始结果也会使Excel不堪重负甚至崩溃。在这种情况下，对结果进行筛选不仅是明智的，而且是必要的。进行筛选操作允许我们根据预先假设的标准，从最初的结果中分离出一些子集，以帮助我们更高效地找到需要的东西。

一个相当充分地利用了这些不同选项的“筛选/合理化”用户窗口可能看起来像这样：



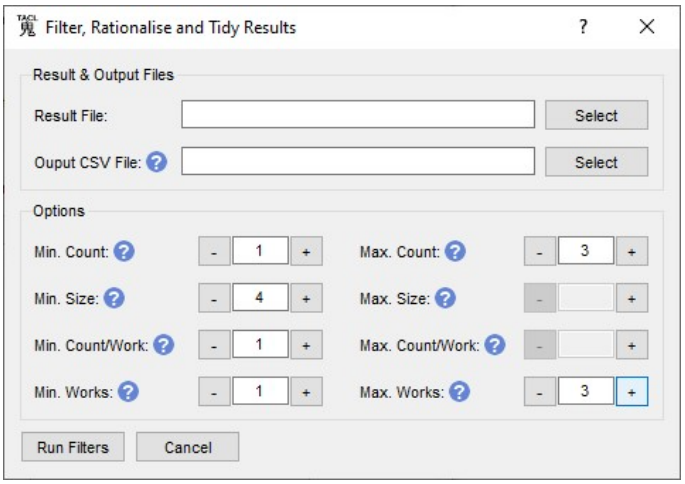

在筛选中使用什么样的参数取决于诸如研究者正在进行的测试的类型、所要寻找的东西，以及要处理的原始结果的数量等因素。根据我们的经验，这些参数更多要通过实践不断尝试总结出来。不过，如果说差异运算（difference）测试一般是用来找出文本风格的，交叉运算（intersect）测试一般是用来发现文本的内容来源的，那么就有可能为每种类型的测试推荐一些一般的准则——至少可以把这些参数当作进一步实验的起点。

1）交叉运算（Intersect）：

如果说交叉运算测试的主要目的是发现一个文本与另一个或多个文本之间的借用关系，一般来说，我们会对较长的字符串更感兴趣，这些字符串只出现在少数作品中。较短的字符串更可能是常见的词汇或短语；同时出现在许多作品中的字符串也是类似的情形。极限的情况是只有唯一的匹配，即一个字符串只在全部藏经中的两个文本中各出现一次（一个例子就是上面提到的76-gram，见第46页）。类似这样的字符串往往可以成为“铁证”，确切地证明一个文本借用了另一个文本。

基于这一原理，对于交叉运算（intersect）测试，我们建议先设置以下初始参数（然后如果发现它们得到的结果太多或太少，可以进行调整）。

— 最小长度 Min. size：4
— 最大计数 Max. count：3
— 最大作品计数 Max. works：3

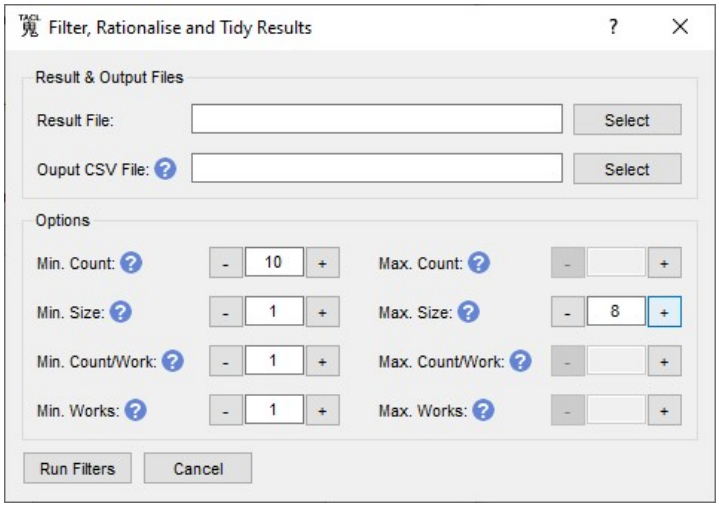



2)差异运算（Difference）：

与交叉运算不同，差异测试的主要目的是发现代表译者风格的特征性字符串。根据定义，我们视为风格证据的特征字符串会重复出现——也就是说，在其他条件相同的情况下，一个字符串在某一特定作者的作品中重复出现的次数越多，它作为该作者的特征性就越强烈。这种重复出现的字符串通常比较短(单独的词，或短的搭配和短语)。在这种情况下，较长的字符串和出现次数较少的字符串都不太可能成为有用的证据。因此，对于典型的差异运算（difference）测试，我们建议从以下参数开始：

— 最大长度 Max. size：6-8

— 最小计数 Min. count：10




然而，在实际应用中，取决于语料库的大小，最小计数（Min. count）需要随时调整。比如，在分析一个非常大的语料库时，如果最小计数为10，可能会有太多的结果，这时就需要加大最小计数的值；而当语料库较小时，则可能需要减小最小计数的值。

3）结果交叉运算（Supplied Intersect）

一般来说，结果交叉运算（Supplied Intersect）测试会将至少一个差异运算（Difference）测试的原始结果作为输入，而且这种测试通常与典型的差异运算测试的目的相同（发现作者或译者风格特征字符串）。由于这些原因，我们建议对结果交叉运算（Supplied Intersect）的筛选/合理化（Filter/Rationalize）设置与差异运算（Difference）大致相同的参数。

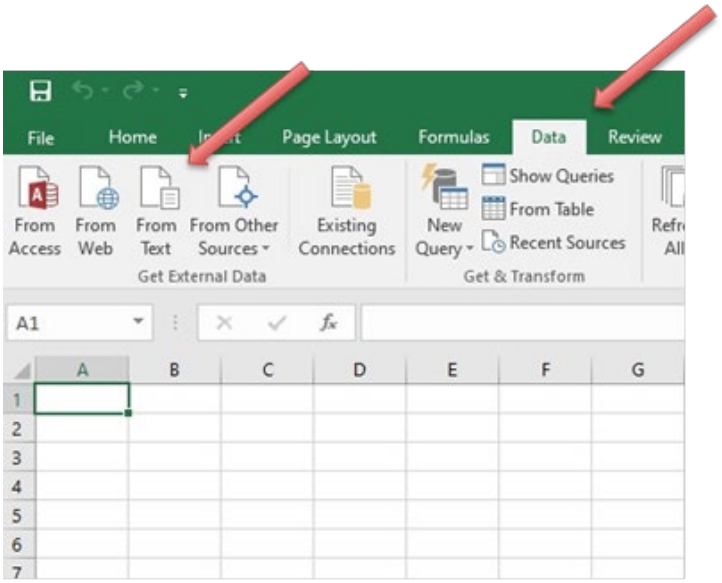
另请注意，把结果按大小和计数筛选可以与我们将在下面推荐的Excel排序规则结合使用。这意味着，在您开始人工分析之前，并非绝对有必要让您的筛选结果完全“正确”——您通常可以通过在Excel中仅查看一组已排序的原始结果的一部分来弥补结果数量太多的问题。还要注意，如果你发现最初的筛选设置产生了太多或太少的结果，你可以随时回到TACL GUI，运行一个新的筛选/合理化（Filter/Rationalize）操作，调整使之更加严格或更加宽松。

# 工作步骤七：处理结果(1)：在Excel（或类似的工具）中导入和排序

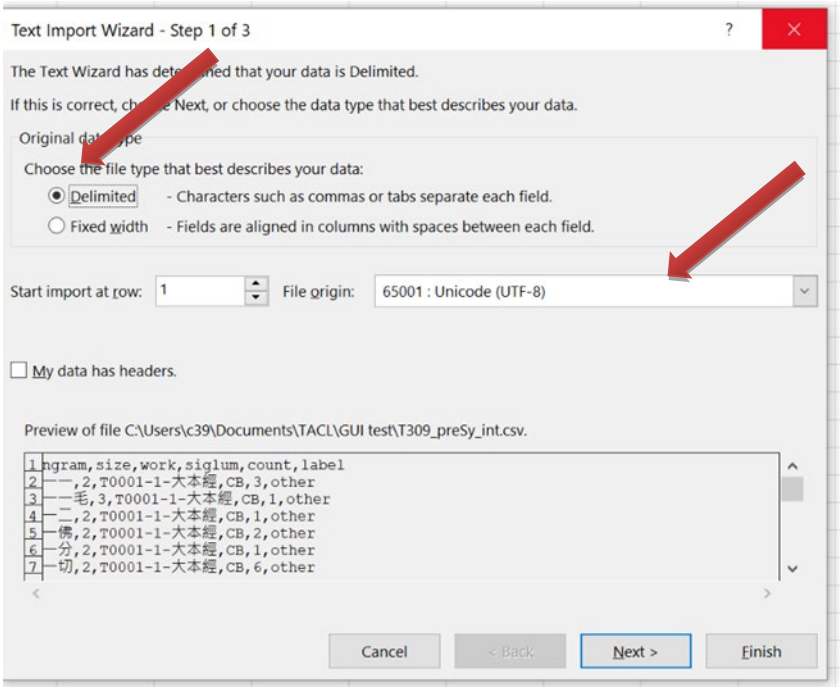
TACL GUI以.csv文件（“逗号分隔值”）的形式输出结果。处理.csv结果的一种简单方法是使用Excel（或类似的工具，例如LibreOffice或Apache OpenOffice的Calc，以及WPS。更高级的用户可能有自己处理.csv文件的方法，但我们将在此引导用户使用Excel处理结果。

请注意，我们这里所有的示例都是用Excel 2016准备的。如果你的Excel版本不同，你的屏幕布局也可能与我们描述和展示的略有不同。

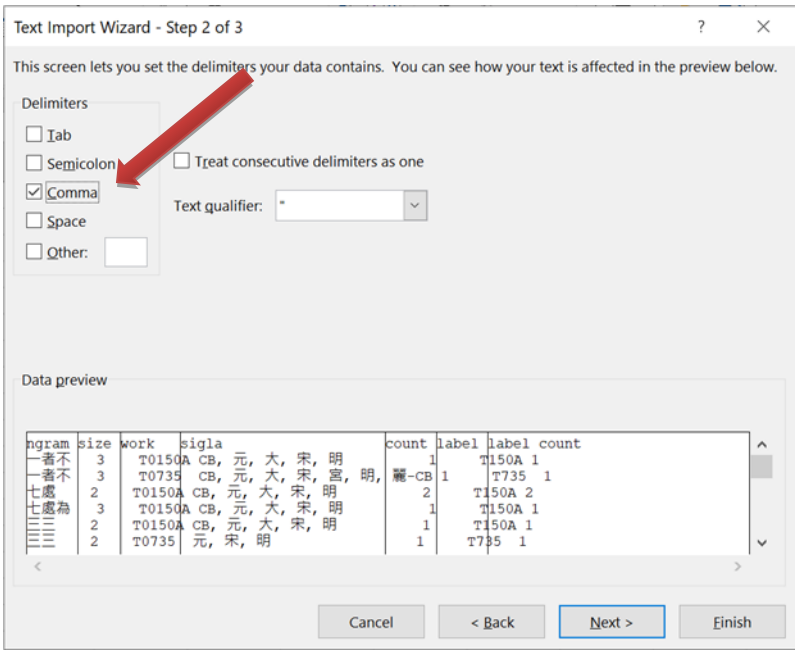
在Windows中，Excel被设置为打开.csv文件的默认应用程序。不幸的是，仅仅通过双击不可能正确打开一个.csv结果文件——该文件内容将显示为乱码。我们需要先打开Excel，转到页面顶部菜单栏中的“数据”（Data）选项卡，单击“来自文本”（From Text），然后导航到您的.csv文件。



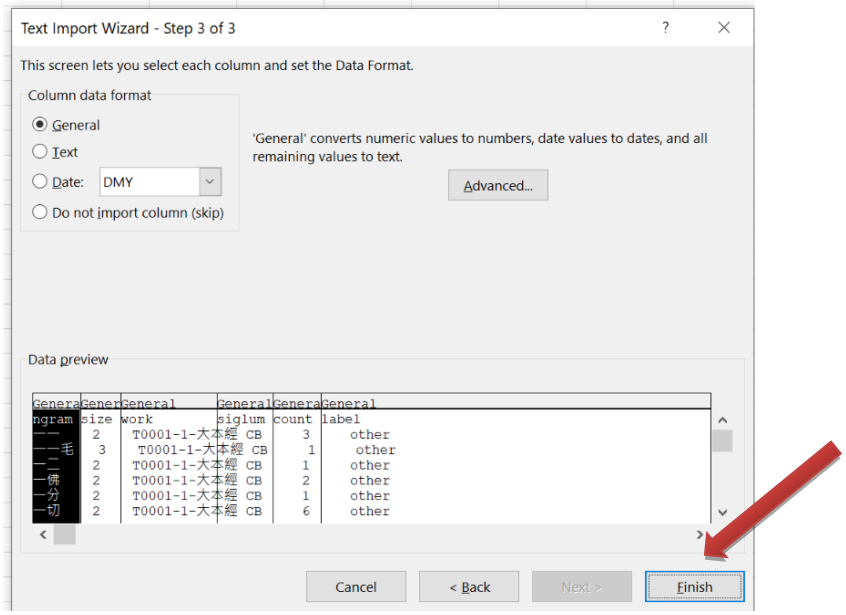

接下来，你将看到弹出“文本导入向导”（Text Import Wizard）的对话框：




在“原始数据类型”（original data type）下，选择“分隔”（Delimited）。对于“文件来源”（File origin），一定要选择“65001: Unicode (UTF-8)”，以确保正确显示汉字——否则，得到的会是乱码。然后点击“下一步”（Next）。

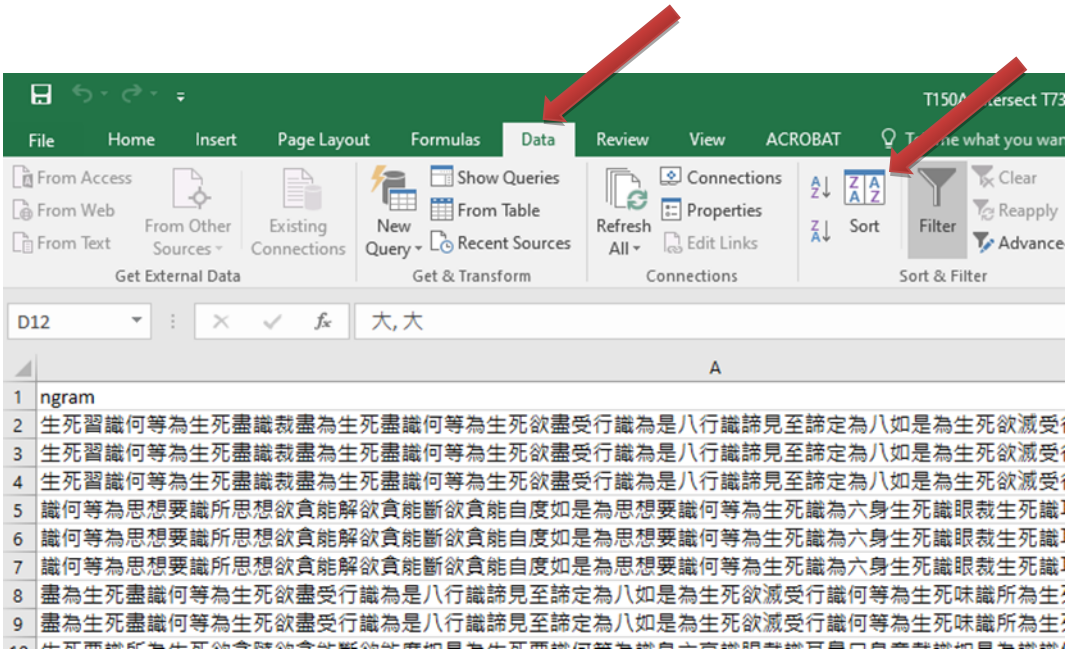



在“分隔符”（Delimiters）下，选择“逗号”（Comma）。然后点击“下一步”（Next）。

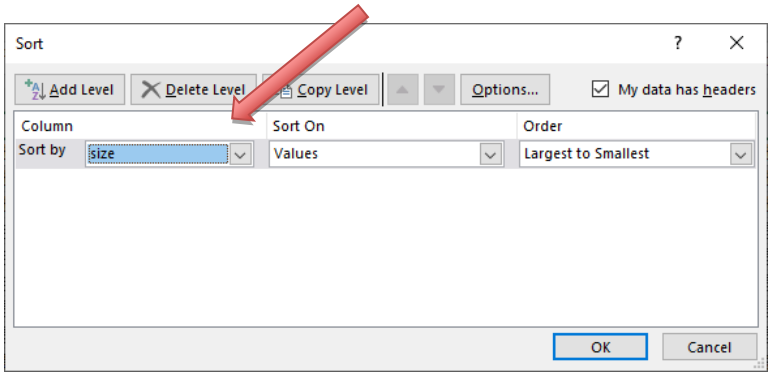



最后，只需单击“完成”（Finish）。

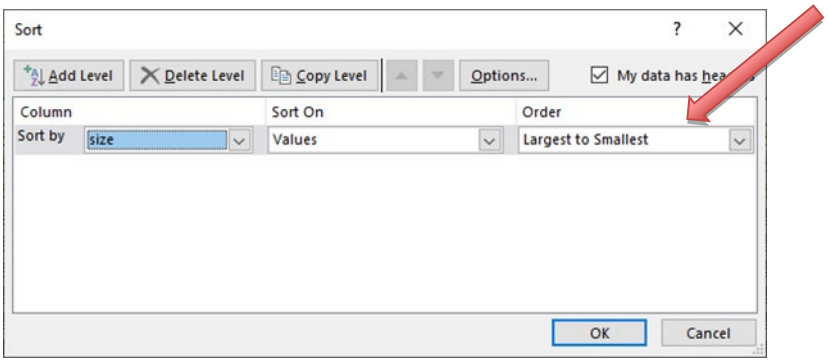
将结果导入Excel后，你可以利用“排序”（Sort）功能更加高效地分析结果。排序位于“数据”（Data）选项卡中：



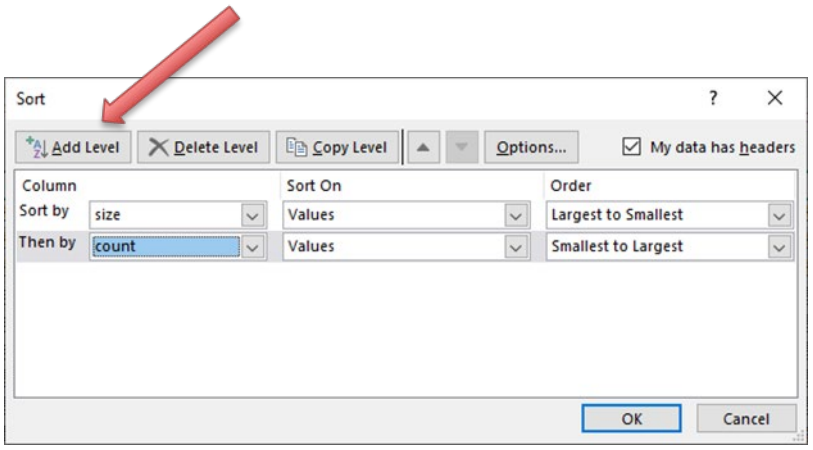

在“排序”（Sort）的设置中，你可以选择任何列标题（例如n-gram、作品work、长度size、计数count），将其指定为排序标准。



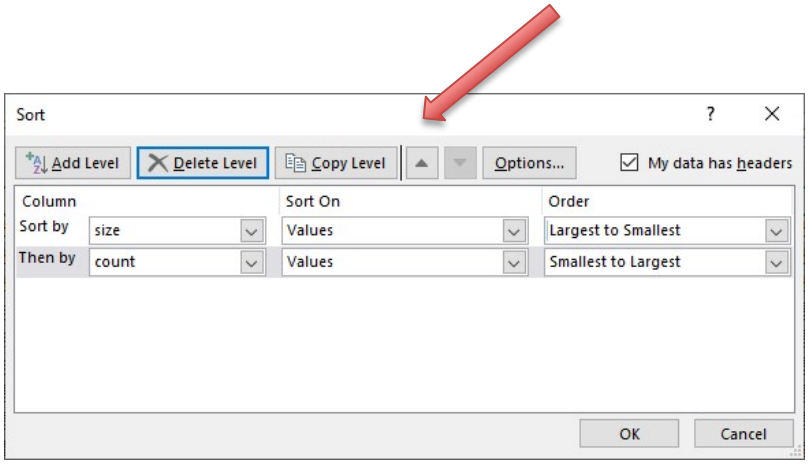

在右侧，你可以指定排序是按升序还是降序（或者，当数据是字符串时，按字母顺序向前或向后等）。




你还可以添加排序级别，这意味着你可先按首要条件排序，然后在按该排序的结果中，再按次要条件排序，依此类推。




如果你想调整级别，可以使用“向上”和“向下”箭头：



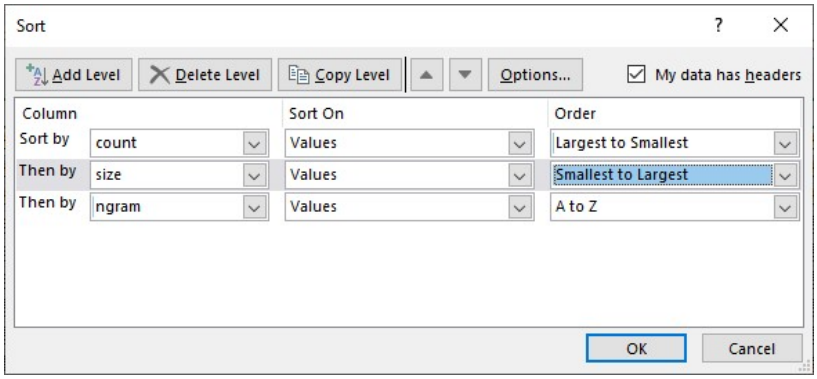

我们建议使用以下排序规则，具体取决于您正在分析的 TACL 测试结果的类型：

差异运算（Difference）：

— 按计数（count）降序（descending，从大到小）

— 按字符串长度（size）升序（ascending，从小到大）

— n-gram

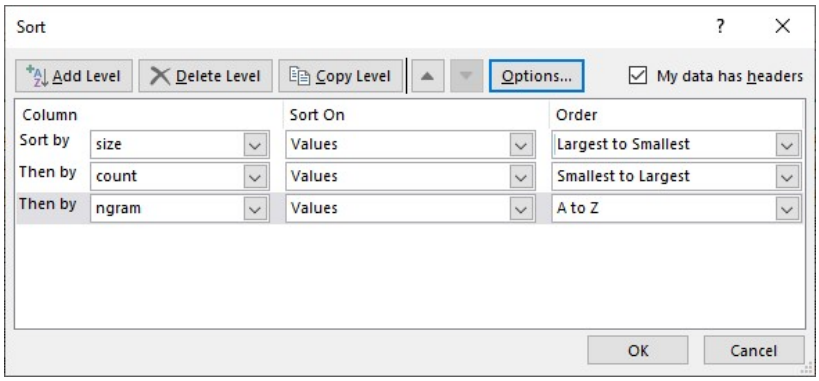



交叉运算（Intersect，包括结果交叉运算Supplied Intersect）：

— 按字符串长度（size）降序（descending，从大到小）

— 按计数（count）升序（ascending，从小到大）

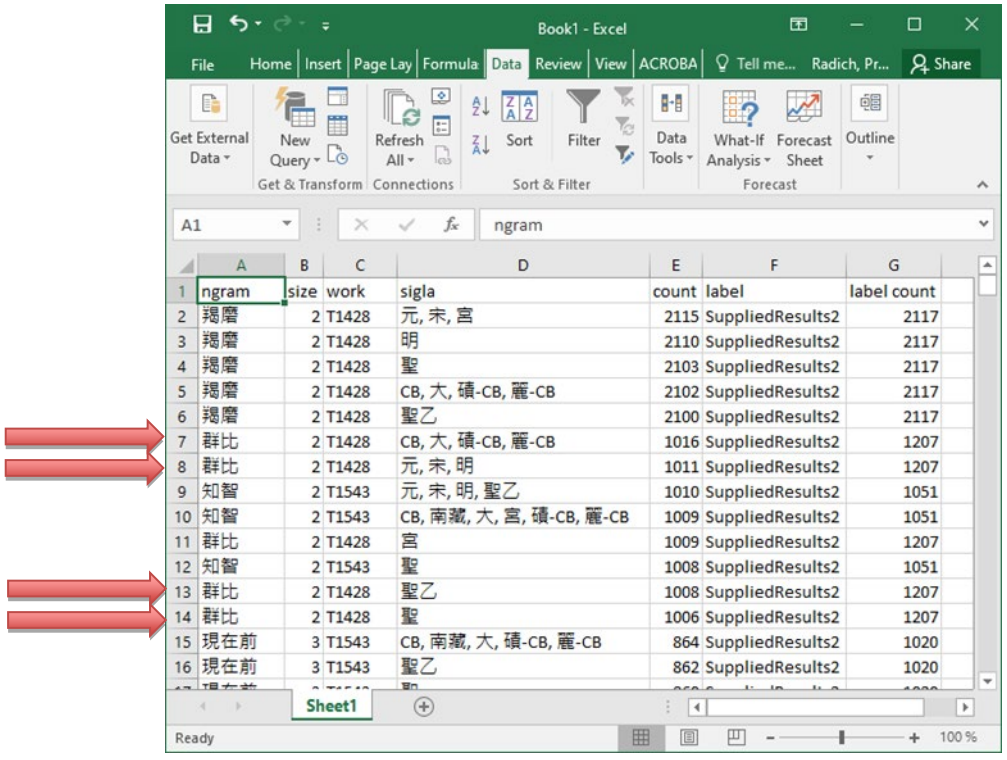
—— n-gram




搜索（Search）：按“n-gram数目”（number of n-grams）降序（descending，“从大到小”）排序。

最后，我们在这里再介绍一个Excel中的排序技巧，它可以提高我们浏览结果的效率，找到最可能有证据意义的字符串。

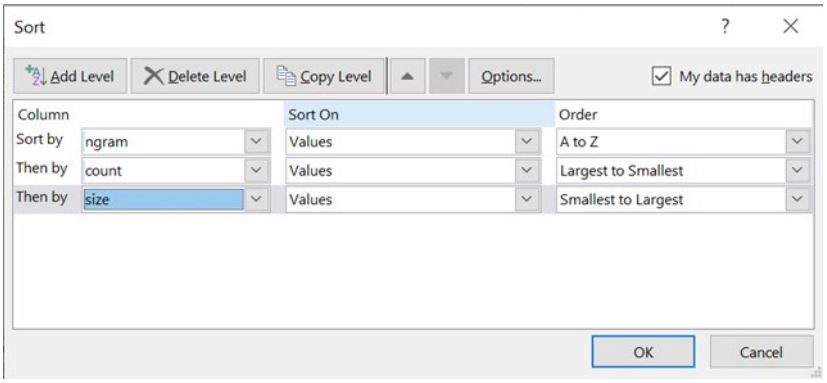
当使用上述的排序规则时，我们偶尔会遇到这样的问题：包含同一个n-gram的多个条目会分散在结果的不同部分。这个问题可以通过一个稍微复杂的排序规则来解决，该规则保留了上述的排序条件，但是仍然将同一n-gram的所有条目组合在一起。以“结果交叉运算”（Supplied Intersect）测试的结果（"Ekottarikagama intersect ZhuFonian diff Madhyamagama"，即“例1”）为例，如果简单地按n-gram计数（count）降序排序，会得到以下结果：



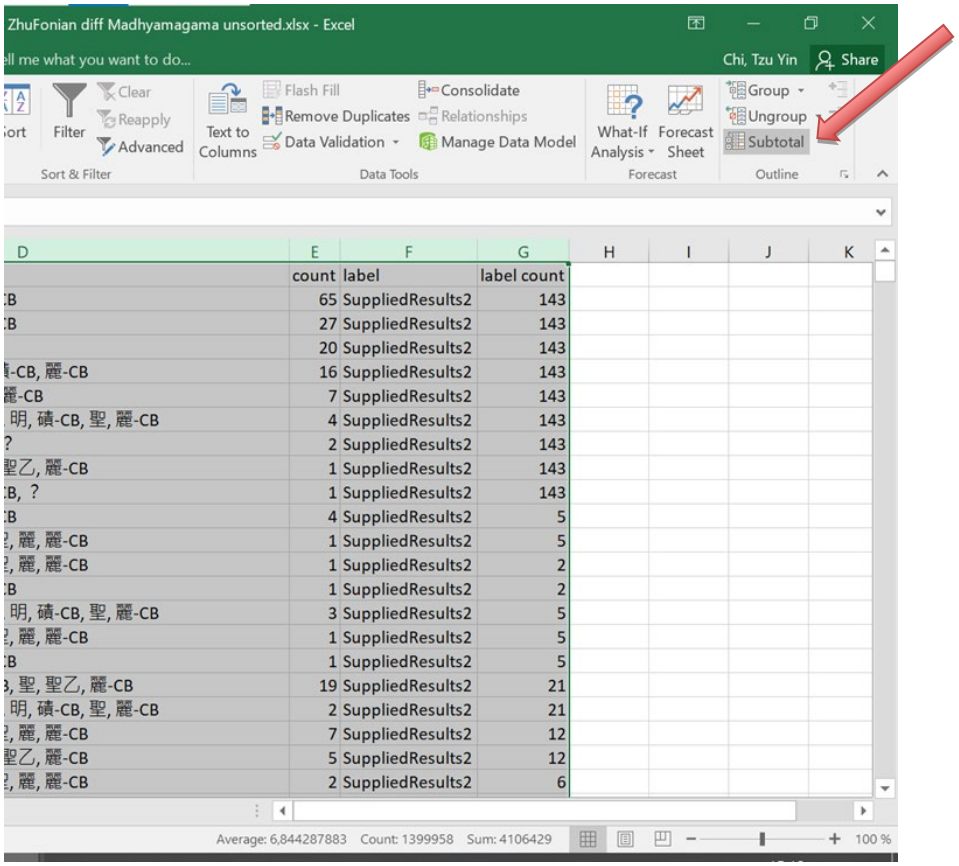

| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 1 | ngram | size | work | sigla | count | label | label count |
| 2 | 羯磨 | 2 | T1428 | 元, 宋, 宮 | 2115 | SuppliedResults2 | 2117 |
| 3 | 羯磨 | 2 | T1428 | 明 | 2110 | SuppliedResults2 | 2117 |
| 4 | 羯磨 | 2 | T1428 | 聖 | 2103 | SuppliedResults2 | 2117 |
| 5 | 羯磨 | 2 | T1428 | CB, 大, 磧-CB, 麗-CB | 2102 | SuppliedResults2 | 2117 |
| 6 | 羯磨 | 2 | T1428 | 聖乙 | 2100 | SuppliedResults2 | 2117 |
| 7 | 群比 | 2 | T1428 | CB, 大, 磧-CB, 麗-CB | 1016 | SuppliedResults2 | 1207 |
| 8 | 群比 | 2 | T1428 | 元, 宋, 明 | 1011 | SuppliedResults2 | 1207 |
| 9 | 知智 | 2 | T1543 | 元, 宋, 明, 聖乙 | 1010 | SuppliedResults2 | 1051 |
| 10 | 知智 | 2 | T1543 | CB, 南藏, 大, 宮, 磧-CB, 麗-CB | 1009 | SuppliedResults2 | 1051 |
| 11 | 群比 | 2 | T1428 | 宮 | 1009 | SuppliedResults2 | 1207 |
| 12 | 知智 | 2 | T1543 | 聖 | 1008 | SuppliedResults2 | 1051 |
| 13 | 群比 | 2 | T1428 | 聖乙 | 1008 | SuppliedResults2 | 1207 |
| 14 | 群比 | 2 | T1428 | 聖 | 1006 | SuppliedResults2 | 1207 |
| 15 | 現在前 | 3 | T1543 | CB, 南藏, 大, 磧-CB, 麗-CB | 864 | SuppliedResults2 | 1020 |
| 16 | 現在前 | 3 | T1543 | 聖乙 | 862 | SuppliedResults2 | 1020 |

“群比”的条目是彼此分开的，因为我们先按计数（count）排序，但这些条目没有相同的计数，而计数在同一范围内的其他n-gram的条目就会被插在它们中间（事实上，其他n-gram的条目也是分散的，但处于这个截图之外）。

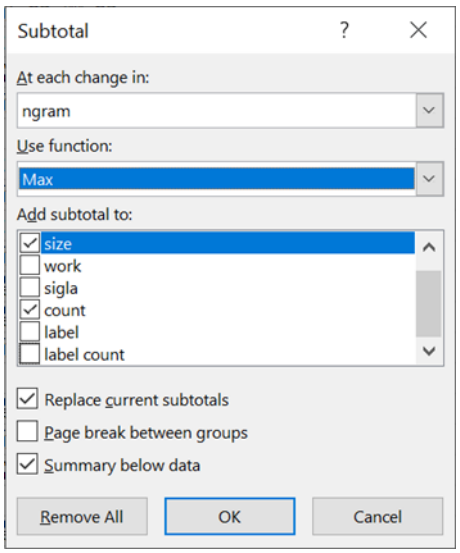
为了解决这个问题，我们建议采用以下排序规则：首先，按n-gram排序，然后（像通常的差异difference测试结果处理一样）按计数（count）降序（descending）排序，然后（可选）按长度（size）升序（ascending）排序。



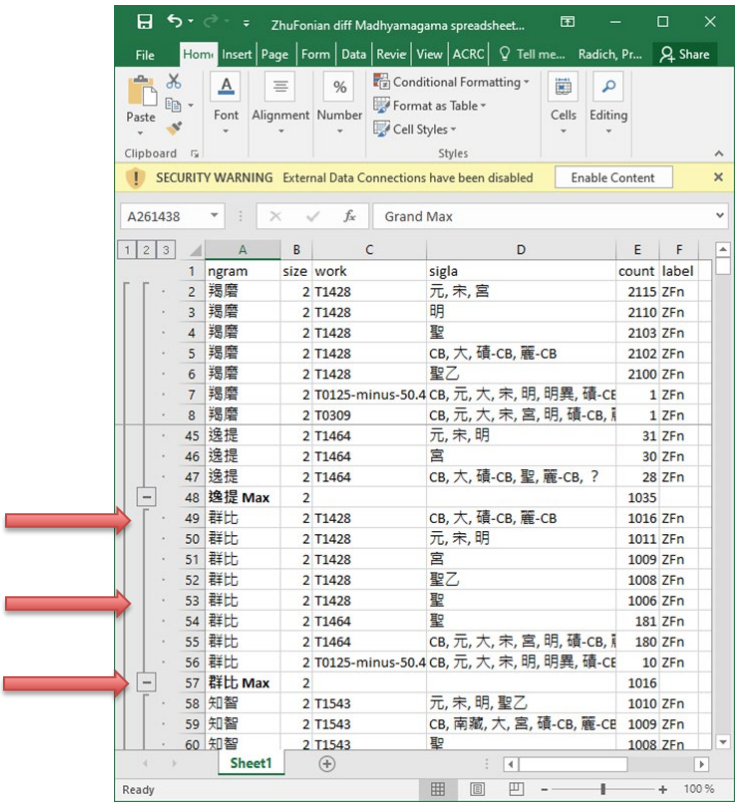

然后选择所有数据，进入“数据”（Data）选项卡，在工具栏的最右端，点击“小计”（Subtotal）。



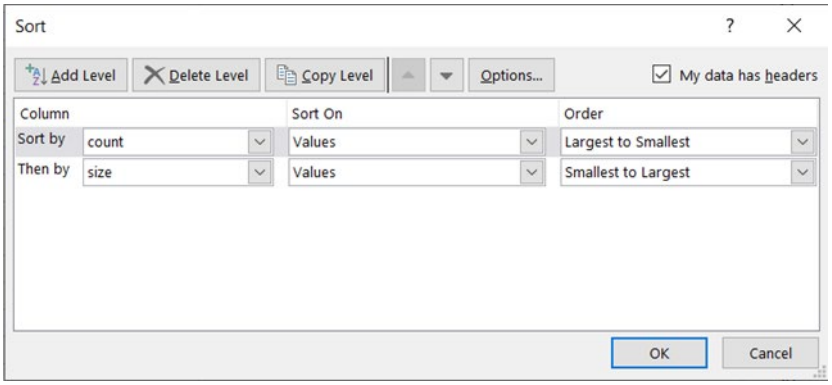
在“小计”（Subtotal）窗口中，“在每次变化时”（At each change in），选择“ngram”；在“使用函数”（Use function）中，根据你需要的分析证据优先次序，可以选择“取最大值”（Max）或“总和（Sum）”——“取最大值”按每个n-gram的最大计数顺序列出；“总和”按总计数顺序列出。然后，在“添加小计到（Add subtotal to）”下，选择“计数（count）”。保留默认勾选的“替换当前小计”（Replace current subtotals）和“摘要置于数据下面”（Summary below data）。点击“确定（OK）”。



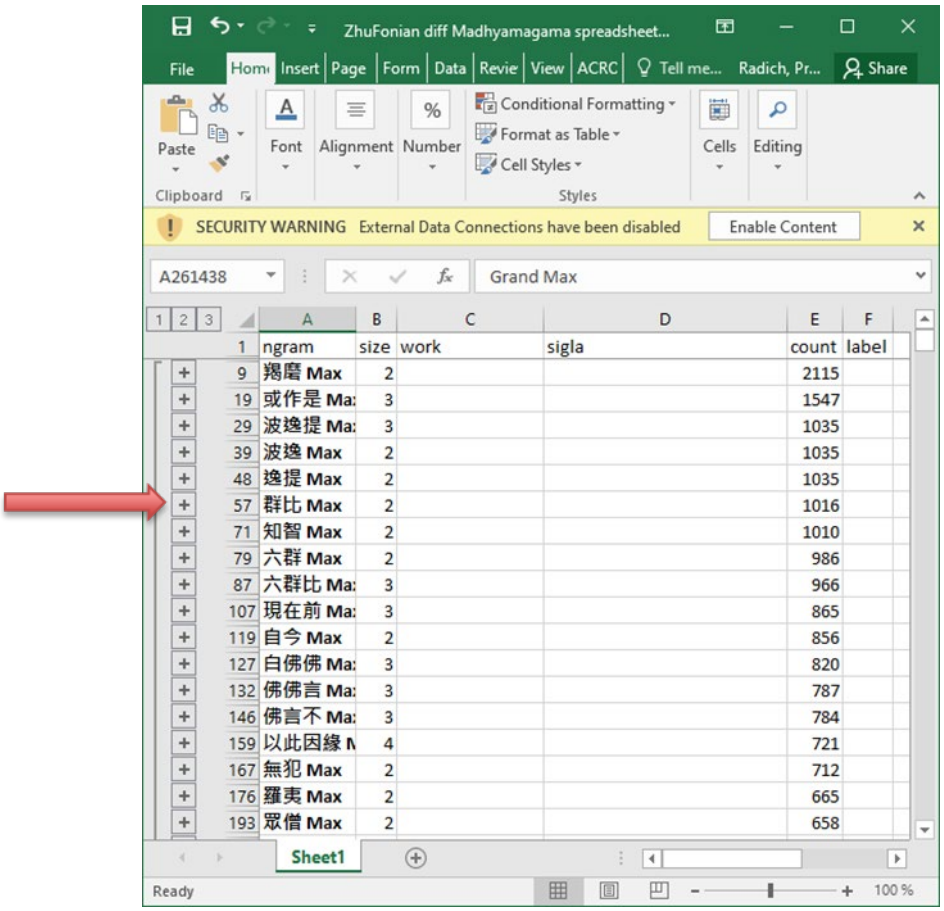

点击“确定”后，Excel开始重新进行排序。如果有大量的数据，排序可能需要一段时间。最终，当它完成时，结果应该是这样的：



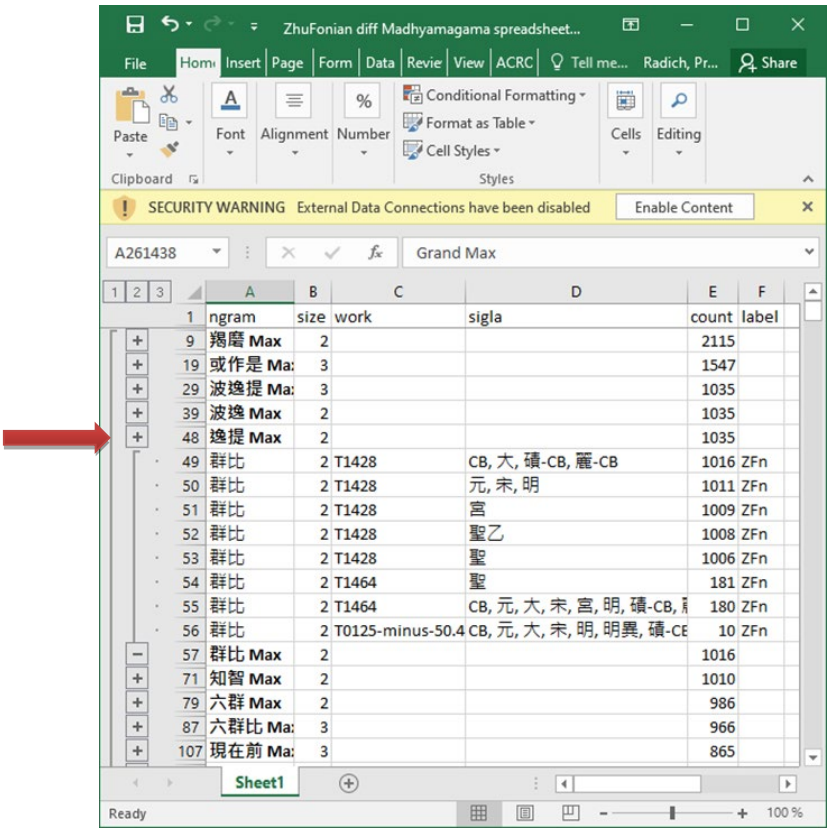
现在，同一n-gram的所有条目都被分组放在一起。此时，点击电子表格左上角的“2”，将每个n-gram子组折叠起来，然后再点击“排序（Sort）”按计数降序排序。在对话窗口中，你将看到你在“小计（Subtotal）”操作之前第一次使用的排序方案（见上文第77页）。删除“ngram”级别，留下“计数（count）”和“长度（size）”（不过“长度[size]”可选可不选）。这样产生的排序将按照“取最大值（Max）”或“总和（Sum）”计数，这取决于你先前选择的是哪一个。




现在，不仅同一n-gram的所有条目被分组在一起，而且其下边的子组也按计数最大值/总计数的降序整齐地排序。这样，你就可以一次性浏览更多的条目：

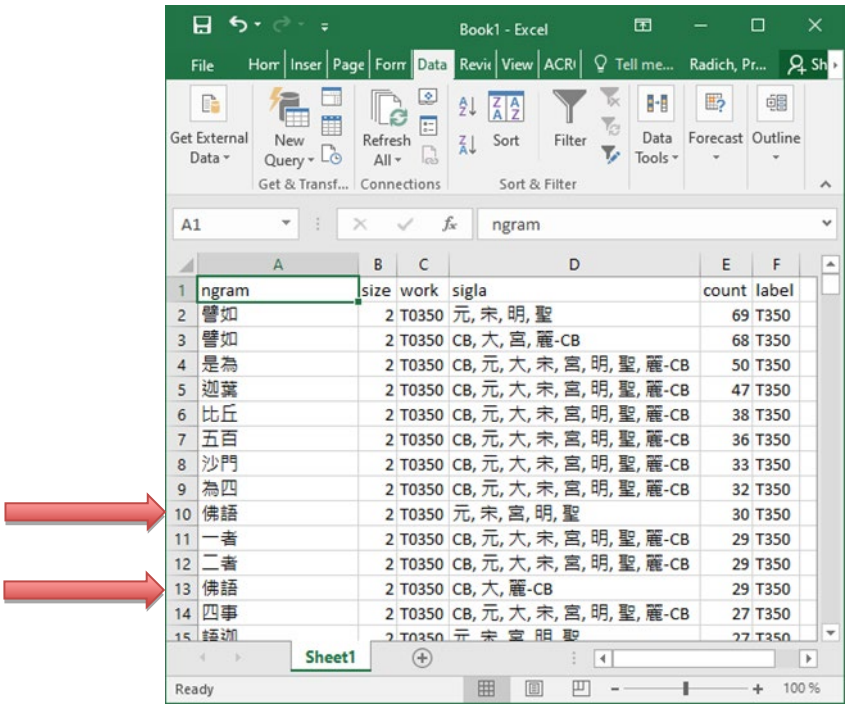


要检查一个分组下的单个条目，只需点击“2”下的加号，即可展开该条目。

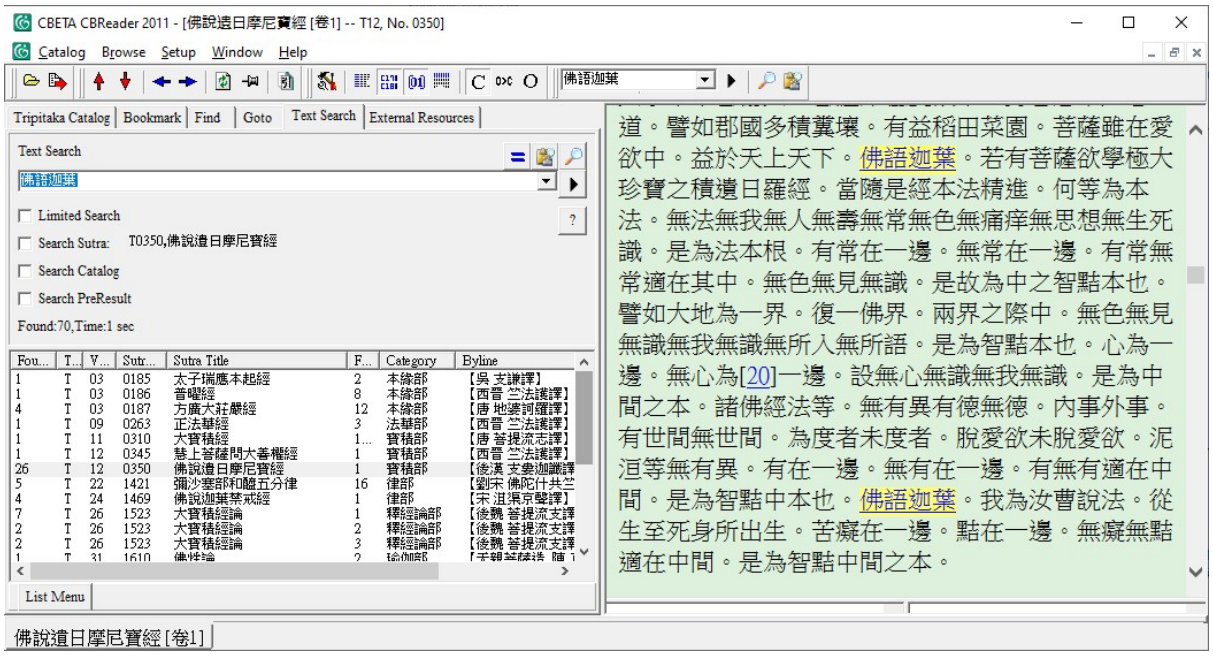



# 工作步骤八：处理结果(2)：回到上下文中检查结果

最后，需要再次强调的是，TACL的原始结果通常还不能用作典型学术问题的“证据”。它们一般仍要由语文学专家参考字符串在文本中的含义和上下文进行分析。（是否可以利用原始的TACL结果来构建坚实的论点，而无需人工定性分析这一干预步骤，这本身尚且是一个悬而未决的研究问题。）比如前面提到的例子，在对支娄迦谶（Lokakṣema）的《佛說遺日摩尼寶經》T350的测试中，结果中可以看到“佛語”一词出现在两行中：




这意味着“佛語”在结果中出现29次或30次（取决于T350的底本）。这是一个差异测试，用来找出可能将支娄迦谶与另一个译者区分开来的文体特征。当我们看到这样孤立的结果时，可能很容易认为“佛語”是名词fóyǔ，意思是“佛陀的言说”（至少，我自己第一次看到它时就是这么想的）。然而，在上下文中：




我们看到，事实上，“佛語”是动词Fó yù，“佛陀对[迦叶]说”。事实证明，在T350中的所有“佛語”都是如此。因此，为了区分翻译风格，区分fóyǔ和Fó yù就很重要——但由于这两个词/搭配在形式上是完全相同的，TACL无法区分，只有人类在上下文中才能分辨。

我们在《TACL方法指南》（TACL Methods Guide）中更详细地描述了这一阶段的分析需要注意的事项。

## 超越图形用户界面（GUI）

对于有技术头脑的用户，或者已经通过使用GUI对TACL产生兴趣，并希望深入学习和使用的用户，这里提供一点补充信息。TACL最初是为在命令行环境中（command line）使用而编写的。命令行版本的程序为用户提供了比TACL图形界面更广泛和更灵活的功能。TACL程序也可以通过API访问，并被其他程序代码调用，这意味着除了在命令行上直接使用外，还可以通过编写新的程序用于在更大规模的操作中协调多个TACL操作。底层的TACL代码是用Python编程的。它是自由软件，意味着用户可以访问代码的所有细节，并且在技术上和法律上都可以自由地修改它并以任何方式使用它。完整的并保持更新的TACL代码在GitHub公布（https://github.com/ajenhl/tacl）。TACL的文档（针对底层的TACL程序，而不是这个GUI）可以在这里找到（https://pythonhosted.org/tacl/）。

## 术语词汇表

以下术语按英文字母顺序列出。每个词条中斜体字的术语在其他词汇条目中有解释。

asymmetric Difference不对称差异运算：见“差异运算”（*Difference*）。另见“不对称差异运算”（Asymmetric Difference），第50页.

catalogue目录文件：一个纯文本文件，用于告诉TACL一个作品（*work*）应归于哪个文本组或语料库（*corpora*）。目录文件中的每一行包含两部分：一个语料库（*corpus*）中作品的标识符（*identifier*）和标签（*label*）。参阅“目录”（Catalogues）,第34页。

concatenated test串联测试序列：由两个或多个差异运算（*Difference*）或交叉运算（*Intersect*）操作组成的TACL测试，将其结果结合在一起。参见“结果交叉运算”（*Supplied Intersect*）。

corpus语料库：（corpus的复数形式是corpora）用户为TACL分析目的定义的一个或一组文本（单文本语料库也是可能的和有用的）。语料库通过目录文件（*catalogue*）中的标签

（*label*）定义。语料库必须采用TACL所需的特定格式。基本语料库可供下载。见“工作步骤一：语料库”（Corpora），第20页。

.csv：“逗号分隔值格式comma separated values”——TACL（和GUI）输出结果的文件格式。结果文件中的信息用逗号分隔。逗号告诉计算机在读取文件时在哪里信息单位结束——例如，由此它可以知道导入Excel时要将信息放入哪个单元格。

database数据库：TACL生成的数据库，存储语料库（*corpus*）中每个文本中出现的所有*n-gram*的列表，以及每个n-gram的出现次数。随后的TACL操作（差异运算*Difference*、交叉运算*Intersect*、搜索*Search*等）都基于数据库中的信息。请参阅“工作步骤二：数据库”（Database），第23页起。

Difference差异运算：用来找出比较双方中仅出现于一方而另一方没有的*n-gram*。（即n-gram只出现于某个指定的语料库*corpus*中，由目录文件*catalogue*中的标签*label*定义）。通过限定“不对称”（asymmetric）操作，输出结果可以限定为只针对比较双方中的一方。见“差异运算”（Difference），第48页。

Filter/Rationalize筛选/合理化：一组后处理操作，允许用户筛选原始结果，以滤掉可能不需要或与研究目标无关的结果，并专注于最有可能感兴趣的结果。在TACL GUI中，在专门的“筛选/合理化”对话窗口中进行操作。请参阅“筛选/合理化”（Filter/Rationalize），第64页。

identifier标识符：用于命名语料库中作品（*work*）的子文件夹，以及在TACL目录文件（*catalogue*）中用于识别作品的名称或缩写。最典型的标识符是作品的《大正藏》编号（或其他藏经集的相应编号，如《续藏经》），有时需要进行修改，以反映文本和语料库结构的变化（比如在修改后的“Radich语料库”中）。例如，《七佛經》T2的标识符为“T0002”。

Intersect交叉运算：一种用来查找参与比较的双方（或所有参与比较的各方）共有的所有*n-gram*（即出现在所分析的所有语料库*corpora*中，语料库由目录文件*catalogue*中的标签*label*定义）的TACL测试。参阅“交叉运算（Intersect）”，第40页。

label标签：由用户定义的、用于标识文本组或语料库（*corpus*）的标签，出现在目录文件（*catalogue*）中一行的末尾，将该行列出的作品分配给相关文本组或语料库。例如，

在下边这样的一行中，

```
T0309 ZhuFonian
```

“T0309”是TACL语料库中《十住斷結經》T309的标识符（*identifier*），而“ZhuFonian”是一个将该作品分配给名为“ZhuFonian”（代表“竺佛念”）的语料库的标签。

n-gram：长度为n的“字符串”（对我们来说是指一串汉字）。例如，2-gram是两个字符长的字符串（例如“如来”）；7-gram则是长7个字符的字符串（例如“無上平等最正覺”）。在TACL中，用户必须指定要收录到数据库中的n-gram长度范围。

“Radich corpus”Radich语料库：我们为TACL定制的一个语料库（*corpus*），包含CBETA数字化的《大正藏》文本文件，并经过修改以反映文本-历史研究的最新发现。请参阅“语料库”（*Corpora*），第20页；以及“（2a）下载Radich《大正藏》语料库的完整数据库”，第29页。

Search搜索：TACL测试，查找给定语料库（*corpus*，由目录文件*catalogue*定义）中所有作品中的所有指定*n-gram*（在用户提供的纯文本文件中列出）。请参阅“搜索”（Search），第60页。

string字符串：一系列连续的字符。就我们的目的而言，“字符串”与*n-gram*同义（见*n-gram*）。

Supplied Intersect结果交叉运算：一种TACL测试，用来找出来自其他TACL操作的两个结果文件（交叉运算*Intersect*或差异运算*Difference*）之间的重合部分，即找到所有同时出现于两个输入的结果文件中的*n-gram*。见“结果交叉运算”（Supplied Intersect），第57页。

TACL：用户通过TACL GUI访问的软件工具套件。请参阅“背景简介”，第6页起。

“T-X语料库”：已预先为TACL整理就绪的佛教藏经语料库，包括《大正藏》和《续藏经》语料库的文本，每个文本都采用CBETA的原始数字化形式（即没有经过类似“Radich语料库”的修改）。参见“工作步骤一：语料库”（Corpora），第20页。

witness底本：一部作品（*work*）的单个版本。不同的底本在其所传承的文本的细节上会有种种不同，这些不同之处可以用TACL加以分析。例如，在TACL语料库中，《大正藏》中的一个作品可能有多个版本的底本，这些底本的文本文件根据《大正藏》的校勘注释被标记为“宋”“元”“明”等版本。

“work”“作品”：即一个“文本”（text），例如，与一个《大正藏》编号相对应的单个文本。我们使用“作品”（work）一词是为了能够区分作品和“底本”（*witness*）。在用TACL分析时，我们也可以将一个大的文本分成多个“作品”——例如，为了更好地分析《六度集(經)》T152，我们把它按章节/故事分成多个独立的“作品”。详情请见经过修改后的“Radich语料库”（Radich corpus）。

# 基于TACL的研究出版物

我们会随时追踪使用 TACL 完成的学术研究，并及时更新以下出版物列表并对其进行补充。也希望世界各地的用户能与我们联系以提供有关此类出版物的信息。

下列的多数由 Radich 撰写或合著的出版物都可以在其 academia.edu 的主页下载（https://uni-heidelberg.academia.edu/MichaelRadich）。


Funayama Tōru 船山徹 [Chuanshan Che]. “*Da fangbian Fo bao’en jing* bianzuan suoyinyong de Hanyi jingdian”《大方便佛報恩經》編纂所引用的漢譯經典. Translated by Wang Zhaoguo 王招國. *Fojiao wenxian yanjiu* 佛教文獻研究 2 (2016): 175-202.

Lin Qian 林乾 and He Shuqun 何书群 [Michael Radich]. "Zhu Fonian suo 'yi' dasheng jingdian de jisuanji fuzhu wenben fenxi yanjiu" 竺佛念所'译'大乘经典的计算机辅助文本分析研究. *Shijie zongjiao wenhua* 世界宗教文化 (2020), no. 6: 16-22.

Lin Qian and Michael Radich. “A Computer-assisted Analysis of Zhu Fonian’s Original Mahāyāna Sutras.” *Buddhist Studies Review* 38, no. 2 (2021): 145-168.

Radich, Michael. “Tibetan Evidence for the Sources of Chapters of the Synoptic *Suvarṇaprabhāsottama-sūtra* T664 Ascribed to Paramārtha”. *Buddhist Studies Review* 32, no. 2 (2015): 245-270.

Radich, Michael. “On the Sources, Style and Authorship of Chapters of the Synoptic *Suvarṇaprabhāsottama-sūtra* T664 Ascribed to Paramārtha (Part 1).” *Annual Report of The International Research Institute for Advanced Buddhology* 17 (2014): 207-244.

Radich, Michael. “On the *Ekottarikāgama* 增壹阿含經 T 125 as a Work of Zhu Fonian 竺佛念.” *Journal of Chinese Buddhist Studies* 30 (2017a): 1-31.

Radich, Michael. “Problems of Attribution, Style, and Dating Relating to the ‘Great Cloud *Sūtra*s’ in the

Chinese Buddhist Canon (T 387, T 388/S.6916).” In *Buddhist Transformations and Interactions: Papers in Honor of Antonino Forte*, edited by Victor H. Mair, 235-289. Amherst, NY: Cambria Press, 2017b.

Radich, Michael. “A Triad of Texts from Fifth-Century Southern China: The **Mahāmāyā-sūtra*, the *Guoqu xianzai yinguo jing*, and a *Mahāparinirvāṇa-sūtra* ascribed to Faxian.” *Journal of Chinese Religions*, 46, no. 1 (2018): 1-41.

Radich, Michael. "Kumārajīva's 'Voice'?" In *China and the World – the World and China: A Transcultural Perspective*, edited by Barbara Mittler, Joachim & Natascha Gentz and Catherine Vance Yeh, 131-145. Deutsche Ostasienstudien 37. Gossenberg: Ostasien Verlag, 2019a.

He Shuqun 何書群 [Michael Radich]. "Zhu Fahu shifou xiuding guo T474? 竺法護是否修訂過T474?" *Foguang xuebao* 佛光學報, New Series, 5, no. 2 (2019b): 15-38.

Radich, Michael. “Was the *Mahāparinirvāṇa-sūtra* 大般涅槃經 T7 Translated by ‘Faxian’? An Exercise in the Computer-Assisted Assessment of Attributions in the Chinese Buddhist Canon.” *Hualin International Journal of Buddhist Studies: E-journal* 2, no. 1 (2019): 229-279. Reprinted in *From Xiangyuan to Ceylon: The Life and Legacy of the Chinese Monk Faxian (227-422),* edited by Jinhua Chen and Guang Kuan, 374-424. Singapore: World Scholastic Publishers, 2020a.

He Shuqun 何書群 [Michael Radich]. “*Da banniepan jing* (*Mahāparinirvāṇa-sūtra*, T no. 7) shifou you ‘Faxian’ suoyi? Jisuanji xiezhu yiren kanding 《大般涅槃經》（Mahāparinirvāṇa-sūtra, T no. 7）是否由「法顯」所譯？計算機協助譯人勘定.” In *Xiantangshan yu Faxian wenhua: Hanseng Faxian (337-422) qi shengping yu yichan guoji yantaohui lunwenji* 僊堂山與法顯文化： 漢僧法顯(337-422)其生平與遺產國際研討會論文集, edited by Miaojiang 妙江, Chen Jinhua 陳金華 , and Ji Yun紀贇, 337-380. Singapore: World Scholastic Publishers, 2020. Chinese version of Radich 2019/2020.

Radich, Michael and Anālayo Bhikkhu. “Were the *Ekottarika-āgama* 增壹阿含經 T 125 and the *Madhyama-āgama* 中阿含經 T 26 Translated by the Same Person? An Assessment on the Basis of Translation Style.” In *Research on the Madhyama-āgama*, edited by Dhammadinnā, 209-237. Dharma Drum Institute of Liberal Arts Research Series 5. Taipei: Dharma Drum Publishing Corporation, 2017.

# 致谢

以下人员以各种方式为开发 TACL GUI 提供了帮助，这里我们表示衷心感谢：Matthias Arnold, Bai Jinghao 白景皓, Merlin Beutlberger, Marcus Bingenheimer, Jonas Buchholz, Chen Ruixuan, Rafal Felbur, Amanda Jack, Huang Mengji, Jamie Norrish, Jiang Tianren, Lin Xueni, Qiu Jie, Manuel Sassmann, Sun Hao, Wang Fengyu, Wang Chenyi, Wang Xinru.