文章编号：1003-0077（2017）00-0000-00

基于历时语料库的在线词典编纂系统设计

吴先[1,2] 胡俊峰*[1,2]

（1. 北京大学 信息科学技术学院，北京 100871；
2. 北京大学 计算语言学教育部重点实验室，北京 10087）

摘要：语料库语言学是借助大规模语料库对语言现象进行发现、挖掘的研究学科，目前已经存在很多在线语料库辅助语言学家的研究。该文提供了一个按时间分片进行管理的语料库，并基于它提出了一个由社区维护的在线词典编纂系统，该系统将语料库查询结果动态结合在了被编辑的词条中。该文还介绍了一个多义词词义发现和层次化聚类算法，用以自动生成一个默认的词条框架。该文将概述词典编纂系统的总体情况，重点介绍了系统的设计和使用方法。

关键词：词典编纂；历时语料库；系统设计；词义发现

中图分类号：TP391 **文献标识码：**A

Design of An Online Lexicographic System Based on Diachronic Corpus

Xian Wu[1,2], and Junfeng Hu*[1,2]

(1. School of Electronics and Computer Science, Peking University, Beijing 100871, China;
2. MOE Key Laboratory of Computational Linguistics, Peking University, Beijing 100871, China)

Abstract: Corpus linguistics is a research discipline that discovers linguistic phenomena by means of large-scale corpus. At present, there are many online corpora-assisted linguists. This paper provided a corpus managed by time slice, and based on it, proposed a community-maintained online lexicography system that dynamically combines corpus query results into edited terms. This paper also introduces a polysemous word meaning discovery and hierarchical clustering algorithm to automatically generate a default term frame. This article will provide an overview of the overall lexicographic system and highlight the design and use of the system.

Key words: lexicography; diachronic corpus; system design; word sense discovery

0 引言

语料库是针对特定领域文本，对其进行科学取样和规范化加工后得到的大规模电子文本库。在近半个世纪，随着计算机存储容量、计算能力的发展，语料库的规模、应用技术、覆盖面和获取方式得到迅速发展[1]。

现当代的语言学研究者们利用计算机辅助工具对词语进行了词义的分析，借助语料库中的大量真实例句，对词所处的句子类型、上下文搭配（词性搭配、具体词语搭配，包括前搭配和后搭配的分情况讨论等）、担当的句子成分进行分析，进而分

收稿收稿日期：2019-07-31；**定稿日期：**2019-08-10
基金项目：大规模汉语历时语料库建设及词汇语义变迁研究（61472017）
*：本文通讯作者

析出其表达功能、使用习惯等更感性的结论[2][3]。在教学方面，语料库语言以其优于教科书语言的，以语言能力为基础、语料高真实性、高覆盖面的特性，也承担着提供真实语言素材、为教材编纂提供支持、补充教科书的任务[4]。在词典编纂任务上，语料库也起到了辅助性作用，一般会要求语料库提供词频统计、索引生成、语法分析、语体分析、搭配分析和语义分析等功能[5]。

历时语料库是在普通的语料库的基础上，加入时间轴，在时间维度上对语料进行划分并分析的语料库。借助历时语料库，语言学家们可以快速沿着一个时间段的切面，从真实的语料出发观察语言现象并加以横向对比，从而挖掘出词语语义、语言风格、词语用法甚至语法等随时间的变化并举出例证。

在传统的词典编纂任务中，语言学家需要从大量的真实语料中发现、挖掘语言现象；然后从中统计、挑选出支持这一发现的数据和例子。为了优化这个工作流，我们设计了一个交互式语料库查询机制，通过简单的命令从语料库中直接获取数据和智能应用的计算结果。融合这个机制，我们实现了一个基于历时语料库的在线词典编纂系统。我们创造性地允许词条作者将查询语句嵌入到词条中，并将查询结果动态实时显示在词条页面，进行简单的筛选和样式修改即可直接展示给使用者。

1 相关工作

现在市面上已经有很多颇为成熟的语料库和基于它们的检索系统。我们在构建历时语料库时比较了相关的技术实现，在设计检索系统时也参考了其基本功能设计。

北京语言大学语料库中心(BCC)是以汉语为主、兼有其他语种的语言大数据的在线检索系统和语言应用基础平台。其语料库总字数约 150 亿字，包括：报刊（20 亿）、文学（30 亿）、微博（30 亿）、科技（30 亿）、综合（10 亿）和古汉语（20 亿）等多领域语料，是可以全面反映当今社会语言生活的大规模语料库。BCC 可以通过云服务 API 调用的方式为其他研究和应用工作提供便利[1]。

英国国家语料库 BNC 收录了来自各种来源的书面和口头语言样本，旨在代表 20 世纪后期英国英语的广泛的横截面。最新版本是 2007 年发布的 BNC XML 版本。BNC 的书面部分占 90%，包括地区和国家报纸，覆盖全年龄和兴趣的专业期刊，学术书籍和流行小说，已发表和未发表的信件和备忘录，学校和大学散文摘录等等。口语部分（10%）由非正式交谈（由不同年龄，地区和社会阶层的志愿者以人口均衡的方式记录）和不同背景下收集的口语进行转写，题材从正式的商业或政府会议到广播节目和电话通话均有覆盖。

英国国家语料库 BNC 本身并没有提供检索和查看功能，但是有大量基于此语料库的检索引擎，例如 BYU-BNC、BNCWeb 和 IntelliText。这些检索引擎提供了面向普通需求的简单检索语法，例如通配符、不完全的正则、词性限制、词语前缀、词序列、句型检索等；也有实现数据库相关的复杂检索语法，例如 BNCWeb 采用的 CWB corpus 数据模型，采用的检索语言是 CQP 语言[6]。

现有系统在语料库系统设计方面树立了良好的标杆。但是在使用方面，现有的语料库系统从发起查询到得到查询结果，完成了一次独立的工作流；与词典编纂的工作流存在较大的独立性。在了解了词典编纂的基本研究方法后，我们将一些统计指标融入在自己设计的简化编程语言中，并加入了一些基于基本统计指标的进一步分析结论，以简化和辅助语言学家的工作。

同时，我们的语料库系统设计中保留了对基本检索式的支持，以满足基本用户需求；在应用接口层面，我们在系统中留有提供计算服务的模块，可以随时接入新的算法应用。

2 历时语料库的构建实践

2.1 语料加工

我们使用了北京大学计算语言所提供的 1986-1995 十年的人民日报标注语料库作为原型系

统的数据源。这是基于严格验收过的 1998 年人民日报的词语切分和词性标注结果，以相似的处理流程在更大规模生语料上自动标注的结果[7]。

词性标注采用了《现代汉语语法信息词典》中规定的 26 个词类代码：名词 n、时间词 t、处所词 s、方位词 f、数词 m、量词 q、区别词 b、代词 r、动词 v、形容词 a、状态词 z、副词 d、介词 p、连词 c、助词 u、语气词 y、叹词 e、拟声词 o、成语 i、习用语 l、简称 j、前接成分 h、后接成分 k、语素 g、非语素字 x、标点符号 w。除此之外，还有名词 n 的扩展类型：专有名词的分类标记 ，即人名 nr，地名 ns，团体机关单位名称 nt 和其他专有名词 nz[8]。

2.2 存储方式

针对语料库系统的“修改少、查询多、大多数查询可控”的特点，我们采用了 MyISAM 引擎的关系型数据库对原始语料以文档、句子和词为单位进行存储。其中文档为存储语料来源、日期等元信息的基本单位；句子保存了语料的完整内容和词性标注结果；词是用于倒排索引的单位，保存了其位于句子中的位置和偏移量。数据库表的定义如表 1、表 2、表 3：

表 1 Document 表结构

字段名	含义
id	自增主键
date	文档的日期
source	来源

表 2 Sentence 表结构

字段名	含义
id	自增主键
content	内容
part_of_speech	分词与词性标注
pos	位于文档中的位置
fk_document_id*	所在文档的 id 外键

*：所有 fk 开头的字段代表外键的语义，需要与真实对象的 id 对应，并按需添加索引。

表 3 XXXInvertedIndex 表结构

字段名	含义
id	自增主键
token	词项
start_offset	位于句子中的起始偏移**
end_offset	位于句子中的结束偏移
pos	位于句子中的位置
fk_sentence_id	所在句子的 id 外键

*：XXX 表示不同类型的倒排索引。为了支持不同类型的标记进行混合检索，我们建立了平行的倒排索引表以支持不同类型的标记，目前支持的标记类型有词语、词性和词义。

**：偏移是以字为单位的，位置是以词为单位的。

在进行检索时，按需将不同类型的 XXXInvertedIndex 表按照 offset 或 pos 字段的顺序进行连接，即可实现诸如“d+开展+运动_s1”（其中“_s1”表示前缀词语的第一个词义）的混合检索式。

2.3 多义词标签提取

我们采用了一个基于非参数 skip gram 模型(NP-MSSG)的多义嵌入算法[10]从原始语料中无监督地提取多义词的词义嵌入。这为我们建立词义倒排索引表和在词典编纂系统中自动产生词义提供了基础。

NP-MSSG 模型是在基础的 skip gram 模型的基础上，将一个词语原本各有一个的上下文向量(global vector)和词向量(sense vector)拆分成了 1 个全局向量（用于构成上下文向量）、k 个上下文聚类中心（对应上下文向量）和 k 个词义向量（对应 k 个词义）。训练过程如算法 1 描述。

算法 1 训练 NP-MSSG 模型

1: 输入：$w_1, w_2, \ldots, w_T, d, \lambda, N, thresh$。
2: 随机初始化$v_s(w,0)$和$v_g(w)$，零初始化$\mu(w,0)$。
3: for $t = 1, 2, \ldots, T$ do
4: $R_t \sim \{1, \ldots, N\}$
5: $c_t = \{w_{t-R_t}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+R_t}$

6: $v_{context}(c_t) = \frac{1}{2*R_t}\sum_{c\in c_t} v_g(c)$

7: $s_t =$

$$\begin{cases} k(w_t)+1, \text{if } max_{k=1,\dots,k(w_t)}\left\{sim\begin{pmatrix}\mu(w_t,k),\\ v_{context}(c_t)\end{pmatrix}\right\} < \lambda \\ k_{\max}, \qquad \text{otherwise or } k == thresh \end{cases}$$

8: 上下文向量c_t被加入词语w_t的第s_t个聚类中心，更新聚类中心$\mu(w_t, s_t)$

9: $c_t' = Noisy_{Samples(c_t)}$

10: 梯度下降更新$v_s(w_t, s_t)$，以及c_t和c_t'中词语的全局向量。

11: end for

12: 输出：$v_s(w,s), v_g(w)$和聚类中心$\mu(w,k)$。

在获得词义向量和聚类中心的同时，我们也得到了每个词在语料中每次出现时被赋予的词义 id。

但是根据 H Shi 等人的讨论，这种方式提取出的多义词词义会出现伪多义的情况[11]**错误!未找到引用源。**，即存在本应是同一个词义，但是在训练时聚类中心和词义中心都分裂成多个的现象。在工程上，我们采用后处理的方式减少这一现象出现的频率。

我们利用标注好词义的语料作为输入数据（北京/ns → 北京_s0/ns），计算所有词语（词义）的 PMI 向量，向量长度为词义词典大小。在这个 PMI 向量后拼接了一个词性模板向量。词性模板是一个词性三元组，定义为“前一个词词性-中心词词性-后一个词词性”。其指示向量为词性模板向量，反映了其用法特征。利用这个向量计算每两个词义间的余弦相似度，并进行层次化聚类[12]。以同一个词的两个词义的树上距离作为词义合并的度量，小于一定阈值则视为同一个词义，将其合并。

表 4 列举了一些通过层次化聚类树上距离发现的伪歧义现象。

表 4 伪歧义现象举例

词语	词义 id		近义词
	合并前	合并后	
十分	0	0	比较、相当、 较为、非常、很、更、可谓
	1	0	更为、极为、颇为、最为、尤为、格外、煞是
水分	0	0	水蒸气、废气、碳酸气
	1	1	漏洞、弊病、疑点、破绽、微词
	2	1	诟病、蛛丝马迹、隐忧、争议、顾虑

在“十分”的例子中，我们看到同样作为描述程度的两个副词词义在经过合并后成为了一个。而在“水分”的例子中，其作为“物体内所含的水”的含义和“比喻不实的成分”的含义的区别被保留了下来。

2.4 数据统计

10 年的人民日报总共出现词语 124168407 个•次，其中出现频率大于 20 的词语有 75212 个。这 75212 个词语总计包含词义 155471 个，平均每个词含有 2.067 个词义；其中，有 45102 个词语含有 3 个或 3 个以上词义，2429 个词语含有 5 个或 5 个以上词义。拥有最多词义的词为“把”、“用”和“还”，达到了算法设置的 K 的最大阈值 10。

3 历时语料库检索功能

3.1 检索系统体系结构

我们的系统包含四个模块，如图 1 所示。

各模块的定义和职责如下：

- 协调模块，用于协调各模块的运作。同时也提供预处理功能，支持通过脚本，从本地导入满足数据规范的数据，在存储引擎中建立持久存储与索引，完成其他计算模块所要求的预处理（比如句子向量），输出数据的存储位置和操作日志。
- 前端交互模块，通过 web 服务，为普通用户提供数据库的全功能、友好的检索接口/页面。输入为检索要求，输出为检索的结果、数据统计及相关图表和下载链接。
- 存储和检索模块，为预处理和前端提供存储和检索支持，并加入基本的数据统计功能。输入为前端定义的检索要求，输出为具体存储引擎支持的指令和对响应结果的后处理。
- 计算模块，用于进行计算密集型任务的处理，比如同义句检索。输入和输出与各模块具体功能相关。

图 1 检索系统体系结构

扁圆代表操作人群，圆角矩形代表软件模块，箭头代表请求或数据流向，数字代表一次完整请求的请求/数据流。

3.2 基本检索功能

语料库实现的检索功能有：单检索、共现检索、PMI 变迁检索和词性 PMI 变迁检索。

以单检索为例，用户输入检索词并点击搜索按钮后，系统将会按年份返回词频的柱状图；通过点击柱状图，用户可以查看该年份包含该词语的例句（如图 2）。在例句详情中，用户可以查看句子内容、词性标注结果和近义句等相关信息。

图 2 单检索示例

近义句是由构成该句子全部词语的词义向量计算平均值得到句子向量，然后利用 Approximate Nearest Neighbors 算法实现快速计算的[13]。这部分运算由本系统的计算模块负责执行，计算模块已拥有预计算的局部敏感哈希模型，载入模型并运算得到近义句 id 后通过 thrift 协议将结果返还协调模块，再去通过存储模块得到句子详情。这是本系统的典型执行过程和数据流。

3.3 PMI 检索与展示方案

历时语料库的一大特点为可以根据语料库上词语的某个统计量的历时变化反映该词语在词义、词语运用、感情色彩等方面随时间的变化。我们接下来以词语的常用搭配为例，展示本系统的历时查询功能。

点互信息（PMI）是一个可以反映词语和词语之间是否存在共现倾向的指标[14]。PMI 的计算式为

$$pmi(w_1, w_2) = \log \frac{p(w_1, w_2)}{p(w_1) * p(w_2)} \#(1)$$

我们按年份计算出给定词语的前 k 个 pmi 最高的词，可以得到这个词语在每个年份中的惯用搭配，进而发现它在用法与词义上的变化。

展示方面，我们给出了词云和力学布局两种可视化方案。在词云方案中，文字的大小表示 pmi 的绝对大小，颜色深度表示当年与前一年 pmi 的差异。也就是说，文字颜色越深，越是表示这个词在这一年突然出现的搭配，以引起用户注意，如图 3。

力学布局是将统计量转化为物理参数后计算出的稳定的力学布局。在这个力学布局中，我们引入了环状力、斥力、碰撞力和引力。环状力用来模拟共现词与中心词的 pmi 大小（其引力点位于物体-圆心连线与指定半径圆环的交点处）；斥力和碰撞力用于使浮动的词语尽可能相互分开且不出现重叠；引力用来刻画共现词之间的相似性关系。整个力学系统在加入阻尼后将会收敛在一个稳定的状态，我们将最终的稳定结果展示给用户。相似度较高的词语之间将会以半透明的实线连接，以表示词语的聚类关系，如图 4。

图 3 词云效果展示

图 4 力学布局展示

在切换年份选择器后，柱状图和上述两个可视化结果都会发生相应的变化，用户可以任意选择年份，以通过动画效果直观地看到词语统计量的变化，如图 5。

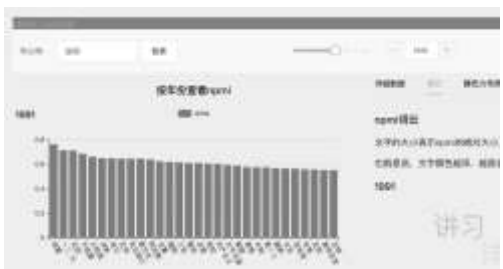

图 5 年份选择器

4 交互式语料库设计

4.1 概念

语料库语言学的一个常见的研究方法，是通过构造各种查询语句，得到研究对象词语的用例，总结归纳得出结论，并且可以以例句加以佐证。这种查询模式在现有的语料库中也有做到，比如 BNCWeb 的查询语法，是通过类似通配符和大括号匹配的方式进行复杂查询的（比如{eat} ++* up，表示 up 在 eat 后 3~4 个词位）；IntelliText 则是提供了一个表单来辅助构建这个复杂的字符。而且一次会话只能处理一种查询，想要对结果进行比较或是并排查看，是需要自己做额外的工作的。

我们的目的是为语言研究者提供好的工具平台，从语言学研究者的角度出发，优化词典编纂的工作流，以词条为核心，尽量多地展示其信息，以及方便地扩展出更多查询。

为此，我们提出了交互式语料库的概念，用户可以在一个类似于笔记本的环境中（后面会以“笔记本”来代指一个完整的会话），同时完成思路的记录、内容的撰写、数据的查询和查询结果的嵌入。

4.2 体系结构

笔记本式的交互系统参照 Jupyter 项目（一个将编程从基于命令行的交互扩展至网络应用的项目，其应用可以捕捉整个计算流程：开发、文本工作、执行代码、分享结果）的体系，保留了其简化结构：Web 应用程序、笔记本文件和运行核心[15]，如图 6。

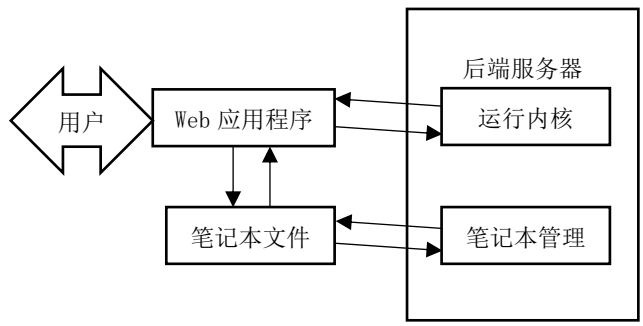

图 6 交互式语料库体系结构

笔记本的管理位于服务器上，为每一个用户开辟了一个存储区，支持笔记本的添加、删除、获取、修改、复制、重命名、保存记录点等操作。

笔记本中命令的执行是依赖于内核进行的。内核位于服务器上，与笔记本本身的内容相互独立，但是又通过“连接”的概念，将前端看到的一个笔记本与一个内核关联起来，用于保存变量等工作区内的信息，以及执行从前端发送过来的命令。

最后是 Web 应用程序（也就是前端页面）。笔记本是只负责保存用于显示的内容的文件，在 Web 应用程序负责渲染与展示；同时，Web 应用程序可以将一个笔记本与一个内核连接起来，使命令具有可以执行的环境。当前端发送了命令至内核，并获取了内核的输出结果后，由前端负责修改笔记本中相应的单元格内容，并负责向笔记本的管理端发起保存请求。在服务器后端，笔记本与内核是不会通信的。

4.3 命令语法

Kernel 所支持的语法是基于函数的简化语言。如帮助中所说，即使是变量赋值和回显等语法，在 Kernel 中也是基于函数执行的。

目前支持的命令有词频统计、词义频统计、近义词、多义词近义词义、词语例句、词义例句等。用户在输入命令并执行后，前端将会根据执行结果的类型渲染出合适的效果，如图 7。

图 7 交互式语料库命令示例

5 在线词典编纂系统构建

5.1 概念

为了实现优化的词典编纂工作流，在交互式语料库之上，我们构建了一个在线词典编纂系统。语言学家可以借助这个系统，快速搭建属于自己的词条，在同一个工作环境中完成词条内容的撰写、更新维护、数据查询与导入，我们把这个环境称为“词条”。

5.2 词条整体操作定义

5.2.1 创建词条

目前一个词条包含标题、词条词语、内容（cells）、公开标记位和创建时间、修改时间；

5.2.2 浏览词条

对于属于自己的词条和公开标记位设置为 True 的词条，可以点开后进行编辑；

对于不属于自己的词条，修改后无法进行保存，需要另存为/复制成自己的词条副本；

5.2.3 编辑词条

对于属于自己的词条，可以进行编辑，具体操作在下一节定义；

5.2.4 共享/公开词条

将公开标记位设置成 True/False，以设置词条的可视范围，公开的词条可以视为发布给社区和全部游客；

5.2.5 继承/复制词条

从其他作者的词条中拷贝出一个副本，将其所有者变成自己，并进行更改，这个继承/复制操作会被系统记录，用于原作者追踪自己的后续分支；

5.2.6 反继承/合并词条

原作者将自己的词条与他人后续的修改版本进行合并。

5.3 词条编辑操作定义

5.3.1 内容编辑

可以编辑词条中的标题内容、文字性内容和查询语句。查询语句是基于交互式语料库技术实现的特殊的单元格，查询类型的单元格将会根据输入指令的返回值类型调整其展示方式和不同的样式修改选项，例如对于图表类返回值，用户可以继续修改图标的颜色、类型、图例等显示风格方面的信息，如图 8。单元格之间可以重新调整顺序、插入和删除。

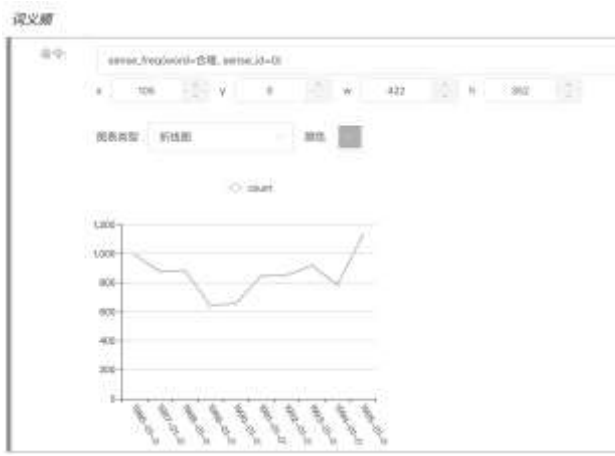

图 8 查询语句单元格

5.3.2 词条初始化

刚创建的词条需要初始化后才能进行编辑。

初始化可以选择生成一个默认的词条框架，我们的系统将会为这个词条自动填入各级标题、读音，并按照利用多义词嵌入模型挖掘出的词义，自动产生 k 个子标题，并在其后列出相应的词义频、近义词和词义例句。

5.3.3 快速更新查询结果

我们的查询类型单元格可以支持从后端服务器取得任意类型的数据，这个数据也很可能是由后端服务器即时演算产生的。因此，我们的词条也支持检查查询类型单元格的结果是否是最新版本的功能，当词条收到了后端服务器的更新通知，将会提醒用户并询问是否重新执行相关查询。这一操作只会影响数据内容，而不会影响已修改的样式数据与其他单元格内容。

5.4 词条继承与合并

用户每次复制/另存为词条都会为原词条产生一条衍生记录，原作者可以根据这个记录追踪自己词条的后续发展，也可以选择合并后续修改的版本。

词条的合并算法是以单元格为单位，为两个词条进行最小编辑距离的计算，并记录下编辑操作。其中，增加单元格的代价为 1，删除单元格的代价为 1，替换单元格的代价是一个代价函数，使差异比较大的两个单元格的替换代价大于 2，以使算法可以得到增加+删除的结果；但同时这个函数的最小值是 0，以保证在发生单元格的完全匹配时，可以直接通过这个函数得到 0 代价。

对于两个单元格的比较，我们采用了复杂度为 $O(n+m)$的 QGRAM 算法[16]，其值计算方式为，给定 n，分别为两个字符串计算 n-gram 频率作为特征向量，两个向量的L1范数为在给定n下的两字符串 QGRAM 值。

于是，我们设置的代价函数为

$$\text{cost} = \text{FACTOR} * \frac{\text{distance}}{\text{len(s1)} + \text{len(s2)} - 4} \#(2)$$

其中 distance 为 n=3 时的 qgram 距离,FACTOR 取 3，分母是为了把 distance 归一化到 0-1 之间（需要减 4 是因为 n=3 时，3-gram 的数量是字符数量-4）。

在这个算法下，我们可以得到一个由 match, add, delete 和 substitute 构成的操作序列，使当前词条转化成被比较词条。冲突的单元格用不同颜色的底纹表示其操作：add 为蓝色、delete 为红色、substitute 为绿色（其中原始版本为蓝色，修改版本为黄色），如图 9。用户在合并模式的词条下需要手动解决全部合并冲突以重新进入正常模式继续进行词条编辑。用户可以选择为连续的相同冲突采取同种操作。

图 9 合并模式展示

6 结语

我们利用可获取到的带有时间标记的大规模语料构建了一个历时语料库，为之开发了一个简单的能够体现语料库历时特征带来的收益的检索系统，并保持了后续接入更多应用与算法的能力。同时我

们提出了交互式语料库的概念，并基于它开发了一个在线词典编纂系统，希望为广大的语言学研究者提供更高效、统一的工作平台。

这项工作的后续研究将会着眼于更智能的默认词条框架的生成，使多义词的词义提取更精确，以及捕捉到更多语言现象。

参考文献

[1] 荀恩东，饶高琦，肖晓悦，等. 大数据背景下 BCC 语料库的研制[J]. 语料库语言学，2016，3(1)：93-118.

[2] 白荃，岑玉珍. “轻易” 的语义及用法分析[J]. 语言教学与研究，2007 (5)：76-81.

[3] 孙德金. 现代汉语书面语中的代词 “其”[J]. 语言教学与研究，2010 (2)：55-62.

[4] 陈露，韦汉. 英语口语语料库在英语口语教学中的作用[J]. 外语电化教学，2005 (3)：23-26.

[5] 钱厚生. 语料库建设与词典编纂[J]. 辭書研究，2002，2002(1)：58-68.

[6] Hardie A. CQPweb—combining power, flexibility and usability in a corpus analysis tool[J]. International journal of corpus linguistics, 2012, 17(3): 380-409.

[7] 俞士汶，段慧明，朱学锋，等. 北京大学现代汉语语料库基本加工规范[J]. 中文信息学报，2002，16(5)：51-66.

[8] 俞士汶，朱学锋，王惠，等. 现代汉语语法信息词典规格说明书[J]. 中文信息学报，1996，10(2)：1-22.

[9] 段慧明，徐国伟，胡国昕，等. 大规模汉语标注语料库的制作与使用[J]. 语言文字应用，2000 (2)：72-77.

[10] Neelakantan A, Shankar J, Passos A, et al. Efficient Non-parametric Estimation of Multiple Embeddings per Word in Vector Space[C]//Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2014: 1059-1069.

[11] Shi H, Li C, Hu J. Real Multi-Sense or Pseudo Multi-Sense: An Approach to Improve Word Representation[J]. CL4LC 2016, 2016: 79.

[12] He S, Zou X, Xiao L, et al. Construction of Diachronic Ontologies from People's Daily of Fifty Years[C]//LREC. 2014: 3258-3263.

[13] Wolff J. Approximate nearest neighbor query methods for large scale structured datasets[J]. 2016.

[14] Bouma G. Normalized (pointwise) mutual information in collocation extraction[J]. Proceedings of GSCL, 2009: 31-40.

[15] Ragan-Kelley M, Perez F, Granger B, et al. The Jupyter/IPython architecture: a unified view of computational research, from interactive exploration to communication and publication[C]//AGU Fall Meeting Abstracts. 2014.

[16] Ukkonen, E. (1992). Approximate string-matching with q-grams and maximal matches. Theoretical computer science, 92(1), 191-211.

吴先（1994—），硕士研究生，主要研究领域为自然语言处理。

E-mail：wuxian94@pku.edu.cn

胡俊峰（1967—），通信作者，博士，副教授，主要研究领域为计算语言学。

E-mail：hujf@pku.edu.cn