Technical Note

A Novel Web Application for Rapidly Searching the Diagnostic Case Archive

Scott Robertson

Department of Anatomic Pathology, Cleveland Clinic, Cleveland, OH, USA

Submitted: 14-May-2020 Revised: 24-Jun-2020 Accepted: 31-Aug-2020 Published: 24-Dec-2020

Abstract

Academic pathologists must have the ability to search their institution's archive of diagnostic case data. This ability is foundational for research, education, and other academic activities. However, the built-in search functions of commercial laboratory information systems are not always optimized for this activity, leading to delays between an initial search request, and eventual results delivery. To solve this problem, a novel web-based search platform was developed, named Pathtools, which allows our staff and trainees to directly and rapidly search our diagnostic case archive. Pathtools was built with open-source components and features a web-based user-interface. Pathtools uses an SQL database which was populated with anatomic pathology case data going back to 1980, and contains 4.2 million cases (as of July 31, 2020). Pathtools has two major modes of operation, "Preview Mode" and "Research Mode." Since deployment in February of 2019, Pathtools carried out 33,817 searches in Preview Mode, averaging 0.72 s (standard deviation = 1.7) between search submission, and on-screen display of search results. In Research Mode, Pathtools has also been used to produce data sets for research activity, providing the data used in many abstracts and manuscripts our investigators submitted recently. Interestingly, 75% of search activity is from trainees during their preview time. In a survey of residents and fellows, 83% used Pathtools during the majority of their preview sessions, demonstrating an important role for this resource in trainee education. In conclusion, a web-based search tool can rapidly and securely provide search capability directly to end-users, which has augmented trainee education and research activity in our department.

Keywords: Education, pathology reports, python, text search, web application

Introduction

The pathology department's archive of diagnostic case data is a valuable resource. Our archive, for example, contains the diagnostic text of more than 4-million anatomic pathology cases compiled over four decades. It represents a treasure-trove of information that can be leveraged for research, education, and a variety of other scholarly activities. However, the extent to which this data can be used for these activities depends on the ability for users to efficiently query the archive. Unfortunately, the search functions of commercial laboratory information systems (LISs) often have cumbersome user-interfaces (UIs) which are difficult for the average pathologist to use. This is something we experienced with our own LIS (Cerner CoPath Plus v2014), as some amount of technical knowledge is needed to design and execute effective searches. Furthermore, searches of the production database can impact the entire system. Using our LIS, we found that multiple concurrent searches could stress the production database. Therefore, to mitigate risk to clinical operations, our institution restricts LIS search functions to a small team of analysts, who carry-out searches on behalf of the pathologist. Unfortunately, this results in a significant lag-time between a pathologist's initial search request, and eventual results delivery – usually requiring >24 h.

The aim of this project was to create a novel search platform which would allow our pathologists to directly and expeditiously query the case archive. To do this, a web-based graphical UI was used, which anyone with basic computer skills can use effectively. Open-source software components were used to minimize software costs, maximize code transparency, and avoid reliance

Access this article online

Quick Response Code:

Website: www.jpathinformatics.org

DOI: 10.4103/jpi.jpi_43_20

Address for correspondence: Dr. Scott Robertson, Department of Anatomic Pathology, L25, Cleveland Clinic, 9500 Euclid Avenue, Cleveland, OH 44195, USA.
E-mail: roberts10@ccf.org

How to cite this article: Robertson S. A novel web application for rapidly searching the diagnostic case archive. J Pathol Inform 2020;11:39.

Available FREE in open access from: http://www.jpathinformatics.org/text.asp?2020/11/1/39/304778

on a single commercial vendor. Finally, the system was designed in close collaboration with our institution's legal, HIPAA and IRB stakeholders, to ensure data security and adhere to good data governance practices. This search platform was named "Pathtools."

Implementation Details

Web design

The Pathtools website was written in Django, an open-source web application framework written in Python (Django version 2.1.2, Python version 3.5.2).[1,2] The site is served by Apache2 running on a Linux (Ubuntu 16.04) server behind our institution's firewall. A signed SSL certificate was installed enabling HTTPS encrypted communication between the server and client. The UI was designed around jQuery QueryBuilder, a Javascript plugin which provides a graphical UI to design searches.[3] This UI was chosen because of its intuitive interface. Searches can be relatively simple, using only one or two parameters [Figure 1]. However, more complex searches can also be designed, mixing together various fields, operators, and logic types [Figure 2]. The choice of available operators depends on the underlying data type for that field [Table 1]. Many different fields are searchable, including the primary diagnostic text (consisting of the final diagnosis, comment, and addendum), gross description, intraoperative diagnosis, synoptic text, patient age, patient sex, attending pathologist, case-type (surgical pathology, cytology, bone marrow), accession date, consult case status and region. On, execution, QueryBuilder transforms the user's input into an SQL query statement, which is sent to the database.

Result Delivery Modes and Data Governance Issues

The workflow for results delivery was designed in close collaboration with our institution's HIPAA, IRB and legal stakeholders, who emphasized the desire to give users the minimum amount of data needed for their specific purpose. Looking at the different use-cases for the system, we designed Pathtools to work in two modes, "Preview Mode" and "Research Mode," which differ in the amount of results the user receives and the data elements the user has access to. For most use-cases, the user needs only a few data elements.

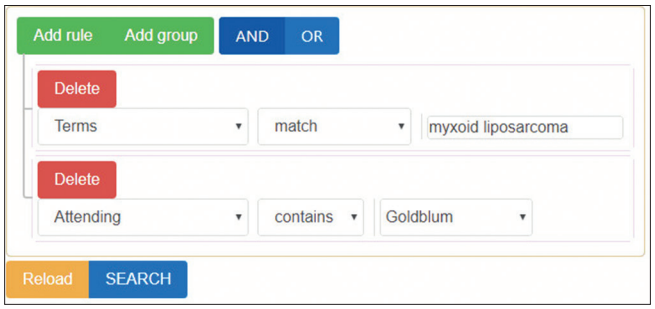

Figure 1: Search example. The user searches for cases containing the term "myxoid liposarcoma" signed out by Dr. Goldblum

For example, most searches are performed by trainees during their preview time (see results section). For these searches, the diagnostic text is most important; demographics and assigned pathologist are also provided. Finally, the case-number is displayed, as the user may want to retrieve the slides from our archive. Therefore, in its default configuration, "Preview Mode," Pathtools retrieves only a limited data set from the database (diagnostic text, patient demographics, accession number, and assigned pathologist), and displays the results on-screen. However, Pathtools can also be used for research, "Research Mode," in which patient identifiers are can be obtained. Using Research Mode, the user can request Microsoft Excel data extracts for IRB-approved research projects.

Preview Mode Technical Details

For Pathtools' default configuration, Preview Mode, the results are displayed on-screen, and a processing script highlights the user's diagnostic terms within the returned text, allowing the user to quickly determine the relevance of each result [Figure 3]. Text highlighting was coded by building a list of compiled regular expression objects (using Python's built-in regular expression module, "re") for each search term. Then, as results arrive from the database, Pathtools loops through the text and uses a substitution command (re.sub) to flank each term with a set of HTML "mark" tags (<mark> [term] </mark>). Therefore, the user's browser will highlight the user's terms in yellow.

The search parameters can be changed after each search so the user can quickly iterate their parameters to optimize their search results. On-screen results are limited to the most recent 100 matching results (using the SQL clauses "TOP [100]" and

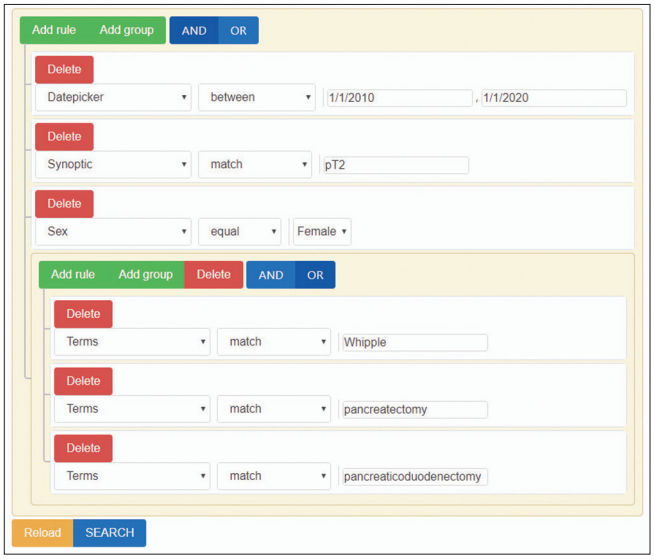

Figure 2: More complex search example. The user searches for pancreatic surgical resections staged as pT2 in a specific date range. "pT2" must be present in the synoptic text of the case. Note that the nested subgroup uses "OR" logic (shaded dark blue) to capture any of three different types of pancreas resection. The top-level logic remains "AND" type (shaded light blue)

Table 1: Database fields, input types and supported operators

Field name	Input type	Supported operators
Final, Comment, Addendum Text	Text Box	Contains, Does Not Contain
Gross Description	Text Box	Contains, Does Not Contain
Clinical Information	Text Box	Contains, Does Not Contain
Synoptic	Text Box	Contains, Does Not Contain
Intraoperative Diagnosis	Text Box	Contains, Does Not Contain
Specimen Type	Drop-down list (Surg Path, Cyto, Bone Marrow)	Equals, Does Not Equal
Attending Pathologist	Drop-down list (containing all active pathologists)	Contains, Does Not Contain
Age	Text Box (with integer validation)	Greater, Less Than
Sex	Drop-down list (Male, Female)	Equals, Does Not Equal
Accession Date	Text Box (with date validation)	Before, After, Between
Consult Case Flag	Drop-down list (Inside Case, Outside/ Consult Case)	Equals, Does Not Equal
Region Selector*	Drop-down list (Ohio, Florida)	Equals, Does Not Equal

*The database contains records from two distinct regions of Cleveland Clinic operations, Ohio and Florida. This selector allows the user to filter based on region

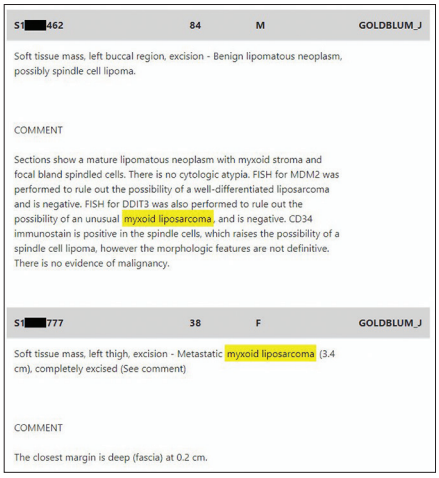

Figure 3: Format of results displayed to user on-screen. Results are displayed on-screen within an HTML table. A shaded header row displays accession number (partially redacted here for publication), patient age, patient sex, and attending pathologist. The next row contains the final diagnosis, as well as any comments or addendums, if present. A processing script highlights the user's search terms by flanking each term with a set of HTML "mark" tags (<mark> [user_term] </mark>)

"ORDER BY accession date DESC"). On-screen results were limited for two reasons. First, for most use-cases, users only need a few relevant search results. Second, limiting search results to 100 cases minimizes execution time. Therefore, a user could search for a commonly used term like "helicobacter" and receive results in 0.51 s. Whereas a comprehensive search of the database (without the "TOP [100]" SQL clause) returns 158,043 cases and takes 1 min, 36 s to complete.

Research Mode Technical Details

The UI of Research Mode is essentially identical to that of Preview Mode, using the QueryBuilder Javascript plugin. However, there are two key differences between the two modes. First, in Research Mode, two different SQL commands are sent to the SQL database when the user submits the search. The first query is identical to the one sent in Preview Mode, and retrieves the diagnostic text, patient demographics, accession number and assigned pathologist fields for the most recent 100 results. These data are displayed on-screen. However, a second search query is also sent, which determines how many total matching cases are in the database, without returning any fields (using "SELECT COUNT [*]…"). For example, searching for the string "helicobacter" in Research Mode takes 0.95 s, returning a preview of the first 100 results which appear on-screen (diagnosis, demographics, cases number, and assigned pathologist), and also displaying the total number of matching cases in the database: 154,043. The second search, which enumerates how many total cases are in the system, is useful for search optimization, as the user may want to narrow or expand their search based on this information.

The second difference from Preview Mode is the ability to save the list of matching cases. After the results are returned, the user may click the "Capture Cases" button. After the user provides a name, Pathtools retrieves the user's last search and sends it to the database again, but with two alterations. First, only the case id field (accession number) is returned (using "SELECT case id"). Second, the "TOP (100)" clause is removed, so all matching cases are retrieved. Using the list of matching cases, Pathtools will build a Django database object called a "Case Set" (which is essentially a list of accession numbers with a user-defined name). The Case Set is saved in Django's database and is linked to the user's Django profile (making it accessible only to the user). The user may return to the saved Case Set at any time, using the Case Set Dashboard Page (see below).

Case Set Dashboard and Data Extraction Workflow

The "Case Set Dashboard" is a page which displays all the user's saved Case Sets. The user can request a Microsoft Excel data extract by clicking on a Case Set. The user is taken to a web-form which asks the user to select which data elements they want to extract. The user is instructed to select only the data elements which have been approved by the IRB for collection. This form also provides a location to upload the user's IRB-approved data collection protocol in pdf format, which lists the approved data elements. After the download request is submitted, the request is forwarded to an independent

analyst, who adjudicates the request. The analyst checks the download request against the investigator's IRB protocol, to ensure that the requested data elements are approved for collection. Assuming the request conforms to the IRB protocol, the analyst executes the request in Pathtools. Upon request execution, Pathtools internally builds a Microsoft Excel file using Pandas library functions (a Python data analysis library).[4] The analyst places the Excel file in a secure network folder that is accessible only to the requesting user. Importantly, the data extract is never sent as an E-mail attachment.

The presence of an independent analyst in the research data extract workflow was required by our legal department and serves as a "gatekeeper" between the user and the data. Currently, we have a pool of four analysts who process these requests, and are the same analysts who fulfill search requests requiring our production LIS, Copath. The analysts are independent, in that they are not subordinate to the pathologists or trainees in any way, and are employed by a different section of our institute. Finally, our legal advisor did not require a gatekeeper in the Preview Mode search workflow, as patient identifier fields are not visible to the user.

Analyst Only Workflow

End-users may not wish to design searches directly. Many users prefer to submit search requests through our legacy search-request web portal, which is the traditional way a Copath search would be initiated. Using a web-form - which is unrelated to Pathtools - the user describes the data they want, the date range of their search, and can upload a pdf copy of their IRB-approved data collection sheet. These search requests are forwarded to the analyst team, who are the same analysts who fulfill Pathtools requests. Once the request arrives, the individual analyst can decide which search tool is best for fulfilling the request. If they chose Pathtools, they use "Research Mode" to design the search, similar to how an end-user would. Finally, they execute the search request, download the data extract, and place the file in a secure network location accessible only to the end-user.

Assembly of Case Sets by Patient or Accession Number

Besides text-based searches, Pathtools also allows the user to build Case Sets by supplying lists of patients or accession numbers. This functionality is important for at least two use-cases. First, an investigator may have a list of patients that are known to have some particular disease, and want to identify the pathology cases for these patients. For this, Pathtools has a "Patient/Case Upload Portal" page which features a large text box, into which, the user may paste a list of patients, using either MRNs or Name/DOB combinations [Figure 4]. The user selects which type of identifiers they are providing using a drop-down list and submits the query. Pathtools will search the database and identify all cases associated with the list of patients. The system will tell the user how many matching

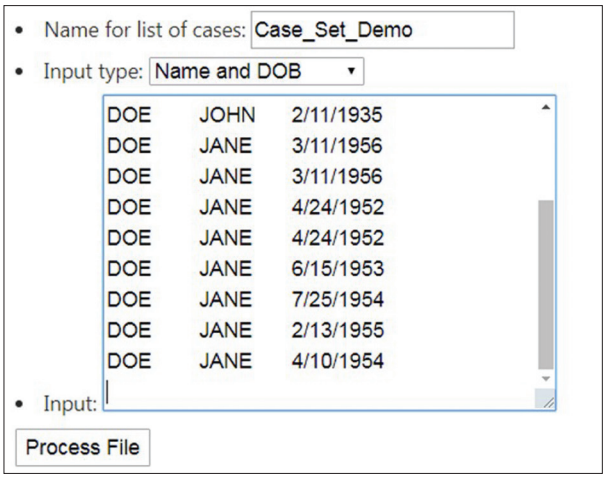

Figure 4: Patient/Case Upload Portal. The user may build Case Sets by supplying lists of patients or case accession numbers. "Input type" can be set to MRN, Name/DOB combinations, or accession numbers

cases were found, and which patients had no matching cases in the system, if any. The list of cases is saved as a Case Set and will be available in the Case Set Dashboard.

For a second use-case, an investigator may have a set of accession numbers compiled from a different source (e.g., Copath and research databases) but want to obtain a data extract using Pathtools. For this use, the process is similar and also uses the "Patient/Case Upload Portal" page. The user pastes a list of accession numbers into the text box, and selects "accession numbers" from the drop-down list. Pathtools will identify all matching cases in the system, and notify the user which cases could not be located, in any. The list of cases is saved as a Case Set and will be available in the Case Set Dashboard.

Database Design and Population

The database was built into Microsoft SQL Server 12.0 and is hosted in our institution's secure datacenter. The database consists of a single SQL table, which uses the case's accession number as the primary key, and has 13 fields for each case [Table 1]. A full-text index was generated containing the following columns: Diagnostic Text (which comprises final diagnosis, comment and addendum text), Gross Description, Clinical Information, Synoptic, and Intraoperative Diagnosis.

The database was populated with data extracts from Copath. Unfortunately, Copath has no utility for large scale bulk data extraction. Copath does have a variety of built-in utilities that generate reports for operational purposes and a collection of utilities that perform natural language searches. Given these options, we chose to use a natural language search utility. While not well-suited to the purpose, it produced data in tab-delimited text format, which could be easily processed. The search parameters had to be carefully calibrated to capture a limited number of cases with each run – exceeding about 3000 cases

would sometimes cause the search to fail, and the Copath client to close. Therefore, each individual search targeted a specific case-type and date range. After each search, the date range would be adjusted to capture the next time period. Using this approach, cases were downloaded in reverse-chronological order from 2019 to 1980. Several "rounds" of searches took place over the course of several months. Surgical pathology cases were captured first, followed by cytology and bone marrow cases. While the search utility was designed for natural language searches, the text parameter was left blank, so that every case in the targeted date range would match. The search utility produced tab-delimited text files. These files were processed with a pipeline of Python scripts, which parsed out the targeted data elements and generated SQL statements to insert the data into the database.

Database Update Technical Details

It is important for the Pathtools database to be up to date, so users can search for recent cases. To do this, we used Copath build-in reporting functions to create automated daily extracts of surgical pathology, cytology and bone marrow cases. The reports are run nightly and the data is downloaded to a secure network location. Then, an automated process (initiated by Linux crontab), processes the files and uploads the data to the SQL database.

Pathology reports often change over time in Copath. For example, the initial report in the system may be the resident's diagnosis. The report will probably change somewhat after the staff pathologist sees the case, before signing it out. After sign-out, the case may acquire one or more addendums. Finally, the case may be amended at any point in the future. Therefore, the database update workflow should account for all of these scenarios, and attempt to pull in the most recent information. To address this, three different sets of Copath reports are run nightly that capture cases in three different date "windows." The first set captures cases accessioned in the past 4 days (the newest cases). The second sets captures cases accessioned from 30 to 34 days previous. The last set captures cases accessioned 150–154 days previous. The limit of a 4-day window is imposed by the Copath search utility, as searching for a larger date range sometimes causes the search to fail. This workflow is designed to capture each case multiple times, as the case passes through each "window." For each case, the process checks to see if the case is already in the database. If the case is new, the data is added using the SQL "INSERT" command. If the case is already present in the database, the old data are replaced by the new data with the "UPDATE" command. It is important to note that there may be a significant lag-time between a change to the case in Copath, and when this change is reflected in Pathtools, depending on its relation to each "window". Furthermore, changes that occur to the case after 154 days will not be detected by Pathtools.

Registration and Authentication

To access Pathtools, users must register for an account, which is a three-step process. A user visits the registration page and fills out a form. An E-mail is sent to the user's institutional E-mail account containing a confirmation link, which they must accept. Finally, a confirmation E-mail is sent to a site admin (currently the only site admin is the author), who must also confirm the account creation. This final step is necessary to ensure that only appropriate personnel gain access to the site. Site access is restricted to pathology staff, trainees, and researchers working within the department and who already have access to our LIS.

Django's authentication suite was augmented and customized to comply with our institution's authentication and password management policy – which specifies various details pertaining to password expiration, password history, password quality, and lock-out functionality. Django's built-in code was sufficient to support some of these policies: Password length requirement, password quality requirements, and password validation against a dictionary of weak passwords. However, a third-party module was needed to implement our password history policy (password cannot match user's last five passwords). Finally, custom modification of Django's local source code was necessary to enable lock-out functionality (a user is locked-out after 10 failed attempts) as well and our password expiration policy (90 days).

Pathtools Benchmarking Against Copath

A modified version of Copath's build-in "InfoMaker Wizard for Natural Language Download" utility was used to search for cases containing the term "stomach" in either the final diagnosis, diagnostic comment or addendum fields. The "text type to return" parameter was also set for final diagnosis, diagnostic comment and addendum. The date-range (accession date) was manually adjusted for each test. The remaining parameters were left in their default state (matching all values). In Pathtools, Preview Mode was used to design a similar search ("stomach" in either the final, comment or addendum). However, for this test, the Pathtools search cap was removed ("TOP [100]" SQL clause) so that all matching cases would return from the database, making it more comparable to the Copath search. Twelve different date-ranges were tested for each system: 1 week, 2 weeks, 3 weeks, 6 weeks, 2 months, 4 months, 6 months, 8 months, 10 months, 12 months, 16 months, 24 months. Three trials were performed for each data-range, in each system. Mean and standard deviation (SD) were calculated. Student's *t*-test was used to compare Pathtools against Copath for each date-range, with statistical significance considered $P < 0.05$.

Results

Pathtools is a very frequently used resource in our department. Pathtools was deployed on February 19, 2019, and has carried out 33,817 searches in Preview Mode as of July 31, 2020. Preview

Mode searches are quick, averaging 0.72 s (SD = 1.7) between search initiation and results return. Pathtools is also frequently used for research. From February 19, 2019 to July 31, 2020, Pathtools generated 338 data extracts for research. These data were used to generate multiple abstracts for USCAP 2020.[5-7]

For Research Mode, detailed metrics (e.g., turnaround time) are only available for cases since the beginning of 2020 (comprising 154 data extracts). Thirty-six percent of these data extracts were initiated by end-users using Research Mode. For requests submitted during the analyst's normal working hours (7 a.m. to 4 p.m.), the median turnaround time (from data extract request to results delivery) was only 36 min (n = 41). Only four requests required >24 h to fulfill. For cases submitted in the evening, or on the weekend, the request was fulfilled the next business day. Interestingly, most of the data extracts (64%) originated from analysts working in the "analyst only workflow" (searches designed by an analyst on behalf of an end-user). For these requests, the turnaround time could not be calculated, as the system which collects these requests is outside of Pathtools, and does not keep detailed event logs.

We analyzed how many cases are in Pathtools compared to Copath, to determine how representative Pathtools' database is of Copath's. We compared the number of cases in each database from 1980 to 2019 [Figure 5]. The analysis focused on surgical pathology, bone marrow and nongynecologic cytology cases. Gynecologic cytology specimens were excluded from analysis. This is because the Copath utility used for data extraction only captures cases that were signed-out by a staff pathologist, and does not extract gynecologic cytology specimens that were released by a cytotechnologist (e.g., conventional pap smears and ThinPrep pap tests). Therefore, these cases are known to be absent from Pathtools. For the remaining cases, we found that Pathtools has 3.86 million cases compared to Copath's 4.5 million cases (86%). Pathtools has the best coverage in the most recent years (e.g., 98% in 2019), with coverage worsening gradually for years extending further in the past. The worst year is 1986, in which Pathtools has only 867 of 4021 cases (22%).

We benchmarked the performance of Pathtools against Copath in executing a text search. This task was meant to measure the computational processing time of each system, in other words, the time it takes for the back-end database system to process a search query, and return results to the requesting front-end program. Searches were designed in each system to identify cases containing the word "stomach" in the diagnostic text (final diagnosis, comment, and addendum text) in a specific date-range. To measure performance over a wide variety of data-set sizes, twelve different date-ranges were measured, ranging from 1 week to 2 years. For each test, the duration of the search, and the number of matching cases were recorded. Pathtools outperformed Copath in almost all tests, proving significantly faster ($P < 0.05$) than Copath in all date ranges >1 week [Figure 6]. Copath could not complete the search for the longest date-range (24 months) as the client closed after several minutes of searching in each attempt. In contrast, none of the Pathtools searches failed. Finally, Pathtools matched more cases than Copath at each date-range; overall retrieving 52% more cases than Copath.

Search requests by trainees (residents and clinical fellows) make up the majority of the searches, comprising about 75% of overall search activity. The trainees were anonymously surveyed to determine how often they use Pathtools during their preview time and how useful they think Pathtools is an educational aid [Table 2]. Out of 18 respondents (a mix of residents and clinical fellows), 44% use Pathtools more than 90% of their preview days, 39% use it in 50%–90% of preview days while 17% use it less than half of preview days. All 18 respondents "strongly agreed" or "agreed" that Pathtools "helped me write diagnoses that require less editing by the attending pathologist" and that "Pathtools is an important educational aid."

Discussion

The main function of the LIS is to manage a laboratory's workflow and deliver timely results for clinical management.[8] However, the LIS should also be able to perform queries of

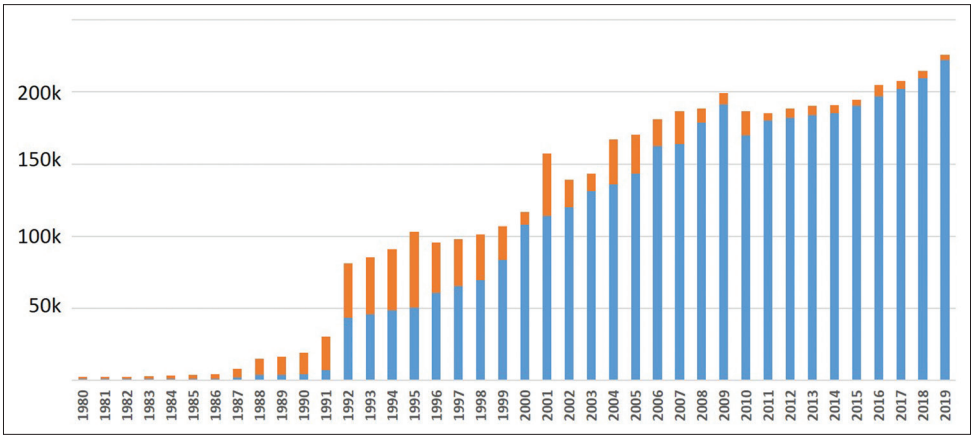

Figure 5: Comparison of the case numbers in Pathtools compared to Copath. Tops of orange bars indicate number of cases in Copath. Blue bars show number of cases in Pathtools. Analysis restricted to surgical pathology, nongynecologic cytology, and bone marrow cases

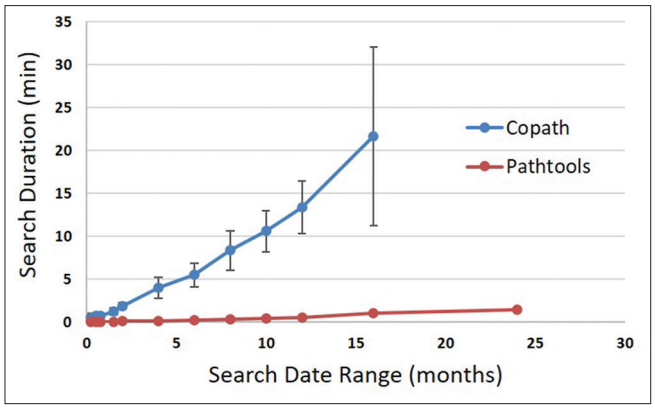

Figure 6: Benchmarking Pathtools against Copath in similar search task. The time to complete a search task was determined for 12 different date-ranges (from 1 week to 24 months). Three trials were conducted for each date-range. Error bars indicate standard deviation (error bars too small to be visible for Pathtools data points). Copath unable to complete 24 months search

Table 2: Pathtools trainee user survey (*n*=18)

Respondents	*n* (%)
How often do you use Pathtools when previewing AP cases?	
Never	0
<50% of preview days	3 (17)
50%-90% of preview days	7 (39)
More than 90% of preview days	8 (44)
Pathtools helps me write diagnoses that require less editing by the attending pathologist	
Strongly agree	12 (67)
Agree	6 (33)
Neither agree nor disagree	0
Disagree	0
Strongly disagree	0
Pathtools is an important educational aid	
Strongly agree	14 (78)
Agree	4 (22)
Neither agree nor disagree	0
Disagree	0
Strongly disagree	0

Survey conducted anonymously via surveymonkey.com. Survey invitation sent to 42 trainees with 18 responses (43%). AP: Anatomic pathology

the database, which is essential for report-building, tracking QA/QC indicators, supporting data mining operations, and case-finding for research and educational purposes.[9] With our current LIS, we noted a distinct "functionality gap" in its ability to deliver case-finding functionality to the end-user.[10] Therefore, the aim of this project was to use open-source software to fill this gap, providing our staff pathologists, trainees, and research investigators with the ability to rapidly query the anatomic pathology case archive.

Pathtools is an improvement over our previous system, which relied on Copath's built-in search functions. Most importantly, Pathtools searches are significantly quicker than Copath searches. Our benchmarking tests showed that computational processing time is faster in Pathtools over Copath. Second, we observed that end-users using "Research Mode" to design searches and request data extracts would receive their results in a median time of 36 min. Unfortunately, this could not be directly compared with our previous data extraction workflow using Copath, as turnaround time is not tracked. However, speaking anecdotally, turnaround time is usually >24 h.

In the research context, speedy searching allows the investigator to quickly estimate how many cases of a particular diagnostic entity are in our archive. This shortens the time between an investigator's initial hypothesis, and an estimate of the number of relevant cases in the system. Our investigators can iterate through many research ideas quickly, to find ones that are likely to be the most fruitful. Furthermore, Pathtools is available directly to end-users, who can design searches themselves. In contrast, Copath's search functions are restricted to a team of analysts at our institution, who design and execute searches on behalf of the pathologist. Sometimes, this arrangement would result in an extended "back and forth" between the analyst and the pathologist, as the search would need to be optimized several times to capture the desired results. This process was often inefficient, leading to the creation of many unused date-sets, and consuming the time of the pathologist and the analyst alike.

In an educational context, rapid searching is also important. Our trainees have limited preview time each day (sometimes just 1 h a day). Therefore, searches need to be quick, to match the pace in which the trainee previews cases. In Preview Mode, the trainee can execute a search and receive results in <1 s. This activity is not possible with Copath, as the search functions are not accessible by our trainees. However, even if this were not the case, Copath searches are much slower and are ill-suited for this specific use-case.

Curiously, the benchmarking tests showed that Pathtools returned more cases than Copath in a similar search task. For example, in the 16-month test, Pathtools returned 21,364 cases containing the term "stomach" while Copath returned only 16,869 cases. This is particularly unexpected given that Pathtools was entirely populated with data extracted from Copath. Examining these results in more details, we found that the specific Copath natural language search utility we used for benchmarking did not match cases originating from our Florida campus, explaining the size discrepancy between the two data-set sizes. This was not expected behavior for this utility, as it should match cases regardless of "region." While we were unable to correct this problem, given that Pathtools matched more cases at each date-range, it does not argue against our conclusion that Pathtools is the faster system.

Interestingly, searches by trainees make up 75% of all search activity. Almost all of these searches happen during a trainee's preview time. At our institution, trainees have dedicated time to preview cases prior to sign out with the attending. Our trainees are encouraged to preview as many cases as they

can, and enter their diagnoses into the LIS. Our trainees use Pathtools during their preview time as an educational aid, one that helps them craft their diagnostic text. This points to important fact of our profession. Namely, a pathologist's ability to write diagnostic text is a key aspect of our jobs, as this is our primary mode of communication with the clinician. It is not merely sufficient to recognize a diagnosis, but it is also essential that we communicate this clearly to the clinician in writing. Furthermore, pathology diagnoses are not always definitive, so a pathologist needs to select words carefully, to imbue the text with the appropriate confidence. A well-written diagnosis leads to efficient patient care. A poorly-worded one generates confusion and the possibility of inappropriate treatment. Unfortunately, there are few resources that help trainees develop this particular skill. While numerous books are generally close at hand to help the trainees with morphologic and clinicopathologic descriptions of various pathologic entities, writing or dictating the diagnosis is often a matter of trial and error. For the pathology trainee, this is a daunting task, and the learning curve is steep. The ability to rapidly query the diagnostic archive is a great help in this regard, and Pathtools has become a resource our trainees rely on.

The prime weakness of Pathtools is that its database is only a partial copy of our clinical LIS database. There are two reasons for this. First, the data model for Pathtools is very simple, comprising a single SQL table. The data model works well for surgical pathology cases with fields for diagnostic text (which is composed of the final diagnosis, comment, and addendums), gross description, synoptic text and intraoperative diagnosis. The model also works for most cytology and bone marrow cases (though none of these use the synoptic or intraoperative diagnosis fields). However, some case-types, like autopsy reports, do not fit into this scheme. Similarly, Copath "procedure reports" also have a distinct data structure (we use this case type to attach the results of molecular or ancillary testing). Finally, the Copath utility that was used for data extraction did not match cytology cases that were released directly by a cytotechnologist without pathologist review (e.g., conventional pap smears and ThinPrep pap tests). A significant amount of additional work would have been required to adapt the data model to accommodate these additional cases and we did not have the resources to do this.

Second, even for targeted case-types, some cases were not successfully extracted; Pathtools has only 86% of the cases of Copath for targeted case-types. This occurred for several reasons. First, a pipeline of Python scripts processed the tab-delimited files obtained from the Copath search utility, and uploaded them to the SQL database. Some cases fail this procedure for various reasons, including nonstandard patient name format, unexpected accession number format, and the presence of characters within the text which are used as delimiters by the processing scripts. If an error occurred in any of these steps, the processing algorithm would skip the case. Furthermore, the data extraction process seemed to perform worse in older cases, with a gradual worsening in case yield going back in time. It is unclear if these older cases failed because of errors in the text parsing pipeline, or if the Copath search utility did not reliably extract this older data.

Interestingly, our institution is investing considerable resources in developing an enterprise data vault (EDV) which should address this weakness. This is an effort to centralize and integrate data from multiple operational systems across our enterprise. In the near-future, our pathology data (both clinical and anatomic) will be comprehensively integrated into the EDV. The EDV is implemented in Teradata (San Diego, CA, USA), a SQL-type relational database system. EDV tables can serve as a centralized source of data for a wide variety of applications across our enterprise. Therefore, Pathtools could be adapted in the future to query the EDV, rather than using its own stand-alone database. This would be an improvement, enabling the user interact with our institution's most reliable and comprehensive data source, thus, addressing Pathtool's biggest shortcoming.

A second drawback of Pathtools relates to its simplistic search method, which does not take advantage of Natural Language Processing (NLP). Specifically, Pathtools text searches rely on the SQL predicate "CONTAINS" which returns only exact matches to the user's search terms-though matching is not case-sensitive. For example, a search for "Crohn's disease" will not match "Crohn disease." A search for "signet ring cell adenocarcinoma" will not match "signet ring cell carcinoma." Pathtools could be improved by leveraging NLP methods to improve the yield of relevant search results and to filter out unwanted results. For example, a medical ontology, a database of related concepts including their various names and relationships, could be used to identify synonyms for the user's search terms.[11] Therefore, a user could search for "Crohn's disease" and the system would know to include results which mention "Crohn disease." Second, "negation detection" could be used to filter out search results that contain a negation of the user's search term.[12] For example, a user may want to find cases of Barrett's esophagus with dysplasia. However, searching Pathtools with the terms "Barrett's" and "dysplasia" will identify any Barrett's case which contains the string "negative for dysplasia", which is not what the user wants. Perhaps, the next iteration of Pathtools will take advantage of advancements in NLP to produce a more powerful search tool.

Conclusion

Pathtools is a web-based search platform optimized for the academic pathologist. Most importantly, the platform is easy to use and fast, delivering search results in seconds. This is a heavily-used resource in our department. The majority of searches are performed by our trainees, who use it during their preview time to help them write diagnoses. Pathtools also supports departmental research activity and has generated data sets for a variety of projects.

Financial support and sponsorship

Nil.

Conflicts of interest

There are no conflicts of interest.

References

1. Django; 2013. Available from: https://djangoproject.com. [Last accessed on 02 Jan 2020].
2. Python Software Foundation. Python Language Reference, version 3.5. Available from: https://www.python.org. [Last accessed on 06 Jan 2019].
3. jQuery QueryBuilder. Available from: https://querybuilder.js.org/. [Last accessed on 06 Jan 2019].
4. McKinney W. Data structures for statistical computing in python. In: Proceedings of the 9th Python in Science Conference. Austin, TX; 2010. p. 51-6.
5. Blank A, Cox R, Doxtader E, Fuller L, McKenney J, Mukhopadhyay S, *et al*. Morphologic patterns of prostate cancer metastases to the lung in cytology and histology: A review of 30 case. Mod Pathol 2020;33:460-601.
6. Shetty S, Joehlin-Price A, Habeeb O. Immunohistochemistry for MDM2 is a useful tool in the diagnosis of malignant brenner tumors? Mod Pathol 2020;33:1164-337.
7. Thomas M, Robertson S, Joehlin-Price A. Size of lymph node metastasis significantly affects overall survival but not recurrence-free survival in a cohort of 182 single-institution endometrial carcinomas. Mod Pathol 2020;33:1164-337.
8. Cucoranu IC. Laboratory information systems management and operations. Surg Pathol Clin 2015;8:153-7.
9. Park S, Parwani AV, Aller RD, Banach L, Becich MJ, Borkenfeld S, *et al*. The history of pathology informatics: A global perspective. J Pathol Inform 2013;4:7.
10. Gershkovich P, Sinard JH. Customizing laboratory information systems: Closing the functionality gap. Adv Anat Pathol 2015;22:323-30.
11. Cai T, Giannopoulos AA, Yu S, Kelil T, Ripley B, Kumamaru KK, *et al*. Natural language processing technologies in radiology research and clinical applications. Radiographics 2016;36:176-91.
12. Yim WW, Yetisgen M, Harris WP, Sharon WK. Natural language processing in oncology review. JAMA Oncol 2016;2:797-804.