Article

Health Informatics Journal

Health Informatics Journal
2014, Vol. 20(4) 288–305

DOI: 10.1177/1460458213494032
jhi.sagepub.com

University of California, Irvine–Pathology Extraction Pipeline: The pathology extraction pipeline for information extraction from pathology reports

Naveen Ashish, Lisa Dahm and Charles Boicey
University of California, Irvine, USA

Abstract

We describe Pathology Extraction Pipeline (PEP)—a new Open Health Natural Language Processing pipeline that we have developed for information extraction from pathology reports, with the goal of populating the extracted data into a research data warehouse. Specifically, we have built upon Medical Knowledge Analysis Tool pipeline (MedKATp), which is an extraction framework focused on pathology reports. Our particular contributions include additional customization and development on MedKATp to extract data elements and relationships from cancer pathology reports in richer detail than at present, an abstraction layer that provides significantly easier configuration of MedKATp for extraction tasks, and a machine-learning-based approach that makes the extraction more resilient to deviations from the common reporting format in a pathology reports corpus. We present experimental results demonstrating the effectiveness of our pipeline for information extraction in a real-world task, demonstrating performance improvement due to our approach for increasing extractor resilience to format deviation, and finally demonstrating the scalability of the pipeline across pathology reports for different cancer types.

Keywords

Clinical decision-making, databases and data mining, decision-support systems, evidence-based practice, information and knowledge management

Objective

Clinical research requires cohort identification, typically in terms of very precise and detailed inclusion and exclusion criteria based on patient demographics, as well as disease and diagnosis characteristics. Much of the raw information needed for cohort identification resides in text documents such as pathology reports or clinical notes and is in a format that is called *unstructured* data,

Corresponding author:
Naveen Ashish, The Institute for Neuroimaging and Informatics, Keck School of Medicine of USC, University of Southern California, 2001 North Soto Street - Room 102, Los Angeles, CA 90032.
Email: nashish@loni.usc.edu

that is, as natural language text or notes. The importance of capabilities for tasks such as cohort identification has driven the development of automated information technologies[1–8] for converting unstructured collections of text such as pathology reports or clinical notes into more structured representations. A retrieval capability on top of such a structured representation would be very useful to researchers, where one could, for instance, ask for

> "Patients with a surgical pathology report containing undifferentiated lymphoepithelioma-like gastric carcinoma." or for "Patients with a surgical pathology report containing spindle cell carcinoma of the breast, grade 3, margin(s) positive, node(s) positive."

We present an end-to-end approach and implemented solution to this problem, with an implementation in the context of a large clinical data warehousing effort at the University of California, Irvine (UCI) Medical Center, keeping an initial focus on pathology reports in the cancer domain. Our solution is an automated information extraction pipeline called the Pathology Extraction Pipeline (PEP) that takes pathology reports in semistructured text format, extracts the fields of interest, and populates a data warehouse with the extracted data. The pipeline itself has been developed within the open-source Unstructured Information Management Architecture (UIMA)[9] framework and as part of the Open Health Natural Language Processing (OHNLP) (Open Health Natural Language Processing Consortium: https://cabig-kc.nci.nih.gov/Vocab/KC/index.php/OHNLP), a community focused on health informatics. We have built upon relevant information extraction *resources* offered by OHNLP and contributed our enhanced solutions back to OHNLP. While we have leveraged existing OHNLP capabilities, we have also significantly enhanced the state-of-the-art in realizing the PEP. Specifically, our contributions include the following:

- A fully functional and operational pipeline for information extraction of detailed information elements from cancer pathology reports.
- An algorithm and implemented technique to enable *extractor resilience*—where for certain kinds of fields, the extractor is actually resilient to minor deviations from an expected format of text.
- Experimental results in the context of a real-world information extraction task where we have assessed several aspects including extraction accuracy, scalability, and performance improvement.

Background and significance

The data

A pathology report from our corpus is illustrated in Figure 1. Such reports are *partially* structured in that different sections and subsections have been delineated with section headings and paragraphs and labeled appropriately. The color annotations in the figure indicate some of the many fields that we are interested in extracting from such pathology reports. The set of these fields is derived from the cancer protocol published by the College of American Pathologists (CAP)[10] that establishes the detailed information items that must be recorded and present in such reports. Figure 1(b) shows a model (in JavaScript Object Notation (JSON)) of the various fields we want to extract for pathology reports from breast cancer. As we can see such fields range from items that are explicitly labeled and present as the content of different sections and subsections in the text (for instance, fields such as Microcalcifications and Pathologic Staging) to items that are embedded

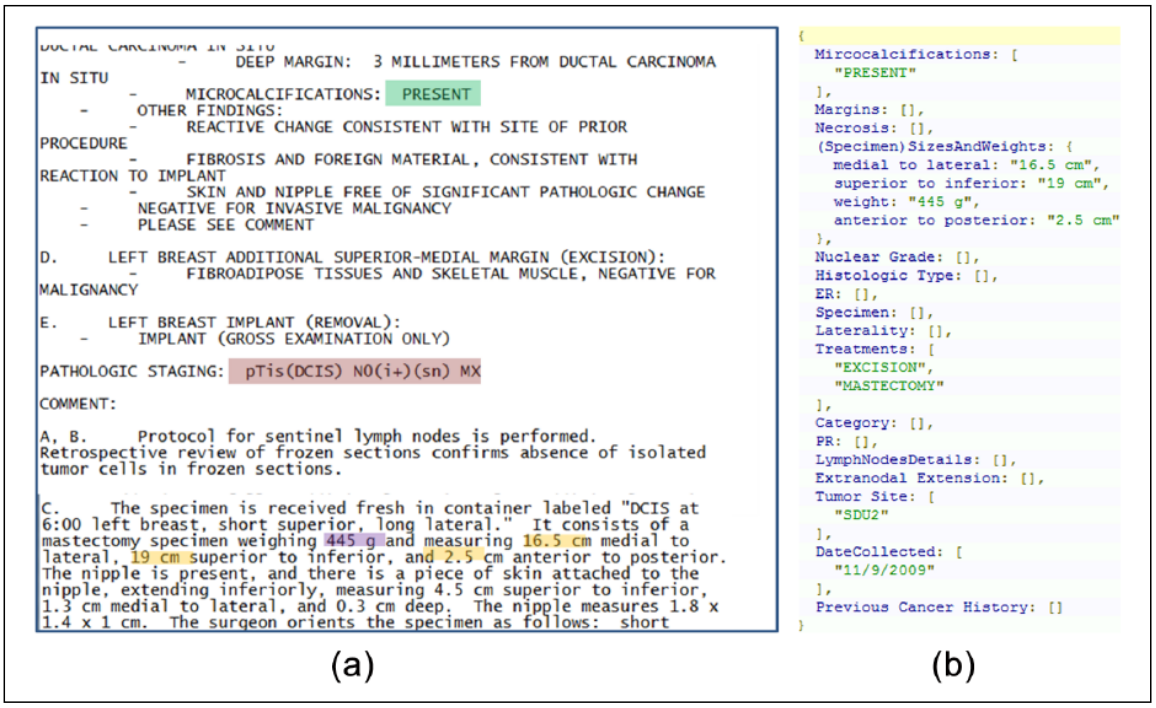


Figure 1. (a) Pathology report and (b) extraction model.

within the natural language text paragraphs (e.g. the various dimensions and weight of a specimen). The task of the PEP is to take a pathology report as shown in Figure 1(a) and automatically derive a structured representation, as shown in Figure 1(b), which can then be used to populate a structured database.

Applicable tools

We now describe the primary frameworks and systems that PEP is built upon.

UIMA is a general purpose information extraction development platform that allows developers to build custom information extraction applications. UIMA's basic building blocks are *annotators*. Annotators are modules that analyze text. The fundamental data structure in UIMA is the *common analysis structure* referred to as "CAS." The CAS is an object that provides access to the artifact being analyzed, that is, the text, and also the current analysis that has been extracted from it. An annotator takes CAS objects as input, and the output is also a CAS object that contains the metadata distilled by that annotator. For instance, an annotator could take a CAS object that has a text artifact of an entire postal address and annotate that to provide another CAS object that has the address elements such as street, city, state, zip, and country individually annotated. Figure 2 provides an illustration of the type system for this example. uima.tcas.Annotation is a *built-in* UIMA type and the uima.tcas.address annotation is an annotation that *extends* from uima.tcas.Annotation. The various fields in an address, namely, street, city, state, zip, and country are *attributes* of the uima.tcas.address annotation.

Medical Knowledge Analysis Tool pipeline (MedKATp[1]) is a system developed specifically for automatically populating the primary elements of the Cancer Disease Knowledge Representation Model[1] from pathology-free text reports. This cancer Knowledge Representation Model[1] has the

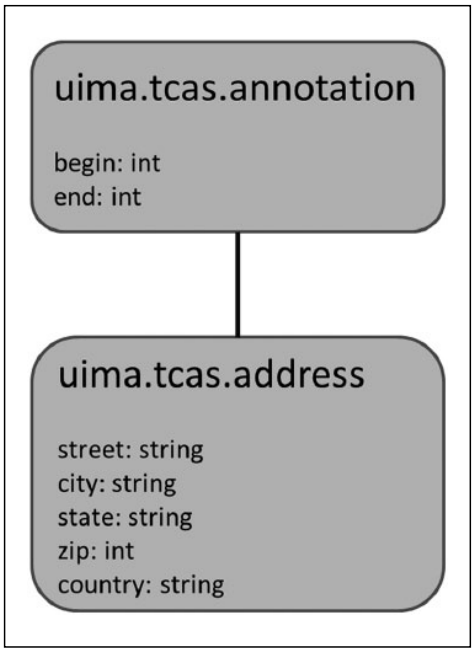


Figure 2. Type system illustration.

classes of anatomical site, histology, grade value, dimension, and stage,[1] and each class in turn has multiple attributes describing the class details. MedKATp is built upon the UIMA framework. In addition to the cancer knowledge representation model, MedKATp provides certain implemented resources for automatically extracting elements such as information on tumor sites, cancer grades, lymph node details, histology information, and the mention of numerical quantities in pathology reports. While these elements as such do not form the detailed fields that we are interested in extracting from the reports, they form an excellent initial basis for the details that we are interested in finally extracting. We have thus chosen to realize PEP by building upon MedKATp for the following key reasons:

- The cancer knowledge representation model of MedKATp, as well as some resources in the system that can actually populate elements of that model, form a good starting point for the fields we want to extract.
- We can further customize as well as add to MedKATp using the annotator configuration and development capabilities provided by UIMA.
- Any resource customization and additional development that we do over MedKATp (and thus in UIMA) can be used as a resource by the MedKATp and UIMA (OHNLP) community.

Materials and methods

As MedKATp already comes with resources for populating some elements of the cancer knowledge model, we consider its applicability for our task in detail. The fields we want to extract fall in three categories with respect to the MedKATp applicability:

Category 1. Where the fields can be extracted as we want, by MedKATp as is.

Table 1. MedKATp applicability.

Category	1	2	3
Fields	Lymph nodes sampled	Capsule invasion	Gleason score
	Lymph nodes detected	Lymph invasion	TNM stage
		Chronic inflammation	Laterality
		Vascular invasion	Extraprostatic extension
		Perineural invasion	PIN
		Surgical margin	Specimen length
		Seminal vesicle	Specimen width
		Necrosis	Specimen height
		Category	Specimen weight
		ER	
		PR	

MedKATp: Medical Knowledge Analysis Tool pipeline; TNM: tumor, node, metastases; PIN: prostatic intraepithelial neoplasia.

Category 2. Where MedKATp provides a resource that can extract that kind of field, however, MedKATp configuration is required (details below).
Category 3. Where we have to customize a MedKATp annotator or write additional annotators to extract the fields.

Table 1 provides a list of (22) fields of interest we want to extract from cancer pathology reports, where the fields are grouped into different categories depending on the applicability of MedKATp to extract them.

The development of our complete solution involved the following four phases:

1. *MedKATp application*. We set up MedKATp to extract the fields it can, as is or with configuration.
2. *Custom annotators*. We then added some custom annotators, based on natural language processing (NLP) and pattern analysis, to extract the fields we could not in the initial phase.
3. *Abstraction layer*. We added an *abstraction layer* to make it easy for developers to configure the system to extract new fields of interest.
4. *Resilience*. We developed and implemented an approach to make the extractor more resilient to format deviations.

We describe these phases in more detail.

Application

MedKATp already contains resources to extract lymph node details. Thus, for Category 1 field, we used the MedKATp resource as is. MedKATp further has generic resources for annotators for tasks such as recognizing sections and subsections in the text.

Figure 3 shows the (extensible markup language (XML)) configuration file in MedKATp for configuring the section extraction annotator. The fields in Category 2 can also be extracted if we can extract the appropriate sections and subsections in the text. We thus configured the section

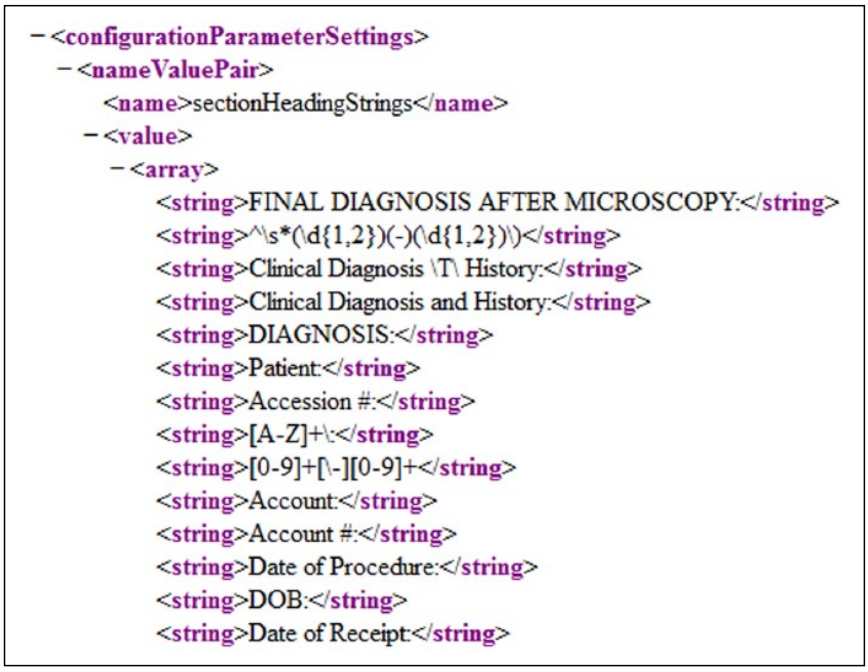


Figure 3. MedKATp configuration file.
MedKATp: Medical Knowledge Analysis Tool pipeline.

extractor in MedKATp with the section label names and patterns appropriate for our particular dataset.

Custom annotators

In the second phase, we added additional annotators and capabilities to the extraction pipeline to address the above limitations. These include the following:

Association formation (annotator). Table 2 illustrates (exactly) the 10 different ways we have found specimen details to be described in our corpus of reports.

Our overall approach, described in algorithm Associate Value Dimension, is to (1) extract the (single and unambiguous) weight value and associate it with the weight measure, (2) to exploit "X" demarcators where present in the sentence, and (3) to associate values with dimensions by analyzing the parse tree of the sentence with the specimen details. We have employed the Stanford Parser[11,12] for this purpose. For the sentence "It consists of a mastectomy specimen without axillary tail weighing 407.7 g and measuring 17.5 cm medial to lateral, 14.5 cm superior to inferior and 3.5 cm anterior to posterior." the parser provides a parse tree in *a typed dependency collapsed relational representation*[13] as shown below.

Algorithm Associate Value Dimension provides the pseudo-code representation of the specimen value dimension association, which has the following key steps: (0) split sentence based on "X" demarcators if available and associate value dimension in individual split portion(s) (lines 1–2 in pseudo-code); (1) parse the (remainder) sentence to create a typed dependency collapsed representation of the parse tree (line 6); (3) create a graph representation where the relation instances from the parse tree relational representation that have a token that is either (a) a numerical value or (b)

nsubj(consists-2, It-1)	amod(lateral-19, medial-17)
root(ROOT-0, consists-2)	dep(lateral-19, to-18)
det(specimen-6, a-4)	amod(cm-16, lateral-19)
amod(specimen-6, mastectomy-5)	num(cm-22, 14.5-21)
prep_of(consists-2, specimen-6)	dobj(measuring-14, cm-22)
amod(tail-9, axillary-8)	conj_and(cm-16, cm-22)
prep_without(consists-2, tail-9)	amod(inferior-25, superior-23)
dep(tail-9, weighing-10)	dep(inferior-25, to-24)
num(g-12, 407.7-11)	amod(cm-22, inferior-25)
dobj(weighing-10, g-12)	num(anterior-29, 3.5-27)
dep(tail-9, measuring-14)	nn(anterior-29, cm-28)
conj_and(weighing-10, measuring-14)	dobj(measuring-14, anterior-29)
num(cm-16, 17.5-15)	conj_and(cm-16, anterior-29)
dobj(measuring-14, cm-16)	prep_to(anterior-29, posterior-31)

Table 2. Specimen details.

The specimen consists of one prostate weighing 106.1 g and measuring 4.1 cm from apex to bladder neck, 6 cm from left to right, and 3.7 cm from anterior to posterior.
The prostate weighs 35.4 g and measures apex to bladder neck 7.5 cm × left to right 4.5 cm anterior to posterior 2.5 cm.
It consists of a prostatectomy specimen weighing 70.1 g and measuring 3.7 cm apex to bladder neck × 6 cm left to right × 3.5 cm anterior to posterior.
The prostate weighs 53 g and measures apex to bladder neck 3.3 cm, left to right 5.4 cm, and anterior to posterior 3 cm.
The prostate weighs 30.7 g and measures apex to bladder neck 2.6 cm × left to right 3.5 cm × anterior to posterior 2.7 cm.
The prostate weighs 65.2 g and measures 3.5 cm apex to bladder neck, 4.8 cm left to right, and 3.5 cm anterior to posterior.
The specimen consists of one prostate weighing 66.4 g and measuring apex to bladder neck 3 cm, from right to left 6.5 cm, and from anterior to posterior 4 cm.
The specimen weighs 35.3 g and the prostate measures apex to bladder neck 2.6 cm, left to right 3.5 cm × anterior to posterior 3 cm.
The specimen weighs 35.3 g and the prostate measures apex to bladder neck 2.6 cm, left to right 3.5 cm × anterior to posterior 3 cm.
The specimen consists of one prostate measuring from apex to bladder neck 3.6 cm, from left to right 4 cm, and from anterior to posterior 3 cm.

the dimension name (or part of) form nodes of the graph (9–13); (4) create edges between the nodes that have common unit or dimension (part) tokens (15–21); (5) extract associations from the graph (23–31).

The final step of extracting associations is illustrated in the graph accompanying the pseudo-code, where we first associate values and dimensions that appear together in the same node in the graph (node 1, line 24 in pseudo-code), then associate values and dimensions that appear in a pair of nodes connected by an edge in the graph (edge 2, line 28), and finally associate any remainder values to remainder dimensions (node 3, line 31).

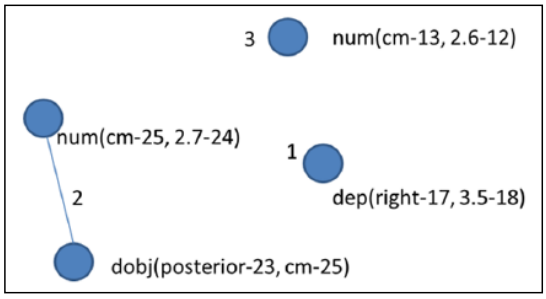
The approach is similar to the RelEx system[14] for extracting interactions between genes and proteins from free text in biomedical literature. RelEx is also a system that extracts information

ALGORITHM: Associate Value Dimension

INPUT: S = sentence containing specimen weight and size details.
OUTPUT: Correctly associated values and dimensions.

/* numNode, is a node which has a numerical measure eg., num(cm-16, 17.5-15) */
/* whichMeasureNode is a node that contains (part of) the dimension name, eg., dep(lateral-19, to-18)*/

```
1.  W← assignSpecimenWeight(S)
2.  VD = VD U {W}
3.
4.  splitAndAssociate("X",S)
5.
6.  parseTreeRelational ← parse(S)
7.  G ← createGraph(parseTreeRelational)
8.
9.  for (each node N ε G) {
10.       markedNode ← markNodes(N)
11.                 //mark nodes as numNode, whichMeasureNode or null
12.       if (markedNode!=null) nodeSet = nodeSet U {markedNode}
13. }
14.
15. for (each markedNode_i, markedNode_j in nodeSet) {
16.       commonToken ← getCommonTokenIfAny(markedNode_i,markedNode_j)
17.
18.       if (isUnit(commonToken) OR isMeasure(commonToken)) {
19.                 createEdge(markedNode_i,markedNode_j)
20. }
21. }
22.
23. for (each markedNode_i ε G) {
24.       if (hasValue(markedNode_i) AND hasMeasure(markedNode_i)) VD ← VD U
    associate(markedNode_i,1)
25. }
26.
27. for (each edge E ε G) {
28.       VD ← VD U associate(E,2)
29. }
30.
31. VD ← VD U associateRemainder(G)
32. return VD
```



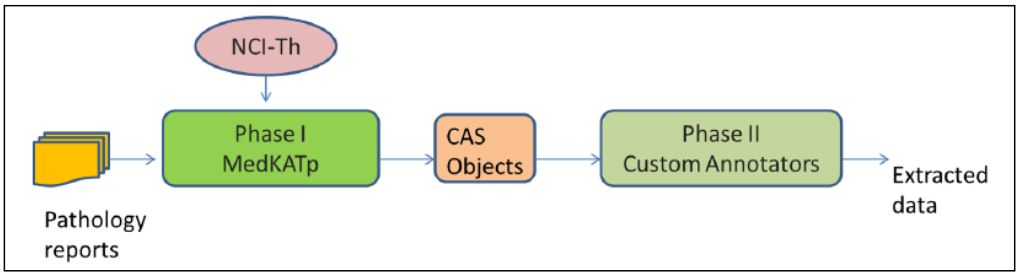


Figure 4. PEP information extraction pipeline.
PEP: Pathology Extraction Pipeline.

from the analysis of the parse tree of the parse of a sentence; this system is however targeted at extracting particular relationships (gene protein interactions), whereas Associate Value Dimension does similar analysis on the parse tree (relational representation) to associate numerical values with dimensions.

Decomposition (annotator). We developed code to decompose complex fields (such as Tumor, Node, Metastases (TNM) staging) into constituent values of interest. Considering a field such as TNM stage as an example, for a value such as T1aM0N0, we are interested in delineating values of the T, N, and M components, that is, 1a, 0, and 0, respectively, in this example. Such decomposition is not handled by MedKATp as is. The code in this annotator is based on regular expression–based pattern matching to decompose and identify the individual elements in such fields.

Concept detection (dictionary plug-in). We integrated the National Cancer Institute (NCI) Thesaurus,[15] an ontology of cancer terms, into the PEP. This integration involved translating the NCI Thesaurus into an equivalent representation using the MedKATp (i.e. UIMA) ontology format.[1] The NCI Thesaurus is a reference terminology and biomedical ontology. It covers vocabulary of over 250,000 terms for clinical care, translational and basic research, and public information and administrative activities. The NCI Thesaurus provides definitions, synonyms, and other information on nearly 10,000 cancers and related diseases, 8,000 single agents and combination therapies, and a wide range of other topics related to cancer and biomedical research.

The PEP architecture is illustrated in Figure 4. The results of the initial MedKATp pass—that is, sections and subsections identified, dimensions and numerical quantities, or concepts spotted, are stored as intermediate CAS objects. In the second phase, the information is read from the intermediate CAS objects, and further customized processing (as described above) is done to extract the fields of interest.

Abstraction layer

MedKATp and the UIMA framework are indeed a very useful starting point for our work, given the pathology information extraction built-in capabilities provided by the tool. However, being able to use UIMA to configure MedKATp appropriately for specific field extraction even in the pathology domain requires a developer level understanding of UIMA. A developer has to become familiar with the UIMA concept of analysis engines and learn how to create or configure such engines. To simplify this, we have developed an *abstraction layer* on top of MedKATp, where a user can provide a high-level specification of the extraction. The abstraction layer then automatically generates a

Table 3. Specifications.

Specification
Field: STAGING, Label: "Pathologic Staging"
Field: TNM, Pattern: "T[a-z0-4][ ]*N[x0-3][ ]*M[x0-1]"

TNM: tumor, node, metastases.

UIMA annotator from such a specification. For instance, the specification in the first row in Table 3 specifies that the content of a section with the label "Pathologic Staging" becomes a value for the field "STAGING." The specification in the second row states that any text conforming to the regular expression pattern provided in the specification becomes a value for the field "TNM."

At this point, the specifications are of two kinds: (1) specify a section by its label name and (2) provide a regular expression pattern for a field. Currently, the interpretation is (1) to consider the label content for a labeled section as the text up to the *next* label or end of report if no following labels and (2) to match the pattern regular expression over the *entire* report text. A usage evaluation within the data warehousing group at the UCI medical center found all users to find the above high-level configuration capability significantly easier and faster to use as opposed to configuring UIMA annotators. This evaluation was conducted with four users. Each user was provided with the following: (1) documentation on UIMA, specifically on configuring annotators, the type system, and building analysis engines and (2) documentation on using the abstraction layer. They were asked to configure the system for extracting 10 given fields (based on section labels or patterns) using (1) UIMA and (2) the abstraction layer. Each user was handed a survey questionnaire at the end with the following questions: (1) total time for configuration, (2) perceived difficulty (scale of 1–5), (3) preference for system (scale of 1–5), and (4) overall usability. We make the claim of preference for the abstraction layer based on this feedback. We see this as consistent with the observation by Nadkarni et al.[16] that NLP frameworks, such as UIMA, are still more apt for programmers and not really commoditized for use by a larger population.

Resilience

The extraction of almost all fields relies on certain consistencies in the text format, for instance, many fields appear to have fixed labels across all reports. This label is provided as configuration information. However, in our dataset, we have found some deviations from such common or "expected" labels (Table 4).

Table 4 illustrates such deviations, which are present in 10% of the reports. With such deviations, the extractor misses extracting the field, and thus, the overall recall for the field suffers. We have designed and implemented an approach based on machine learning for this problem.

The central idea is to *learn* a pattern for the data field instances and use that to identify label deviations and then extract the missed data instances. The primary steps (Figure 5) are as follows:

1. For a given field, we first extract and collect the instances of the field that we can, across the entire set of reports. Basically, this is a first pass of the extractor where extraction is done with the system as is, and for each field, the extracted instances are collected as list.
2. We then use the extracted instances in the first pass above as *labeled examples* to *learn extraction pattern(s)* for that field. Essentially, we induce regular expressions that best

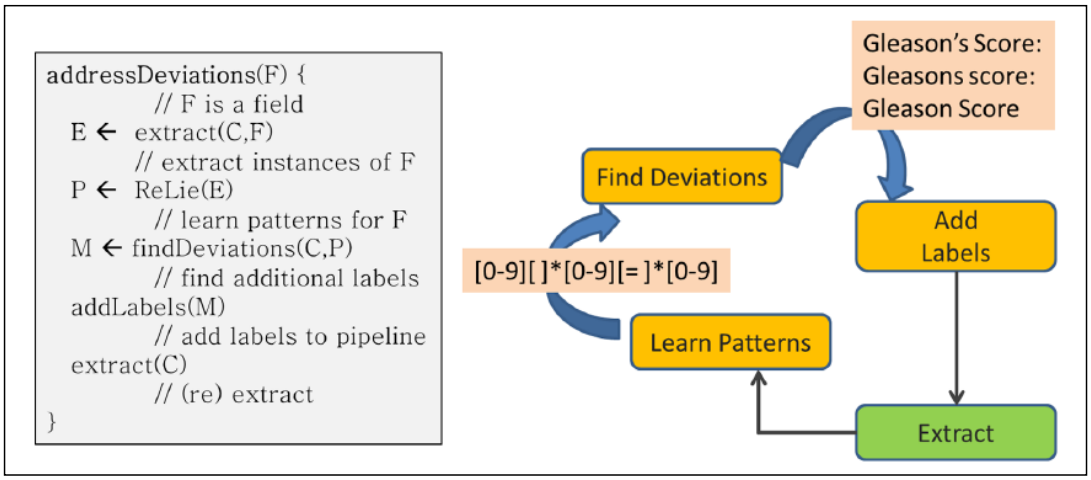


Figure 5. Addressing format deviations using learning.

Table 4. Format deviations.

Common (expected) label	Deviations
Gleason score	Gleason's score Gleasons score Gleason score
TNM staging	TNM Stage TNM STAGING

TNM: tumor, node, metastases.

describe and cover the set of instances extracted. Specifically, we have implemented and used the ReLie algorithm[17] to induce regular expressions from a set of field instances. The ReLie algorithm formulates the regular expression learning problem as search optimization problem over a space of possible regular expressions.

3. We then use the learnt regular expressions to identify missed instances and the labels for the missed instances. Such labels correspond to the deviations from the expected format.
4. Finally, the deviant labels are added to the annotators in the PEP, and a *second* extraction pass is done to extract the previously missed instances.

Results

We conducted the following experimental evaluations with PEP:

1. Evaluating the accuracy of automatic information extraction, in the context of prostate cancer pathology reports.
2. Evaluating the accuracy (recall) improvement achieved when incorporating our approach to addressing format deviations.
3. Evaluating the scalability of PEP, in moving across different types of cancer pathology reports.
4. Processing throughput rate.

For the first evaluation, we compared a structured database of protocol fields created from unstructured pathology reports using our automated system, with data for the same fields for the same patients from the *manually curated* Strategic Partnering to Evaluate Cancer Signatures (SPECS) clinical database. The UCI SPECS consortium (Strategic Partners for the Evaluation of Predictive Signatures of Prostate Cancer)[17] is an observational clinical trial in progress at UCI. The SPECS clinical database[17] contains extensive clinical annotation for up to 19 years for over 1500 cases. It is a structured database that contains all the variables of case report forms, results of the pathology reports for examination of the prostatectomy tissue, subsequent treatment histories, as well as post-surgery prostate-specific antigen (PSA) values and other variables. The data dictionary for the database contains over 580 items covering annotation for 5468 patients. The SPECS database was curated by three annotators with an inter-annotator-agreement (IAA) of 0.96 and forms the "truth database" for our automated extraction experimental evaluations.

Extraction accuracy for fields

Table 5 provides the evaluation of the accuracy of PEP for automated extraction over pathology reports. The key details regarding the experiment are as follows:

- The evaluation dataset is a corpus of 400 pathology reports for prostate and breast cancer from the UCI Medical Center.
- We have measured the accuracy of extraction, *per field*, in terms of strict precision and recall.
- For any field f
  - $Precision\ Pf = \dfrac{(\#\ of\ correctly\ extracted\ instances)}{(\#\ of\ instances\ extracted)}$
  - $Recall\ Rf = \dfrac{(\#\ of\ correctly\ extracted\ instances)}{(Total\ \#\ of\ instances)}$
  - $(Traditional)\ F-measure(F1) = \dfrac{2.Pf.Rf}{Pf+Rf}$
- The total number of instances = 400
- An instance is said to be correctly extracted if the extractor correctly extracts the field *as well as* the values within the field.
- For instance, for a TNM score, we also want the extractor to correctly identify the individual T, N, and M values.
- The above measures were computed using some custom code for this evaluation that compared the extractor output for the 400 reports with the corresponding fields from a (comma-separated value (CSV)) dump of records of the corresponding reports from the SPECS database. Some minor normalization had to be done to account for format differences such as a field value of "PRESENT/ABSENT" in the pathology reports being reported as "T/F" or "1/0" in the SPECS database.

Table 5 displays the extraction accuracy measures for 22 fields of interest conducted over 400 pathology reports. As shown, the precision and recall are over 85% for most fields. With such

Table 5. Extraction evaluation results.

Field	Gleason score	TNM stage	Capsule invasion	Lymph invasion	Chronic inflammation	Vascular invasion	Perineural invasion	Surgical margin	Seminal vesicle
Precision	0.87	0.95	1.0	0.93	1.0	0.93	0.99	0.88	0.81
Recall	0.77	0.85	1.0	0.88	0.99	0.87	0.92	0.94	0.89
F-score	0.81	0.9	1.0	0.91	0.99	0.90	0.95	0.91	0.85
Field	Extraprostatic extension	PIN	Lymph nodes sampled	Lymph nodes detected	Specimen length	Specimen width	Specimen height	Specimen weight	
Precision	1.0	1.0	0.91	0.89	0.9	0.89	0.93	0.91	
Recall	1.0	0.86	0.93	0.89	0.68	0.72	0.82	0.86	
F-score	1.0	0.93	0.92	0.89	0.78	0.8	0.87	0.89	
Field	Laterality	Necrosis	Category	ER	PR				
Precision	0.88	0.93	0.87	0.91	0.9				
Recall	0.81	0.88	0.79	0.86	0.87				
F-score	0.84	0.9	0.83	0.89	0.89				

TNM: tumor, node, metastases; PIN: prostatic intraepithelial neoplasia.

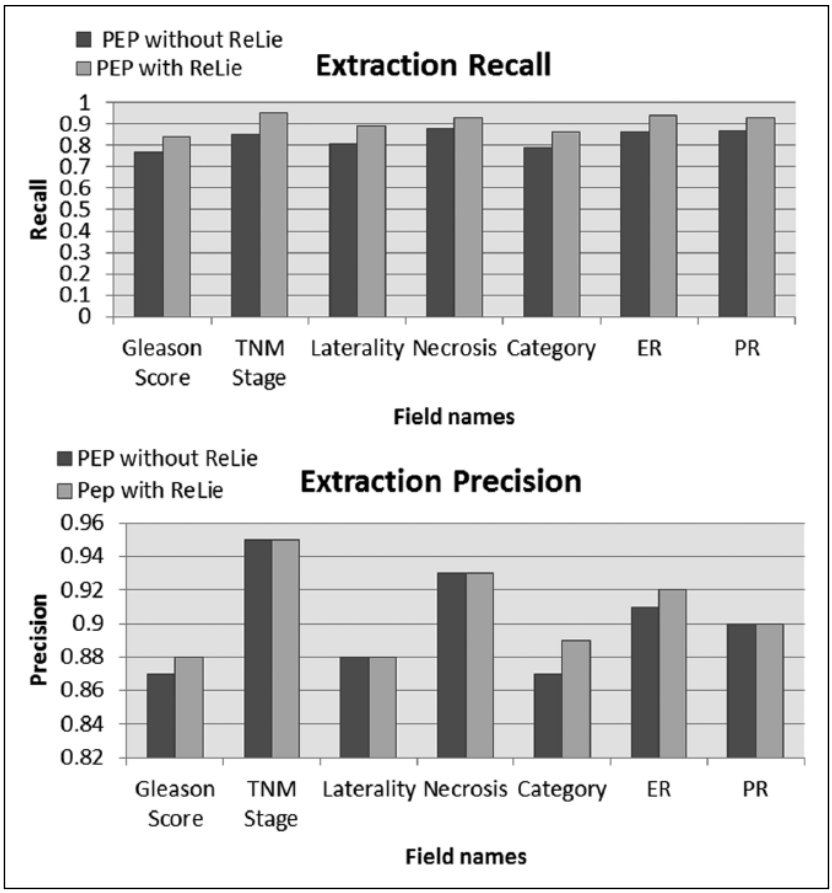


Figure 6. (a) Recall improvement with ReLie integration and (b) precision impact with ReLie integration.

accuracy, PEP can provide a curation pipeline for data warehouse population, where extracted results can be populated to the warehouse database after manual review.

Recall improvement

The next evaluation was to assess any improvement achieved in recall by incorporating our algorithm for addressing format deviations. Recall improvement was achieved in 7 out of the 22 fields. The improvement is achieved in only a subset of the 22 fields as only 7 of the fields *had* some format deviation across the dataset we used.

The results are shown in Figure 6(a) for certain fields from the prostate cancer and breast cancer domains. The results illustrate that our approach indeed increases the recall for each field over the recall that can be achieved by just using MedKATp. Figure 6(b) shows the accompanying impact on precision which remained the same, and in some cases, even slightly increased, with the integration of ReLie.

McNemar's Test

Table 6 shows the data we collected to assess the impact of our algorithm for addressing format deviations. The data are collected over multiple instances of multiple fields, where the terms

Table 6. McNemar's Test.

	After: incorrect (Ai)	After: correct (Ac)	Row total
Before: incorrect (Bi)	205	73	278
Before: correct (Bc)	22	801	823
Column total	227	874	1101

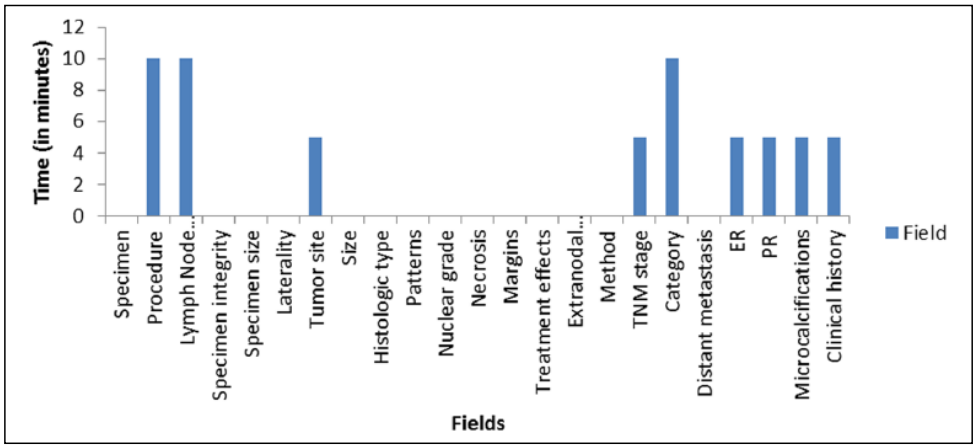


Figure 7. Time for extractor configuration for new cancer types.

"correct" and "incorrect" refer to correctly and incorrectly extracted field values, respectively. As the McNemar Test statistic (with Yates' continuity correction) is

$$\chi^2 = \frac{\left[\left(AcBi - AiBc\right) - 0.5\right]^2}{(AcBi + AiBc)}$$

Applying the data in Table 6, we have

$$\chi^2 = \frac{\left[\left(73 - 22\right) - 0.5\right]^2}{(73 + 22)} = 26.84$$

The value of 26.84 is very significant, that is, extremely unlikely from the distribution implied by the null hypothesis, and thus validates the effectiveness of integrating a ReLie-based resilience approach.

Scalability

Finally, we evaluated the scalability of our system across various types of pathology reports, specifically reports on different types of cancer. The first version of the PEP system was built taking prostate cancer reports as the data and the extraction of prostate cancer–related fields as the extraction task. Figure 7 illustrates the effort (in minutes for each field) that it took a developer to expand the PEP system for prostate cancer to reports (and fields) from *breast* cancer. As Figure 7 illustrates, a majority of the fields required no additional effort, and the effort for any of the other

(breast cancer–specific) fields was quite minimal. The entire exercise was completed in less than an hour. We achieved similar results, that is, less than an hour customization time, for a third dataset from *lung* cancer. We must mention that the extraction accuracies for fields in the new datasets (breast and lung cancer) were very similar to what we achieved for prostate cancer and were within 5% of the extraction accuracy achieved for prostate cancer for each field.

Throughput

PEP is part of an operational pipeline for the UCI medical center clinical data warehouse. To keep the data up to date, new pathology reports will be fed nightly to our system for extraction and addition to the data warehouse. To assess the scale at which we can process such reports, we evaluated the throughput rate per report. The throughput rate was 5.1 s per document on a CentOS VM server with 8GB RAM and 500 GB storage, using a single-thread configuration. The throughput rate for just MedKATp is 4.8 s per document.

Discussion

Real-world use of PEP

The PEP has been committed to the OHNLP effort and is now available as open-source software from the OHNLP community. This has been done with the intention of providing an extraction pipeline with additional capabilities to the OHNLP community for use as is, or for enhancement as appropriate. In addition, the PEP has been deployed in production use in the extract, transform, and load (ETL) pipeline for the UCI Medical Center clinical data warehouse, which is a multidimensional data warehouse of clinical data developed and maintained for UCI clinicians, administrators, and researchers.

Related work

There are many systems and research efforts that relate, at different levels, to what we have developed in PEP. The area of automated information extraction from clinical and biomedical text has seen many efforts and systems[1–4,6–8] in the past decade. From a higher level perspective, most approaches have used a combination of techniques from pattern matching, lexical analysis, NLP, and machine learning. Systems have also been developed for very specific automated information extraction tasks such as identifying medications or diagnosis mentions in the text. With respect to the issue of development frameworks to build (an extraction system) upon, clinical Text Analysis and Knowledge Extraction System (cTAKES[2]) is a system built upon UIMA as well. However, the functionality offered by cTAKES is at a more generic processing level, namely, it provides annotators for things such as sentence boundary detection, tokenization, and part-of-speech tagging. Buckley et al.'s[8] study is an exploratory study to ascertain the feasibility of using NLP to extract clinical information from breast pathology reports. The study was conducted using NLP software from Clearforest—a text analytics company. The scope of the automated extraction here is limited to identifying the mention of diagnoses in the reports. Our goal, as demonstrated above, is to extract a much wider and more comprehensive set of fields from the pathology reports, namely, all fields of interest as identified by the CAP protocols. The system that is most related to our work is MedKATp, so much so that it forms the baseline framework for this work. We have applied it by configuring it as well as further customized and added our own annotators for extraction. Our specific contributions in PEP can be summarized as follows:

- An application of MedKATp for protocol fields extraction: We have developed a complete MedKATp application for the extraction of detailed CAP protocol fields for various types of cancer pathology reports and have also evaluated the application for a real-world task.
- Customization: We have further developed custom annotators that perform deeper processing to extract protocol fields that we could not get with MedKATp application as is.
- Abstraction layer: As a software engineering contribution, we have provided an abstraction layer where non UIMA developers can easily configure the system to extract new pathology-related fields of interest.
- Resilience: We incorporated an approach to make the extractor more resilient to occasional label format deviations—thus increasing the accuracy (recall) that is achieved.
- Operationalization: PEP is in operational use at the UCI medical center.
- OHNLP contribution: We have contributed PEP as a resource for community use to the OHNLP consortium.

In conclusion, we report on an enhanced information extraction system that can handle extraction of multiple fields with high accuracy. The resulting system is being deployed in a real clinical informatics application and also provides a foundation for further research in improving such systems.

Funding

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

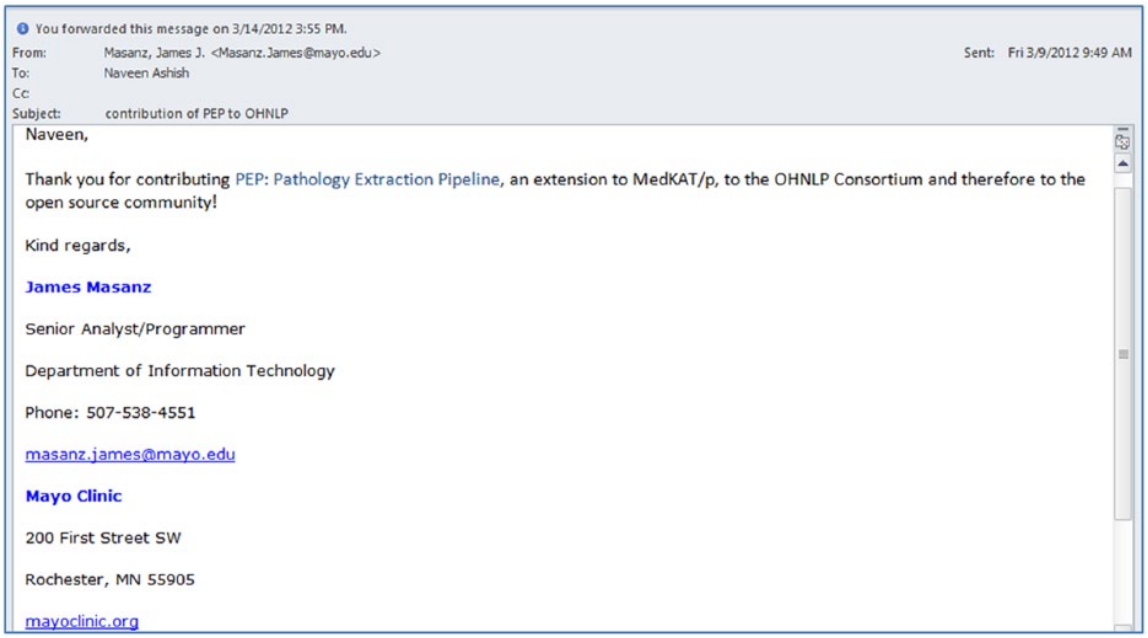
References

1. Coden A, Savova G, Sominsky I, et al. Automatically extracting cancer disease characteristics from pathology reports into a Disease Knowledge Representation Model. *J Biomed Inform* 2009; 42: 937–949.
2. Savova GK, Masanz JJ, Ogren PV, et al. Mayo clinical Text Analysis and Knowledge Extraction System (cTAKES): architecture, component evaluation and applications. *JAMIA* 2010; 17: 507–513.
3. Crowley RS, Castine M, Mitchell K, et al. caTIES: a grid based system for coding and retrieval of surgical pathology reports and tissue specimens in support of translational research. *JAMIA* 2010; 17: 253–264.
4. Rosenbloom ST, Denny JC and Xu H. Data from clinical notes: a perspective on the tension between structure and flexible documentation. *JAMIA* 2011; 18: 181–186.
5. Meystre SM, Savova GK, Kipper-Schuler KC, et al. Extracting information from textual documents in the electronic health record: a review of recent research. *IMIA Yearb Med Inform* 2008; 47: 128–144.
6. D'Avolio LW, Nguyen TM, Goryachev S, et al. Automated concept-level information extraction to reduce the need for custom software and rules development. *JAMIA*. Epub ahead of print 22 June 2011. DOI: 10.1136/amiajnl-2011-000183.
7. Patrick JD, Nguyen DH, Wang Y, et al. A knowledge discovery and reuse pipeline for information extraction in clinical notes. *JAMIA*. Epub ahead of print 7 July 2011. DOI: 10.1136/amiajnl-2011-000302.
8. Buckley JM, Coopey SB, Sharko J, et al. The feasibility of using natural language processing to extract clinical information from breast pathology reports. *J Pathol Inform* 2012; 3: 23.
9. Gotz T and Suhre O. Design and implementation of the UIMA Common Analysis System. *IBM Syst J* 2004; 43(3): 476.
10. College of American Pathologists (CAP). College of American Pathologists, http://www.cap.org/ (2011).

11. Klein D and Manning CD.Accurate unlexicalized parsing. In: *Proceedings of the 41st meeting of the Association for Computational Linguistics*, Stroudsburg, PA, 2003, pp. 423–430.
12. Klein D and Manning CD. Fast exact inference with a factored model for natural language parsing. In: *Advances in neural information processing systems 15 (NIPS 2002)*. Cambridge, MA: MIT Press, 2003, pp. 3–10.
13. De Marneffe M-C, MacCartney B and Manning CD. *Generating typed dependency parses from phrase structure parses. In: LREC* 2006,Genoa, Italy.
14. Fundel K, Küffner R and Zimmer R. RelEx—relation extraction using dependency parse trees. *Bioinformatics* 2007; 23(3): 365–371.
15. Sioutos N, de Coronado S, Haber MW, et al. NCI Thesaurus: a semantic model integrating cancer-related clinical and molecular information. *J Biomed Inform* 2007; 40(1): 30–43.
16. Nadkarni PM, Ohno-Machado L and Chapman WW. Natural language processing: an introduction. *JAMIA* 2011; 18(5): 544–551.
17. Li Y, Krishnamurthy R, Raghavan S, et al. Regular expression learning for information extraction. In: *Proceedings of the conference on empirical methods in natural language processing (EMNLP '08)*, October 2008, Waikik,i Hawaii, 2008, pp. 21–30. Stroudsburg, PA: Association for Computational Linguistics.

Appendix 1

Email confirmation from OHNLP


You forwarded this message on 3/14/2012 3:55 PM.

From: Masanz, James J. <Masanz.James@mayo.edu> Sent: Fri 3/9/2012 9:49 AM
To: Naveen Ashish
Cc:
Subject: contribution of PEP to OHNLP

Naveen,

Thank you for contributing PEP: Pathology Extraction Pipeline, an extension to MedKAT/p, to the OHNLP Consortium and therefore to the open source community!

Kind regards,

James Masanz

Senior Analyst/Programmer

Department of Information Technology

Phone: 507-538-4551

masanz.james@mayo.edu

Mayo Clinic

200 First Street SW

Rochester, MN 55905

mayoclinic.org