


Unlocking Social Media and User Generated Content as a Data Source for Knowledge Management

James Meneghello, Optika Solutions, West Perth, Australia

Nik Thompson, Curtin University, Perth, Australia

 <https://orcid.org/0000-0002-0783-1371>

Kevin Lee, Deakin University, Victoria, Australia

Kok Wai Wong, Murdoch University, Murdoch, Australia

Bilal Abu-Salih, The University of Jordan, Amman, Jordan

ABSTRACT

The pervasiveness of social media and user-generated content has triggered an exponential increase in global data. However, due to collection and extraction challenges, data in embedded comments, reviews and testimonials are largely inaccessible to a knowledge management system. This article describes a KM framework for the end-to-end knowledge management and value extraction from such content. This framework embodies solutions to unlock the potential of UGC as a rich, real-time data source. Three contributions are described in this article. First, a method for automatically navigating webpages to expose UGC for collection is presented. This is evaluated using browser emulation integrated with automated collection. Second, a method for collecting data without any a priori knowledge of the sites is introduced. Finally, a new testbed is developed to reflect the current state of internet sites and shared publicly to encourage future research. The discussion benchmarks the new algorithm alongside existing techniques, providing evidence of the increased amount of UGC data extracted.

KEYWORDS

Content Discovery, Data Acquisition, Data Manipulation, Knowledge Management, Social Mining, User-Generated Content, Web Data Extraction

INTRODUCTION

User-generated content (UGC) is a promising data source for analytical applications, providing up-to-date information about a wide range of topics. There is an increasing focus on the use of UGC on Social Networking Sites (SNS) in areas including emergency management (Robinson, Power, & Cameron, 2013) and opinion mining (Maynard, Bontcheva, & Rout, 2012). While previous research has analyzed data from micro-blogging SNS such as Twitter, few studies have used traditional SNS (Meneghello, Thompson, & Lee, 2014; Wandhöfer et al., 2012) or embedded UGC such as user comments and reviews (Cao, Liao, Xu, & Bai, 2008). Since the shift to Web 2.0, static content on websites has increasingly been replaced by UGC that is not easily retrieved, resulting in a large amount of UGC being unavailable for knowledge discovery and mining.

DOI: 10.4018/IJKM.2020010105

Copyright © 2020, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

Web Data Extraction (WDE) algorithms have had success in extracting template-based data from web pages (e.g. Arasu & Garcia-Molina, 2003; Ferrara, De Meo, Fiumara, & Baumgartner, 2014). However, page design and rendering on the internet has changed significantly since many WDE algorithms were developed, requiring newer structure detection techniques (Blanvillain, Kasioumis, & Banos, 2014). Modern pages are often dynamically rendered using Asynchronous JavaScript and XML (AJAX), which many WDE algorithms do not support. Furthermore, UGC contains embedded user comments or replies, but this is unpredictable and poses contextual challenges for extraction.

UGC presents an excellent opportunity for providing data about a wide range of topics, both historical and real-time. However, although UGC is timely and relevant, its semi-structured or unstructured nature poses a challenge for knowledge extraction. There are no standard methods of performing data collection of UGC embedded in web-pages. This contrasts to UGC on SNS, which is usually accessible via Application Programming Interfaces (APIs) - specifically made for applications to query stored data. The work described in this paper aims to address the need for a method of extracting UGC in a standardized and reliable way for use in analytical applications and Knowledge Management systems.

This paper presents a complete data extraction framework able to automatically navigate pagination structures, expansion links and render dynamic DOM trees. This framework aims to unlock the value of UGC stored within web pages and to address the shortfalls of existing WDE algorithms. This framework is established to maximize the amount of useful data that may be extracted. UGC structures can then be isolated and filtered to provide a standardized interface for reliably retrieving UGC. This work is placed into context by incorporating a Knowledge Management (KM) framework as a data analytics paradigm (Jennex & Bartczak, 2013). The framework facilitates the implementation of filters, processes, and technologies to satisfy the aim of data analytics and provide actionable intelligence.

THEORETICAL BACKGROUND

Proliferation of UGC

Since the emergence of Web 2.0, the role of web browsers has evolved from enabling users to send content through basic e-mail or chat applications, to now supporting advanced electronic platforms such as Online Social Networks (OSNs). OSNs enable users to share videos, photos, and instant conversations. These platforms provide an important means by which communities can grow and consolidate, allowing individuals or groups to share interact with others (Abu-Salih, Wongthongtham, Chan, & Zhu, 2019). Several modern computing applications, for instance; online education, health, music and entertainment rely on the content generated by OSNs (Althoff, Jindal, & Leskovec, 2017). This is evidenced by the dramatic usage increase of these platforms for networking and communication (Shouhong & Hai, 2018). Pew Research Center reported that 74% of American adults in Jan 2018 used OSNs for social interactions compared to 5% in 2005 (Gramlich, 2019). In Australia, the usage statistics of OSNs in Jan 2017 indicated around 2.8 million Twitter active users, 14.8 million visits to YouTube and 4.0 million Snapchat active users (Social Media News, 2019). Such a dramatic uptake of the online social platforms has brought people together with shared interests, thoughts, and aims.

Web Data Extraction

WDE algorithms are designed to extract semi-structured data from web pages have been in development for the last 15 years (Arasu & Garcia-Molina, 2003; Atzeni & Mecca, 1997; Cao et al., 2008; Crescenzi, Mecca, & Merialdo, 2001; B. Liu, Grossman, & Zhai, 2003; Miao, Tatemura, Hsiung, Sawires, & Moser, 2009; Zhai & Liu, 2005). WDE algorithms have had great success in scraping search result pages in order to facilitate web crawling (Fang, Xie, Zhang, Cheng, & Zhang, 2018; B. Liu et al., 2003), as well as collecting e-commerce data (such as competitor prices) (Zhai & Liu, 2005) database

detail pages (Thamviset & Wongthanavas, 2014a), automatic data extraction from template pages (Yuliana & Chang, 2018), and news articles (Gupta, Mittal, Bishnoi, Maheshwari, & Patel, 2016).

WDE algorithms generally use programs known as wrappers to extract data from a web page into a relational form. Initially, wrapper development was manual and required a developer to write code to extract desired fields from a single page (Atzeni & Mecca, 1997; Crescenzi & Mecca, 1998; Kejriwal, 2019). Most wrappers are not generic, and each new data source required the development of a new wrapper. As this represented a significant development and maintenance cost, automated techniques were highly sought after. However, web information extraction is not a trivial task and rapid evolution of page styles and poor quality of web pages impedes the effort to extract relevant content (Sahoo & Parthasarthy, 2018). This requires advanced and sophisticated approaches to effectively tackle this issue by providing an automated way to crawl and analyze such heterogeneous data lakes using applicable frameworks to manage them.

Automated WDE algorithms can be classified based on their method of function and the manner of data input. Functionally, these algorithms work either by comparing the page DOM trees (a tree representation of a web-page content and structure), (e.g. Ferrara & Baumgartner, 2011; Jindal & Liu, 2010; Liu et al., 2003) by examining and mining the text directly (Thamviset & Wongthanavas, 2014a, 2014b), or by recognizing visually-similar page regions (Liu, Meng, & Meng, 2005; Liu, Meng, & Meng, 2010). Some algorithms also combine several of these techniques to improve accuracy, such as using text mining to locate data and tree comparisons to identify the enclosing environment (Thamviset & Wongthanavas, 2014a). Similar techniques can be used to collect UGC, thus UGC extraction can, therefore be considered a subfield of Web Data Extraction - henceforth, User Generated Content Extraction (UGCE).

Incorporation of Knowledge Management

Knowledge is information in action which is contextual, relevant, and useful (Davenport & Prusak, 1998). It is also called intellectual capital or intellectual assets (Stewart & Kirsch, 1991). Knowledge can be classified into two forms, namely tacit and explicit (Nonaka & Takeuchi, 1995; Polanyi, 2009). Explicit knowledge represents documented knowledge which is objective, rational and technical, and can be distributed or transformed into a process or a strategy. Tacit knowledge is a store of subjective or observed learning experiences. It comprises insights, expertise, understanding, skill sets, and organizational culture (Rainer, Cegielski, Spletstoeser-Hogeterp, & Sanchez-Rodriguez, 2013). Knowledge Management (KM) is a process that helps organizations apply important knowledge that is part of the organization's memory, thereby improving decision-making and organizational effectiveness accordingly (Jennex, 2007). A KM System, a cardinal enabler of KM (Dorasamy, Raman, & Kaliannan, 2013), enhances and expedites KM, utilizing best practices, to capture, refine, store, manage and disseminate knowledge (Jennex, 2007).

The wealth of UGC presents a unique opportunity for organizations to reveal new knowledge and insights in this data abundance and ultimately to increase their revenues. Hence, there is an imperative need to capture, load, store, process, analyze, transform, interpret, and visualize such manifold social datasets to develop meaningful insights. This can be achieved by incorporating a KM framework leveraging the proliferation of UGC to help companies to integrate their internal processes, the UGC's data lake and KM system to provide actionable intelligence. Determining a consolidated KM strategy enables identifying the crucial components necessary to achieve this aim (Jennex, 2017; Jennex & Bartczak, 2013).

In this context, a thread of efforts has steered towards extracting knowledge from UGC to inform actionable intelligence (Abu-Salih, Wongthongtham, & Chan, 2018; Hultgren, Jennex, Persano, & Ornatowski, 2016). As noted, OSNs have already been extensively used as a powerful tool to promote knowledge extraction and management in several domains (Arularasan, Suresh, & Seerangan, 2018; Kasemsap, 2019; Nishikant, Prabin Kumar, & Shashi Kant, 2018). Given such an impact, understanding ways to broaden this scope and extra data from new sources such as UGC is of

interest to many practitioners and researchers alike. In fact, even though identifying, reviewing, and interpreting social content consumes substantial time and effort (Chang, Diaz, & Hung, 2015), it still attracts wide interest due to the potential to apply KM to obtain high quality content, and actionable intelligence in many disciplines including politics (Cruz, 2019), e-commerce (Schaupp & Bélanger, 2019), e-learning (Hosseingholizadeh, Sharif, & Kouhsari, 2018), and healthcare (Surendro, Satya, & Yodihartomo, 2018). Other efforts have provided unconventional and advanced perceptions to frame this constant growth of UGC, alongside other Big Data islands. For example, Jennex (2017) presented a revised version of the traditional KM pyramid that incorporates Big Data, Internet of Things (IoT) and Business Intelligence (BI) to provide an overarching paradigm toward better decision support strategies.

This research incorporates a KM framework (Jennex, 2017; Jennex & Bartczak, 2013) to provide a systematic basis for UGC extraction and analysis incorporating KM approaches. The work presented in this paper describes an automated method to extract UGC in a standardized format so that web pages such as news articles, blogs and other media can be unlocked and made accessible as viable data sources. This will greatly broaden the scope of data available for use in analytical applications and support platforms designed to use this knowledge for better decision making.

PROPOSED UGC EXTRACTION ARCHITECTURE

UGC can represent current events, peoples' views and thoughts on issues or their current environment. These comments, statuses, messages, and tweets often possess a common set of attributes. While different sources may provide differing metadata for a social event, the minimal set usually includes an origin, a timestamp, and some message content. An origin can be a person, an account-name or commonly a pseudonym, while a timestamp can be a date, a time, a relative time or any combination. Content usually contains the detail about the event occurring - a message between friends, a comment on a topic under discussion or an update on the status of a user. Figure 1 shows an example of UGC, with each user comment containing each of these attributes - name, date, and content.

The UGC present on such pages is similar to that on SNS, except it tends to be anchored around a particular subject or content. This makes page-based UGC extremely useful for user-based analytics such as intent or sentiment analysis. UGC is already a major consideration in e-Commerce research (Cheong & Morrison, 2008) often having significant effects during product selection. While data from SNS is frequently used, inaccessibility of page-based UGC is an obstacle to providing more effective social analytics.

Figure 1. Embedded UGC from a popular news website, ABC Australia

Vincent: 29 May 2015 7:48:30am
Witnessing the celebration of opportunistic imbecility paraded as policy initiative by these two leads one irrevocably to the decision that democracy in this country is broken.
Reply Alert moderator

barsnax: 29 May 2015 8:14:07am
Question time has been broken for years. It's a great example of what is wrong with parliament and an embarrassment to watch.
We need an independent speaker in the house.
Reply Alert moderator

The wealth of UGC presents a unique opportunity for organizations to reveal new knowledge and insights and ultimately to increase their revenues. Knowledge management frameworks (Jennex, 2017; Jennex & Bartczak, 2013) provide a systematic approach to UGC extraction and analysis and place this work in context. Three primary steps have been identified that must be considered when developing an automated UGC extraction (UGCE) process:

- **Data Collection and Acquisition:** a method of automatically interacting with and collecting data from dynamic sources without any *a priori* knowledge of the source structure;
- **Information Discovery and Extraction:** an algorithm to detect commonly-occurring data structures and extract location and data types of fields contained within, while avoiding nested data complications, and a rule-based filter that ensures only UGC is extracted from the page, which enhances precision.
- **Actionable Intelligence:** the steps as mentioned earlier establish the foundation for subsequent data analytics leveraging the UGC extraction approach.

Figure 2 below illustrates the system architecture that includes the main phases followed in this research incorporating the KM framework. This paper tackles the first two phases (Data Collection and Acquisition, and Information Discovery and Extraction). Actionable intelligence phase will be addressed in our future research. The following sections detail the developed UGC extraction approach.

DATA COLLECTION AND ACQUISITION APPROACH

Before running any data extraction algorithms on a data source, significant preparation is required. Because AJAX and dynamic DOM manipulation are in heavy usage on modern sites, it is no longer possible to simply issue an HTTP request and save any raw page data returned. The advent of the “single page website” requires that any method of extracting UGC needs to fully emulate a user by emulating the browser rendering, JavaScript engine and user interaction in order to expose linked UGC hidden behind pagination fields.

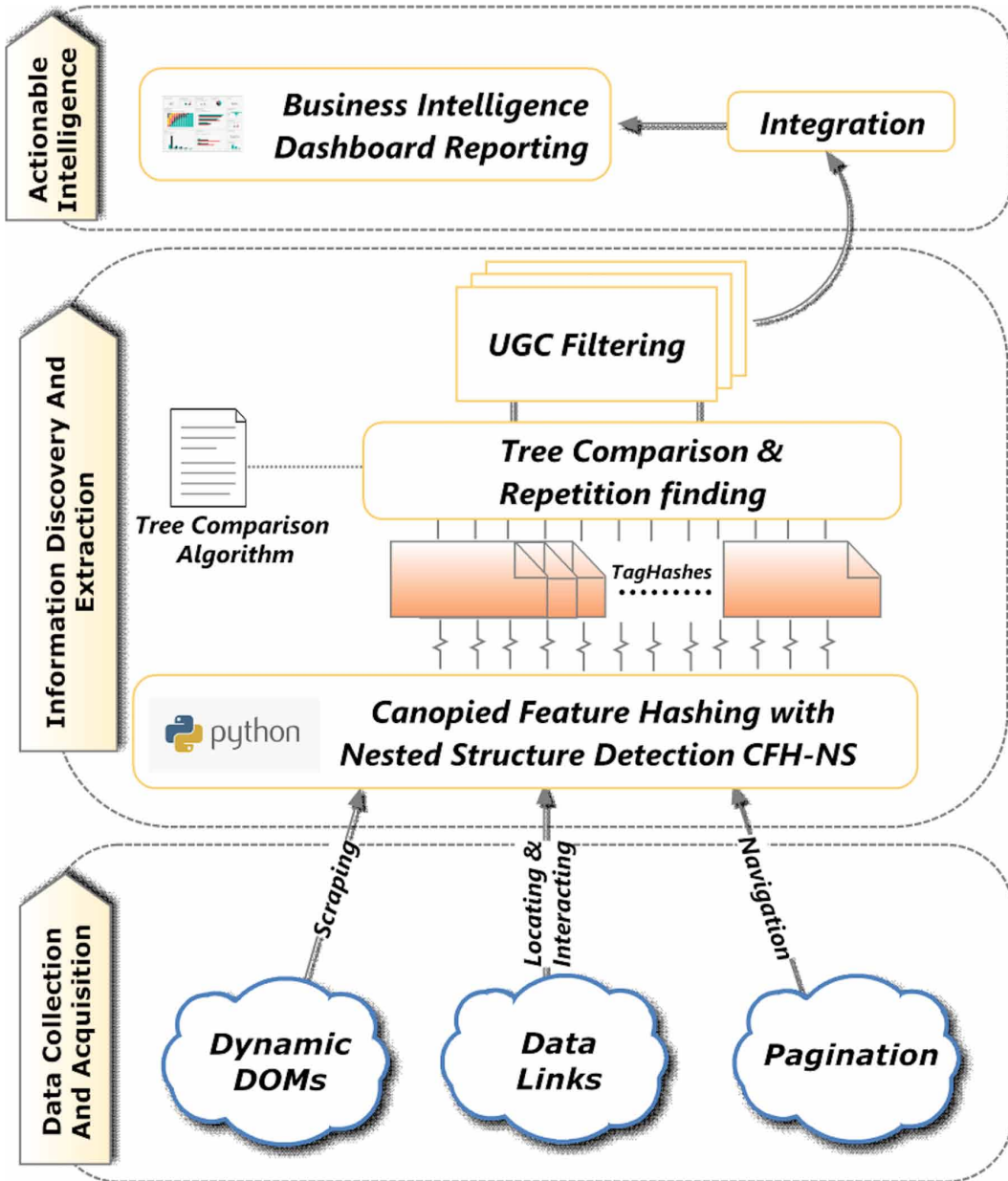
Pagination serves to limit the size of requests required to dynamically render pages, but also to restrict the amount of content available to the user. While a user may not want to view 600 comments at once, applications analyzing social comments desire every available piece of data and therefore require a method of identifying and triggering pagination and expansion elements (emulating a real user clicking through to open all comments or pages) to collect comprehensive data. Figure 3 depicts this process.

Fortunately, the frequent use of Cascading Stylesheet (CSS) and JavaScript frameworks has led to similarities between pagination structures and expansion links. By designing broad rules, most of these structures can be identified regardless of page design. Three primary structures have been identified and are handled by our algorithm:

- **Pagination:** Pagination elements generally consist of a numeric sequence containing links to data pages, sometimes within a select box, as seen in Figure 4.
- **Expansion Link:** An expansion link is often interleaved throughout nested comment trees, and can be clicked by a user to expand the replies within that thread.
- **Redirection Link:** A redirect link is used on some pages when comments are completely isolated from the page, and provides a link to another page consisting primarily of social content.

To effectively collect the raw data required to extract UGC, the collection process must be robust and can handle the above structures properly. The data collection described above is also compatible with legacy data extraction algorithms not designed to operate on dynamic DOM trees.

Figure 2. System architecture incorporating KM framework



Each part of this process is described in the following sections.

Scraping Dynamic Pages

A method of emulating user interaction and standard browser processes (such as JavaScript and page rendering) is required. Automated approaches do not need to visually present the page, so a screenless solution is preferable. To satisfy these requirements, PhantomJS a browser emulator capable of dynamically rendering pages, including handling any JavaScript requests was employed (Hidayat, 2013).

Figure 3. The proposed architecture for pagination handling

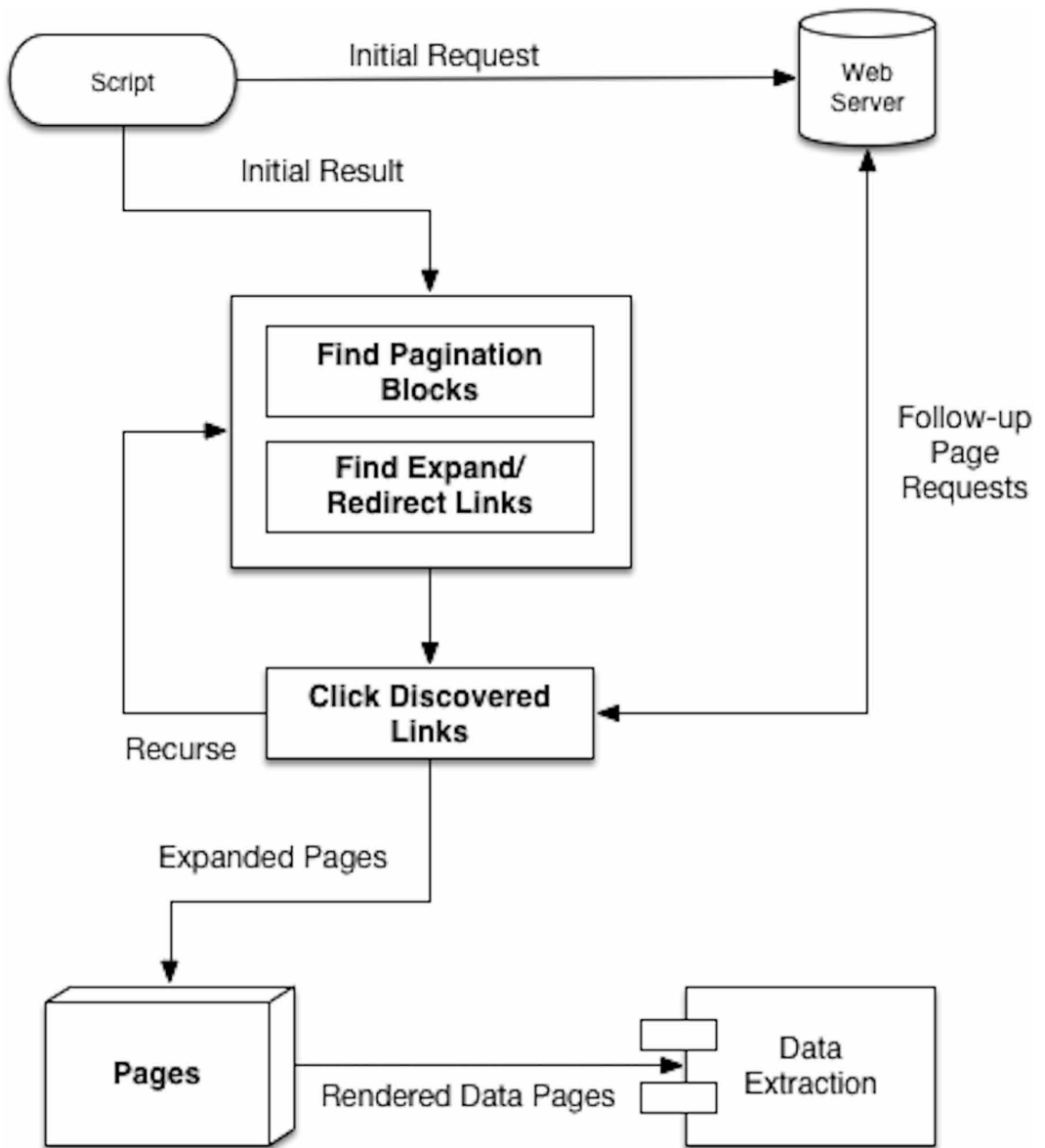


Figure 4. Examples of pagination elements taken from popular news websites and online forums



Locating and Interacting with Data Links

In order to extract all possible data from a page, user interaction with pagination and expansion elements needs to be simulated. The interaction should cause the browser emulator to dynamically modify the DOM tree, which can later be parsed by extraction algorithms.

To facilitate this interaction emulation process, Selenium, a real-time interactive web framework (Holmes & Kellogg, 2006), was used. Selenium can emulate user interaction while rendering pages. Combined, PhantomJS and Selenium provide a virtual browser and testing agent that can load and render content and interact with it realistically.

Discovering Data Links

Content expansion elements can be identified by searching for elements containing certain text patterns. Links that lead to additional data are often identified by a qualifier followed by a certain noun, such as “more comments”. While there are several synonyms to “more” and “comments”, a regular expression can match combinations of these words in order. Similar principles can be applied to languages other than English.

The example Python code presented in Figure 5 will generate a regular expression that can be applied to a page to locate link candidates. The candidates can then be filtered to exclude any words that can lead us to unwanted content. Of the discovered elements, those that can respond to user interaction are stored.

Once appropriate link elements have been discovered, Selenium is used to simulate a user “click” on these elements. The resulting data from this interaction, either a new page or dynamically-loaded content, is cached in the set of page content for later use.

Navigating through Pagination

Identifying and interacting with pagination elements on a page is more difficult than handling the data links described above. While some pagination blocks contain names that can be matched, many structures do not contain relevant identifiers. Thus, to detect these blocks, the algorithm instead searches for features that may suggest the presence of a pagination structure.

Figure 6 presents a basic algorithm designed to locate these structures. Pagination blocks typically consist of a list of elements containing consecutive ordered integers representing data pages. These numbers can either be the values of individual clickable elements or option fields within a select box, which are both common designs for pagination blocks. Structures that contain value lists following this structure are located, as these identify pagination blocks. These blocks may also contain “previous”, “next” or “all” buttons, which can be “clicked” by the algorithm to emulate a real user. For instance, the “next” button cycles through every available page, while the “all” button redirects to a view showing all available data.

The pagination block can be described using a syntax known as an identifying XPath which allows the block to be located again easily on new pages. Links such as “next” or “all” are expressed as direct relative XPath, while numbered elements are expressed as grouped relative XPath. This XPath is then passed to Selenium, which handles the interaction.

Using this process, the required interaction with the web-page is completely automated to gather UGC by emulating user interaction. Once all pagination structures have been found and expanded a full set of page content is available. This can then be used in the Data Extraction process presented in the next section.

Figure 5. Locating and interacting with data links

```
1 COMMENT_MULTIPLE_WORDS = ['read', 'more', 'all', 'show']
2 COMMENT_LINK_WORDS = ['comments', 'replies']
3
4 COMMENT_WORD_REGEX = regex.compile(
5     r'.*{ }.*|.*{ }.*'.format(
6         r'(?={})'.format(
7             r'|'.join(
8                 [r'.*\b{ }\b'.format(word) for word in
9                 COMMENT_MULTIPLE_WORDS]
10            ),
11            r'(?={})'.format(
12                r'|'.join(
13                    [r'.*\b{ }\b'.format(word) for word in
14                    COMMENT_LINK_WORDS]
15                ),
16                r'(\d+\b({}))'.format(r'|'.join(COMMENT_LINK_WORDS))
17            ), regex.I
18        )
19
20 """ Generates:
21
22 .*
23 (?=.*\bread\b.*\bmore\b|.*\ball\b|.*\bshow\b)
24 (?=.*\bcomments\b|.*\breplies\b)
25 .*
26 |
27 .*(\d+\b(comments|replies)).*
28
29 """
```

UGC DISCOVERY APPROACH

CFH-NS (Canopied Feature Hashing with Nested Structure Detection) is a new UGC extraction algorithm developed during this research to specifically target user-generated content. Our example implementation uses the Python (Van Rossum & Drake, 2003) language and libraries to parse the DOM tree (web-page structure).

Figure 6. Simplified example of finding a numbered pagination block

```
1: function FINDPAGINATIONNUMBERS(domtree)
2:   for each tag in domtree do
3:     values ← tag.children.strings
4:     digits ← LIST(value for value in values if value is an integer)
5:     spans ← INTSPAN(digits)    ▷ converts [1,2,3,5] into [1-3,5]
6:     if length(digits) > length(spans) then          ▷ found consecutive ints
7:       if SORTED(digits) == digits then    ▷ ints appeared in order
8:         links ← buttons or links in tag.children
9:         if length(links) ≥ length(digits) then
10:           pagination ← tag
11:           return pagination
12:         end if
13:       end if
14:     end if
15:   end for
16: end function
```

To parse page structures, the algorithm operates by breaking a DOM tree into individual branches, anchoring at different nodes within the tree. An example DOM tree for a single social comment is presented in Figure 7 in which the desired anchor would be the top-level LI tag. To locate similar structures, the algorithm matches the structure derived from that branch with other branches in the tree. A quick and effective means of comparing these branches is through the generation of a canopied feature hash, or TagHash.

Canopied Feature Hashes

TagHashes express the structure of a tree and its associated tags in a simplified manner, stripping out unique features. Rather than attempting to match element attribute values directly (which may include unique attributes such as an ID) the existence of attributes is matched, specifically ID and name. This provides a level of differentiation not present if only matching tag types, and assists in tree comparison.

TagHashes are expressed internally as a Python dictionary and are serialized as a JavaScript Object Notation (JSON) string. JSON is used due to simple bidirectional serialization, and Python dictionaries have the additional benefit of supporting tree comparison functions. Each TagHash is a simplified representation of the DOM tree for that structure, generalized to allow for partial-tree matching while retaining some differentiating features. Each TagHash is effectively a signature for that part of the web-page structure.

To match similar structures and identify as the maximum amount of UGC structures as possible, a method of comparing DOM structures is required. Several techniques have been used in previous

Figure 7. An example DOM tree for a single comment

```
1 <li>
2   <a id="m_ucMessageDisplay1548290_m_anchMessageAnchor"
   ↪ name="m1548290"></a>
3   <h3 class="">Patrick:</h3>
4   <p class="date">29 Jan 2015 3:15:55pm</p>
5   <p>Abbott displays all the hallmarks of a highly
   ↪ delusional right-man. He appears egotistical in the
   ↪ extreme and it should now be obvious to all that he
   ↪ is an extremely dangerous individual and one who
   ↪ should never be in a position of power, let alone
   ↪ being leader of a nation</p>
6   <p>
7     <span>
8       <a class="popup"
       ↪ href="NewMessage.aspx?b=69& t=12532">
9         Reply
10      </a>
11    </span>
12    <span>
13      <a class="popup"
14      ↪ href="AlertModerator.aspx?b=69& m=1548290">
15        Alert moderator
16      </a>
17    </span>
18  </p>
19  <ul></ul>
</li>
```

research (Hultgren et al., 2016; Jindal & Liu, 2010; Zhai & Liu, 2005). Our initial prototypes used a simple tree-comparison algorithm to find identical matches between structures. Due to the structurally-variable nature of UGC and page design, this was subsequently refined to handle partial matches.

Excellent results were achieved by using a text representation of each TagHash and computing the difference between them using standard text comparison tools. This approach includes the ability to handle unaligned or partial matches in the page trees. Further, text representations ensure that any differences (whether an attribute change, a full branch change or a node change) are easily distinguishable. Differences between TagHashes are represented internally by “+” or “-” denoting node or branch addition/subtraction. This provides an efficient way of performing scored comparisons by assigning each action (tag modification, tag addition/subtraction, branch addition/subtraction) with a penalty score. By comparing this penalty with the number of tags involved in the comparison, it is possible to assign a score or similarity percentage between two trees. Trees are deemed similar if over a specified threshold score. This allows for granular control of similarity scores for different trees

(Bille, 2005). An appropriate threshold can be designated depending on application requirements for signal-to-noise: a higher threshold will match only very similar structures and can potentially miss data from some sites with less structured designs.

Finding Repeating Structures

To detect repeating UGC structures on a page, a list of TagHashes is first built. This can be built using every possible tag on a page, or restricted to a subset of tag types likely to hold UGC items. The TagHash list class contains additional functionality to automatically group structures based on their TagHash, and can then output common structural XPath. The TagHash list also provides the ability to apply custom filters over the data to ensure that the identified structures contain certain items - this is particularly useful when dealing with UGC.

With a populated TagHash list, structural XPath can be generated to match discovered structures. This XPath can then be used to locate structures to extract data from additional pages.

UGC Filtering

The structure-matching process yields a set of similar data structures, but little information as to what they contain. As UGC is the primary focus of this process, it is desirable to filter the detected data records to only those that are likely to contain UGC. These principles can be applied to other data types as well – for example, filters might be developed to limit the data records to just those containing e-commerce data.

As discussed, UGC structure typically contains at least three elements: an origin, a timestamp, and a message. Corresponding datatypes are a string (origin), a date (timestamp) and text (message). Datatypes can thus be associated with fields within the discovered structures and filters designed to match these types.

Type Discovery

Parsing the content to determine the datatype of a single field is a significant task. A set of three words could be a short string or a short block of text. For UGC, this difference is important as a string can represent a user id or location, while text usually indicates message content. To ensure that fields are not misclassified, the individual fields must be considered alongside the matching fields from other structures to provide context.

With multiple values available per-field, the probability that a field is of a certain datatype can be determined and compared to a predetermined threshold. Using this method of probability-based type-checking, the reliability of field data typing can be improved.

To perform this datatype determination, the value of each descendant tag in a structure is first extracted by iterating through the whole data tree. This presents a further challenge: if a UGC structure is nested within a parent structure, such as replies to a comment, fields may be within several levels of nested structure. To ensure accuracy of datatyping, the TagHash of structures is compared against their ancestors - if a match exists, a nested structure has been discovered.

Some types require more work to validate than others. Text can be discovered by checking for the presence of newline characters or linefeeds and having high average word counts, while strings are short and generally do not contain punctuation. Dates can be particularly problematic, as they can be represented in standard ways (e.g. ISO8601), as times (e.g. 5:12 am) or in relative terms (e.g. 16 minutes ago).

The type determinations for each instance of the field are represented as Boolean values and used to determine the percentage of instances that identify as each type. If any of these types exceeds the predetermined threshold, the field is considered to be of that type. Once an appropriate type has been decided, this may be flagged based on the relative field location and the position of the data (e.g. within a certain attribute, or the text value of the tag).

During this process, additional data alignment and cleaning is also performed. If the field is only aligned with a few structures, it may be optional and gets flagged as such. Fields that have constant templated values are discarded. Any tag that contains a series of text-like tags is noted as a text parent, and children are discarded from the list - this prevents individual paragraphs being included as separate fields.

Field Cleansing

With each valid field sorted into type-buckets, a filter checking for the existence of certain types can be applied. For UGC, a string, date, and a text field are required; if the structure satisfies these requirements it is appended to a list of discovered UGC fields. Using the discovered set of UGC structures and their fields, data is extracted and aligned. In this phase of filtering, the most appropriate instance of each type is automatically selected. Once completed, the resulting structure and list of fields are generated into a wrapper.

Filtered Extraction

This wrapper can then be used to extract data from other pages made from the same templates. Figure 8 shows the data ultimately extracted using the wrapper generated in the previous step. The output data contains the detected UGC, which includes the correct datatype and is structured consistently. The content is injected into a structure with appropriately named fields. Following any optional filtering or cleaning, the UGC data can now be considered to be in a structured form. This may be easily imported and used in analytical tools or stored in traditional relational database systems.

EVALUATION

The effectiveness of the new CFH-NS algorithm was evaluated by comparing it with algorithms from prior research. To test these algorithms, each was run against a testbed of HTML data and extracted structures were counted to determine recall and precision. To evaluate the effectiveness of automated interaction algorithms, the amount of data collected using traditional HTTP requests was compared with the data collected using the interaction and rendering techniques.

Testbeds for web data extraction algorithms already exist, such as TBDW (Yamada, Craswell, Nakatoh, & Hirokawa, 2004), ViNTs (W. Liu et al., 2005) and others. Each of these testbeds contains a set of pages that contain data records, such as Search Record Results or List Records. Evaluating an algorithm against these testbeds is relatively simple - the algorithm is run against the stored data and the number of detected data records is compared with the number that is known to exist in the testbed.

Two testbeds commonly-used in previous research (TBDW and ViNTs) were compiled in 2004 and 2005, and represent typical web design techniques for that period. However, web design has dramatically changed since then. HTML standards have evolved to encourage better structure while introducing higher complexity, which significantly alters the requirements for web data extraction algorithms. Modern pages also rely heavily on user interaction to render content, making the mechanisms implemented in many WDE algorithms non-functional.

These changes have effectively rendered the standard evaluation testbeds inappropriate, as they no longer represent the modern internet. Regardless, they would be unsuitable for evaluation in this context: user-generated content was seldom embedded into pages during the time period in which the testbeds were compiled. The only sites that tended to contain UGC were message boards, whereas the shift to Web 2.0 ideologies has driven the expansion of primarily user-generated content pages in many domains. These fundamental changes required not only the development of new extraction technique but also of new evaluation techniques including a testbed. This testbed and supporting data are available to the public domain to support future research and to enable researchers to directly compare their findings with the results described in this manuscript.

Figure 8. Data extracted from a document using the wrapper generated

```
1  [
2    {
3      'other': [],
4      'content': ['Abbott displays all the hallmarks of a highly
5        ↪ delusional...'],
6      'datetime': ['29 Jan 2015 3:15:55pm'],
7      'name': ['Patrick']
8    },
9    {
10     'other': [],
11     'content': ["Every footy team needs a head-kicker but you
12       ↪ don't make him captain"],
13     'datetime': ['29 Jan 2015 3:47:38pm'],
14     'name': ['Tony']
15   },
16   {
17     'other': [],
18     'content': ['@Tony:\nTony Abbott displayed all of his
19       ↪ head kicking prowess as...'],
20     'datetime': ['29 Jan 2015 4:17:03pm'],
21     'name': ['JohnC']
22   },
23   {
24     'other': [],
25     'content': ['Like'],
26     'datetime': ['29 Jan 2015 6:07:58pm'],
27     'name': ['Arthur']
28   },
29 ]
```

METHODOLOGY

To test the CFH-NS algorithm against existing solutions, a new testbed was compiled that is more representative of the current state-of-the-art in web design and development methodologies.

A set of candidate URLs was crowdsourced, with participants directed to provide sites that they commonly visited that used UGC in some way. The candidate URLs provided covered news sites, social media, blogs, and online stores. This crowdsourced list was then filtered to exclude sites that met any of the below criteria:

1. The presence of an inappropriate amount of UGC (i.e. less than 10 items or more than 1000 items).
2. Multiple sites with duplicate structures, e.g. blogs operating on the same software with similar design themes.
3. Obscure sites or those that are unrepresentative of the state of the internet, e.g. sites that haven't undergone design upgrades in the last decade.
4. Sites on unreliable servers that didn't always return page data in a timely fashion.

After filtering, a total of 49 URLs remained. The pages require various amounts of interactivity to retrieve data, consisting of different types of pagination and expand/redirect link types and covering a wide range of design strategies for these elements. The page designs also vary, with some using tag types in structures (such as list elements), and others preferring to use separate HTML sections for all content (which makes extraction significantly more difficult). The site designs represent modern web design and are representative of popular content on the internet. These include a mixture of sites that require dynamic interaction or statically presented content.

While the older testbeds from prior research are unsuitable for use in this evaluation, the algorithms themselves are not. Although designed to detect simple data records such as search results and e-commerce data tables, UGC constructs share many of the same qualities and legacy WDE algorithms should be able to detect (but not necessarily isolate) UGC. For these legacy WDE algorithms, only the effectiveness of structure detection is evaluated.

Furthermore, some of the algorithms from prior research come with their own data collection processes that are not able to handle dynamic content. Where this was the case, our data collection method was used to extract the dynamic data and supply it to those algorithms, to allow for a fairer side-by-side comparison.

User Generated Content Extraction

Web Data Extraction algorithms come in two broad types: those designed to operate on single pages, and those designed to operate on sets of similar pages. As the CFH-NS algorithm is designed to operate on a single page, it was benchmarked against other algorithms of this type.

An attempt was made to acquire the code for several WDE algorithms by contacting the authors of previous research; however limited replies were received. Few algorithms are provided online, and not all are still functional. Hence, the algorithms being evaluated are Data Extraction Based on Partial Tree Alignment (DEPTA) (Zhai & Liu, 2005) and Bottom-Up Wrapper (BUW) (Thamviset & Wongthanavas, 2014a), both of which are publicly available.

CFH-NS can split up fields into their appropriate type and identify fields relevant to UGC, which neither DEPTA nor BUW was designed to do, so this functionality was not evaluated. The DEPTA and BUW algorithms are evaluated slightly differently. Neither algorithm provides the ability to filter UGC, so this process was performed manually upon the various record types returned by each algorithm. If a discovered record represented a single social comment that contained the necessary data, it was deemed a successful extract. Any records that contained nested data, missed necessary fields or represented a field within a record were considered a failed extract. Records that were unrelated to UGC were discarded and not included in the evaluation, and are not reflected in recall or precision.

DEPTA provides a Java binary that can be operated on a set of collected data, but BUW only provides a live web interface and performs its own collection. DEPTA was passed a full set of post-rendered data and interacted content, but BUW was unable to perform these duties on its own. To account for these problems, two sets of tests were completed. Using data procured, interacted and rendered by CFH-NS, the performance of both CFH-NS and DEPTA was evaluated for extracting UGC structures. Then, using their own collection mechanisms, DEPTA and BUW were again evaluated. In keeping with evaluation practices in previous research, results for recall and precision are provided: "All Results" describes recall and precision if including collection or WDE failures as part of the

results, while “Success Only” excludes failures and only determines recall and precision for pages in which at least one result was found.

The recall represents the number of relevant records available on a page that was successfully discovered, while precision represents the number of discovered elements that were relevant. F-score is calculated using a standard balanced algorithm that represents the harmonic mean of precision and recall:

$$F_1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

These results are presented in Figure 9. Actual Results-Paginated (AR-P) is the number of available results from the paginated dataset, produced by the new CFH-NS algorithm. Actual Results (AR) is the number of available results without any interaction required. The figures in each algorithm’s column represent the total number of results found and the number of relevant results. DEPTA-L and BUW-L are the “live” versions of each algorithm, and represent the results if the algorithms perform their own data collection rather than using CFH-NS’s collected set.

As seen in the table, CFH-NS provides a significant increase in recall compared to DEPTA using the paginated set, while retaining similar precision. This also represents a significant improvement in both recall and precision compared to BUW, though BUW performs admirably when only counting successful tests. When comparing all results, CFH-NS represents a major improvement in F-score.

A large part of the difference in recall between CFH-NS and the live versions of DEPTA and BUW is due to dynamic DOM parsing. Neither algorithm is able to dynamically render the web page structures or retrieve the UGC on pages. By counting the URLs in which both DEPTA and BUW fail to retrieve data, it is possible to determine how many pages use dynamic mechanisms to render social content. 51% of sites tested require dynamic parsing to retrieve UGC, indicating that algorithms without the ability to deal with this problem are unable to collect significant amounts of UGC.

CFH-NS represents a major improvement in the ability to collect UGC from web pages. This technique can be used to develop a reliable API for accessing previously-inaccessible UGC, broadening the scope of social data analytics applications. This is particularly useful for advertising and product-based analytics, as CFH-NS can be used to generically mine product reviews and user comments in the context of the product. It can also be used to mine real-time blogs and microblogs, providing a broader set of data sources for event detection.

Figure 9. Content extraction experimental results

	AR-P	CFH-NS	DEPTA	AR	DEPTA-L	BUW-L
Total	3155	3454/2981	853/764	1675	149/124	704/593
All Results:						
Recall		94.5%	24.2%		7.4%	35.4%
Precision		89.2%	89.6%		83.2%	84.2%
F-score		0.92	0.38		0.14	0.50
Success Only:						
Recall		94.5%	48.1%		48.1%	71.6%
Precision		89.2%	89.6%		83.2%	84.2%
F-score		0.92	0.63		0.61	0.77

Dynamic Interaction

To test the effectiveness of automatic dynamic interaction, the number of records available on a page without interaction is compared against the number of records available using interaction. Some URLs in the dataset normally used pagination, but there was not enough UGC present to be paged. Of 49 URLs, 10 (20.5%) contained data hidden behind pagination or expansion links. In addition, 19 (38.8%) had pagination code, though there was not enough UGC on the selected pages to trigger its usage. In total, 29 (59.2%) sites required navigation of pagination to extract available data. This proportion of sites requiring complex user interaction to access data highlights the need to develop better methods of automated interaction, a field that has received little attention to date.

These pages and the results of automated interaction are presented above. As shown in the table, the automated interaction substantially increased the amount of UGC retrieved in all tests, yielding in total 428% more UGC. This suggests that a significant volume of web data is currently inaccessible to many existing data collection mechanisms.

Interestingly, sites that require automated interaction are very often those that use pagination to handle their heavy traffic or large volumes of UGC. These missed opportunities may actually be the most useful and desirable candidates for UGC mining. Being able to access this previously-unavailable data using CFH-NS is a significant step forward, and allows for more extensive mining of UGC to unlock its analytical value.

Figure 10. Additional data retrieved through CFH-NS's automated interaction techniques

URL	AR-P	AR	Increase
http://newsfeed.gawk...	42	20	210.0%
http://the-toast.net...	41	27	151.9%
http://www.adelaiden...	101	50	202.0%
http://www.cracked.c...	592	34	1741.2%
http://www.dailytele...	72	50	144.0%
http://www.destructo...	219	50	438.0%
http://www.dpreview...	193	176	109.7%
http://www.smh.com.a...	47	10	470.0%
http://www.theage.co...	545	10	5450.0%
http://www.tripadvis...	20	10	200.0%
Totals	1872	437	428.4%

Limitations and Future Work

The three techniques presented in this paper have some limitations that would provide a good basis for future work.

The automated interaction algorithm is subject to certain phrases or design styles being used in the source code of the pages, and does not function well outside these general cases - a more robust method of detecting expansion and redirection links may provide access to more data.

The CFH-NS algorithm does not perform well with very loose tag structures, particularly those that rely heavily on inline HTML tags. It was developed to work with well-designed page structures, which causes performance to suffer on legacy designs. Combination of the algorithm with those that perform text-based context extraction (e.g. BUW) could improve recall and precision on certain sites.

Finally, the probabilistic datatype determination technique created is a simple threshold-based model and could benefit from the use of more complex statistical methods, though the accuracy of the current model performs adequately for this implementation.

There are additional broader opportunities for future work. These include; (i) implementing the actionable intelligence layer indicated in the KM framework; (ii) improving recall and more precise selection of data; and (iii) using intelligent algorithms to locate new UGC sources.

CONCLUSION

Regardless of the application domain, every KM system requires data. In the modern Internet, there is a rich and diverse set of data in the form of user-generated content, which is largely inaccessible to KM techniques due to the data extraction challenge. The work described in this paper addresses this issue and presents a new algorithm for data extraction from such sources. The paper incorporates a KM framework as a paradigm for this knowledge management and data value extraction. The incorporated framework presents three contributions to the field of UGC and social media analytics. The first contribution involves a method for automatically navigating generic sites to extract UGC. This overcomes the limitations of traditional web data extraction methods by fully emulating the actions of a user, thus revealing data elements hidden by JavaScript or pagination. The second contribution, the CFH-NS algorithm, makes it possible to discover and extract nested UGC page structures while classifying UGC fields. Thirdly, as the research revealed the need for a more modern testbed, this has been developed and shared in the public domain to invite and support future research. All contributions described in this paper have been empirically evaluated, both with existing testbeds if applicable, as well as with the new testbed described above.

CFH-NS, the UGC extraction algorithm, represents a new algorithm capable of automatically extracting UGC from web pages. Results demonstrate that overall data recall is significantly improved compared to existing data extraction algorithms. The user emulation and interaction techniques can significantly increase the amount of UGC collected from web pages by automatically navigating UGC pagination while the data typing and classification algorithms provide a standardized interface to the UGC.

The combined use of these techniques increases the reach of UGC mining to semi-structured content including news articles and comments, forums and sites that do not provide an API for retrieving data. This data is made accessible through a standard interface that enables more direct integration into analytical applications. This work may be integrated into a full framework designed to integrate diverse social data sources and provide a unified interface for performing social analytics. This is the ultimate objective: to realize the full potential of UGC and social data as a low-cost and real-time data source, capable of providing actionable insights on a global scale.

REFERENCES

- Abu-Salih, B., Wongthongtham, P., & Chan, K. Y. (2018). Twitter mining for ontology-based domain discovery incorporating machine learning. *Journal of Knowledge Management*, 22(5), 949–981. doi:10.1108/JKM-11-2016-0489
- Abu-Salih, B., Wongthongtham, P., Chan, K. Y., & Zhu, D. (2019). CredSaT: Credibility ranking of users in big social data incorporating semantic analysis and temporal factor. *Journal of Information Science*, 45(2), 259–280. doi:10.1177/0165551518790424
- Althoff, T., Jindal, P., & Leskovec, J. (2017). Online actions with offline impact: How online social networks influence online and offline user behavior. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM. doi:10.1145/3018661.3018672
- Arasu, A., & Garcia-Molina, H. (2003). Extracting structured data from web pages. In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*. ACM. doi:10.1145/872757.872799
- Arularasan, A., Suresh, A., & Seerangan, K. (2018). Identification and classification of best spreader in the domain of interest over the social networks. *Cluster Computing*, 1–11.
- Atzeni, P., & Mecca, G. (1997). Cut and paste. In *Proceedings of the 16th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*. ACM. doi:10.1145/263661.263678
- Bille, P. (2005). A survey on tree edit distance and related problems. *Theoretical Computer Science*, 337(1), 217–239. doi:10.1016/j.tcs.2004.12.030
- Blanvillain, O., Kasioumis, N., & Banos, V. (2014). Blogforever crawler: techniques and algorithms to harvest modern weblogs. In *Proceedings of the 4th International Conference on Web Intelligence, Mining and Semantics (WIMS14)*. Academic Press. doi:10.1145/2611040.2611067
- Cao, D., Liao, X., Xu, H., & Bai, S. (2008, January). Blog post and comment extraction using information quantity of web format. In *Asia Information Retrieval Symposium* (pp. 298-309). Springer.
- Chang, W.-L., Diaz, A. N., & Hung, P. C. (2015). Estimating trust value: A social network perspective. *Information Systems Frontiers*, 17(6), 1381–1400. doi:10.1007/s10796-014-9519-0
- Cheong, H. J., & Morrison, M. A. (2008). Consumers' reliance on product information and recommendations found in UGC. *Journal of Interactive Advertising*, 8(2), 38–49. doi:10.1080/15252019.2008.10722141
- Crescenzi, V., & Mecca, G. (1998). Grammars have exceptions. *Information Systems*, 23(8), 539–565. doi:10.1016/S0306-4379(98)00028-3
- Crescenzi, V., Mecca, G., & Merialdo, P. (2001). *Roadrunner: Towards automatic data extraction from large web sites*. VLDB.
- Cruz, C. (2019). Social Networks and the Targeting of Vote Buying. *Comparative Political Studies*, 52(3), 382–411. doi:10.1177/0010414018784062
- Davenport, T. H., & Prusak, L. (1998). *Working knowledge: How organizations manage what they know*. Harvard Business Press.
- Dorasamy, M., Raman, M., & Kaliannan, M. (2013). Knowledge management systems in support of disasters management: A two decade review. *Technological Forecasting and Social Change*, 80(9), 1834–1853. doi:10.1016/j.techfore.2012.12.008
- Fang, Y., Xie, X., Zhang, X., Cheng, R., & Zhang, Z. (2018). STEM: A suffix tree-based method for web data records extraction. *Knowledge and Information Systems*, 55(2), 305–331. doi:10.1007/s10115-017-1062-0
- Ferrara, E., & Baumgartner, R. (2011). Automatic wrapper adaptation by tree edit distance matching. In *Combinations of Intelligent Methods and Applications* (pp. 41-54). Springer.
- Ferrara, E., De Meo, P., Fiumara, G., & Baumgartner, R. (2014). Web data extraction, applications and techniques: A survey. *Knowledge-Based Systems*, 70, 301–323. doi:10.1016/j.knosys.2014.07.007

Gramlich, J. (2019). Facts about Americans and Facebook. *Pew Research Center*. Retrieved from <http://www.pewresearch.org/fact-tank/2019/02/01/facts-about-americans-and-facebook/>

Gupta, K., Mittal, V., Bishnoi, B., Maheshwari, S., & Patel, D. (2016). AcT: Accuracy-aware crawling techniques for cloud-crawler. *World Wide Web (Bussum)*, 19(1), 69–88. doi:10.1007/s11280-015-0328-2

Hidayat, A. (2013). Phantomjs. PhantomJS. Vers 1.7 [Computer software].

Holmes, A., & Kellogg, M. (2006). Automating functional tests using selenium. In *Proceedings of the Agile Conference*. Academic Press.

Hosseingholizadeh, R., Sharif, A., & Kouhsari, M. (2018). PKM Tools for Developing personal knowledge management skills among university students. *International Journal of Information Science and Management*, 16(1).

Hultgren, M., Jennex, M. E., Persano, J., & Ornatowski, C. (2016). Using knowledge management to assist in identifying human sex trafficking. In *Proceedings of the 2016 49th Hawaii International Conference on System Sciences (HICSS)*. Academic Press. doi:10.1109/HICSS.2016.539

Jennex, M. E. (2007). What is knowledge management? In *Knowledge management in modern organizations* (pp. 1–9). Hershey, PA: IGI Global. doi:10.4018/978-1-59904-261-9.ch001

Jennex, M. E. (2017). Big Data, the Internet of Things, and the Revised Knowledge Pyramid. *ACM SIGMIS Database: the DATABASE for Advances in Information Systems*, 48(4), 69–79. doi:10.1145/3158421.3158427

Jennex, M. E., & Bartczak, S. E. (2013). A revised knowledge pyramid. *International Journal of Knowledge Management*, 9(3), 19–30. doi:10.4018/ijkm.2013070102

Jindal, N., & Liu, B. (2010). A generalized tree matching algorithm considering nested lists for web data extraction. In *Proceedings of the 2010 SIAM International Conference on Data Mining*. Academic Press. doi:10.1137/1.9781611972801.81

Kasemsap, K. (2019). Professional and business applications of social media platforms. In *Social Entrepreneurship: Concepts, Methodologies, Tools, and Applications* (pp. 824-847). Hershey, PA: IGI Global. doi:10.4018/978-1-5225-8182-6.ch042

Kejriwal, M. (2019). Information Extraction. In *Domain-Specific Knowledge Graph Construction* (pp. 9–31). Cham: Springer International Publishing. doi:10.1007/978-3-030-12375-8_2

Liu, B., Grossman, R., & Zhai, Y. (2003). Mining data records in web pages. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. doi:10.1145/956750.956826

Liu, W., Meng, X., & Meng, W. (2005). ViNTs: Visual information aNd Tag structure based wrapper generator. Retrieved from <http://www.data.binghamton.edu/vints/testbed.html>

Liu, W., Meng, X., & Meng, W. (2010). Vide: A vision-based approach for deep web data extraction. *IEEE Transactions on Knowledge and Data Engineering*, 22(3), 447–460. doi:10.1109/TKDE.2009.109

Maynard, D., Bontcheva, K., & Rout, D. (2012). Challenges in developing opinion mining tools for social media. *Proceedings of the @ NLP can u tag# usergeneratedcontent*, 15-22.

Meneghello, J., Thompson, N., & Lee, K. (2014). Towards social media as a data source for opportunistic sensor networking. In *Proceedings of the 12th Australasian Data Mining Conference (AusDM 2014)*, Brisbane, Australia.

Miao, G., Tatemura, J., Hsiung, W.-P., Sawires, A., & Moser, L. E. (2009). Extracting data records from the web using tag path clustering. In *Proceedings of the 18th International Conference on World Wide Web*. Academic Press. doi:10.1145/1526709.1526841

Nishikant, B., Prabin Kumar, P., & Shashi Kant, S. (2018). Knowledge discovery from vernacular expressions: An application of social media and sentiment mining. *International Journal of Knowledge Management*, 14(1), 1–18. doi:10.4018/IJKM.2018010101

Nonaka, I., & Takeuchi, H. (1995). *The knowledge-creating company: How Japanese companies create the dynamics of innovation*. Oxford university press.

Polanyi, M. (2009). *The tacit dimension*. University of Chicago Press.

- Rainer, R. K., Cegielski, C. G., Spletstoesser-Hogeterp, I., & Sanchez-Rodriguez, C. (2013). *Introduction to information systems*. John Wiley & Sons.
- Robinson, B., Power, R., & Cameron, M. (2013). A sensitive twitter earthquake detector. In *Proceedings of the 22nd International Conference on World Wide Web*. Academic Press.
- Sahoo, P., & Parthasarathy, R. (2018). An efficient web search engine for noisy free information retrieval. *The International Arab Journal of Information Technology*, 15(3), 412–418.
- Schaupp, L. C., & Bélanger, F. (2019). Social commerce benefits for small businesses: An organizational level study. In *Social Entrepreneurship: Concepts, Methodologies, Tools, and Applications* (pp. 1237-1255). Hershey, PA: IGI Global.
- Shouhong, W., & Hai, W. (2018). Social-Media-Based Knowledge Sharing: A Qualitative Analysis of Multiple Cases. *International Journal of Knowledge Management*, 14(1), 19–29. doi:10.4018/IJKM.2018010102
- Social Media News. (2019). Social Media Statistics Australia. Retrieved from <https://www.socialmedianews.com.au/social-media-statistics-australia-january-2017/>
- Stewart, T. A., & Kirsch, S. (1991). Brainpower. *Fortune*, 123(11), 44–50.
- Surendro, K., Satya, D. P., & Yodihartomo, F. (2018). Integrated Social Media Knowledge Capture in Medical Domain of Indonesia. *Telkomnika*, 16(4), 1846–1856. doi:10.12928/telkomnika.v16i4.8320
- Thamviset, W., & Wongthanavas, S. (2014a, 30 Jul-1 Aug). Bottom-up region extractor for semi-structured web pages. In *Proceedings of the International Computer Science and Engineering Conference (ICSEC)*, Khon Kaen, Thailand. Academic Press. doi:10.1109/ICSEC.2014.6978209
- Thamviset, W., & Wongthanavas, S. (2014b). Information extraction for deep web using repetitive subject pattern. *World Wide Web*, 17(5), 1109–1139. doi:10.1007/s11280-013-0248-y
- Wandhöfer, T., Taylor, S., Walland, P., Geana, R., Weichselbaum, R., Fernandez, M., & Sizov, S. (2012). Determining Citizens' Opinions About Stories in the News Media. *JeDEM-eJournal of eDemocracy and Open Government*, 4(2), 198-221.
- Yamada, Y., Craswell, N., Nakatoh, T., & Hirokawa, S. (2004). Testbed for information extraction from deep web. In *Proceedings of the 13th international conference on World Wide Web*. Academic Press. doi:10.1145/1013367.1013468
- Yuliana, O. Y., & Chang, C.-H. (2018). A novel alignment algorithm for effective web data extraction from singleton-item pages. *Applied Intelligence*, 48(11), 4355–4370. doi:10.1007/s10489-018-1208-0
- Zhai, Y., & Liu, B. (2005). Web data extraction based on partial tree alignment. In *Proceedings of the 14th international conference on World Wide Web*. Academic Press. doi:10.1145/1060745.1060761

James Meneghello is a Principal Data Scientist and Software Development Manager with Optika Solutions in Western Australia. He provides high level consultancy and solution development for multinational clients primarily in the oil and gas industry. Nik Thompson is a Senior Lecturer in the School of Information Systems at Curtin University in Western Australia. He holds MSc and PhD degrees from Murdoch University and teaches in the area of Computer Security and Information Systems. His research interests include affective computing, human-computer interaction and information security.

Kevin Lee is a Senior Lecturer in Software Engineering and Internet of Things (IoT) at the School of Information Technology in Deakin University, Australia. Before joining Deakin University in 2018 he held the position of Senior Lecturer in Computing and Technology at Nottingham Trent University, UK (2015-2018). Previously, he held the position of Senior Lecturer in Information Technology at Murdoch University, Australia (2010-2015). He was a postdoctoral research fellow at the University of Mannheim, Germany (2009-2010). He was a research associate at the University of Manchester, UK (2006-2009). He received his PhD in Computer Science from Lancaster University, UK in 2006. He is currently also an appointed external examiner for the Department of Computing and Technology, Ulster University, for the Computing suite of degrees (2017-2018). He has over 70 publications in high quality journals and conferences, with a H-Index of 13 and over 1100 citations. His research has mainly been in the Distributed Systems and Autonomous Computing areas. He has published in the areas of IoT, Cloud computing, robotics, embedded systems, Big Data, P2P network monitoring and physiological computing. His current main active research area is in IoT and its integration with Cloud computing in support of dynamic IoT applications.

Kok Wai Wong received his B.Eng (Hons) and Ph.D. degree, in the School of Electrical and Computer Engineering from Curtin University of Technology, Australia, in 1994 and 2000 respectively. He is a senior member of IEEE since 2004. He is currently working as an Associate Professor at the School of Engineering and Information Technology in Murdoch University, Australia. He has published over 150 articles in journals and conference proceedings. He is the Western Australia Chapter Chair for the IEEE Systems, Man, and Cybernetics Society in 2015 and 2016. His research interests include intelligent data analysis, computational intelligence, game technology, and technology for healthcare and well-being. Bilal Abu Salih received the B.Sc. degree in Computer Science from Qatar University, Qatar, and the M.Sc. degree in Computer Science from Al-Balqa Applied University, Jordan. He received his Ph.D. in 2018 from the School of Information Systems, Curtin University, Australia. His current research interests include big social data analytics, trust, semantic analytics and distributed computing.