ISSN 2311-4738 (print), ISSN 2413-3000 (online)

УДК 004.053:519.681 DOI: 10.20998/2413-3000.2018.1277.5

О. Ю. ЧЕРЕДНІЧЕНКО, М. А. ГРИНЧЕНКО, А. В. ВАСИЛЕНКО, О. М. МАТВЄЄВ

МЕТОД ПОШУКУ ТА АНАЛІЗУ ДАНИХ З ІНТЕРНЕТ РЕСУРСІВ ДЛЯ ФОРМУВАННЯ АКТУАЛЬНИХ ВИМОГ ДО КАНДИДАТІВ

У статті розглянуті питання екстракції даних з Web-ресурсів на прикладі збору інформації щодо вакансій. Виділено три основні взаємодіючі сторони цього процесу: джерело даних, база даних та експерт. Розглянуто основні проблематичні сторони процесу видобування даних, а саме: наявність декількох джерел даних; представлення даних різними мовами; видобування даних з різних форматів файлів; багаторазові повторювані операції і безперервні оновлення. Проаналізовано та визначено переваги та недоліки таких методів WebMiningяк:аналіз DOM дерева, парсинг рядків, використання регулярних виразів, XML парсинг та візуальний підхід. У статті застосовано метод аналізу DOM дерева з використання XPath. Запропоновано використання методу компараторной ідентифікації для моделювання процесу видобування даних. Представлено приклад застосування наведеного підходу для ідентифікації певної вакансії на сайті пошуку роботи. Розроблено тезаурус вимог роботодавців та налаштовано роботу парсера.

Ключові слова: видобування даних, парсинг, компараторна ідентифікація, веб-сторінка, експерт, вакансія.

О. Ю. ЧЕРЕДНИЧЕНКО, М. А. ГРИНЧЕНКО, А. В. ВАСИЛЕНКО, А. Н. МАТВЕЕВ

МЕТОД ПОИСКА И АНАЛИЗА ДАННЫХ ИЗ ИНТЕРНЕТ РЕСУРСОВ ДЛЯ ФОРМИРОВАНИЯ АКТУАЛЬНЫХ ТРЕБОВАНИЙ К КАНДИДАТАМ

В статье рассмотрены вопросы экстракции данных с Web-ресурсов на примере сбора информации о вакансиях. Выделены три основные взаимодействующие стороны этого процесса: источник данных, база данных и эксперт. Рассмотрены основные проблематичные стороны процесса добычи данных, а именно: наличие нескольких источников данных; представления данных на разных языках; добычи данных из различных форматов файлов; многократные повторяющиеся операции и непрерывные обновления. Проанализированы и определены преимущества и недостатки таких методов WebMining как: анализ DOM дерева, парсинг строк, использование регулярных выражений, XML парсинг и визуальный подход. В статье применен метод анализа DOM дерева с использованием XPath. Предложено использование метода компараторной идентификации для моделирования процесса добычи данных. Представлен пример применения указанного подхода для идентификации определенной вакансии на сайте поиска работы. Разработан тезаурус требований работодателей и настроена работа парсера.

Ключевые слова:добыча данных, парсинг, компараторная идентификация, веб-страница, эксперт, вакансия.

O. YU. CHEREDNICHENKO, M. A. GRINCHENKO, A. V. VASYLENKO, O. M. MATVIEIEV

THE METHOD OF DATA SEARCH AND ANALYSIS FROM THE INTERNET RESOURCES FOR THE FORMATION OF ACTUAL REQUIREMENTS FOR CANDIDATES

The article deals with the issues of data extraction from Web-resources on the example of gathering information on vacancies. There are three main interacting parts of this process: data source, database, and an expert. The main problematic aspects of the data mining process are the availability of several data sources; data representation in different languages; extraction data from different file formats; multiple updating of repetitive operations and data. The advantages and disadvantages of Web Mining methods were analyzed and defined. They are DOM tree analysis, line parsing, usage of regular expressions, XML parsing and visual approach. Method of DOM tree using XPath was applied in the paper. The method of comparator identification for modeling the data extraction process was proposed. The component, which receives the search topic and the search start page, carries out a thematically directed extraction. The comparator compares the extracted word from the page with the words of the search model. The application of the above-mentioned approach is presented for identifying a vacancy on the job search site. The thesaurus of employers' requirements is developed. Words-indicators of the required vacancies are presented in three languages. The parser work was set up. The parser processes the documents and retrieves the data used to fill a particular data model. The developed module works as follows. It begins to work with obtaining an array of necessary pages from the selected Web site. The next step is the analysis of Web page's structure. Then it is necessary to get the content of a specific HTML page, which contains the necessary information for its further retrieval and processing. As a result "vacancy model" is developed. The model should include the following elements: vacancy title; date of adding a job to the site; the city where the applicant needs to work; requirements for the candidate; applicant duties; working conditions. Extraction of requirements, liabilities, and conditions was defined as the most problematic area, whereas the same information can be presented in a different way. In order to unify requirement experts were engaged.

Keywords: data mining, parsing, comparative identification, web page, expert, vacancy.

Вступ. Кількість інформації доступної через мережу Інтернет постійно зростає. На жаль, вилучення корисного вмісту з цієї величезної кількості даних залишається відкритим питанням. Відсутність стандартних моделей даних і структур змушує розробників створювати рішення з нуля.

Видобуток знань є важливим завданням у багатьох компаніях і дослідницьких проектах, які вимагають даних, розміщених в Інтернеті, щоб зберігати їх, аналізувати або продавати третім особам. Це завдання вимагає розуміння макета даних і того, що потрібно витягти. У деяких випадках використання описів метаданих або моделей даних може допомогти зрозуміти структуру даних. На жаль, ця інформація недоступна в більшості випадків.

Вилучення знань здійснюється в спеціальних рішеннях. Зазвичай ці рішення включають збір, аналіз, перенесення і зберігання даних. Розробники мають справу з двома різними проблемами: технічною складністю аналізу даного документа і розумінням семантики інформації, що міститься в цьому документі.

Роль експерта все ще потрібна в багатьох ситуаціях, коли розробники не мають правильних

фундаментальних знань. Це змушує розробників витрачати дорогоцінний час, поглинаючи знання експерта. В інших напрямках є багатообіцяючі рішення, що використовують методи машинного навчання. Проте підвищення точності вимагає збільшення складності системи, що неможливо реалізувати в багатьох проектах.

Таким чином, дослідження в напрямку створення технологій видобування знань та програмних рішень в різних предметних галузях задля цілей підвищення ефективності бізнесу є актуальним завданням.

Метою даної роботи є дослідження методів екстракції даних з веб-сторінок, використання яких дозволить підвищити ефективність прийняття рішень шляхом збільшення корисної інформації. В статті розглядається застосування методів екстракції даних на прикладі збору даних щодо вимог роботодавців.

Екстракція даних. Для відображення даних в Інтернеті використовується спеціальна мова. HTML стала найбільш поширеною мовою Інтернету. Проте, HTML не надає ніякого механізму, який полегшує автоматичний аналіз існуючих документів. Це обмеження не дозволяє відрізнити контент від макета і семантики даних.

Кілька стандартів, таких як RDF, RDFS і OWL, були розроблені для забезпечення загального синтаксису для визначення моделей даних. Ці рішення дозволяють визначати онтології, які підтримують запити. Ці технології зазвичай не зрозумілі для розробників, які спочатку ігнорують процес семантичної анотації при розробці HTML-сторінок. Щоб спростити цю проблему, використовують підхід Schema [1], що визначає словник понять, таких як люди, місця, події та продукти, та дозволяє анотувати дані, що містяться в документі HTML.

Дизайн веб-сторінок може приховувати дані від пошукових систем. Використання динамічного контенту, CAPTCHA, приватних веб-сторінок, сценаріїв або незв'язаного контенту серед інших призводить до створення DeepWeb [2]. Простим прикладом є використання веб-сторінок, які виконують пошукові запити по базі даних. Інформація, що міститься в базі даних, не може бути проіндексована пошуковою системою, оскільки для цього потрібно, щоб програмний рушій взаємодіяв з формою пошуку, задавав параметри пошуку і розумів семантику даних, що повертаються. Комерційні пошукові системи, такі як Google, Bing або DuckDuckGo, розробляють свої інструменти з чітким акцентом на індексування так званої поверхневої мережі. Це змушує думати, що більша частина інформації, що міститься в Інтернеті, не індексується. Ця проблема була освітлена проектом MEMEX DARPA [3], в якому показана спроба індексації інформації, що міститься в DeepWeb.

Обробка даних, що містяться на будь-якій з цих веб-сторінок, передбачає певний ступінь взаємодії з людиною (заповнення форми пошуку, взаємодія зі сценарієм і т. д.). Після завантаження необроблених даних він перетвориться в певний формат, який може бути збережений в базі даних для подальшого аналізу. При вирішенні проблеми вилучення знань більшість рішень розробляються з нуля, займаючись видобуванням даних, аналізом і зберіганням. У випадку з декількома джерелами даних складність проблеми зростає до тих пір, поки вона не стане неможливою для вирішення.

Отже, можемо ідентифікувати три основні елементи в будь-якій задачі видобування знань. Перший – джерело даних, що містить відповідну інформацію (наприклад, веб-сторінку). Другий – база даних, яка призначена для зберігання даних (наприклад, MySQL). Третій – експерт, який може визначити, як перетворити дані з джерела в базу даних. Перетворення між джерелом і базою можна вважати виконаним автоматично. Незалежно від рівня автоматизації, роль експерта потрібна для того, щоб вставити деяку вихідну семантику щодо даних до початку екстракції. Крім того, експерт відповідає за визначення правильності вилучення даних.

При роботі з будь-яким проектом видобування веб-знань є кілька аспектів, які необхідно враховувати. Зазвичай в існуючих проектах ігноруються деякі аспекти, такі як:

1. Кілька джерел даних.
2. Більшість існуючих рішень розглядають тільки джерела даних, написані однією мовою.
3. Витяг даних з різних форматів файлів: HTML є найбільш поширеним форматом даних. Однак можуть бути присутніми інші формати, такі як XML, DOC або PDF.
4. Багаторазові повторювані операції і безперервні оновлення.

Постановка завдання. Існування DeepWeb зареєстровано в 1998 році [4]. З тих пір в деяких роботах представлені рішення для автоматичного доступу та індексації цих даних. Raghavan і Garcia Molina [2] розробляють набір модулів, які дозволяють заповнювати бази даних існуючими даними, що зберігаються за формою пошуку. Їх підхід дозволяє вручну визначити набір міток, які можна використовувати для ідентифікації цінної форми пошуку. Якщо виявлена цінна форма пошуку, система запускає запит і заповнює базу даних після маркування відповідних даних. В роботі [5] спробували вирішити ту ж проблему з особливим наголосом на виявлення максимальної кількості документів в системах, доступних тільки через форму пошуку. В роботі [6] розроблено мову, специфічну для домену, так звану DEQUE, яка дозволяє перетворювати запити стилю SQL в запити HTML-форми. Google представила своє власне рішення [7], метою якого є пошук найбільш інформативних запитів для зменшення трафіку при збільшенні масштабованості сканера.

Інше сімейство скануючих рішень дозволяє користувачеві визначити стратегію сканування. Scrapy [8] – це бібліотека Python, призначена для спрощення процесу сканування. Інші рішення, такі як

проект Nutch [6], більш орієнтовані на великі масштабовані сканери і дозволяють поширювати стратегії сканування. Останнім часом все більше число рішень SaaS пропонують онлайн-інтерфейси для визначення стратегій сканування [10, 11]. Ці платформи зазвичай інтегрують плагіни, які дозволяють ідентифікувати ті елементи з веб-сторінки, які повинні скануватися, в браузера. Спрощення процесу обходу може обмежувати визначення політик обходу. Однак використання хмарних ресурсів дозволяє розгортати рішень для сканування протягом невеликого часу.

Прямим рішенням для екстракції знань є розробка парсеру, який обробляє документи і отримує дані, що використовуються для заповнення деякої моделі. Такий підхід може бути достатнім для невеликих обсягів даних з використанням відомих структур даних. Найпростішим рішенням, прийнятим багатьма проектами, є використання XQuery [12] або регулярних виразів для визначення точного шляху до цільового елементу. Цей підхід не дуже стійкий до структурних змін шаблону документа.

Іншим популярним підходом є використання розширених перетворень мови таблиць стилів (XSLT) [13], який забезпечує уніфікований синтаксис для запису правил перетворення між сумісними з XML мовами. У базовій формі HTML в основному сумісний з XML, тому цей підхід може бути застосований до HTML. Цей підхід більш стійкий, ніж XQuery до структурних змін, але його зазвичай дуже складно налагоджувати. Ще одна проста методика, дуже практична в невеликих проектах, – це спрощена версія [14] або HTML. Це дозволяє спростити синтаксис HTML, видаливши всі елементи, крім основного HTML синтаксису і форматування. В інших сценаріях розумним вибором може бути використання тільки більш складних рішень.

В останні роки інтенсивно зростає проблема екстракції знань. Перше сімейство рішень вивчило використання доменних мов для визначення того, як дані повинні бути вилучені. Рішення, подібні до тих, які представлені в [15, 16], використовують мову вилучення декларативної інформації для визначення планів вилучення даних. Аналогічним чином використовуються набори правил. У цих рішеннях якість екстракції особливо залежить від навичок операторів визначати правила вилучення. Друге сімейство рішень досліджує використання методів машинного навчання для поліпшення вилучення інформації. Ці рішення ґрунтуються на використанні моделей виведення, які намагаються побудувати відносини для даного набору даних.

Підходи до видобування даних. WebMining– це процес отримання даних з веб-ресурсів, який, як правило, має більше практичну складову ніж теоретичну. Основна мета WebMining– це збір даних (парсинг) з подальшим збереженням в потрібному форматі. Фактично, завдання зводиться до написання HTML парсерів, розглянемо цей процес більш детально.

Є кілька підходів до вилучення даних:

1. Аналіз DOM дерева, використання XPath.
2. Парсинг рядків.
3. Використання регулярних виразів.
4. XML парсинг.
5. Візуальний підхід.

Розглянемо всі підходи більш детально.

Аналіз DOM дерева, використання XPath ґрунтується на аналізі DOM дерева. Використовуючи цей підхід, дані можна отримати безпосередньо за ідентифікатором імені або інших атрибутів елемента дерева (таким елементом може служити параграф, таблиця, блок і т.д.). Крім того, якщо елемент не позначений будь-яким ідентифікатором, то до нього можна дістатися по якомусь унікальному шляху, спускаючись вниз по DOM дереву або йдучи по колекції однотипних елементів.

Переваги цього підходу:

- можна отримати дані будь-якого типу і будь-якого рівня складності;
- знаючи розташування елемента, можна отримати його значення, прописавши шлях до нього.

Недоліки такого підходу:

- різні HTML/JavaScript движки по-різному генерують DOM дерево, тому потрібно прив’язуватися до конкретного движка;
- шлях елемента може змінитися, тому, як правило, такі парсери розраховані на короткочасний період збору даних;
- DOM-шлях може бути складний і не завжди однозначний.

Наступним еволюційним етапом аналізу DOM дерева є використання XPath, тобто шляхів, які широко використовуються при парсінгу XML даних. Суть даного підходу в тому, щоб за допомогою деякого простого синтаксису описувати шлях до елементу без необхідності поступового руху вниз по DOM дереву. Даний підхід використовує усіма відома бібліотека jQuery і бібліотека HtmlAgilityPack.

Незважаючи на те, що парсинг рядків не можна застосовувати для написання серйозних парсерів, слід звернути на нього увагу. Іноді дані відображаються за допомогою деякого шаблону (наприклад, таблиця характеристик мобільного телефону), коли значення параметрів стандартні, а змінюються лише їх значення. У такому випадку дані можуть бути отримані без аналізу DOM дерева, а шляхом парсингу рядків. Використання набору методів для аналізу рядків іноді (частіше – простих шаблонних випадках) більш ефективне ніж аналіз DOM дерева або XPath.

Регулярні вирази і парсинг XML необхідно використовувати тільки для отримання даних, які мають строгий формат – електронні адреси, телефони і т. д. Ще одним неефективним підходом є розгляд HTML як XML даних. Причина в тому, що HTML рідко буває дійсним, тобто таким, що його можна розглядати як XML дані. Бібліотеки, які реалізували такий підхід, більше часу приділяли перетворенню HTML в XML і вже потім безпосередньо парсингу даних. Тому краще уникати цей підхід.

В даний момент візуальний підхід знаходиться на початковій стадії розвитку. Суть підходу в тому, щоб користувач міг без використання програмної мови або API «налаштувати» систему для отримання потрібних даних будь-якої складності і вкладеності. В даний час існує велика кількість доступних веб-сканерів в проектах з відкритим вихідним кодом.

Метод компараторної ідентифікації. Найбільш перспективним сьогодні стає використання моделей і методів, інформаційних технологій, що базуються на результатах, отриманих при розв'язанні проблем штучного інтелекту.

Знання, на основі яких експерт приймає рішення, можна формалізувати за допомогою методів теорії інтелекту, зокрема методу компараторної ідентифікації [18]. Компаратор реалізує предикат $K(y_1, y_2, ..., y_m) = t$, що відповідає відношенню K, в якому знаходяться вхідні сигнали $y_1, y_2, ..., y_m$. При цьому t – це двійкова реакція компаратора, $t \in \sum, \sum = \{1, 0\}$. До входів компаратора підключені своїми виходами ідентифіковані інформаційні процеси $r_1, r_2, ..., r_m$. Інформаційні процеси представляють механізми сприйняття вхідних фізичних сигналів $x_1, x_2, ..., x_m$. Компаратор разом із підключеними до нього інформаційними процесами називається ідентифікованим об'єктом. Предикат об'єкта $P(x_1, x_2, ..., x_m) = t$ виражається у вигляді $P(x_1, x_2, ..., x_m) = K(r_1(x_1), r_2(x_2), ..., r_m(x_m))$. Сигнали $y_1 = r_1(x_1), y_2 = r_2(x_2), ..., y_m = r_m(x_m)$ є внутрішніми станами об'єкта, недоступними для спостереження.

Результатом формального опису будь-якого об'єкта мовою алгебри предикатів завжди є деякий предикат $P(x_1, x_2, ..., x_m)$. Він має виражати деяке цілком визначене відношення P, яке представляє собою множину всіх наборів предметів $x_1, x_2, ..., x_m$, що задовольняють рівнянню $P(x_1, x_2, ..., x_m) = 1$. Саме це відношення виражає структуру об'єкта, який описується. Якщо для опису деякого предметного простору S мають одночасно виконуватися декілька відношень, то це приводить до кон'юнкції відповідних предикатів.

Класична задача ідентифікації полягає у тому, що по вхідному x і вихідному y сигналам об'єкта визначити функцію $y = F(x)$ перетворення сигналу цим об'єктом. Таку ідентифікацію звуть прямою, оскільки вона здійснюється при безпосередньому доступі до вихідного сигналу об'єкта. Однак у ряді випадків виникає необхідність у непрямій ідентифікації об'єкта, коли у дослідника немає прямого доступу до вихідного сигналу. Багато задач цього типу можна вирішувати методом компараторної ідентифікації об'єкта [18, 19]. Даний метод дозволяє викладати основні положення теорії інтелекту дедуктивним способом, виходячи виключно з фактів, які можна фізично спостерігати, він добре зарекомендував себе при обробці інформаційних об'єктів різних рівнів [18].

Універсум елементів U – усі можливі сигнали, ознаки, дані з бази даних [18], ключові поняття, дескриптори, які входять до складових інформаційної системи і т. д. Вводяться предикатні змінні $L_1, L_2, ..., L_k$, які зв'язуються логічними рівняннями. Ці рівняння є вихідними постулатами метода компараторної ідентифікації. З них, як з аксіом, дедуктивно виводяться залежності, які характеризують внутрішню структуру елементів універсаму U та предикатів $P_1, P_2, ..., P_k$.

Ментальні моделі пошуку задаються на базі компаратора. Компонент, отримуючи тему пошуку і сторінки для початку пошуку, здійснює тематично направлену екстракцію на основі оцінки перспективності веб-сторінки, ментальна модель реалізує компаратор, який порівнює видобуті зі сторінки слова у певних структурних елементах із словами моделі пошуку.

Нехай E – множина структурних елементів веб-сторінки, W – множина слів. Тоді $R_{SEARCH} \subseteq E \times W$ – бінарне відношення «використовується для пошуку». Нехай $E_q \subseteq E$ – множина елементів веб-сторінки, які обрані для оцінки та $W_q \subseteq W$ – множина слів, які відповідають темі пошуку. Бінарне відношення $R_{SEARCH} = \{(e_{qi}, w_{qj}) \mid e_{qi} \in E_q, w_{qj} \in W_q\}$ задає пари «елемент-слово», для яких слова належать множині слів, що відповідають темі та елементи належать множині елементів, обраних для розгляду.

Нехай $w_{pj} \in W_p$ – множина слів, видобутих із веб-сторінки. Тоді предикат, який оцінює бінарні пари елемент-слово:

$$P_w(e_{qi}, w_{pj}) = \begin{cases} 1, \text{ якщо } (e_{qi}, w_{pj}) \in R_{SEARCH}, \\ 0, \text{ якщо } (e_{qi}, q_{pj}) \notin R_{SEARCH}. \end{cases}$$

Предикат, який визначає наявність контрольних слів в певному елементі:

$$P_e(e_{qi}) = P_w(e_{qi}, w_{p1}) \vee P_w(e_{qi}, w_{p2}) \vee ... \vee P_w(e_{qi}, w_{pn}).$$

Оцінка веб-сторінки об'єднує оцінки за кожним елементом та визначається предикатом:

$$P_q = P(e_{q1}) \vee P(e_{q2}) \vee ... \vee P(e_{qs}).$$

Графічна інтерпретація заданих предикатів представляється у вигляді дводольних графів (рис. 1).

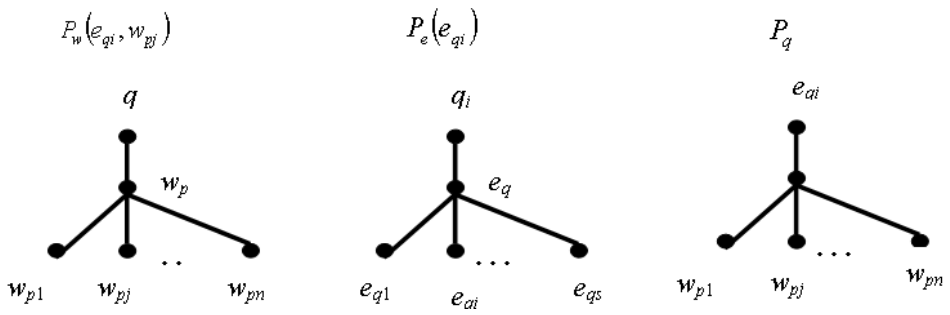

Рис. 1 – Дводольні графи предикатів

Результати. Розглянемо використання описаних методів для видобування даних щодо вакансій, які розміщено роботодавцями у відкритому доступі в мережі Internet. На етапі проектування програмного забезпечення розроблений структурований XML-словник ключових слів на трьох мовах, який включає в себе слова-індикатори необхідних вакансій. Подібне рішення дозволяє ідентифікувати серед усіх вакансій на сайті лише необхідні. Словник є базовим для певного напрямку. Нові слова, які найбільш часто зустрічаються та мають відношення до вибраної галузі, будуть марковані як можливі індикатори та запропоновані користувачеві для додавання в словник. Це дозволяє підтримувати словник ключових слів в актуальному стані.

Модуль «Уніфікація вимог» представляє собою сукупність алгоритмів, методів та підходів для роботи з різноманітними Веб-ресурсами пошуку роботи (наприклад, rabota.ua, work.ua та ін.) на яких представлено перелік вакансій по запиту користувача, категорії професій, даті додання вакансії до ресурсу і т.д. Основною задачею даного модуля є уніфікація вимог до обраної професії, отриманих з деякого Веб-ресурсу пошуку роботи. Дане програмне забезпечення написано з використанням мови програмування Java.

Для прикладу роботи модуля, на початку обрано сайт rabota.ua (рис. 2). Оскільки з швидким ростом кількості технологій для побудови Веб-сайтів, зростає кількість можливостей написання цих сайтів, то виникає проблема добування інформації при диференціації структури Веб-сторінок у різних Веб-ресурсів. Оскільки задача видобутку конкретної інформації з Веб-ресурсів, які мають різну структуру, не входить до даної роботи, тому обрано вище зазначений сайт для проведення експериментів.

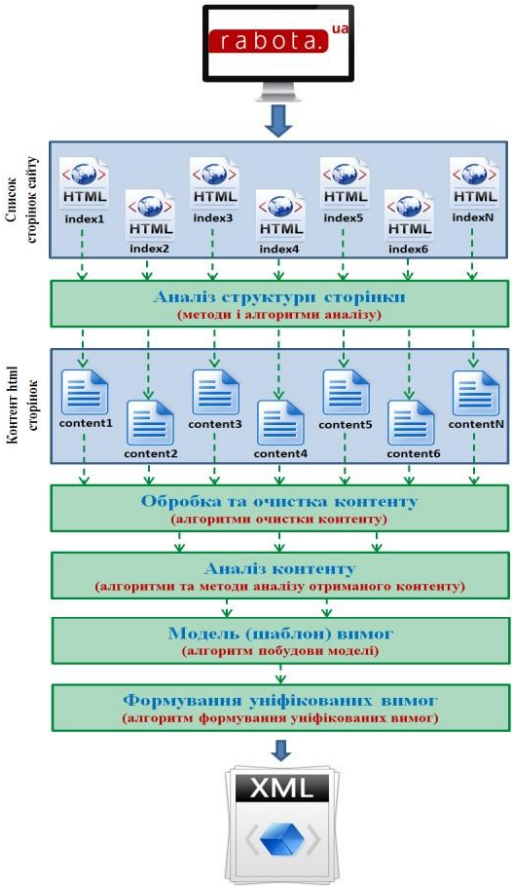

Рис. 2 – Схема застосування запропонованого підходу

Робота модуля починається з добування масиву необхідних сторінок обраного Веб-сайту (rabota.ua, в даному випадку) для подальшого аналізу їх структури. Під «необхідними сторінками» розуміємо те, що треба дістати ті Веб-сторінки, на яких знаходиться корисна для користувача інформація, тобто, в конкретному випадку, – це перелік вакансій різноманітних професій та детальний опис вакансії.

Після виявлення необхідних сторінок, слід перейти до аналізу їх структури. Структура визначається за допомогою експерта. На даний час, сайт rabota.ua має блочну структуру, де перелік вакансій представлений у вигляді таблиці (тег tbody) з одним рядком (tr) і однією коміркою (td) у якій знаходиться тег опису вакансії (article), а також декілька блоків (div) з відповідними атрибутами (class) та інформацією про вакансію (заголовок, опис, місце, дата додання вакансії). З даних елементів цікавить заголовок вакансії, який представлений у вигляді посилання на Веб-сторінку детального опису обраної вакансії.

Сторінка детального опису вакансії представлена у вигляді блочної структури, де знаходяться дані про вимоги, обов'язки, умови роботи та інші відомості про вакансію.

Наступний крок передбачає отримання контенту визначеної html-сторінки, на якій знаходиться необхідна інформація для її подальшого вилучення та обробки. Тобто, на даному кроці отримуємо необхідні дані у вигляді структури html.

Після чого проводиться обробка та очистка раніше отриманого контенту. Цей крок передбачає видалення усіх html-тегів з отриманого, на попередньому кроці, контенту. Для парсингу html-сторінки використовуємо бібліотеку Jsoup. Вона має безліч методів для роботи з html. Надалі, маємо "чисту" інформацію про необхідну вакансію.

Далі треба проаналізувати отриману інформацію з декількох джерел (Веб-сторінок опису вакансії) для отримання даних, з яких треба розробити модель вакансії. Модель має включати такі елементи, як:

- заголовок вакансії;
- дата додання вакансії на сайт;
- місто, де необхідно працювати;
- вимоги до кандидата;
- обов'язки кандидата;
- умови роботи.

Тому потрібно отримати необхідну інформацію. Для добування необхідних даних знову використовуємо бібліотеку Jsoup. Заголовок вакансії, дата додання вакансії на сайт та місто, можна витягнути дуже легко, так як вони містяться в одному блоці і мають зрозумілі теги та атрибути.

Проблеми з'являються коли треба здобути вимоги, обов'язки та умови. Оскільки дана інформація може бути представлено різноманітно. Це значить, по-перше, що кожна з вакансій може бути описана на різній мові (українська, англійська, російська тощо); назва вище перерахованих елементів може мати різний опис, але єдине значення (наприклад, вимоги – необхідно вміти, що треба знати та ін.). Таким чином, для початку взято вакансії лиш на російській мові, оскільки більшість вакансій описано саме на ній та розроблено алгоритм для видобутку необхідної інформації з обробленого та очищеного контенту.

Суть розробленого алгоритму для видобутку переліку вимог, обов'язків та умов полягає в наступному. Задається масив ключових слів – «требования», «обязанности», «условия» (оскільки пошук вакансій проводиться на російській мові, тому ключові слова задаються саме на цій мові). Далі, береться кожне слово та порівнюється з ключовими словами із опису вакансії. Ці ключові слова виділені тегом . Для кожного ключового слова задається маркер статусу (false за замовчуванням). Коли слово із масиву співпадає з ключовим словом з опису вакансії та, що важливо, після цього слова йде список , то маркер цього слова задається як true. Тобто визначено, що було знайдено, наприклад, перелік вимог. Якщо маємо для кожного слова маркертrue, то автоматично підтверджується, що вакансія співпадає "по шаблону" та можна зберегти знайдені відомості до зовнішнього файлу, наприклад, JSON, для подальшого використання.

Модель вакансії формується за допомогою класу-моделі VacancyModel, який використовує дані із раніше створеного JSON-файлу та зберігається до pdf-файлу у вигляді таблиці.

Маючи усі необхідні дані про вакансію (перелік вимог, обов'язків та умов) слід перейти до уніфікації вимог. Обов'язки та умови зберігаємо для виявлення прихованих даних, котрі можуть знадобитись у майбутньому.

Для формування уніфікованих вимог, на початку, експертом, в Microsoft Excel, було створено початкові уніфіковані вимоги для деяких категорій,що часто зустрічаються, а саме: «Высшее образование», «Английский язык», «Опыт работы», «Работа с пакетом Microsoft Office», «Применение методологий», «Работа в команде», «Работа с заказчиком», «Системы управления проектами», «Дополнительные требования». До кожної із категорій належить перелік семантично схожих текстів. Наприклад, для категорії «Высшее образование» відносяться тексти «Высшее законченное техническое образование», «Образование высшее/незаконченное высшее (желательно техническое)» тощо. Для складання цих вимог, використовувався перелік минулих вакансій за місяць й більше. Тому для формування нових вимог має місце автоматизація на основі вже існуючих уніфікованих вимог, які задані експертом.

Для подальшої уніфікації вимог використовується метод латентно-семантичного аналізу (ЛСА). Даний метод заснований на сингулярному розкладанні матриці з пониженням рангу та призначений для вилучення контекстно-залежних значень слів за допомогою статистичної обробки великих масивів текстових даних за наборами їх частотних характеристик.

Після роботи даного методу отримуємо перелік текстів, що належать, до відповідної категорії.

Множина текстів, яка не відноситься до жодної категорії, залишається для сортування експертом. Оновлений перелік категорій та їх семантично-схожих одиниць зберігається до JSON-файлу для подальшої роботи.

Висновки. Отримані результати підтверджують доцільність використання методу компараторної ідентифікації та аналізу DOM дерева для реалізації програмного забезпечення аналізу даних з Інтернет ресурсів для формування актуальних вимог до кандидатів.

Список літератури

1. Guha R. V., Brickley D, Macbeth S. Schema.org: Evolutionofstructured data on the web. ACM. 2008. № 59 (2). P. 44–51.
2. Raghavan S., Garcia-Molina H. Crawling the hidden web. Proceedings of the 27th International Conference on Very Large Data Bases, VLDB ’01. San Francisco, CA, USA, Morgan Kaufmann Publishers Inc, 2001. P. 129–138.
3. Memex (Domain-Specific Search URL : www.darpa.mil/program/memex (дата звертання : 02 листопада 2017).
4. W3CXMLQuery (XQuery) URL : https://www.w3.org/XML/Query (дата звертання : 04 листопада 2017).
5. XSLTransformations (XSLT) Version 3. URL : https://www.w3.org/TR/xslt (дата звертання : 15 листопада 2017).
6. ApacheNutch URL : http://nutch.apache.org (дата звертання : 18 листопада 2017).
7. Declarative information extraction using datalog with embedded extraction predicates / W. Shen, A. Doan, J. F. Naughton, R. Ramakrishnan // Proceedings of the 33rd International Conference on Very Large Data Bases, VLDB ’07. VLDB Endowment, 2007. P. 1033–1044.
8. Scrapy | A Fast and Powerful Scraping and Web Crawling Framework. URL : http://scrapy.org (дата звертання :25 листопада 2017).
9. Nakashole N., Theobald M., Weikum G. Scalable knowledge harvesting with high precision and high recall. Proceedings of the Fourth ACM International Conference on Web Search and Data Mining, WSDM ’11. New York, NY, USA, ACM. 2011. P. 227–236.
10. From data fusion to knowledge fusion / Xin Luna Dong, E. Gabrilovich, G. Heitz [et al.] // Proc. VLDB Endow. 2014. № 7 (10). P 881–892.
11. Web-scale information extraction in knowitall: (preliminary results) / Oren Etzioni, Michael Cafarella, Doug Downey [et al.] // Proceedings of the 13th International Conference on World Wide Web, WWW ’04. New York, NY, USA, ACM. 2004. P. 100–110.
12. Toward an architecture for never-ending language learning / A. Carlson, J. Betteridge, B. Kisiel [et al.] // AAAI. AAAI Press, 2010.
13. Bing Liu, Kevin Chen-Chuan-Chang. Editorial: special issue on web content mining // AcmSigkdd explorations newsletter. 2004. № 6 (2). P. 1–4.
14. AnanthaBarathi B. Structured information extraction system from web pages // MiddleEast Journal of Scientific Research. 2014. № 19 (6). P. 817–820.
15. Arasu A., Garcia-Molina H. Extracting structured data from web pages // Proceedings of the 2003 ACM SIGMOD international conference on Management of data. ACM, 2003. P. 337–348
16. Chia-Hui Chang, Shao-Chen Lui. Iepad: information extraction based on pattern discovery // Proceedings of the 10th international conference on World Wide WebACM. 2001. P. 681–688.
17. Selenium–WebBrowserAutomation URL : http://www.seleniumhq.org (дата звертання :01 грудня2017).
18. Шабанов-Кушнаренко С. Ю. Компараторная идентификация процессов многомерной количественной оценки. Саарбрюккен, Германия: PalmariumAcademicPublishing, 2015. 217 с.
19. Шабанов-Кушнаренко С. Ю., КудхаирАбедТамер. Разработка метода формирования предикатных моделей прототипов структурированных объектов // СОИ: ХУПС, 2015. № 9 (134). С. 83-87.
20. Шабанов-Кушнаренко С. Ю., Коваленко А. И., Булаенко Д. С. Построение онтологии семантического поиска документов // СОИ: ХУПС, 2015. № 10 (135). С. 156-158.

References (transliterated)

1. Guha R. V., Brickley D, Macbeth S. Schema.org: Evolution of structured data on the web.Commun. *ACM*. 2008, no. 59 (2), pp. 44–51.
2. Raghavan S., Garcia-Molina H. Crawling the hidden web. *Proceedings of the 27th International Conference on Very Large Data Bases*. San Francisco,USA, Morgan Kaufmann Publishers Inc., 2001, pp. 129–138.
3. *Memex (Domain-Specific Search)*. Available at :www.darpa.mil/program/memex. (accessed 02.11.2017)
4. *W3C XML Query (XQuery)*. Available at :https://www.w3.org/XML/Query. (accessed 04.11.2017)
5. *XSL Transformations (XSLT) Version 3.0*. Available at :https://www.w3.org/TR/xslt. (accessed 15.11.2017)
6. *Apache Nutch*. Available at : http://nutch.apache.org/ (accessed 18.11.2017).
7. Shen W., Doan A., Naughton J. F., Ramakrishnan R. Declarative information extraction using datalog with embedded extraction predicates. *Proceedings of the 33rd International Conference on Very Large Data Bases*. VLDB Endowment, 2007, pp. 1033–1044.
8. *Scrapy | A Fast and Powerful Scraping and Web Crawling Framework*. Available at :http://scrapy.org. (accessed 25.11.2017).
9. Nakashole N., Theobald M., Weikum G. Scalableknowledgeharvestingwithhighprecisionandhighrecall. *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*. New York, NY, USA, ACM, 2011, pp. 227–236.
10. Xin Luna Dong, Gabrilovich. E, Heitz G. et al. From data fusion to knowledge fusion. *Proc. VLDB Endow*. 2014, no. 7(10), pp. 881–892.
11. Etzioni O., Cafarella M., Downey D. et al. Web-scale information extraction in knowitall: (preliminary results). *Proceedings of the 13th International Conference on World Wide Web*. New York, NY, USA, ACM, 2004, pp. 100–110.
12. Carlson A., Betteridge J., Kisiel B. et al. Toward an architecture for never-ending language learning. *AAAI*. AAAI Press, 2010.
13. Bing Liu, Kevin Chen-Chuan-Chang.Editorial: special issue on web content mining. *AcmSigkdd explorations newsletter*. 2004, no. 6 (2), pp. 1–4.
14. AnanthaBarathi B. Structured information extraction system from web pages.*MiddleEast Journal of Scientific Research*, 2014, no. 19(6), pp. 817–820.
15. Arasu A., Garcia-Molina H. Extracting structured data from web pages. *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*. ACM, 2003, pp. 337–348.
16. Chia-Hui Chang, Shao-Chen Lui. Iepad: information extraction based on pattern discovery. *Proceedings of the 10th international conference on World Wide Web*.ACM, 2001, pp. 681–688.
17. *Selenium – Web Browser Automation*. Available at :http://www.seleniumhq.org. (accessed: 01.12.2017)
18. Shabanov-Kushnarenko S. Yu. *Komparatornaya identifikatsiya protsessov mnogomernoy kolichestvennuy otsenki* [Comparative identification of multidimensional quantitative estimation processes]. Saarbruecken, Germany, PalmariumAcademicPublishing, 2015. 217 p.
19. Shabanov-Kushnarenko S. Yu., Kudkhair AbedTamer. Razrabotka metoda formirovaniya predikatnykh modeley prototipov strukturirovannikh ob’ektov [Development of the method for the formation of predicate models of structured objects prototypes]. *SOI, KhUPS*, 2015, no. 9(134), pp. 83–87.
20. Shabanov-Kushnarenko S. Yu., Kovalenko A. S., Bulaenko D. S. Postroenie ontologii semanticheskogo poiska documentov [Building an ontology of semantic document search]. *SOI, KhUPS*, no. 10 (135), pp. 156–158.

Надійшла (received) 15. 12.2017

Відомості про авторів /Сведения об авторах /About the Authors

Чередніченко Ольга Юріївна **(*Чередниченко Ольга Юрьевна, Cherednichenko Olga Yuriivna*)** – кандидат технічних наук, доцент, Національний технічний університет «Харківський політехнічний інститут», доцент кафедри програмної інженерії та інформаційних технологій управління; тел.: (067) 754-79-44; e-mail:olha.cherednichenko@gmail.com, ORCID: 0000-0002-9391-5220.

Гринченко Марина Анатоліївна **(*Гринченко Марина Анатольевна, Grinchenko Marina Anatolievna*)** – кандидат технічних наук, доцент, Національний технічний університет «Харківський політехнічний інститут», доцент кафедри стратегічного управління; тел.: (050) 970-82-95; e-mail: marinagrunchenko@gmail.com, ORCID: 0000-0002-8383-2675.

Василенко Артем Вікторович **(*Василенко Артем Викторович, Vasylenko Artem Vyktorovych*)** – Національний технічний університет «Харківський політехнічний інститут», аспірант кафедри програмної інженерії та інформаційних технологій управління; тел.: (099) 342-09-54; e-mail:artyom4ek@yandex.ua, ORCID: 0000-0003-3121-4856.

Матвєєв Олександр Миколайович **(*Матвеев Александр Николаевич, Matvieiev Oleksandr Mykolayovych*)** – Національний технічний університет «Харківський політехнічний інститут», аспірант кафедри програмної інженерії та інформаційних технологій управління; тел.: (096) 352-59-65; e-mail: matwei1970@gmail.com, ORCID: 0000-0001-5907-3771.