

Ekstrakcija podatkov

Enes Fejzoski, Marko Kofol, Elian Mugerli
10. Maj 2024

To poročilo opisuje implementacijo treh različnih pristopov za ekstrakcijo strukturiranih podatkov iz spleta: regularne izraze (regex), XPath in algoritem za samodejno ekstrakcijo vsebine (RoadRunner). Poudarek je na pridobivanju podatkov iz štirih primerjalnih spletnih mest: rtslo.si, bolha.com, overstock.com in sporttv.si.

1. Uvod

Ekstrakcija strukturiranih podatkov iz spleta je ključna naloga za različne aplikacije, vključno z web scrapingom, pridobivanjem informacij in združevanjem podatkov. To poročilo raziskuje tri pogoste pristope za doseganje tega: regularne izraze, XPath in algoritme za samodejno ekstrakcijo vsebine.

Regularni izrazi (regex) so zmogljiva orodja za iskanje in manipulacijo besedila na podlagi vzorcev. So jedrnat in učinkoviti pri pridobivanju določenih formatov podatkov iz HTML kode. XPath je poizvedovalni jezik, posebej zasnovan za navigacijo in ekstrakcijo podatkov iz XML dokumentov, ki se lahko uporabijo za predstavitev HTML strukture. Algoritmi za samodejno ekstrakcijo vsebine, kot je RoadRunner, se osredotočajo na učenje pravil ekstrakcije iz nabora spletnih strani, s čimer avtomatizirajo postopek identifikacije in pridobivanja relevantnih podatkovnih elementov.

2. Izbira dodatnih spletnih strani

Poleg že dodeljenih spletnih domen smo za širitev analize izbrali tudi strani sporttv.si in bolha.com.

The image displays two screenshots side-by-side. The left screenshot shows a sports news article from SportTV.si titled 'Dončić in Dallas izgubili prvo tekmo končnice v Kaliforniji'. The article discusses the Los Angeles Lakers' performance in the NBA playoffs. The right screenshot shows a car listing page from Bolha.com, featuring several Mercedes-Benz models for sale, including a Mercedes-Benz GLC Coupe and a Mercedes-Benz GLK 220 CDI 4-MATIC.

3. Implementacija

3.1. Regularni izrazi

Regularni izrazi (angl. "regular expressions" ali "regex") so izrazi, ki se uporabljajo za iskanje in ujemanje vzorcev v besedilu. Regularni izrazi so implementirani v Pythonu z uporabo knjižnice `re`. Tukaj je primer za pridobivanje podatkov iz novic na `rtvslo.si`:

```
import re

def rtv(html):
    title_regex = re.compile(r'<h1>(.*?)</h1>')

    subtitle_regex = re.compile(r'<div class="subtitle">(.*?)</div>')

    lead_regex = re.compile(r'<p class="lead">(.*?)</p>')

    content_regex = re.compile(
        r'<div *?class="article-body">(.*?)<div class="gallery">',
        re.DOTALL)

    author_regex = re.compile(r'<div class="author-name">(.*?)</div>')

    published_time_regex = re.compile(r'<div
class="publish-meta">\n\t\t(.*?)<br>')

    data = {
        'Naslov': re.findall(title_regex, html)[0],
        'Podnaslov': re.findall(subtitle_regex, html)[0],
        'Uvod': re.findall(lead_regex, html)[0],
        'Vsebina': re.findall(content_regex, html)[0],
        'Avtor': re.findall(author_regex, html)[0],
        'Cas Objavve': re.findall(published_time_regex, html)[0]
    }

    return data
```

Razlaga kode:

1. `import re`: Uvozi knjižnico `re` za delo z regularnimi izrazi.
2. `def rtv(html)`: Definira funkcijo `rtv`, ki sprejme HTML vsebino kot vhod.

3. `title_regex = re.compile(r'<h1>(.*?)</h1>')`: Ustvari objekt regularnega izraza `title_regex` za ujemanje naslova znotraj oznake `<h1>`. `(.*?)` zajame vse znake znotraj začetne in končne oznake `<h1>`.
4. Podobne vrstice sledijo za `subtitle_regex`, `lead_regex`, `author_regex` in `published_time_regex`, vsaka cilja na določene HTML elemente in zajame njihovo vsebino znotraj oklepajev.
5. `re.DOTALL`: Ta zastavica omogoča regexu, da se ujema z znaki nove vrstice (`\n`), kar je potrebno za zajetje vsebine, ki se lahko razteza na več vrstic.

3.2. XPath

XPath je drug pristop za pridobivanje podatkov iz HTML dokumentov. Uporablja potenco osnovano sintakso za lociranje določenih elementov in atributov znotraj drevesne strukture HTML. Tukaj je primer za pridobivanje podatkov iz novic na rtvslo.si:

```
from lxml import html

def rtv(html_content):

    tree = html.fromstring(html_content)

    title_xpath = "//h1/text()"
    subtitle_xpath = "//div[@class='subtitle']/text()"
    lead_xpath = "//p[@class='lead']/text()"
    content_xpath = "//div[@class='article-body']//text()"
    author_xpath = "//div[@class='author-name']/text()"
    published_time_xpath = "//div[@class='publish-meta']/text()"

    title = tree.xpath(title_xpath)[0]
    subtitle = tree.xpath(subtitle_xpath)[0]
    lead = tree.xpath(lead_xpath)[0]
    content = ' '.join(tree.xpath(content_xpath))
    author = tree.xpath(author_xpath)[0]
    published_time = tree.xpath(published_time_xpath)[0]

    data = {
        'Naslov': title,
        'Podnaslov': subtitle,
        'Uvod': lead,
        'Vsebina': content,
        'Avtor': author,
```

```
        'Cas Objavve': published_time
    }

    return data
```

Razlaga XPath nekaterih izrazov:

1. `//h1/text()`: Ta izraz izbere vse elemente `<h1>` kjerkoli v dokumentu (označeno z `//`) in izvleče njihovo besedilno vsebino.
2. `//div[@class='subtitle']/text()`: Izbere vse elemente `div` s pripisanim razredom "subtitle" in izvleče njegovo besedilno vsebino.
3. Podobni izrazi sledijo za `lead`, `content`, `author` in `published_time_xpath`, ki ciljajo na določene elemente glede na njihova imena oznak ali razredov in izvlečejo njihovo besedilno vsebino.
4. `content = ' '.join(tree.xpath(content_xpath))`: Za vsebino uporabimo `tree.xpath` za pridobitev seznama tekstovnih elementov znotraj elementa "article-body". Nato jih združimo z presledkom (" "), da ustvarimo eno samo besedo, ki predstavlja celotno vsebino.

XPath ponuja bolj strukturiran in deklarativen način za navigacijo po drevesu HTML v primerjavi z regularnimi izrazi. Bolje se spopade s kompleksnimi strukturami HTML, vendar zahteva poznavanje XPath sintakse in specifične hierarhije elementov HTML.

3.3. RoadRunner

Tretji pristop za pridobivanje podatkov iz HTML dokumentov je implementacija algoritma RoadRunner. Tega smo razvili s pomočjo grajenja DOM dreves in pripadajočih elementov.

Implementacijo sestavljajo naslednje funkcije:

1. **build_dom_tree**: Ta rekurzivno sestavi DOM drevo iz razčlenjene HTML strani s BeautifulSoup. Hkrati pa preskoči oznake, ki niso prisotne na seznamu `allowed_tags`.
2. **Build_generalized_tree**: Funkcija primerja dve HTML drevesi, predstavljeni z objekti `Node`. Ugotavljanje skladja oznak, neskladja besedil in ponavljajoča se vozlišča. Glede na te primerjave nastavi vozlišča kot izbirna ali ponovljiva.
3. Pomožne funkcije: Skripta vključuje več pomožnih funkcij, kot je **check_repeated_nodes** za preverjanje, ali so vozlišča ponovljena, **check_string_mismatch** za preverjanje neskladij besedil in **print_dom_tree** za izpis generaliziranega DOM drevesa z oznakami za izbirnost in ponovitev.
4. **Print_dom_tree**: Funkcija za izpis DOM drevesa rekurzivno izpiše generalizirano DOM drevo z oznakami za izbirnost in ponovitev, zamaknjeno glede na globino vozlišča.

Spodaj so podani izseki implementacije.

```
class Node:
    def __init__(self, tag, text, classes, children, optional):
        self.tag = tag
        self.text = text
        self.classes = classes
        self.children = children
        self.optional = optional
        self.repeatable = False
```

```
def road_runner(pages):
    trees = []
    for page in pages:
        soup = BeautifulSoup(page, 'lxml')
        tree = build_dom_tree(soup.find("html"))

        trees.append(tree)

    wrapper = build_generalized_tree(trees[0], trees[1])
    print_dom_tree(None, wrapper)
```

```
def build_dom_tree(tag):
    node = Node(tag.name, tag.string if tag.string else None,
tag.attrs.get('class', []), [], False)

    if allowed_tags is None or tag.name in allowed_tags:
        for child in tag.children:
            if isinstance(child, str):
                continue

            if child.name in allowed_tags:
                node.children.append(build_dom_tree(child))

    return node
```

```
def build_generalized_tree(tree1, tree2):
    if tree1.tag != tree2.tag:
        tree1.optional = True
        return tree1

    if check_string_mismatch(tree1, tree2):
        tree1.text = "#PCDATA"

    else:
        if tree1.children:
```

```

        for index in range(min(len(tree1.children),
len(tree2.children))):
            tree1.children[index] =
build_generalized_tree(tree1.children[index], tree2.children[index])

        if len(tree1.children) > len(tree2.children):
            for index in range(len(tree2.children),
len(tree1.children)):
                tree1.children[index].optional = True
        else:
            for index in range(len(tree1.children),
len(tree2.children)):
                tree2.children[index].optional = True
                tree1.children.append(tree2.children[index])
    elif tree2.children:
        # Set children as optional
        tree1.children = [child for child in tree2.children]
        for child in tree1.children:
            child.optional = True

    else:
        return tree1

```

```

def check_repeated_nodes(node1, node2):
    if node1 is None or node2 is None:
        return False

    if node1.tag != node2.tag:
        return False

    if not node1.children and not node2.children:
        return True

    if node1.children and not node2.children or not node1.children and
node2.children:
        return False

    if node1.classes != node2.classes:
        return False

    for index in range(len(node1.children)):
        repeatable = False if index >= len(node2.children) else
check_repeated_nodes(
            node1.children[index], node2.children[index])
        if not repeatable:

```

```
        return False

    return True

def check_string_mismatch(node1, node2):
    if node1.text and node2.text:
        return node1.text != node2.text
    return False
```

Spodaj bodo podani se primeri tesnih izhodov:

1. RTV

[illegible]

y Bands</td></tr><tr><td> </td><td>View

All</td></tr></tbody></table></td></tr></tbody></table><table><tbody><tr>Apparel, Shoes &

Access.<td>Apparel, Shoes & Access.</td></tr><tr>Books

, Movies, CDs, Games<td>Books, Movies, CDs, Games</td></tr><tr>Electronics &

Computers<td>Electronics & Computers</td></tr><tr>Home & Garden<td>Home &

Garden</td></tr><tr>Jewelry, Watch

es & Gifts<td>Jewelry, Watches & Gifts</td></tr><tr>Sports, Travel & Toys<td>Sports, Travel

&

Toys</td></tr><tr>Worldstock<td>Worldstock</td></tr></tbody></table><table><tbody><tr><td>

></td></tr><tr><td></td></tr><tr><td></td></tr><tr><td></td></tr></tbody></table>

able><table><tbody><tr>Shopping Cart & Checkout<td>Shopping Cart &

Checkout</td></tr><tr>T

rack Your Order<td>Track Your Order</td></tr><tr>Your Account<td>Your

Account</td></tr><tr>Help & FAQ<td>Help & FAQ</td></tr><tr>Best Price Guarantee<td>Best

Price Guarantee</td></tr><

/tbody></table><table><tbody><tr>About Us<td>About Us</td></tr><tr>Privacy &

Security<td>Privacy & Security</td></tr><tr>Terms & Conditions<td>Terms &

Conditions</td></tr><tr>Become An

Affiliate<td>Become An Affiliate</td></tr><tr>Business Purchases<td>Business

Purchases</td></tr><tr>Have Products to Sell?<td>Have Products to

Sell?</td></tr><tr>Investor Relations<td>

>Investor

Relations</td></tr></tbody></table></td><td></td><td></td><td><table><tbody><tr><td><table>

><tbody><tr><td><table><tbody><tr><td></td><td></td></tr><tr><td></td><td></td></tr>

</tbody></table></td></tr></tbody></table></td></tr><tr><td><table><tbody><tr><td><table><

tbody><tr><td><table><tbody><tr><td></td></tr><tr>More Info...<td>More

Info...</td></tr></tbody>

y></table></td><td><table><tbody><tr><td><table><tbody><tr><td>List

Price:</td><td>#PCDATA</td></tr><tr><td>Price:</td><td>#PCDATA</td></tr><tr><td>You

Save:</td><td>#PCDATA</td></tr><

/tbody></table></td><td><table><tbody><tr><td></td></tr></tbody></table></td></tr></tbody>

</table></td></tr><tr><td></td></tr><tr><td><table><tbody><tr><td></td></tr><tr>More

Info...<t

d>More

Info...</td></tr></tbody></table></td><td><table><tbody><tr><td><table><tbody><tr><td>List

Price:</td><td>#PCDATA</td></tr><tr><td>Price:</td><td>#PCDATA</td></tr><tr><td>You Sa

ve:</td><td>#PCDATA</td></tr></tbody></table></td><td></td></tr></tbody></table></td></tr>

<tr><td></td></tr><tr><td><table><tbody><tr><td></td></tr><tr>More Info...<td>More Info...</td>

></tr></tbody></table></td><td><table><tbody><tr><td><table><tbody><tr><td>List

Price:</td><td>#PCDATA</td></tr><tr><td>Price:</td><td>#PCDATA</td></tr><tr><td>You

Save:</td><td>#PCDAT

A</td></tr></tbody></table></td><td></td></tr></tbody></table></td></tr><tr><td></td></tr><tr>

><td><table><tbody><tr><td></td></tr><tr>More Info...<td>More Info...</td></tr></tbody></table>

[illegible]

[illegible]

S pridobljenimi rezultati sicer nismo bili najbolj zadovoljni in ima tako podana implementacija veliko prostora za nadgradnjo.