



Tekoälyn hyödyntäminen datankeruussa

5.12.2024

Johdanto

PDF-tiedostot ovat yksi yleisimmistä dokumenttiedostoista, joita käytetään erilaisiin tarkoituksiin, kuten laskuihin, sopimuksiin, e-kirjoihin ja muihin dokumentteihin. PDF-tiedostoissa halutaan myös kerätä dataa, jota voitaisiin käsitellä ja hyödyntää. PDF-tiedostojen käsittelyä vaikeuttaa niiden epälooginen järjestys. Pythonille on kehitetty erilaisia PDF-tiedostojen käsittelykirjastoja, joista valitaan ominaisuuksiltaan hyödyllisin tähän VauhtiData-hankkeen pilottiin. Kolme esimerkkivaihtoehtoa ovat PyPDF2-, pdfminer.six- ja pdfplumber -kirjastot, jotka pystyvät tuomaan dataa ulos PDF-tiedostoista. Tässä hankkeen pilotissa on testattu pdfplumber, joka on kehitetty pdfminer.six -lähdekoodista.

Tekoäly apuna ongelmanratkaisussa

Kun käytetään tekoälypalveluita, kuten OpenAI:n luomaa ChatGPT:ä, on hyvä ottaa huomioon tietoturvallisuus ja välttää lähettämästä sille mitään arkaluontoista, kuten henkilökohtaisia tietoja, salasanoja muita tärkeitä tietoja, jotka ei kuulu kenellekään ulkopuoliselle henkilölle tai edes robotille. ChatGPT kerää käyttäjiltään kaiken tiedon, mitä sille syötetään (Veale, 2023).

Aivan ensimmäiseksi tässä hankkeen pilotissa tekoälyä on pyydetty luomaan ohjelmakoodi, joka prosessoi PDF-tiedostoja tuoden niistä tekstiä komentoriville. Muokkaamalla koodia niin, että siihen lisätään PDF-

tiedostojen polun sijainti, jolla pystytään testaamaan tekoälyn antamaa koodin toimivuutta. Tiedoston lukeminen onnistuu ja käsittelemätön data on esitetty kuvassa 1., joka on Reportronic-ohjelmasta ladatun PDF-tiedoston ulostulo. Kuvasta 1. voidaan huomata, että teksti on epäselvää eikä sitä pystytä käsittelemään eikä siten hyödyntämään.

```
{'0': 'SakuKaarlejärvi', '1': '01Palkkakustannukset', '2': '2  
9.09.23', '3': '5,25', '4': 'RaspberryPi:nkoodintarkistusGith  
ubissajamyöstietokannanasetuksetkatsottuläpi.Pilottiyrityksen  
serveripäivitysontulossajatietokantamuuttopitääpohjustaajahel  
pottaa.PowerBI:nvisuaalisiamuokkauksiätehty.', '5': "Person's  
hourstotal:Specifier'hourstotal:", '6': '100,75100,75', '7':  
'Allhourstotal:', '8': '152,25', '9': '3', '10': '3/', '11':
```

Kuva 1. Kuvankaappaus PDF-tiedoston muunnoksesta tekstiksi.

Johtopäätöksenä on se, että testattua koodia pitää muokata, jotta saadaan laadukkaampaa dataa PDF-tiedostoista. Jotta koodin toimivuutta saataisiin validoitua, niin seuraavaksi on luettava läpi pilotissa käsiteltävien PDF-tiedostojen tekstit ja manuaalisesti kirjoittaen ylös halutut etsittävät tiedot ja niiden toistuvat otsikot. Tekoälyä pyydetään nyt apua ongelmanratkaisussa näiden otsikoiden ja vaaditun datan suhteen.

SA

You

PDF file reading with python:

Regular Expression in PDF occurs. PDF file must be read and there are

"Job list" that starts every PDF page. And it's closest value is 14077.

First page of PDF contains e.g. :

"Job 14077"

"Description 15mm Steel"

"Info1 "

"Info2 "

"Info3 "

"Machine LaserMachineName"

"Material LaserMaterial"

"Thickness 15,00 mm"

"Number of different parts 9"

"Used technologies Default, Cutting, Pulsed"

"Waste 22,05%" "Waste target "

"Cutting time 00:29:13" "Waste cost "

"Non-prod. times 00:01:44" "Total waste weight 547,343 kg"

"Total time 00:30:57"

Info1 - 3 ... can be None value.

These data can be found from PDF file. Can you create regexp Python code RPA read and write it to CSV or JSON to process the data?

Kuva 2. Kysymys ongelmaan ChatGPT:ltä.

Kuvassa 2. tekoälylle on lähetetty viesti sisältäen etsittävät otsikot ja niiden mahdolliset tiedot, jotka halutaan kerätä ja mitä ongelmia halutaan ratkaista Python-ohjelmointikielellä. Tässä esimerkissä on käytetty englannin kieltä promptien kirjoittamiseen omasta tottumuksesta, mutta tällaisessa tilanteessa ChatGPT-työkalu todennäköisesti pystyy hyvin tulkkamaan kysymykset ja vastaukset, jos käytetään suomen kieltä. Erityisesti on huomioitava, että tekoälyltä saadut viestit ja vastaukset eivät ole aina oikein sekä niiden toimivuutta ja luotettavuutta on suositeltava aina tarkistaa.



ChatGPT

To extract the specified data from a PDF file and save it in a CSV or JSON format using Python, you'll need to follow these steps:

1. **Read PDF File:** Use a library like `PyPDF2` or `pdfplumber` to read the text content of the PDF file.
2. **Regular Expression (RegEx) Matching:** Define a regular expression pattern that matches the structure of the information you need to extract.
3. **Extract Data:** Use the regular expression to find and extract the relevant data from the text.
4. **Write to CSV or JSON:** Format the extracted data and write it to a CSV or JSON file using Python's built-in libraries.

Here's an example Python script that demonstrates these steps. Note that this script assumes the structure of the text in the PDF is consistent with the example you provided:

```
python                                                                    Copy code

import re
import json
import csv
import pdfplumber # You may need to install this library

# Define the regular expression pattern
pattern = r'''
    Job\s(?:P<job>\d+)\s*
    Description\s(?:P<description>.*?)\s*
```

Kuva 3. Ongelman ratkaisua.

Kuvassa 3. esitetään ChatGPT:n kirjoittama vastaus kuvassa 2. esitettyyn kysymykseen, jossa aluksi kerrotaan käyttäjälle aluksi suunnitelma datankeruuseen. Siinä esitellään, kuinka luodaan aloituskoodi, jolla pystytään avaamaan PDF-tiedosto ja tuomaan pyydettyjä toistuvia kaavoja, kuten "Job" tai "Description", komentoriville, sekä viemällä saadut tiedot CSV- ja JSON-tiedostoihin.

Tässä esimerkkikoodissa käytetään tekoälyn suosittamaa pdfplumber-kirjastoa, joka perustuu pdfminer.six -lähdekoodiin (jsvine, i.a.). Edelleen testaamalla tekoälyltä saatua koodia käyttäen pilotin PDF-tiedostoja, koodi ei kuitenkaan löydä esitettyjä toistuvia kaavioita, joten koodia tulee välttämättä muokata ja testata. Debuggauksella pyritään selvittämään, että mitä pdfplumber-kirjaston avulla on kerätty PDF-tiedostoista ja mikä olisi mahdollinen ratkaisu ongelmaan. Koodissa on "with pdfplumber.open(pdffile) as pdf", joka avaa PDF -tiedostot ja tämän avulla pystyttiin löytämään ratkaisu ongelmaan.

page.pdf_pages[0].chars

	matrix	fontn...	adv	upright	x0	y0	x1	y1	width	height	size	mcid	tag	object...	page...	ncs	text
0	0.75,0,0	CIDFont...	10.58662...	true	287.0400...	794.6268...	294.9799...	808.9073...	7.939972...	14.28052...	14.28052...	NaN	NaN	char	1	DeviceRGB	J
1	0.75,0,0,0...	CIDFont...	11.61482...	true	294.8942...	794.6268...	303.6054...	808.9073...	8.711120...	14.28052...	14.28052...	NaN	NaN	char	1	DeviceRGB	o
2	0.75,0,0,0...	CIDFont...	11.61482...	true	303.7482...	794.6268...	312.4593...	808.9073...	8.711120...	14.28052...	14.28052...	NaN	NaN	char	1	DeviceRGB	b
3	0.75,0,0,0...	CIDFont...	5.274274...	true	312.4713...	794.6268...	316.4270...	808.9073...	3.955705...	14.28052...	14.28052...	NaN	NaN	char	1	DeviceRGB	
4	0.75,0,0,0...	CIDFont...	5.274274...	true	316.4389...	794.6268...	320.3946...	808.9073...	3.955705...	14.28052...	14.28052...	NaN	NaN	char	1	DeviceRGB	l
5	0.75,0,0,0...	CIDFont...	5.274274...	true	320.4065...	794.6268...	324.3622...	808.9073...	3.955705...	14.28052...	14.28052...	NaN	NaN	char	1	DeviceRGB	i
6	0.75,0,0,0...	CIDFont...	10.58662...	true	324.3740...	794.6268...	332.3140...	808.9073...	7.939972...	14.28052...	14.28052...	NaN	NaN	char	1	DeviceRGB	s
7	0.75,0,0,0...	CIDFont...	6.340553...	true	332.2283...	794.6268...	336.9837...	808.9073...	4.755415...	14.28052...	14.28052...	NaN	NaN	char	1	DeviceRGB	t
8	0.75,0,0,0...	CIDFont...	10.58662...	true	292.2000...	774.3468...	300.1399...	788.6273...	7.939972...	14.28052...	14.28052...	NaN	NaN	char	1	DeviceRGB	1
9	0.75,0,0,0...	CIDFont...	10.58662...	true	300.0542...	774.3468...	307.9942...	788.6273...	7.939972...	14.28052...	14.28052...	NaN	NaN	char	1	DeviceRGB	4
10	0.75,0,0,0...	CIDFont...	10.58662...	true	307.9964...	774.3468...	315.9364...	788.6273...	7.939972...	14.28052...	14.28052...	NaN	NaN	char	1	DeviceRGB	0
11	0.75,0,0,0...	CIDFont...	10.58662...	true	315.9385...	774.3468...	323.8785...	788.6273...	7.939972...	14.28052...	14.28052...	NaN	NaN	char	1	DeviceRGB	7
12	0.75,0,0,0...	CIDFont...	10.58662...	true	323.8807...	774.3468...	331.8207...	788.6273...	7.939972...	14.28052...	14.28052...	NaN	NaN	char	1	DeviceRGB	7
13	0.75,0,0,0...	CIDFont...	6.672278...	true	65.39999...	725.5409...	70.40420...	734.5413...	5.0042085	9.000375	9.000375	NaN	NaN	char	1	DeviceRGB	J
14	0.75,0,0,0...	CIDFont...	7.320305...	true	70.40557...	725.5409...	75.89580...	734.5413...	5.490228...	9.000375	9.000375	NaN	NaN	char	1	DeviceRGB	o
15	0.75,0,0,0...	CIDFont...	7.320305...	true	75.98580...	725.5409...	81.47603...	734.5413...	5.490228...	9.000375	9.000375	NaN	NaN	char	1	DeviceRGB	b

PROBLEMS 14 OUTPUT ROBO TASKS OUTPUT ROBOT DOCUMENTATION ROBOT OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
'e:\code\pdfReadingPy'; & 'C:\Program Files\Python311\python.exe' 'c:\Users\SaKa\.vscode\extensions\ms-python.python-2023.20.0\pythonFiles\lib\python\debugpy\launcher' '55599' '...' 'E:\code\pdfReadingPy\py_pdf_read_test.py'
```

powershell
Python Debug Console
Python Debug Console

Kuva 4. Koodin debuggaamista.

Kuvassa 4. esitetyssä debuggauksessa “pdf” muuttujaa, löytyy kohta “chars”, joka sisältää tiedot “matrix” ja “fontname”, jotka aluksi eivät kerro mitään. Kuitenkin avaamalla Visual Studio Codessa tämän muuttujan “View Value in Data Viewer”, pystytään tarkemmin selaamaan koko datakokonaisuutta ja sen sisältöä. Kuvassa 4. nähdään, että jokainen sana on jaettu kirjaimiin “text” -sarakkeessa ja niille on määritetty x ja y –akselisijainnit. Tämän avulla pystyttäisiin paikantamaan jokaisen halutun tekstin sijainnit ja varmistamaan tarkasti, mistä dataa on kerätty. Antamalla nämä löydetyt tiedot ChatGPT:lle, niin pystyttiin jatkamaan ongelman ratkaisemista.

SA

You

Opening the PDF file reveals data from list "page.pdf._pages[0].chars"

Viewing data with debugging I can find lots of different data points. It list all the pages and "text" data which contains one letter / str -type data.

e.g.

```
000: {'matrix': (0.75, 0.0, 0.0, 0.75, 287.04000075, 797.64004325), 'fontname':  
'CIDFont+F1', 'adv': 10.586629756, 'upright': True, 'x0': 287.04000075, 'y0':  
794.6268523167499, 'x1': 294.979973067, 'y1': 808.90737806675, 'width':  
7.939972317000013, 'height': 14.280525750000038, 'size': 14.280525750000038,  
'mcid': None, 'tag': None, 'object_type': 'char', ...} etc...
```

and the column names from data point "page.pdf._pages[0].chars" :

(id number that is hidden. It starts from 0)

matrix

fontname

adv

upright

x0

y0

x1

y1



Kuva 5. Debuggaamista tekoälyn avulla.

Kuvassa 5. tekoälylle on annettu käsiteltäväksi kuvassa 4. esiintyneitä havaintoja koordinaateista. Tämän tiedon avulla tekoäly rakensi y-akseliin raja-arvoja, jotka määrittävät kerättävän datan sijainnin. Tekoäly on olettanut, että dataa löydetään ainoastaan yhdeltä sivulta kappale kerrallaan, ottamatta huomioon samalla rivillä olevia tekstejä tai dataa. Pelkästään tämän viestin avulla, tekoäly ei kuitenkaan pystynyt kirjoittamaan toimivaa koodia, jolloin taas tulee testata koodia ja kysyä tarkentavia kysymyksiä tekoälyltä.

Here is an improved version of the `extract_text_from_chars` function:

python

Copy code

```
def extract_text_from_chars(chars):
    if not chars:
        return ''

    words = []
    current_word = [chars[0]]
    last_char = chars[0]

    # Use both x and y coordinates to group chars into words and lines
    for char in chars[1:]:
        x_gap = char['x0'] - last_char['x1']
        y_gap = abs(char['top'] - last_char['top'])
        same_line = y_gap < 5 # 5 is a threshold for y-coordinate change

        if same_line and x_gap <= 3: # 3 is a threshold for x-coordinate change
            current_word.append(char)
        else:
            # Combine characters into a word and reset current_word
            words.append(''.join([c['text'] for c in current_word]))
            current_word = [char]

        last_char = char

    # Add the last word
    words.append(''.join([c['text'] for c in current_word]))

    return ' '.join(words)
```



Kuva 6. Raja-arvot X ja Y -akseleille.

Useiden kommentojen ja korjausten jälkeen, tekoälyn tukena kirjoitettu koodi löytää kirjainten ja tekstien sijainnit loogisemmassa järjestyksessä, kuin lähtötilanteessa. Kuvassa 6. koodiin on määritetty raja-arvoja tekstien sijainnille. Ongelmaksi koitui PDF-tiedostojen sivut, jotka sisältävät useita sarakkeita. Ilman rajoja, koodi keräsi samalta riviltä kaikki tiedot, tietämättään sarakkeiden olemassaolosta ja näin luoden hämmentävää dataa. Lisäämällä raja-arvot, huomataan edelleen datan ulostulossa visuaalisia virheitä, jotka tuovat ongelmaa datan käsittelyyn.



ChatGPT

The `re.search` patterns you're using assume that there is a space character `' '` after each piece of data you're trying to capture. However, the actual text might not have these spaces, especially at the end of a line or before a new line starts, which seems to be the case with the `'Thickness'` and `'Machine'` data.

For `'Machine'`, since you're expecting to capture multiple words including numbers, you could adjust your pattern to continue until it hits a known following key or the end of the string. The same goes for `'Thickness'`, where you might want to capture everything until you hit the `'mm'` unit.

Let's refine the regular expressions:

1. For `'Machine'`, we want to capture everything up to but not including the word `'Description'` which seems to follow `'Machine'` data based on the pattern in your document.
2. For `'Thickness'`, we want to capture the number followed by `'mm'`, considering that there may be no space between the number and `'mm'`.

Here are the adjusted patterns:

python

Copy code

```
patterns = {
    # ... other patterns ...
    'machine': r'Machine (.*)?(?=Description)', # Lookahead for 'Description'
    'thickness': r'Thickness (.*)?(?=mm)',      # Lookahead for 'mm'
    # ... other patterns ...
}
```



Kuva 7. Toistuvien kaavojen raja-arvot.

Kuvassa 7. on ChatGPT:n antama vastaus virheellisen datan korjauksesta. Tekoälylle kerrottiin, että se leikkasi “Machine” ja “Thickness” datan useaan osaan. Esimerkiksi “Thickness” oli jaettu välilyönnein ja dataksi tuli “15,00” ja lisäsi ylimääräisen sarakkeen “mm”. Tekoälyn vastaus kertoi, kuinka rajoittaa löydettävä dataa. “Machine” kerää kaiken seuraavaan “Description” otsikkoon asti ja “Thickness” kerää vain numerot ilman “mm” merkintöjä.

Johtopäätös

ChatGPT:n kaltaiselle tekoälylle voidaan suoraan viedä eri tiedostoja, kuten kuvia, videoita ja dokumentteja. Mutta ChatGPT:tä käytettäessä on otettava huomioon arkaluontoisten tietojen käsittely ja niistä syntyvät tietoturvariskit. Toki tänne olisi voinut viedä kaikki PDF-tiedostot ja kysyä, miten tästä kerätään nämä halutut tiedot ja samalla ruokkia mahdollisesti tekoälyn tietokantaa datalla, joka ei kuulu sille. Olisi myös mahdollista käyttää tekoäly palvelua käsittelemään PDF-tiedostoja API-kutsujen avulla, mutta se olisi tässä tapauksessa

resurssien ja myös energian tuhlausta. Kaikki ongelmat eivät vaadi tekoälyä ratkaisemaan ne puolestamme, mutta tekoäly voi auttaa ongelmanratkaisussa.

Tekoälymallit ovat hyödyllisiä koodin kirjoittamisessa ja ongelmanratkomisen avustuksessa. ChatGPT pystyy luomaan alkeellista ohjelmakoodia, koska monet suuret ohjelmat ovat useiden rivien ja tekstien pituisia ja tokenien määrä ei vielä riitä tuottamaan yhdellä viestillä ohjelmakokonaisuutta. ChatGPT:n kaltaiset verkossa toimivat tekoälypalvelut kehittyvät jatkuvasti nopeudeltaan ja myös tuoden enemmän tokeneita sekä uudempaa kerättyä dataa käyttäjille.

Tässä pilotissa on havaittu, että monet koodinpatkät, joita on saatu ChatGPT:ltä, ovat olleet virheellisiä tai eivät toimi lainkaan, vaan vaativat välillä suuria tai pieniä korjauksia. Myös useasti ChatGPT:tä joutuu huomauttamaan, että unohdit koodissa antamasi muuttujat tai arvot. Joskus on hyvä ChatGPT:n kanssa luoda täysin uusi viestiketju ja sinne viedä tuolloin toimiva koodi edellisestä viestiketjusta, että se unohtaisi jotkin asiat, mitkä se välttämättä haluaa lisätä takaisin koodiin, joka ei edes toimi tai tuhoaa koodin toimivuuden.

Positiivisesti tekoäly antaa nopeita pieniä ohjelmakoodeja, joita soveltamalla voidaan päästä nopeasti perille siitä, kuinka esimerkiksi käytetään Pythonille luotua PDFPlumber-kirjastoa ja kuinka nopeasti päästään käsiksi PDF-tiedoston raakadataan. Tekoäly ei opeta koodaamaan, ellei siltä pyydetä erikseen selitystä, miten sen tekemät koodit tai muokkaukset toimivat. Monesti korjausten yhteydessä se vain lisää ilman selitystä joitain tekstiä rivien väliin, tuoden ongelmia, jos ei ole kokemusta koodaamisesta tai debuggaamisesta.

Artikkeli on valmisteltu osana Datasta vauhtia valmistavan teollisuuden (VauhtiData) -hanketta, ja haluamme kiittää hankkeen ja tämän artikkelin rahoittamisesta Etelä-Savon elinkeino-, liikenne- ja ympäristökeskusta. Kyseinen hanke on Euroopan unionin osarahoittamana.

Saku Kaarlejärvi

asiantuntija, TKI

SeAMK

Saku Kaarlejärvi on tietotekniikan insinööri, joka on erikoistunut ohjelmointiin, tekoälyyn ja sulautettuihin järjestelmiin. Hänellä on kokemusta tuotannonohjauksen mittauksessa käytetyistä ratkaisuista, Power BI -raporttien rakentamisesta sekä älyteknologioiden mahdollisuuksien tutkimisesta hyvinvointi- ja terveysaloilla.

Juha-Matti Arola

projektipäällikkö, TKI

SeAMK

Juha-Matti Arola on kokenut projektipäällikkö, joka on erikoistunut EU-rahoitteisiin ESR-, JTF- ja EAKR-hankkeisiin, virtuaalisten työ- ja oppimisympäristöjen kehittämiseen sekä tiedolla johtamisen hyödyntämiseen. Hän toimii oman toimensa ohella SeAMKin asiakastiedonhallinnan, CRM:n, pääkäyttäjänä. Hän yhdistää vahvan teknologisen ja pedagogisen osaamisen tukemaan pk-yritysten ja aluekehityksen innovaatioita.

Lähteet

Jsvine. (i.a) *pdfplumber*. <https://github.com/jsvine/pdfplumber>

Veale, K. (2023) MakeUseOf. *Does ChatGPT Have Privacy Issues?*
<https://www.makeuseof.com/chatgpt-privacy-issues/>