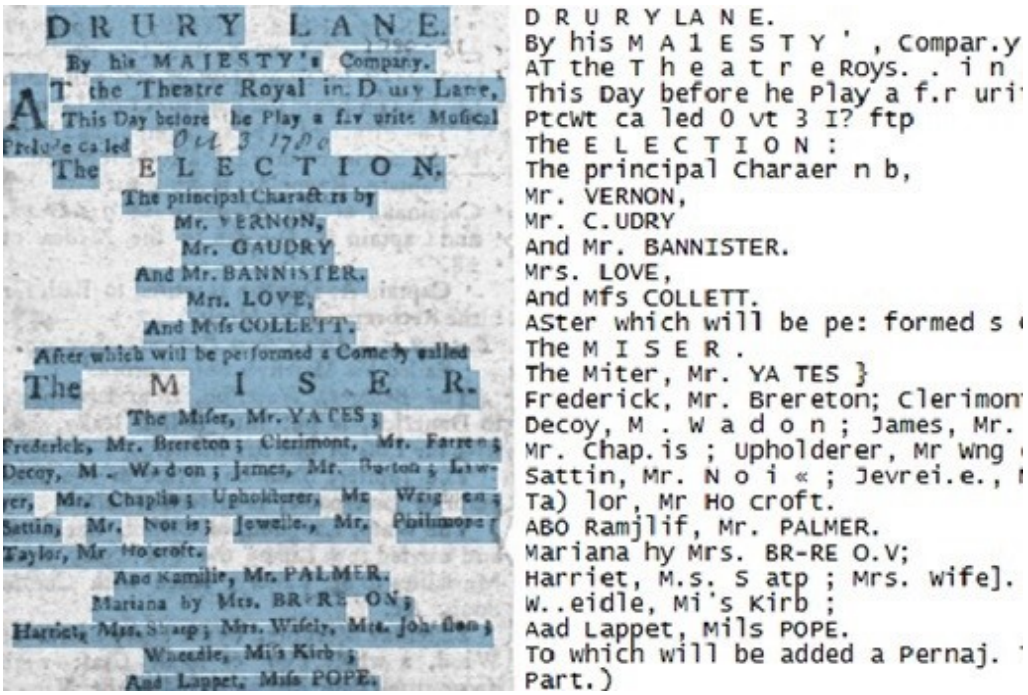
Soft Kompjuting E2 – 2021/22

Nedeljni izazov #2 – OCR
(Rešavanje „Captcha“ problema)

Motivacija

Optičko prepoznavanje znakova (**OCR**) se koristi za konverziju knjiga i dokumenata koji su štampani na papiru u digitalni oblik (tekst). Optičko prepoznavanje znakova - teksta koriste slepe osobe kako bi mogle da "čitaju" štampani materijal uz pomoć OCR softvera i sintetizatora govora.



Postoje razne primene OCR-a u raznim domenima. Sve od izvlačenja teksta iz formalnih dokumenata (razni ugovori, zakoni, izveštaji i slično) do pretvaranja notnih zapisa u audio, prepoznavanjem nota i njihovih pozicija na notnoj skali. Zbog svega navedenog, OCR je često jedan od neizostavnih koraka u procesu **digitalizacije**. U eri informacija, postoji težnja da se svi podaci digitalizuju kako bi se vršilo njihovo sigurnije skladištenje i kasnija obrada.

Kompleksnost OCR rešenja može jako varirati, u zavisnosti od tipa teksta, kao i konteksta iz koga se tekst izvlači. Recimo:

- Izvlačenje (engl. extraction) štampanog teksta je tipičan primer OCR-a
- Izvlačenje rukom pisanog teksta predstavlja dosta složeniji problem zbog same varijacije u rukopisima. Ova oblast se često u literature naziva ICR (intelligent character recognition) i trenutno predstavlja aktivno područje istraživanja. U ovom slučaju se karakteri najčešće dodiruju i preklapaju, što direktno utiče na složenost rešenja.

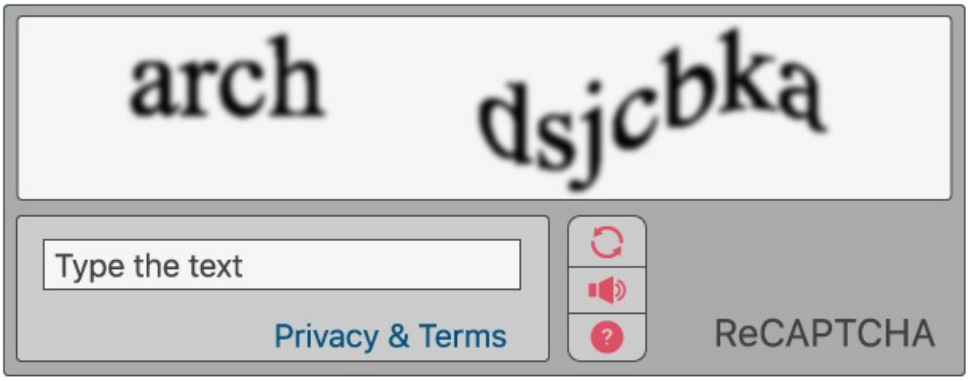
Kompleksnost zavisi i od koteksta u kome se tekst nalazi, kao recimo:

- Ako se tekst nalazi na papiru, to je “najlakši” tip ekstrakcije. Tekst je crne boje, a papir bele pa je zbog kontrasta te dve boje nešto lakše izvršiti segmentaciju.
- Ukoliko je kontekst kompleksniji, to direktno utiče i na kompleksnost sistema. Recimo, tekst se može čitati sa izloga, saobraćajnih znakova, tablica automobila i slično.

Zadatak

Naš ovonedeljni zadatak će biti čitanje teksta sa jednostavnih pozadina (kao što je čitanje teksta sa papira), ali takođe i čitanje teksta sa relativno teških pozadina koje tehnički predstavlja prvu fazu rešavanja Captcha problema.

Kapča (engl. **CAPTCHA**, potvrdni kod) je vrsta izazov-odgovor testa koji se koristi u računarstvu da odredi da li je korisnik čovek ili mašina. Postupak podrazumeva jedan računar (server), koji traži od korisnika da odradi jednostavan test koji je računar sposoban da stvori i oceni. Pošto bi računar trebalo da bude nesposoban da reši taj test, svaki korisnik koji unese tačan odgovor smatra se čovekom. Uobičajene kapče traže od korisnika da unese nekoliko slova koja su prikazana na neki način iskrivljenoj slici.



Skraćenica CAPTCHA dolazi od engleskog Completely Automated Public Turing test to tell Computers and Humans Apart (u prevodu: potpuno automatizovani javni Tjuringov test za razlikovanje računara i ljudi).

Problem koji stvaraju uobičajene kapče jeste što onemogućuju pristup ne samo računarima, već i slepim i slabovidim licima, koja nemaju načina da odgovore na traženi upit. Neki od načina za rešavanje ovog problema, mada se u praksi još uvek relativno retko koriste, jesu omogućavanje slepim ili slabovidim licima da klikom na odgovarajuće dugme zatraže da umesto slikovne pokrenu zvučnu kapču, ili da se obrate administratoru sistema, koji će im omogućiti otvaranje naloga.

Kako vreme prolazi, razvijaju se sve napredniji algoritmi za čitanje kapči, pa i same sličice postaju sve složenije. Ovo s jedne strane znači da ponekad čitanje slova sa slike i za čoveka može da predstavlja veliki izazov, neki sistemi nude korisniku da zameni sliku ukoliko mu je nečitljiva. S druge strane, novi algoritmi razvijeni za razbijanje sistema kapči mogu da unaprede tehnologiju za optičko prepoznavanje znakova (OCR).

Neki osnovni koraci u rešavanju ovog izazova bi mogli biti:

1. segmentacija slike (binarizacija) - sami treba da odredite na koji način je to najbolje uraditi. Budite kreativni,
2. Uklanjanje šuma i post-procesiranje problematičnih situacija (možda spojenih kontura, možda razdvojenih kontura i sl)...,
3. Treniranje neuronske mreže nad zadatim alfabetom,
4. Procesiranje train skupa i čitanje teksta sa svih slika koje se u njemu nalaze.

Za razliku od prošle godine, platforma neće vršili download serijalizovanih fajlova modela, zbog zloupotreba prošle godine. Svi modeli će se ponovo trenirati na platformi, pa vodite računa da imate identične verzije biblioteka OpenCV, Keras i Theano, jer se u suprotnom može desiti da treniranje ili izvlačenje kontura na platformi radi drugačije (različite verzije biblioteka imaju različitu implementaciju svojih metoda). Specifikaciju biblioteka imate u ovom dokumentu, u sekciji **Dozvoljene biblioteke i podešavanje okruženja**.

Izvršavanje svakog pojedinačnog rešenja je ograničeno na **60 minuta**.

Na vežbama smo prošli sasvim dovoljno teorijskih i praktičnih osnova za rešavanje ovog izazova. ***Budite kreativni i primenite ih na svoj način, tako da dobijete što bolje rezultate.***

Format koda za ocenjivanje (isto za sve izazove)

Kod koji upload-ujete u GoogleDrive folder treba da zadovolji neke kriterijume da bi ga platforma za ocenjivanje analizirala na pravi način. Glavna ograničenja su sledeća:

1. **Fajl main.py se mora nalaziti u korenu vašeg foldera.** Ukoliko to nije slučaj, platforma neće biti u mogućnosti da pokrene vaše rešenje i nećete biti ocenjeni.

2. **Fajlove main.py i evaluate.py nije dozvoljeno menjati.** Ovi fajlovi su direktno korišćeni od strane platforme da bi ocenjivanje bilo moguće.

3. **Vaša implementacija treba da bude u fajlu process.py.** U ovom fajlu se nalazi neimplementirana metoda koja ima jasno naznačen ulaz i izlaz. Metoda je automatski uklopljena u ostatak koda (poziva se iz main.py) i nema potrebe da je ručno pozivate. **Vaš zadatak je da implementirate traženu metodu i da obezbedite da vraća ono što se od vas traži.**

4. **Dozvoljeno je kreiranje novih python fajlova, koje možete pozivati iz process.py.** Ukoliko želite da deo koda izdvojite u druge fajlove I da onda kroz python import koristite u process.py, to je dozvoljeno. Dok god poštujete sve prethodne korake, ne bi trebalo biti problema.

5. **U kodu koji okačite na platformu nemojte koristiti sistemske pauze i slične mehanizme koji zahtevaju reakciju korisnika, pošto u tom slučaju rešenje neće biti pokrenuto.**

Pokretanje rešenja i evaluacija

Da biste pokrenuli rešenje na svojoj mašini i proverili kolika je postignuta tačnost, potrebno je uraditi sledeće:

1. Implementirati metodu u **process.py** traženom logikom. Ovaj fajl **ne** pokrećete direktno.
2. Pokrenuti **main.py** (iz pycharm-a na Run, ili iz terminala komandom “python main.py” uz prethodno aktiviranje odgovarajućeg virtuelnog okruženja). Pokretanje main.py fajla će izgenerisati **result.csv** fajl, tako što će pozvati prethodno implementiranu metodu za sve primerke iz skupa podataka.
3. Pokrenuti **evaluate.py** fajl (iz pycharm-a na Run, ili iz terminala komandom “python evaluate.py” uz prethodno aktiviranje odgovarajućeg virtuelnog okruženja). Ovaj fajl će učitati result.csv koji je prethodno generisan i izračunati tačnost. Izlaz ovog fajla je samo broj koji pokazuje procenat tačnosti trenutnog rešenja.

Ocenjivanje (isto za sve izazove)

Ocenjivanje upload-ovanog koda će biti izvršavano iterativno, po sledećim pravilima:

1. Platforma će automatski vršiti download koda, jednom u 24h i vršiti ocenjivanje.

2. U toku jednog dana možete imati neograničen broj upload-a. Ocenjivanje će svakako biti pokrenuto samo jednom na kraju dana i biće ocenjen kod koji u tom trenutku bude u folderu na Google Drive-u.

3. Platforma vrši ocenjivanje za prethodni dan u periodu **od 3:00 iza ponoći do 8:00 ujutru narednog dana**, pa u tom periodu nije dozvoljeno menjanje fajlova.

4. Ukoliko izazov traje 7 dana, studenti tehnički imaju 7 pokušaja da reše izazov. Platforma će ocenjivati kod svaki dan. Na rang listu će se računati **najbolji rezultat** iz svih ciklusa ocenjivanja. Zbog toga je bolje da što ranije rešite izazov, pošto ćete imati više pokušaja da ispravite nešto i postignete još bolji rezultat. Ako bilo koji pokušaj bude detektovan kao plagijat, student dobija godinu dana zabrane polaganja.

5. Svaki dan će studenti dobijati izveštaj u formi txt fajla u svom Google Drive folderu. Ovo se odnosi samo na studente koji su postavili nešto u svoj folder. Izveštaj se generiše svaki dan, bez obzira na to da li ste šta menjali u folderu tog dana. Tako ćete na dnevnom nivou biti ažurirani činjenicom gde se nalazite na rang listi.

6. U izveštaju niko neće imati informaciju gde se tačno nalazi na rang listi. Dobićete informaciju da li se nalazite u TOP 5, TOP 10, TOP 25 ili TOP 50 studenata. Ako ste dobili informaciju da ste u TOP 25, to znači da se nalazite izmedju 11. i 25. pozicije i da možete poboljšati rešenje da popravite rang. Tačan rang će biti objavljen naknadno, tek na kraju izazova.

Dozvoljene biblioteke i podešavanje okruženja

U sklopu ovog izazova je dozvoljeno koristiti sledeće biblioteke uz Python 3.6:

- numpy
- openCV verzija **3.4.1.15** (bilo koja verzija koja počinje sa 3.4.1.x)
- matplotlib
- scikit-learn
- keras verzija **2.1.5**
  - za FeedForward potpuno povezane NM
  - **nije dozvoljeno koristiti konvolutivne mreže, odnosno *Conv* slojeve**
- theano verzija **1.0.4** (kao backend za keras)
- fuzzywuzzy

Preporuka je da koristite “theano” kao backend za keras biblioteku, a ne tensorflow, pošto platforma koristi theano. Nekad tensorflow trening odradi brže, pa da se ne desi da na platformi ne bude dovoljno epoha i da mreža ne bude dovoljno istrenirana (underfitting) samo zato što platforma koristi theano kao backend. Uputstvo za prebacivanje keras backend-a sa tensoflow-a na theano imate u dokumentu sa najčešćim problemima, u sekciji 6.

Instaliranje (možete samo dodati nove biblioteke u okruženje od prvog izazova):

Za kreiranje okruženja i instalaciju biblioteka je potrebno preuzeti najnoviju Anaconda distribuciju sa njihovog zvaničnog sajta i instalirati je. Anaconda postoji za sve moderne operativne sisteme. Nakon instaliranja možete preći na kreiranje virtuelnog okruženja i instaliranje biblioteka u njega.

Detaljniji opis šta virtuelna okruženja predstavljaju možete naći u sklopu **v0** na github repozitorijumu predmeta (https://github.com/ftn-ai-lab/sc-2021-e2/blob/master/v0-priprema/podesavanje-okruzenja.ipynb)

1. Kreirati virtuelno okruženje (iz terminala na Linux i MacOS, ili Anaconda prompt na Win)

   ```
   conda create -n soft-env python=3.6
   ```

2. Aktivirati okruženje (ukoliko ćete fajlove kasnije pokretati iz terminala, a ne iz PyCharm-a)

   ```
   source activate soft-env
   ili
   conda activate soft-env
   ```

3. Instalirati biblioteke (žutom bojom su označene one koje su dodate u izazovu 2)

   conda install -n soft-env -c conda-forge opencv=3.4.1
   conda install -n soft-env -c conda-forge keras=2.1.5
   conda install -n soft-env -c conda-forge theano=1.0.4
   conda install -n soft-env scikit-learn
   pip install matplotlib
   pip install fuzzywuzzy

4. Preuzeti i instalirati pyCharm Community razvojno okruženje, koje je preporuka za razvoj python projekata. Otvoriti projekat koji je deo ovog izazova.

5. Podesiti interpreter za projekat tako što ćete se povezati na python instancu iz prethodno kreiranog virtuelnog okruženja. Uputstvo je nalazi na kraju sledećeg fajla, koji je na github repozitorijumu predmeta (https://github.com/ftn-ai-lab/sc-2021-e2/blob/master/v0-priprema/podesavanje-okruzenja.ipynb)

6. Sve je spremno. Desni klik na odgovarajući fajl i onda Run ili Debug. Ukoliko importovanje cv2 biblioteke puca u skripti, proverite da li ste dobro instalirali openCV i da li ste dobro povezali projekat sa virtuelnim okruženjem.