### **Logistika**

* Predaja se vrši preko github classroom sistema: <https://classroom.github.com/a/OAmMHZwW>
* Sav kod treba da bude u jednom .py fajlu.
* Rok za predaju je 13.12.2024 do kraja dana.

**Zadatak:**Analiza histograma boja u slikama razvrstanim u različite klase (npr. pejzaži, portreti, objekti). Cilj je da se za svaku sliku izračuna histogram boja u RGB sistemu boja, a zatim da se prosečni histogrami porede između klasa, i konačno da se implementira jednostavan klasifikator koji procenjuje kojoj klasi slika pripada na osnovu kosinusne sličnosti sa prosečnim histogramima klasa.

**Napomena:**Rešenje treba da bude implementirano tako da:

* Nigde ne koristi eksplicitno navedenu petlju ili list i dict comprehension.
* Ne koristi funkciju len za određivanje dužine neke kolekcije (ukoliko je potrebno, len implementirati kroz reduce).
* Dozvoljeno je forsirati realizaciju iteratora u list pozivom funkcije list (ukoliko je potrebno, a ako bude potrebno, verovatno će biti posle poziva map).
* Dozvoljena (i preporučena) je upotreba lambda funkcija.

#### **1. Računanje histograma boja u 3D prostoru za jednu sliku (5 poena)**

Napisati funkciju koja kao jedini parametar prima putanju do slike, a vraća po jedan histogram za svaku od RGB komponenti (ukupno tri histograma). Broj binova definisati kao konstantu koja se može koristiti iz globalnog okruženja (dobar izbor za broj binova bi trebalo da se kreće u opsegu 8-16). Posle prebrojavanja histogram je potrebno i normalizovati (podeliti sa ukupnim brojem piksela u slici).

*Format rezultata:*2D niz (npr. numpy matrica) gde svaka dimenzija predstavlja bin za RGB vrednosti, a svaki element beleži frekvenciju pojavljivanja odgovarajuće date vrednosti.

#### **2. Računanje prosečnog histograma po klasama (7 poena)**

Implementirati funkciju koja prima listu slika parova (klasa slike, putanja do slike) i računa prosečan histogram boja za svaku klasu. Za svaku klasu, histogram boja treba da bude prosečan histogram svih slika koje pripadaju toj klasi.

* Koristiti map i/ili reduce kako bi se dobio agregirani histogram za sve slike u jednoj klasi.
* Podeliti ukupni histogram sa brojem slika u klasi da bi se dobio prosečni histogram.
* Dozvoljena je upotreba pythonove funkcije sorted

*Format rezultata:*Rešenje treba da bude implementirano tako da vraća listu torki (klasa, prosečan histogram), gde je prosečan histogram 2Dniz (matrica) sa prosečnim vrednostima.

#### **3. Računanje kosinusne sličnosti (2 poena)**

Implementirati funkciju koja izračunava kosinusnu sličnost između dva histograma boja. Sličnost se računa između 2D matrica histograma koje se prethodno moraju „poravnati” (flatten) u 1D nizove.

* Koristiti map i/ili reduce za izračunavanje skalarnih proizvoda i normi.
* Dozvoljeno je koristiti numpy funkciju flatten kako bi se 2D matrica pretvorila u 1D niz (ali razmislite kako biste to uradili kroz map / reduce)
* Kosinusna sličnost se može izarčunati pomoću numpy funkcija: https://www.geeksforgeeks.org/how-to-calculate-cosine-similarity-in-python/

*Format rezultata:*Funkcija treba da vraća vrednost kosinusne sličnosti između dva histograma.

#### **4. Jednostavan klasifikator zasnovan na histogramima (6 poena)**

Implementirati jednostavan klasifikator koji procenjuje kojoj klasi slika pripada na osnovu poređenja njenog histograma sa prosečnim histogramima klasa. Za svaku sliku koja treba da se klasifikuje, uporediti njen histogram sa prosečnim histogramima svake klase koristeći kosinusnu sličnost, i dodeliti je klasi sa najvećom sličnosti.

* Prvi korak je izračunavanje histograma boja za sliku (iz tačke 1).
* Drugi korak je poređenje sa prosečnim histogramima svake klase (iz tačke 2) koristeći kosinusnu sličnost (iz tačke 3).
* Klasa sa najvećom kosinusnom sličnosti je predviđena klasa za sliku.

*Format rezultata:*Funkcija treba da vraća torku (identifikator slike, predviđena klasa, vrednost sličnosti).

Prilikom izrade projekta odabrati jedan od sledećih skupova slika, i raditi nekoliko klasa (4-5). Dovoljno je da se za svaku kategoriju uzme stotinak slika (veći broj bi trebalo da da bolje rezultate, proverite!), uz par desetina slika za testiranje.

* STL-10: <https://cs.stanford.edu/~acoates/stl10/>
* CIFAR-10: <https://www.cs.toronto.edu/~kriz/cifar.html>
* Caltech101: <https://data.caltech.edu/records/mzrjq-6wc02>

***Napomena:*** *Zadati pristup klasifikaciji je veoma naivan i vrlo verovatno neće dati dobre rezultate! Poenta zadatka je vežba zadatog stila programiranja.*