Univerzitet u Sarajevu
Elektrotehnički Fakultet

Prepoznavanje Oblika i Obrada Slike

Projektni Zadatak br. 1

Studenti:
Muftić *Belma, 1423/17260*
Lemeš *Lamija, 1474/17070*
Krupalija *Ehlimana, 1431/17461*

Odgovorni asistent:
MoE Sumejja Porča

Novembar, 2018

Sadržaj

1 *Dataset*

1.1 Tema projekta

Projekat se bavi analizom fotografija različitih vrsta **krvnih ćelija**. Na slikama se nalaze četiri vrste krvnih ćelija:

1. *Neutrophil*;
2. *Eosinophil*;
3. *Monocyte*;
4. *Lymphocyte*.

Na slikama je potrebno pronaći ćeliju te odrediti kojoj od sljedećih klasa pripada:

1. *Neutrophil*;
2. *Lymphocyte*;
3. Ništa od navedenog (neka druga vrsta krvne ćelije).

Nakon toga potrebno je izdvojiti ćeliju i označiti njenu poziciju na slici.

1.2 Opis *dataset*-a

Dataset se sastoji od **12,444** slika. Među tim slikama nalaze se četiri prethodno opisane klase (odnosno vrste krvnih ćelija). Broj uzoraka svake klase prikazan je u sljedećoj tabeli:

Klasa	**Uzorci za trening**	**Uzorci za testiranje**	**Ukupan broj uzoraka**
Neutrophil	2,499	624	3,123
Eosinophil	2,497	623	3,120
Monocyte	2,478	620	3,098
Lymphocyte	2,483	620	3,103

Za svrhe ovog projekta biti će upotrijebljeno ukupno **90 slika** (po **30 slika** za sve tri klase: *Neutrophils*, *Lymhocythes*, ostalo).

Cijeli projekat dostupan je na *GitHub*-u, na sljedećem linku: `https://github.com/bmuftic1/POOS_Projekat`

2 *DataPrep1*

Iz foldera *DataSet* učitavaju se slike, jedna po jedna, te se regija od interesa treba označiti tako što se nacrta pravougaonik oko ćelije. Pravougaonik se crta tako što se pritisne na gornji lijevi ugao željene regije, a potom se povuče preko slike sve do donjeg desnog ugla regije od interesa. Kada se to uradi, onda će se pojaviti zeleni pravougaonik, te ako se pritisne **C**, slika će izdvojiti regiju i nastaviti dalje, a ako se pritisne **R**, resetuje se slika. Istovremeno se u datoteci *KoordinateROI.txt* pamte koordinate prethodno spomenutih uglova, a struktura datoteke je:

```
NazivSlike x1,y1,x2,y2
```

Slike se spašavaju u folder *ROI*, a kod koji sadrži kreiranje anotacija, kao i slike na kojima je primjenjena maska se nalazi u folderu *Code* pod nazivom *kreiranjeROI.py*. Također, izdvojena je datoteka *spasavanjeSlike.py*, tako da se može koristiti i u narednim zadacima.

3 *DataPrep2*

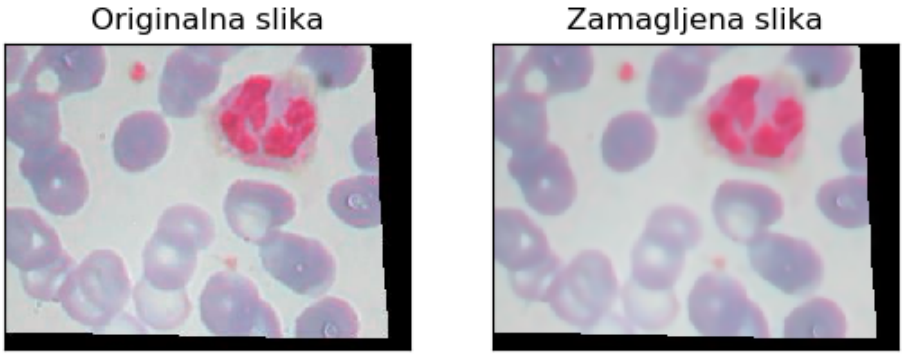
3.1 Uklanjanje šuma

Za uklanjanje šuma (zamagljivanje slike - *blurring*) izvršen je izbor između sljedećih filtera:

- **Filter na bazi prosjeka (*Averaging filter*):** Vrijednost piksela mijenja se sa srednjom vrijednošću svih piksela u oblasti od interesa (na ovaj način zamagljenje slike bude veoma veliko);
- **Filter na bazi statističkog prosjeka (*Mediana filter*):** Vrijednost piksela mijenja se sa medijanom uzorka (efekat zamagljenja je manji);
- **Bilateralni filter:** Pri računanju vrijednosti za zamjenu vrši se računanje prosjeka samo za okolinu nekog piksela, što ne uključuje cijelu oblast od interesa (kao rezultat, ivice će biti očuvane, odnosno neće biti zamagljene, dok će se šum smanjiti u ostalim dijelovima slike)

Za upotrebu je odabran **bilateralni filter**, jer iako je sporiji od ostalih filtera, ne zamagljuje ivice, čije je očuvanje važno pri analizi krvnih ćelija.

Na sljedećoj slici prikazan je prije i nakon vršenja redukcije šuma korištenjem bilateralnog filtera:



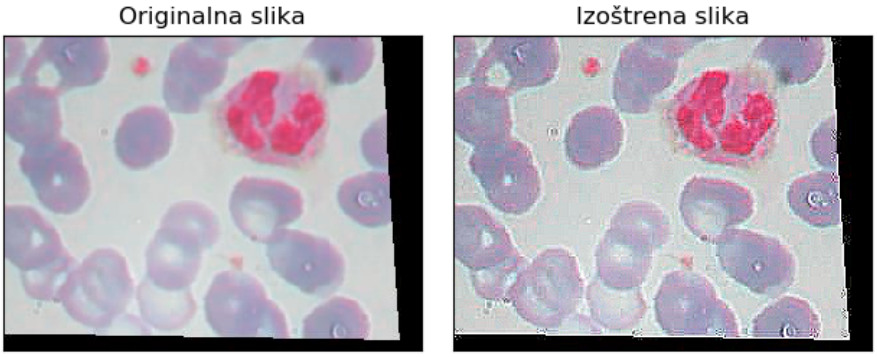
Slika 1: Smanjenje šuma na slici korištenjem bilateralnog filtera

3.2 Maskiranje neoštrina

Za maskiranje neoštrina izabrana je kernel matrica sa vrijednostima:

$$\begin{bmatrix} 0 & -2 & 0 \\ -2 & 9 & -2 \\ 0 & -2 & 0 \end{bmatrix}$$

Na slici ispod prikazan je rezultat izoštravanja u odnosu na originalnu sliku:



Slika 2: Izoštravanje slike korištenjem filtera za maskiranje neoštrina

4 *DataPrep3*

4.1 Poboljšavanje kvaliteta slika

4.1.1 Poboljšavanje kontrasta

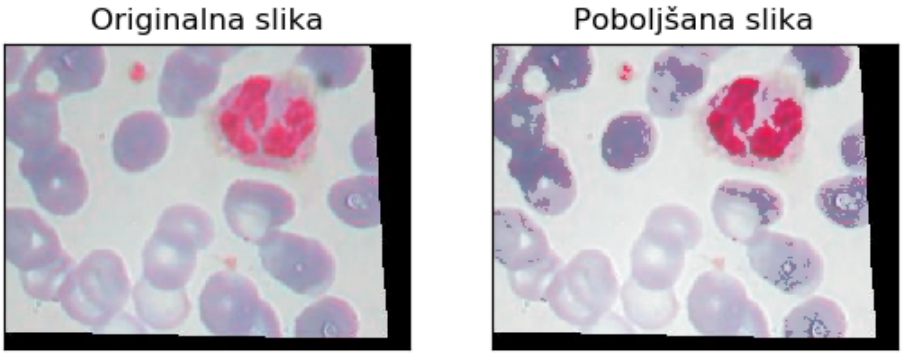
Za poboljšavanje kontrasta slike korištena su tri različita postupka, koji će biti opisani u nastavku.

1. **Aritmetičke operacije**:

   Kako bi se poboljšao kontrast slike, prvenstveno je neophodno pretvoriti RGB u HLS sliku. Zatim se vrši manipulacija nad pikselima koji označavaju *Luminence* u okviru tako transformisane slike. Koriste se aritmetičke operacije nad pojedinačnim pikselima slike, na sljedeći način:

   - Ukoliko je vrijednost piksela manja od srednje vrijednosti svih *Luminence* piksela, vrši se **smanjenje vrijednosti piksela** za iznos faktora koji se može prilagođavati (svijetli pikseli postaju svjetliji za iznos faktora);
   - Ukoliko je vrijednost piksela veća od srednje vrijednosti svih *Luminence* piksela, vrši se **povećanje vrijednosti piksela** za iznos faktora koji se može prilagođavati (tamni pikseli postaju tamniji za iznos faktora).

   Na sljedećoj slici prikazan je izgled slike prije i nakon vršenja poboljšanja za vrijednost faktora `factor = 30`:



Slika 3: Poboljšavanje kontrasta slike korištenjem aritmetičkih operacija

2. **Linearno razvlačenje kontrasta**:

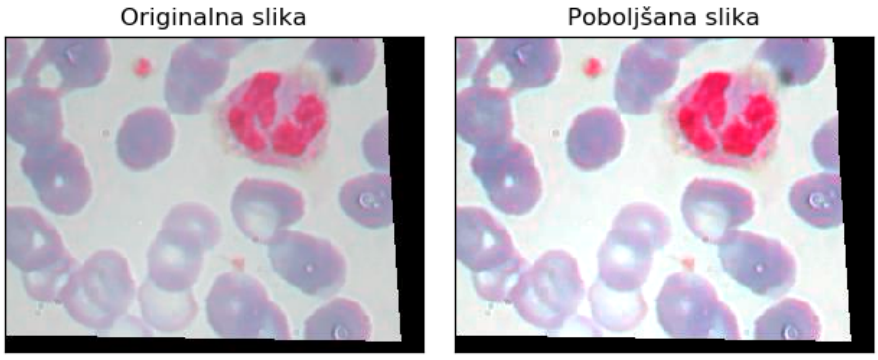
Drugi metod poboljšanja kontrasta slike postiže se manipulacijom nad pojedinačnim kanalima slike. Svaka vrijednost piksela se određuje u zavisnosti od vrijednosti susjednih piksela. Intenzitet se izračunava pomoću formule za normalizaciju:

$$I_o = (I_i - Min_i) * (((Max_o - Min_o)/(Max_i - Min_i)) + Min_o)$$

Gdje su vrijednosti:

- $\texttt{I}_o$ - Izlazna vrijednost piksela
- $\texttt{I}_i$ - Ulazna vrijednost piksela
- $\texttt{Min}_i$ - Minimalna vrijednost piksela u ulaznoj slici
- $\texttt{Max}_i$ - Maksimalna vrijednost piksela u ulaznoj slici
- $\texttt{Min}_o$ - Minimalna vrijednost piksela u izlaznoj slici
- $\texttt{Max}_o$ - Maksimalna vrijednost piksela u izlaznoj slici

Na sljedećoj slici prikazan je izgled slike prije i nakon linearnog razvlačenja kontrasta:



Slika 4: Poboljšavanje kontrasta slike korištenjem linearnog razvlačenja

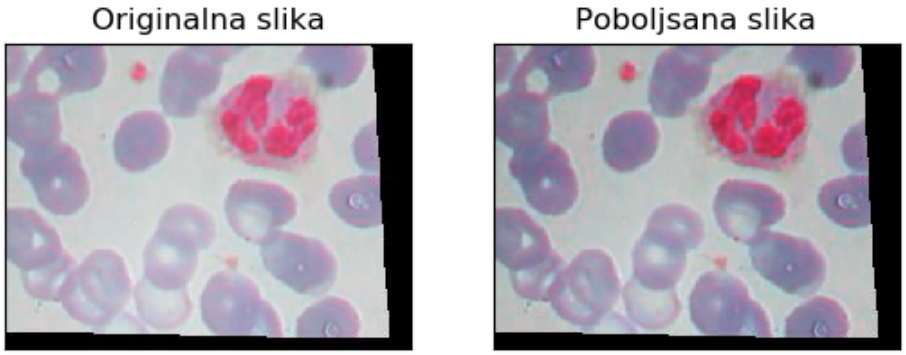
3. ***Gamma* korekcija**:

Za ovaj zadatak korištena je *gamma* korekcija koja transformiše datu sliku prema formuli:

$$NovaSlika = Slika^{\frac{1}{gamma}}$$

gdje je `gamma` vrijednost koju mi zadajemo.

Novu sliku je potrebno korigovati tako da su vrijednosti piksela u opsegu 0 do 255, uključivo. Za vrijednost `gamma` je odabrano 0.7, kako se treba istaći ljubičasti dio regije od interesa. Samo računanje novih vrijednosti piksela je urađeno preko LUT (*lookup table*), gdje su se pikseli originalne slike mapirali u piksele `gamma` korigovane slike prema navedenoj formuli. Rezultat poboljšanja kontrasta je prikazan na sljedećoj slici:



Slika 5: Poboljšavanje kontrasta slike korištenjem *gamma* korekcije

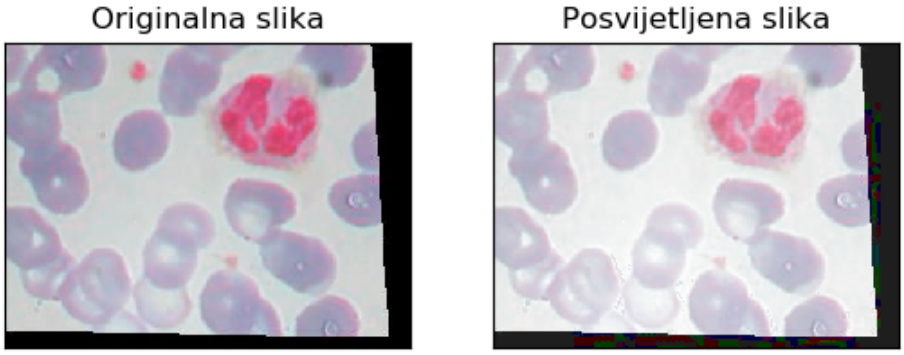
4.1.2 Povećanje osvjetljenja

Za povećanje osvjetljenja slike korištena su tri različita postupka, koji će biti opisani u nastavku.

1. **Aritmetičke operacije**:

   Kako bi se povećalo osvjetljenje slike, prvenstveno je neophodno pretvoriti RGB u HLS sliku. Zatim se vrši manipulacija nad pikselima koji označavaju *Luminence* u okviru tako transformisane slike. Koristi se aritmetička operacija **sabiranja** pojedinačnih piksela slike sa iznosom faktora (koji se može prilagođavati).

   Na sljedećoj slici prikazan je izgled slike prije i nakon vršenja poboljšanja za vrijednost faktora `factor = 30`:

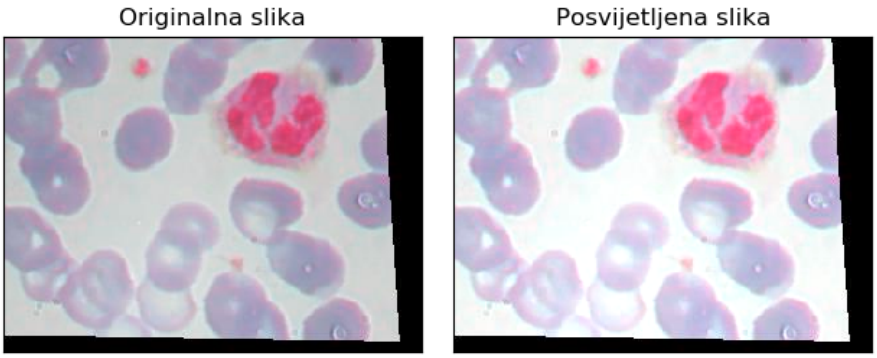


Slika 6: Povećanje osvjetljenja slike korištenjem aritmetičke operacije sabiranja

2. **Linearne transformacije**:

   Drugi metod za povećanje osvjetljenja slike podrazumijeva primjenu linearnih transformacija. Konkretno promjena vrijednosti parametra beta u OpenCV funkciji *convertScaleAbs*.

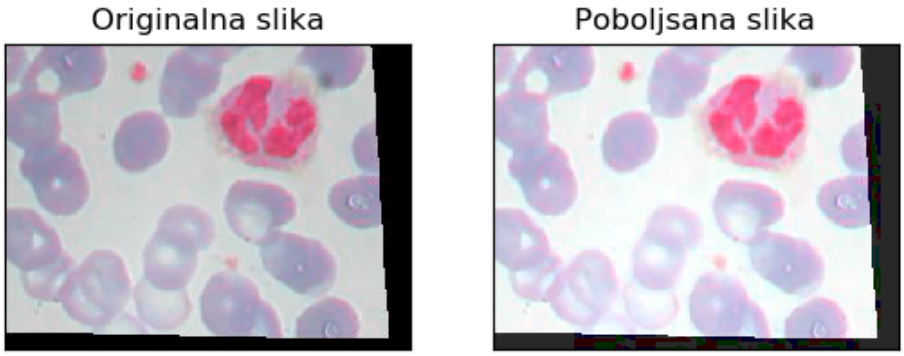
   Na sljedećoj slici je izgled slike prije i nakon povećanja osvjetljenja postavljanjem vrijednosti parametra `beta` na 20 (moguće su vrijednosti od 0 do 100):



Slika 7: Povećanje osvjetljenja slike korištenjem linearne transformacije

3. **Manipulacija HSV slike**:

   Korištena je manipulacija intenziteta treće komponente HSL-a, tj. *lightness*. Faktor kojim se poboljšava osvjetljenje je 40, te u slučaju da zbir trenutne vrijednosti treće komponente piksela i faktora ima vrijednost veću od 255, vrijednost se postavlja na 255, u suprotnom, nova vrijednost postaje pomenuti zbir. Nakon korekcije treće komponente, sve tri komponente se spajaju u HSV sliku, koja se zatim pretvori u BGR vrijednost, i spasi u folder *Brightness*. Sljedeća slika prikazuje promjenu osvjetljenja odabrane slike pomoću manipulacije HSV:



Slika 8: Povećanje osvjetljenja slike korištenjem manipulacije HSV slike

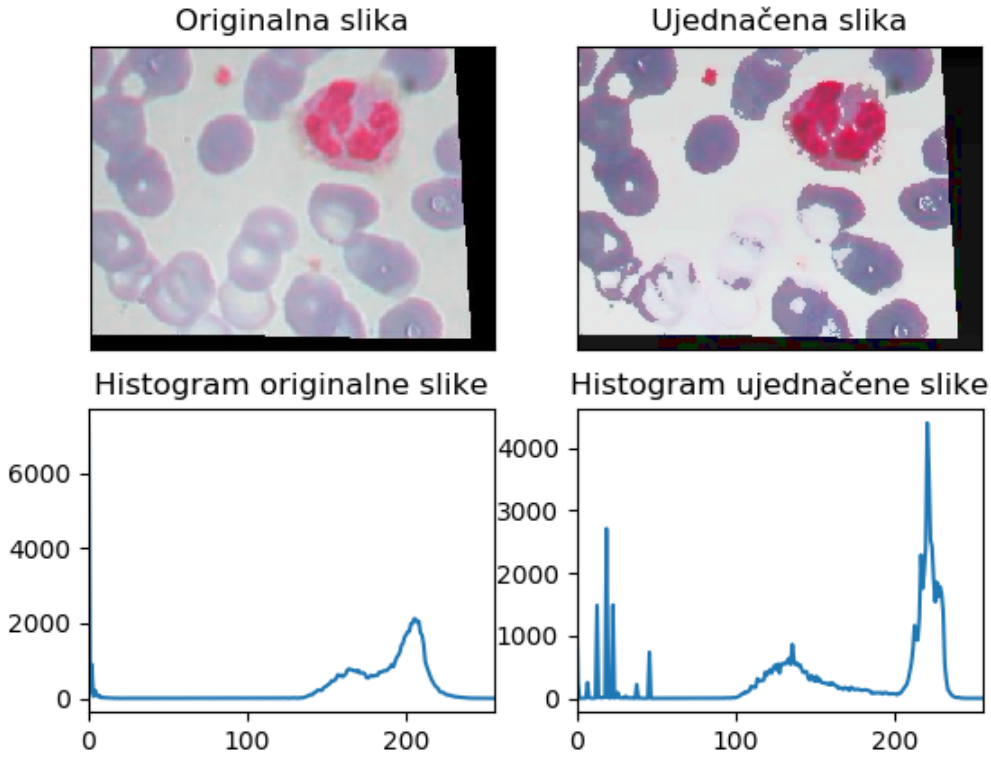
4.1.3 Ujednačavanje histograma

Za ujednačavanje histograma slike korištena su tri različita postupka, koji će biti opisani u nastavku.

1. **Raspodjela vjerovatnoća**:

   Kako bi se ujednačio histogram slike, prvenstveno je neophodno pretvoriti RGB u HLS sliku. Zatim se vrši analiza piksela koji označavaju *Luminence* u okviru tako transformisane slike. Za svaki piksel pronalazi se njegova okolina (čija veličina ovisi o iznosu prilagodljivog faktora), te se zatim pronalazi vrijednost **piksela s najmanjom frekvencijom pojavljivanja** iz te okoline. Zatim se trenutni piksel izjednačava s tom vrijednošću i vrijednost frekvencije piksela povećava. Na ovaj način postiže se ujednačavanje histograma, odnosno preraspodjela vjerovatnoće pojavljivanja piksela u okolinu.

   Na sljedećoj slici prikazan je izgled slike i njenog pripadajućeg histograma prije i nakon vršenja ujednačavanja histograma za vrijednost faktora `factor = 30`:

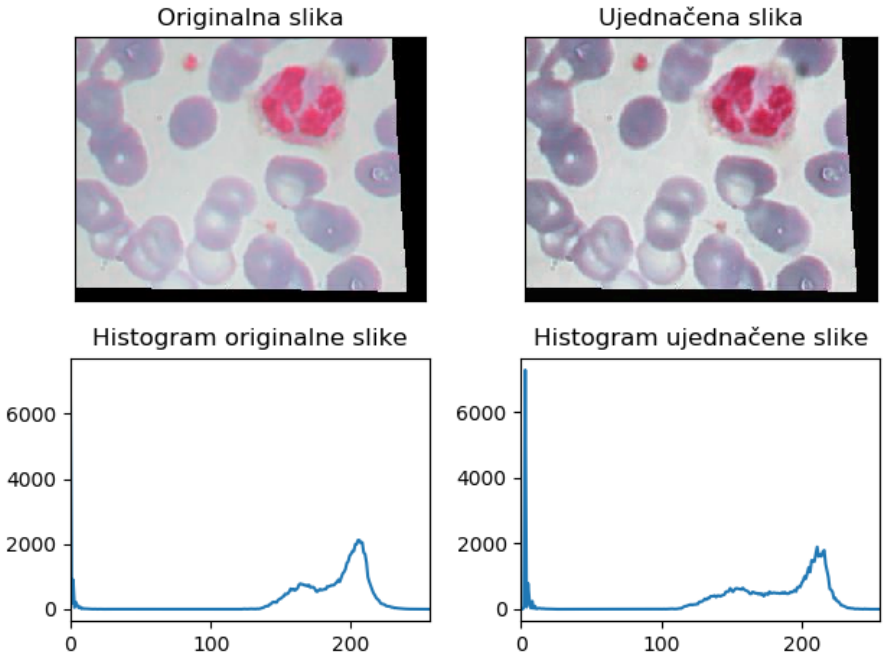


Slika 9: Ujednačavanje histograma slike korištenjem raspodjele vjerovatnoća

2. **CLAHE**:

Drugi metod ujednačavanja histograma slike jeste CLAHE (eng. *Contrast Limited Adaptive Histogram Equalization*). Da bi se omogućilo korištenje ovog tipa ujednačavanja, sliku je bilo potrebno prethodno pretvoriti u *grayscale*.

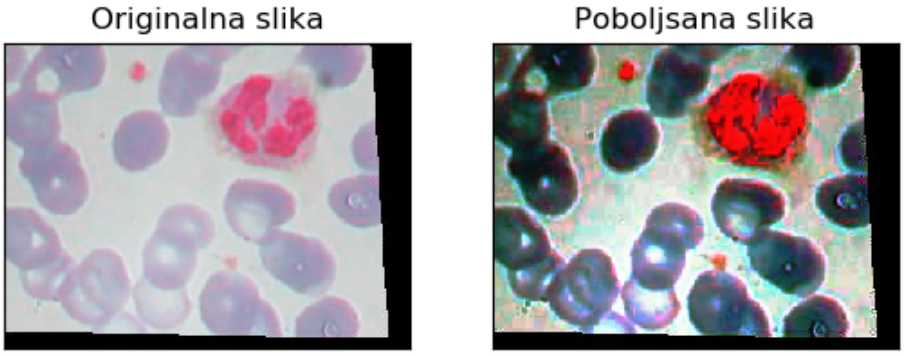
Na sljedećoj slici prikazan je izgled slike i njenog histograma prije i nakon izvršenja ujednačavanja histograma za vrijednost faktora `factor = 10`:



Slika 10: Ujednačavanje histograma slike korištenjem CLAHE

3. `equalizeHist`:

   Za ujednačavanje histograma korištena je funkcija `equalizeHist` iz biblioteke cv2, i to nad svim *channels* od date RGB slike, nakon čega je ista spašena u folder *Histogram*. Na sljedećoj slici je prikazan rezultat primjene ove funkcije:



Slika 11: Ujednačavanje histograma slike korištenjem `equalizeHist`

5 *DataPrep4*

Kreiran je file *podjela.py* koji definiše kako će se *dataset* podijeliti, gdje je odlučeno da podjela bude sljedeća:

- **Validacija (10%)**: Po 3 slike za svaku klasu, dakle ukupno 9 slika;
- **Test (25%)**: Po 7 slika za svaku klasu, dakle ukupno 21 slika;
- **Trening (65%)**: Po 19 slika za svaku klasu, dakle ukupno 60 slika.

Kako su slike poredane tako da se one sa brojevima između 1 i 30 uključivo nalaze u prvoj klasi, one sa brojevima između 31 i 60 u drugoj klasi, a slike sa brojevima u nazivu između 61 i 90 pripadaju klasi tri, tako je moguće izvršiti ravnopravnu podjelu tako da svaka klasa bude zastupljena u svakom skupu. Prvo se kreiraju folderi *Validacija*, *Test* i *Train* (u slučaju da već ne postoje), te se nasumice generišu brojevi između prethodno spomenutih intervala. Slike sa datim brojem se potom kopiraju u odgovarajući folder. Dovoljno je samo pokrenuti datoteku, bez poziva funkcija, kako se kod nalazi u `main` funkciji.

Također, važno je naznačiti da se svaka *.py* datoteka može uključiti kao *.dll* i nije potrebno pozivati određene funkcije unutar njih, već samo pozvati sami modul, te će on izvršiti ono što se nalazi u `main` funkciji (primjer upotrebe se nalazi u *projekat.py*).