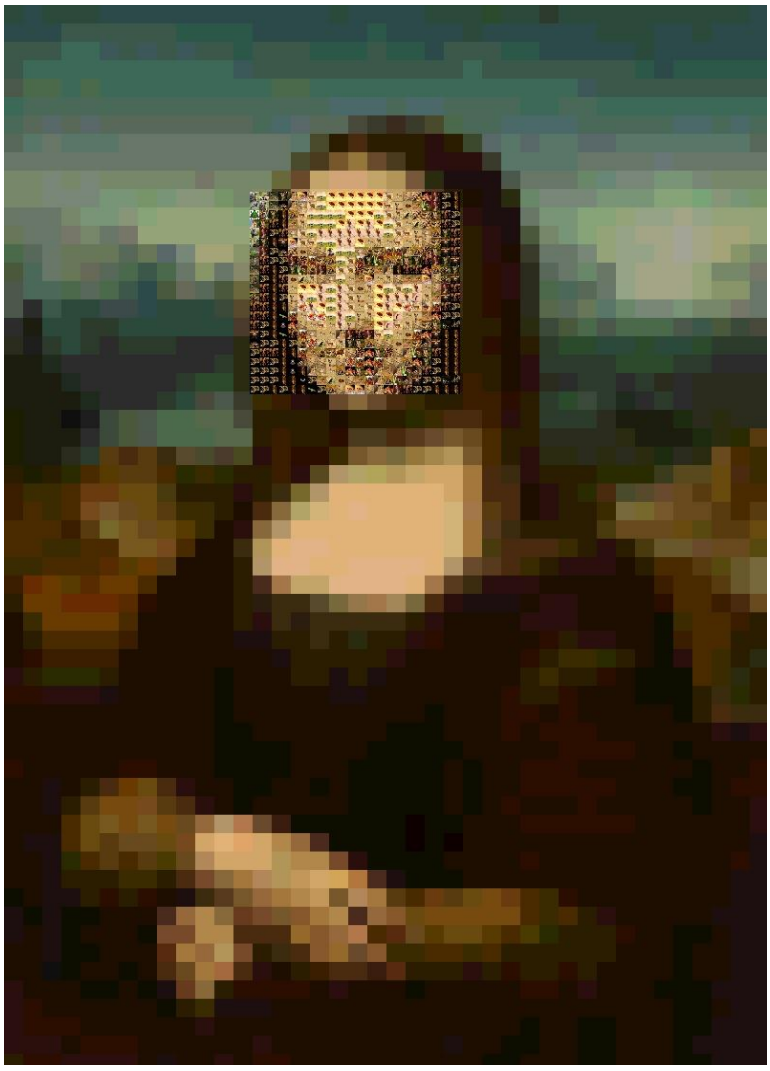
Дигитално процесирање на слика

Тема:

Креирање на мозаик



Сретен Главинчески 226019

Мартина Спировска 221049

СОДРЖИНА

ВОВЕД.....3

1. МОДЕЛ 1.....5

1.1 Начин на извлекување на средна RGB вредност за дадена слика/сегмент ...5

1.2 Избор на слика за сегмент.....6

2. МОДЕЛ 2.....7

3. МОДЕЛ 3.....8

3.1 Графичко претставување на податоци и користење на хистограми.....8

3.2 Споредба на вредности кај хистограми 8

4. МОДЕЛ 4.....11

4.1 Начин на одредување на најзастапена(доминантна) боја кај слика.....11

5. МОДЕЛ 5.....12

5.1 Креирање на мозаик од лица со помош на модел за препознавање на лица.....12

5.2 Фактор на големина.....12

5.3 Фактор на потврда.....12

5.4 Минимална големина.....13

6. МОДЕЛ 6.....15

6.1 Прв пристап.....15

6.2 Втор пристап.....17

ЗАКЛУЧОК.....18

ВОВЕД

Цел на оваа проектна задача е изработка на мозаик. За таа цел се користи слика/примерок од кој што сакаме да креираме мозаик, додека пак како главни компоненти од кои што сакаме да биде составен тој мозаик ќе бидат искористени бази од слики. Во оваа проектна задача ќе бидат разгледани повеќе модели за креирање на мозаик и воедно ќе се разгледа каков мозаик се добива при имплементирање на одреден модел.

Моделите кои се креирани за обработка и добивање на мозаик, се засноваат на неколку различни принципи. Ќе биде разгледано како бројот на пиксели влијае на процесирањето и креирањето на мозаик, како истото влијае на времетраењето на процесот, како може врз основа на користење на хистограми да се донесе одредена одлука, како може одредена карактеристика на нештата да влијае на донесувањето на правилна одлука, како може да се инкорпорира модел за детекција на лица при креирање на мозаик, и како може врз основа на процесирање на различни региони за дадена слика да бидат имплементирани својства со кои може да се направи дистинкција између два соседни региони. (пример лице и позадина за дадена слика)

Врз основа на ова ќе бидат разгледани неколку алгоритми преточени во модели за обработка и креирање на мозаик и тоа:

I. ***Модел 1*** : Определување на средна вредност за RGB на сегмент од сликата примерок ***врз основа на добиените вредности од сите пиксели*** од дадениот сегмент и споредба на добиените вредности за RGB између сегментите на сликата со тие од базата со слики.

II. ***Модел 2***: Определување на средна вредност за RGB на сегмент од сликата примерок ***врз основа на добиените вредности од само дел пиксели*** кои го сочуниваат дадениот сегмент со користење на функцијата random од библиотеката numpy (Овој начин воедно претставува оптимален начин од моделот 1) и споредба на добиените вредности за RGB између сегментите на сликата со тие од базата со слики.

III. ***Модел 3***: Графичко претставување на податоците со помош на хистограми и споредба на вредности за истите.

IV. **Модел 4:** Одредување на најзастапената (доминантна) боја кај сегментите на сликата и сликите од база, и нивна споредба.

V. **Модел 5:** Креирање на мозаик со имплементација на модел за препознавање на лица

VI. **Модел 6:** Надградба на моделот 5 со тоа што ќе се мапираат региони од дадениот примерок и за истите ќе се приспособува големината на сегментите и истото ќе биде изведено на два начина:

6.1 Динамички пристап на подесување на сегментите од регион за креираниот мозаик

6.2 Фиксно подесување на големината на сегментите од регион за креираниот мозаик

Важно е да се напомене дека за да дојде со совпаѓање на сегментот со некоја слика од базата која ја користиме треба нивните средни RGB вредности да се совпаѓаат.

За сите овие модели, се користи JSON фајл со податоци од дадена база на слики при што се олеснува процесот на обработка на истите откако еднаш ќе бидат обработени.

Како слики за споредба и креирање на мозаик се користат две база со слики, Pool од самиот Python и Images како база со различни слики.

Преку работа со овие алгоритми, ќе забележиме дека тие имаат клучни разлики во начинот на работа, а со тоа и продуцираат мозаици со различен квалитет, кој подобар кој полош, во различен временски интервал, кој побрз кој побавен. На крајот, ќе направиме анализа и споредба на перформансите на овие алгоритми, извлекувајќи заклучок кој од нив е подобар избор во одредени случаи.

Сликата која ќе биде предмет за креирање на мозаик

1. МОДЕЛ 1

Алгоритамот под име „Модел 1“ работи на принцип со споредување на блокови/сегменти од слика како и другите алгоритми во продолжение на оваа семинарска работа. Разликата помеѓу алгоритмите/моделите е во начинот на споредба на карактеристики на сегментите, односно излекување на одредени специфични вредности врз основа на критериумот со кои тие се одбрани.

Првиот и основен модел е наједноставен, каде што за секој сегмент се извлекуваат/процесираат вредности за RGB од пиксели кој претставуваат градбени единки на сегментот и притоа се одредува средната RGB вредност за даден сегмент. Таа вредност потоа се споредува со низа вредности од база со слики со кои ќе се креира даден мозаик. Определувањето на средната RGB вредност за дадена слика од база оди на ист начин како и за сегментот од сликата од која треба да се изработи мозаикот.

Целта е да се избере онаа слика од базата која има иста или најблиска вредност со вредноста на самиот сегмент од сликата.

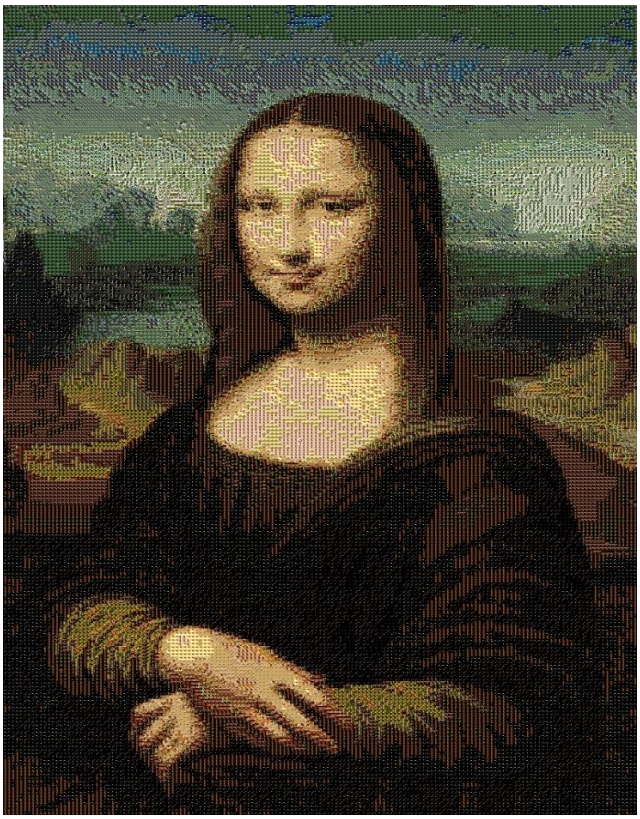
Имајќи го тоа во предвид, основната слика ја делиме на сегменти со еднаква фиксна големина, кои понатаму ќе служат за споредба со сликите од базата со кои ќе се пополнува гридот. Споредувањето на сликите со користење на евклидово растојание. Големината на сегментите влијае врз квалитетот на целата слика, но исто така и врз процесирачката моќ за извлекување на средни RGB вредности на сегментите, споредба на истите и крајно креирање на комплексен мозаик. Поради тие причини во овој алгоритам како и во другите се испитува како големината на сегментот ќе влијае врз крајниот квалитет на сликата и потребното време за да се креира истата. Бидејќи е потребно подолго време за комплетирање на сите операции при процесирање и добивање на мозаик, податоците за RGB вредностите за сликите од база се чуваат во JSON фајл.

1.1 Начин на извлекување на средна RGB вредност за дадена слика/сегмент

Најпрво, потребно е да се изминат ***сите пиксели во дадениот сегмент***, да се сумираат RGB вредностите за секој пиксел и да се најде нивната средна вредност (за поефикасна итерација на пикселите се користи библиотеката numpy). Добиената торка вредности се заокружува на најблискиот цел број. Во овој алгоритам за грид со N x N пиксели, потребно е да се изминат сите пиксели, од каде произлегува дека комплексноста е $O(n^2)$, што значи за големи слики ќе биде потребно значително поголемо време за процесирање.

1.2 Избор на слика за сегмент

Со користење на горенаведената функција добиваме вредности за средна RGB вредност за сите сегменти од гридот како и сите слики од базата со чија помош ја одбираме најсоодветната слика за дадениот сегмент. Тоа се прави со споредба на вредностите користејќи Евклидово растојание т.е наоѓање најблиска вредност со онаа на сегментот. Комплексноста на овој алгоритам е O(к), каде што к е број на слики во базата.

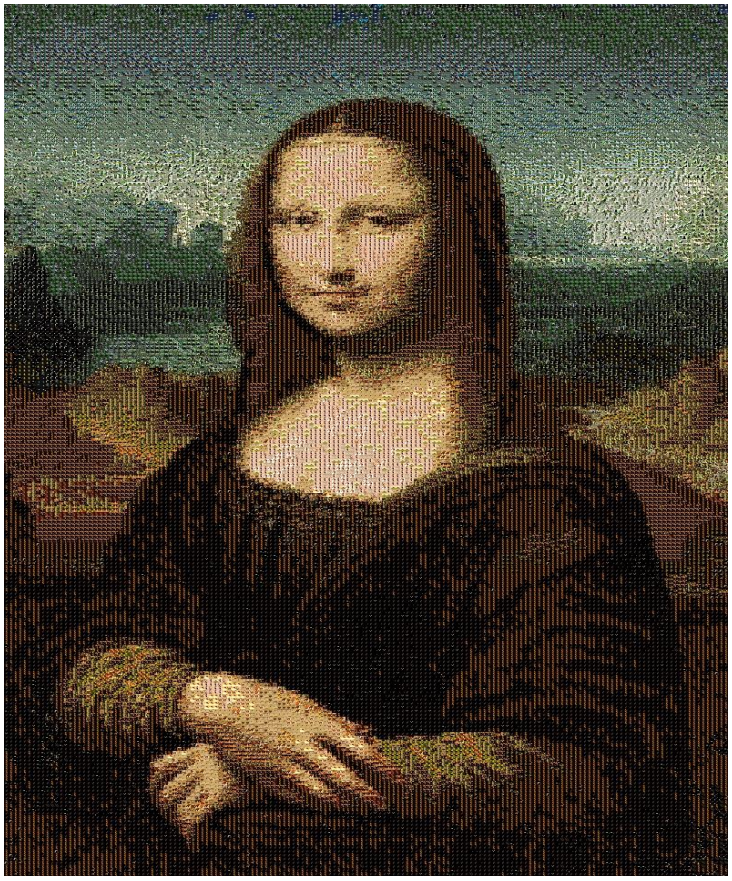


Модел 1

**Како краен резултат е добивање на мозаик со подобар квалитет*

2. МОДЕЛ 2

„Модел 2“ е оптимизирана верзија на “Модел 1“, преку определување на RGB вредностите за дадена торка со ***избирање на случајни пиксели*** од даден сегмент/слика. Во горенаведениот случај Евклидовото растојание за пресметка на вредноста за RGB го користевме најпрво за процесирање на сите пиксели од сегментот, па потоа се бараше најдоброто совпаѓање од сликите од базата. Во овој пооптимален случај се користи функцијата random од numpy со цел да се одберат само одреден дел на пиксели по случаен избор од сегментот и врз основа на нивните вредности за RGB да се најде средната вредност за дадениот сегмент. Па така се заштедува време и ресурси бидејќи врз база на овој пристап не се проверуваат сите пиксели за даден сегмент. Овој случај може да се користи за слики со поголема резолуција при што процесирање на сликата би одело побрзо бидејќи не се процесираат сите пиксели туку само група на пиксели но во исто време како резултат се добива мозаик со нешто послаб квалитет.



Модел 2

**Крајниот резултат во овој случај би бил добивање на мозаик со помал квалитет*

.

3. МОДЕЛ 3

3.1 Графичко претставување на податоци и користење на хистограми

Во овој случај, за процесирање на сликите и податоците за боја на пикселите се користат визуелни графички прикази (хистограми). За секоја од RGB вредностите имаме посебен график, при што од визуелен аспект, е полесно да се увидат областите на интензитет во кој еден пиксел припаѓа и со тоа да се направи еден вид на групирање на самите пиксели. Дополнителен услов кој што е од голема помош е и нормирањето, како услов доколку користиме слики со различна големина при одредена селекција. Поголемите слики имаат поголем број на пиксели и оттука се добиваат различни вредности кои што не се од полза во овој модел. Поради тоа користиме нормирање заради споредбата и совпаѓањето на одредена слика од база со сегмент од главната слика.

3.2 Споредба на вредности кај хистограми

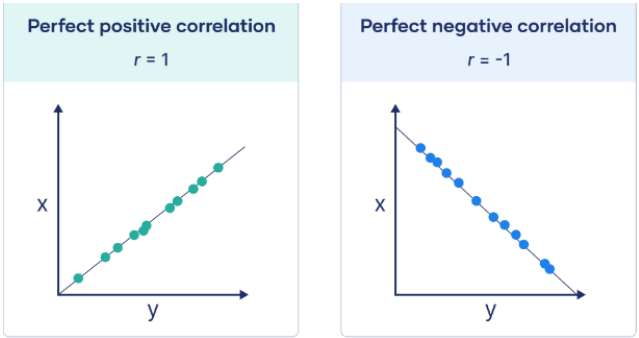
Пеарсонов коефициент на корелација

$$r = \frac{n(\sum xy) - (\sum x)(\sum y)}{\sqrt{[n\sum x^2 - (\sum x)^2][n\sum y^2 - (\sum y)^2]}}$$

Особености со кои се карактеризира оваа релација:

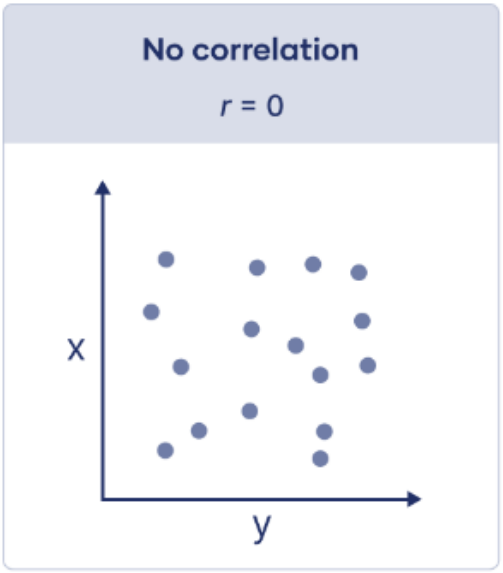
- Коефициентот е секогаш во граници помеѓу -1 и 1

Ако вредноста на коефициентот по апсолутна вредност е 1, тогаш помеѓу X и Y постои линеарна зависност. (слика)



Графичко претставување на коефицинетот на корелација во зависност од максималната и минималната вредност

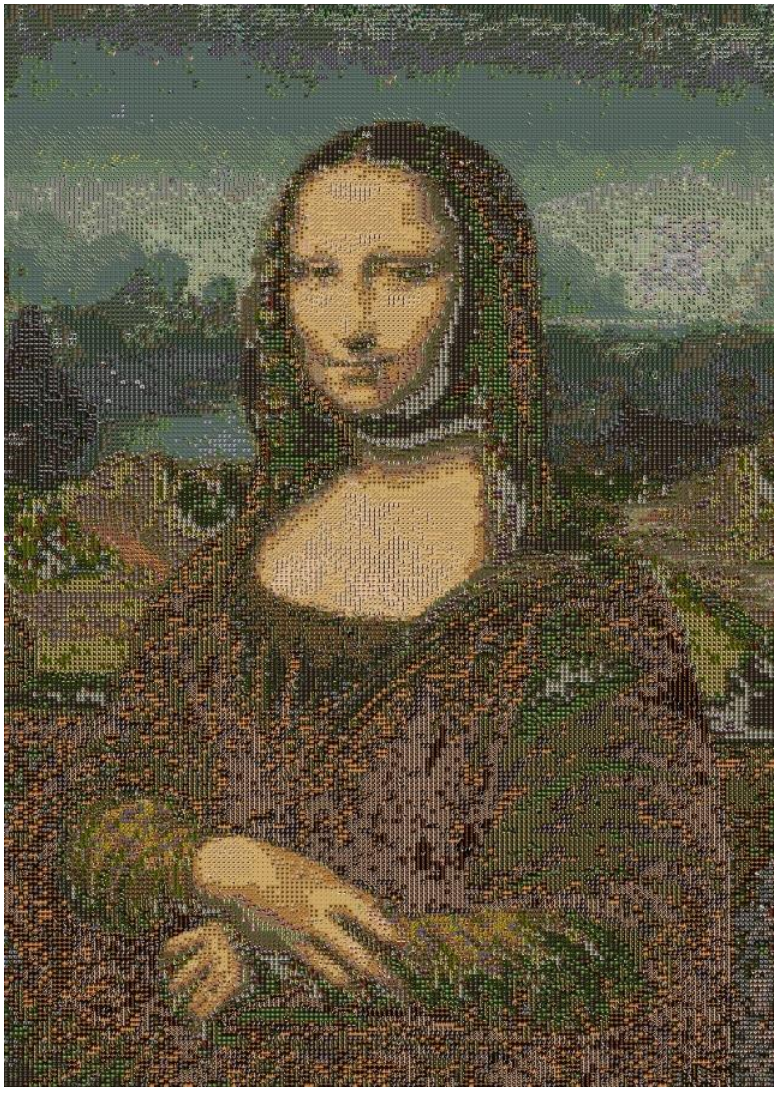
- Од тука следува дека коефициентот на корелација е мерка за линеарната зависност помеѓу X и Y како променливи
- Колку тој коефициент е поблиску до 1, толку линеарната зависност помеѓу X и Y е посилна.
- Кога коефициентот на корелација се доближува до 0, тогаш X и Y се некорелирани. (слика)



Графичко претставување на коефициентот на корелација за вредност 0

Врз основа на вредноста на коефициентот на корелација, вредноста -1 е индикатор дека двата хистограми воопшто не се совпаѓаат, 0 дека нема никаква корелација меѓу распределбите на интензитетите на RGB вредностите, а 1 е индикатор дека двата хистограми апсолутно се совпаѓаат. При имплементација на овој модел, најпрво за секоја слика од базата се добиваат 3 коефициенти на корелациија, соодветно за секоја RGB вредност. Потоа се бара аритметичка средина од добиените коефициенти за дадена слика, од каде што се добива конечниот коефициентот на корелација за секоја слика од базата.

Следно, овие коефициенти од базата со слики се споредуваат со исти такви коефициенти од сегменти/блокови од главната слика од која правиме мозаик. Доколку добиваме совпаѓање, сликата од база се процесира и заменува со сегментот.

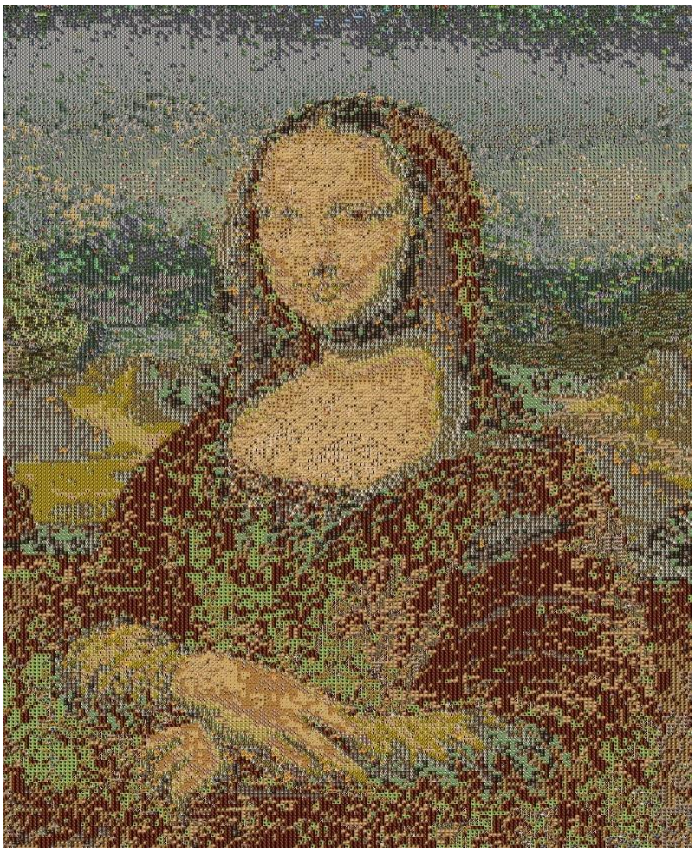


Резултат при користење на хистограм како модел за креирање мозаик

4. МОДЕЛ 4

4.1 Начин на одредување на најзастапена(доминантна) боја кај слика

Сликите во боја обично се во RGB или BGR формат. Бидејќи работиме во Python, а притоа користиме библиотека како OpenCV која сликите ги обработува во RGB формат потребно е да се направи конверзија бидејќи во процесот на мапирање на торките од пикселите за да се осигураме дека користиме ист формат за сите пиксели (сите се во RGB формат, нема пиксели во BGR формат). Во овој случај важно е да се земе во обзир дека сликите од база претставуваат 3Д низи па за нивно полесно процесирање и мапирање се претвараат во 2Д низи. Ова го правиме затоа што многу полесно е да се работи со вектори (2Д низи) отколку со 3Д структури (3Д низи). Откако тоа ќе се изврши, преку бројач се пребројува која од дадените торки е најзастапена. Доколку најзастапената торка за RGB за дадена слика од база се совпаѓа со сегмент/блок од главната слика (модел 1) истата може да се вклучи во креирање на мозаикот.



Модел 4

*процесот на процесирање на главната слика за делење и обработка по сегменти е исти како и во модел

5. МОДЕЛ 5

5.1 Креирање на мозаик од лица со помош на модел за препознавање на лица

Принцип на работа на алгоритамот Haar Cascade Classifier

Процесот на детекција на лица од сликата започнува со конвертирање на BGR боите од сликата (BGR бидејќи работиме со библиотеката OpenCV) во сива варијанта. Ова се прави со цел алгоритамот за детекција на слики побрзо и поефикасно да ги анализира контурите и другите детали поврзани со лицата без притоа да се фокусира на бојата.

При работа со овој алгоритам треба да се обрне внимание на влијанието на неколку фактори.

5.2 Фактор на големина (scaleFactor)

Овој фактор е значаен бидејќи лицата на сликата може да се наоѓаат на различни растојанија од камерата што резултира со различни големини на истите. Поради тоа, со помош на овој фактор алгоритмот ја процесира сликата повеќе пати, почнувајќи од нејзината оригинална големина за потоа истата да биде намалена за одреден процент во зависност од вредноста која ја задаваме (може да варира од 1.01 па се до над 1.5).

Исто така, вредноста на овој фактор е причина за тоа со колкава брзина и точност ќе биде процесирана сликата, како и елементите на неа (во овој случај лица). Ова значи дека доколку користиме коефициенти со помала вредност, дадената слика се процесира повеќе пати, што резултира со поголема точност во однос на детектирањето на сите лица прикажани на сликата, односно грешката моделот да не препознае одредено лице од сликата драстично се намалува. Во спротивен случај, доколку се зададе коефициент со повисока вредност, моделот сликата ќе ја процесира помалку пати што во овој случај би резултирало со поголема веројатност моделот да не препознае одредено лице на самата слика.

5.3 Фактор на потврда (minNeighbors)

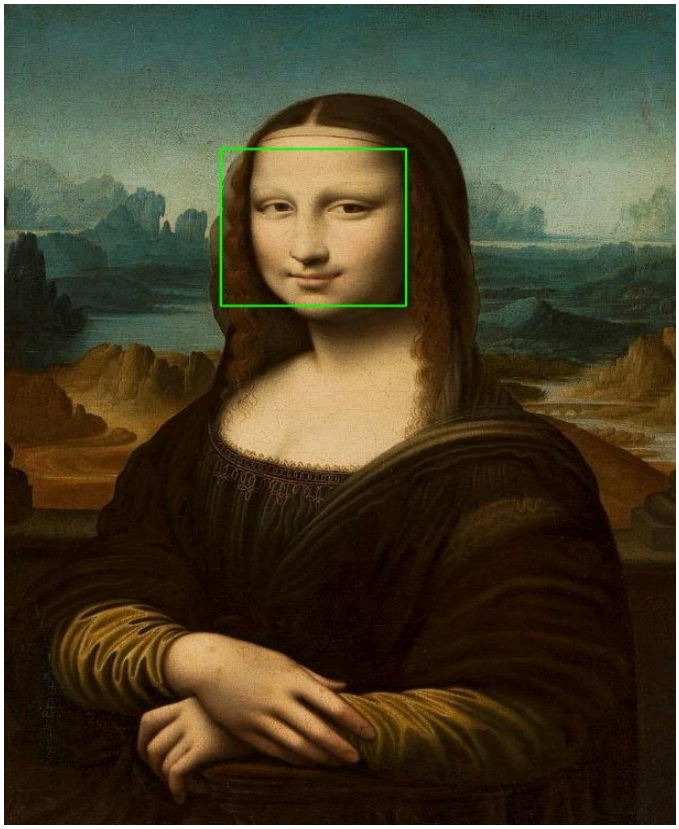
При процесирање на сликата, алгоритмот селектира одредени региони од сликата (во вид на прозорчина) со цел детекција на лицата кои се наоѓаат на истата. Значењето на овој коефициент е тоа што за едно лице да може да биде потврдено за такво од страна на моделот истото треба да биде препознаено повеќе пати (во таканаречените прозорци/региони лицето да биде детектирано во што е можно повеќе такви прозорци при процесот на детекција). Што значи ние одбираме според зададена вредност која се движи од 1 па нагоре колку пати е потребно моделот да го детектира тоа лице за истото биде потврдено како такво. Ако

користиме помали вредности процесот на обработка би бил побрз, значи не ни треба голема проверка за детектирањето но може и да се соочиме и со некоја позитива што значи намалена точност за детекција. Доколку се одберат повисоки вредности го добиваме спротивното.

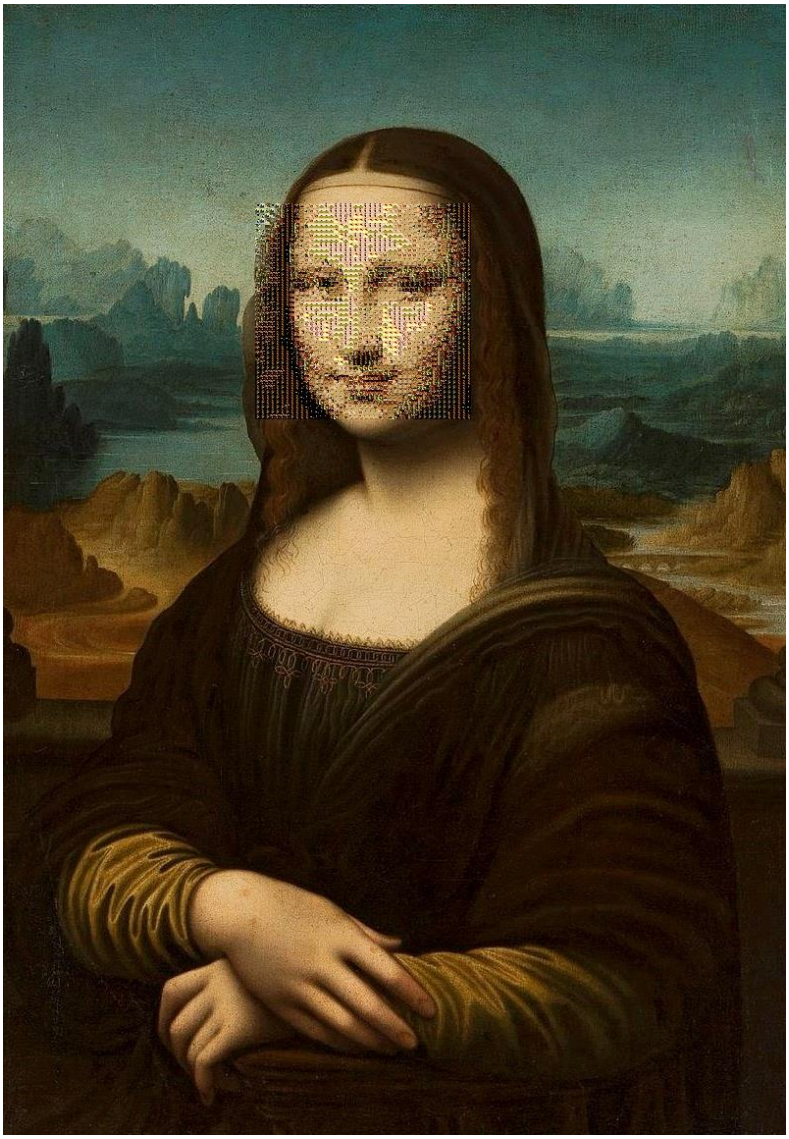
5.4 Минимална големина (minSize)

Овој фактор го ограничува алгоритмот со колкава минимална големина треба да се детектираат лицата. Пред се овој фактор служи за да се избегне детекција на многу мали објекти за да не дојде до забуна кај самиот модел.

Откако лицето од сликата ќе биде детектирано, следува процес на сегментација на истото (во овој случај се фокусираме само на лицето претставено како засебна слика без позадината од оригиналната слика) при што се користи истиот принцип за креирање на мозаик опишан во модел 1.



Резултат при користење на моделот за препознавање на лица



Креирање на мозаик на лице со помош на алгоритам за препознавање на лица

6. МОДЕЛ 6

Принципот на работа на овој алгоритам може да се каже дека е комбинација од претходните модели.

Целта е да се придаде поголема важност на тоа како се пополнуваат различни региони со сегменти/слики кои ги приспособуваме така што ќе направиме разграничувањето измеѓу два региони да биде токму големината на сегментите како главен фактор.

] Овој пат сакаме да креираме мозаик така што лицето од сликата ќе биде составено од региони кои пак се направени од сегменти/слики со помала големина (лице) и позадина која треба да биде составена од сегменти со поголема големина. Со цел полесна детекција на лицата се користи гаусов филтер. Со негова помош доаѓа до отстранување на таканаречените шумови (делови од сликата кои можат да попречат во дектирањето на објекти како што е лице).

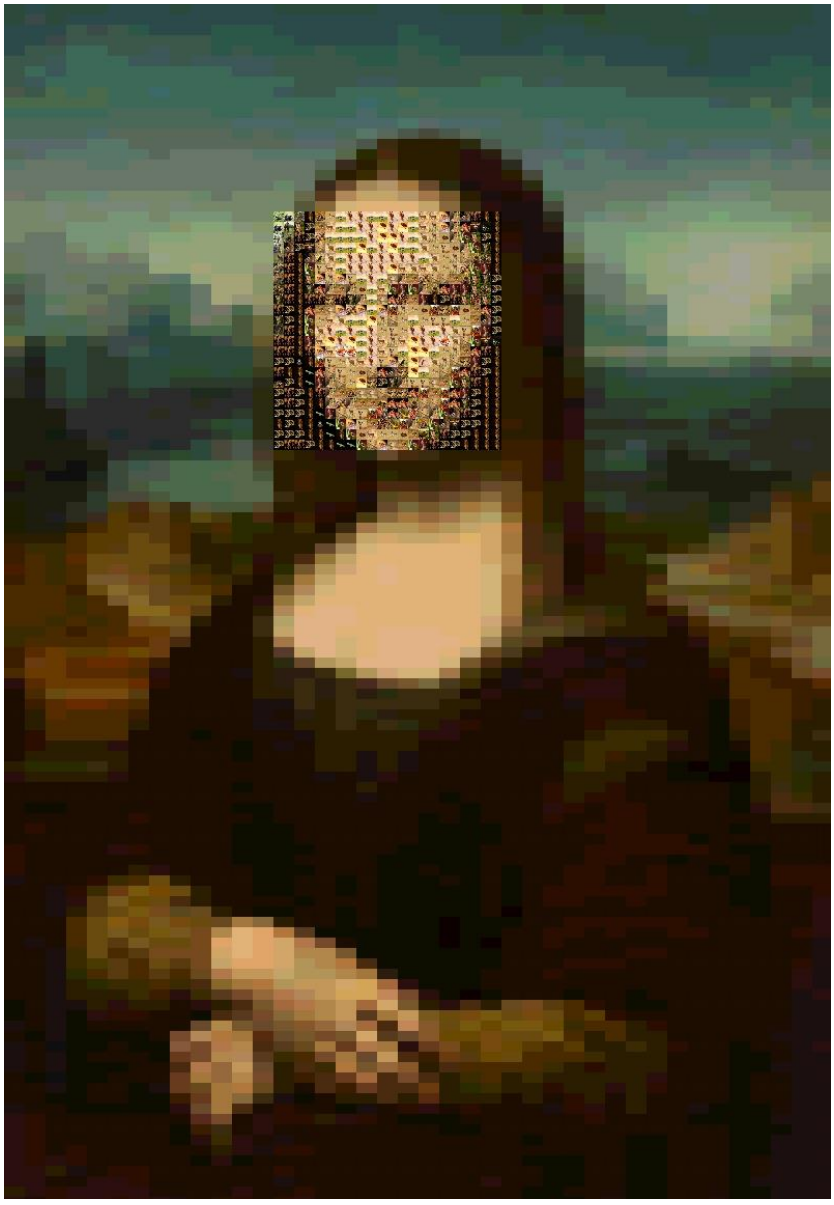
Разликата измеѓу долунаведените случаи е во пополнувањето со слики/сегменти измеѓу региони кои се содржат во еден ред:

6.1 Прв пристап

Откако успешно ќе го детектираме лицето со помош на моделот за детекција (модел 5.1) потребно е да се искористи маска која претставува бинарен формат со што позадината се прикажува во црна боја а лицето во бела боја. Ова се прави со цел за разделување на областите кои се цел на обработка а со тоа се олеснува манипулирањето со истите.

Во овој таканаречен прв пристап, процесирањето и вметнувањето на сликите на местата од сегментите/блокови со одредена големина се одвива на следниот начин: во еден ред од сликата, кој е составен од сегменти на позадината и лицето, пополнувањето оди така што иако имаме одредено точно со колкава големина треба да се сликите за позадината од една страна и лицето од друга страна, овој алгоритам гледа пополнувањето да се изврши на еден оптимален начин така што сликите во тој ред би имале иста висина но различна ширина. Ова се прави затоа што доколку зададените параметри за сликите не го задоволуваат условот за целосно исполнување на даден ред, големината на еден начин повторно се менува од страна на овој алгоритам така што добиваме целокупен ред составен од слики/сегменти. Ова претставува таканаречен динамички начин за пополнувањето на мозаик со слики доколку зададеме различна големина на сликите кои не одговара на форматот за дадена слика од која се креира мозаик.

Па така на пример доколку ја зададеме висина и ширина на сликите за даден регион во случајот позадина на 20x20 а за лицето користиме слики во формат 5x5 тогаш кога настанува пополнување во редот каде имаме поклопување на двата региони, овој алгоритам повторно ја подесува големината на сликите овој пат од позадината на 5x20 на сликите од позадината со цел динамичко пополнување на даденот ред.

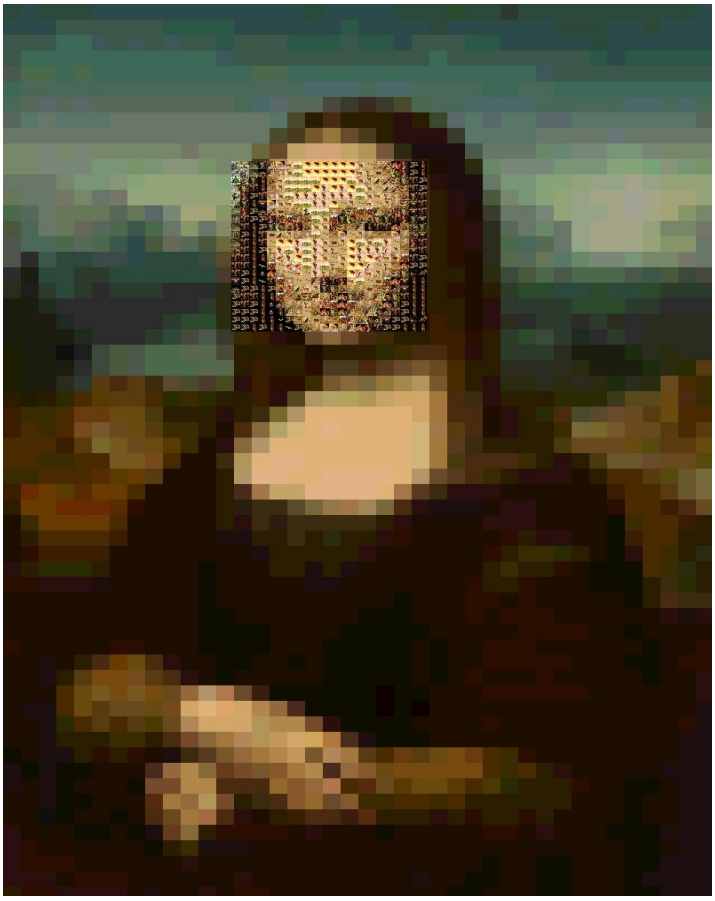


Резултат од првиот пристап

6.2 Втор пристап

Во овој пристап исто како и во првиот, пополнувањето и споредувањето на сликите се прави на ист начин како и во моделот 1 (пресметување на средна вредност на торката за RGB и вредност за евклидовото растојание) и воедно се користи истиот модел за детекција како и порано, но главната разлика тука во однос на првиот пристап е во пополнувањето на регионите со слики со различна зададена големина. Имено овде не се прави динамичко подесување на веќе зададените големини за сегментите/блокови за да се пополни дадена редица, туку пополнувањето со сликите оди во два посебни процеси со фиксните големини кои сме ги задале за даден регион. Во овој случај еден за лицето со помали сегменти/слики и уште еден за позадината на сликата со поголеми сегменти/слики.

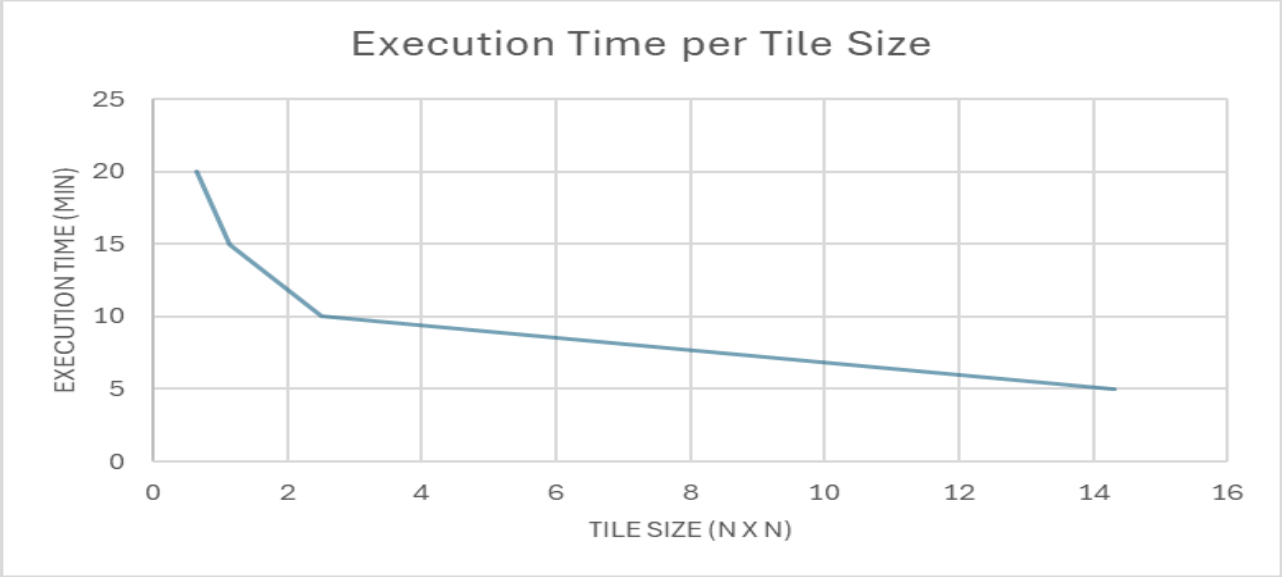
Овој алгоритам е поприкладен доколку сакаме да креираме мозаик од сегменти кои служат како граница измеѓу регионите за дадена слика.



Резултат од вториот пристап

ЗАКЛУЧОК

На следниот график е прикажано како големината на сегментите со кои се дели оригиналната сликата влијае врз времетраењето на процесирање на сликата.



Како што може да се види, колку што сегментите што ги избираме се помали толку повеќе време е потребно да се процесира сликата. Додека во обратен случај, колку сегментите се поголеми толку времето на процесирање е побрзо.

Но исто така, треба да се наведе дека кога избираме помали сегменти, добиваме подобри резултати во однос на квалитетот на сликата, за разлика од тоа кога избираме поголеми сегменти кога квалитетот на сликата е полош.

Како заклучок од оваа проектна задача, може да изведеме дека изборот на алгоритмот, како и некои други карактеристики, има влијание врз перформансите на сликата.

Не можеме да кажеме дека постои идеален алгоритам за креирање на мозаик. Кај некои алгоритми добиваме квалитет на сликата, но губиме време. Кај други, бенефитите од кратко време на процесирање ја плаќаат цената на полош квалитет на сликата.

Важно е да знаеме во одреден случај кои перформански сакаме да ги истакнеме а кои да ги жртвуваме. Знаејќи го тоа, горенаведените алгоритми даваат задоволителни резултати во однос на нашите очекувања.