Nume: Luta Vlad Cristian
Grupa: 341A2

Tema 1

Keras-cv:

Keras-cv este un model generativ de imagini bazat pe ce se intitulează în limba engleză “Stable Diffusion”. În decursul proiectului, am folosit aceasta rețea generativă text-to-image pentru generarea unor imagini, pe baza cărora am realizat procesarea ulterioară.

Aceasta foloseste ca input un prompt (descriere sub forma de text a ce dorim să transpunem în imagini) și o serie de parametrii auxiliari (număr de iterații, număr de imagini). De asmenea, ca back-end am folosit Tensorflow și am realizat rularea script-ului pe GPU, ceea ce a determinat timpi de execuție semnificativ mai mici.

Limitările modelului au inceput sa reiasă în momentul în care am încercat să generăm lucruri de culori care în mod normal nu se regăsesc, spre exemplu: morcovi maronii. Am observat ca rețeaua nu prelua aproape sub nicio forma informația referitoare la culoare, iar în urma mai multor încercări, nu putem spune că a generat 100% culoarea dorită, ci nuanțe de portocaliu spre roșu sau maroniu spre galben deschis.

O posibilă explicație ar fi faptul că portocaliul se apropie foarte mult de maroniu ca nuanță, într-un cât am generat și morcovi de culoare mov, iar acest prompt a dat rezultate mai fidele conform textului. De asemenea, nu am primit output dorit în funcție de numărul de obiecte pe care am dorit să le conțina o imagine.

Yolo V5:

Yolo este o rețea de detecție preantrenată, utilă pentru detecția obiectelor din imagini, dar și din input provenit de la camera sau fișiere video. De asemenea, aceasta poate fi antrenată la alegere pe un anumit tip de obiect conținut într-un dataset personalizat.

Inferența poate fi rulată încă din linie de comandă împreună cu o serie de parametrii, dintre care putem menționa weight-urile, dar și ajustarea valorii de threshold, ce reprezintă sensibilitatea rețelei. Cu cât threshold-ul este mai mare, cu atât rețeaua oferă clasificări de o acuratețe mai sporită, dar identifică mai greu obiectele, iar cu cât threshold-ul este mai mic, rețeaua identifică mai usor obiectele, dar scade acuratețea clasificării obiectului.

Pentru acest proiect, am realizat detecția cu valori ale threshold-ului nu mai mari de cea default, adica 0,25. Rezultatele au fost mulțumitoare în mare parte, dar se poate observa din imaginile obținute că anumite clasificări sunt eronate și în imaginile ce conțin mai multe obiecte, anumite obiecte de interes nu sunt detectate.

O soluție pentru îmbunătațirea rezultatelor pe imaginile generate poate fi antrenarea rețelei cu un set personalizat, generat cu Keras-cv.

Nume: Luta Vlad Cristian
Grupa: 341A2

Spațiul de culoare HSV:

Spațiul de culoare HLS (Hue, Lightness, Saturation) este un sistem de reprezentare a culorilor bazat pe trei componente principale: nuanța (Hue), luminozitatea (Lightness) și saturația (Saturation).

Pentru a converti un sistem de culoare RGB (Red, Green, Blue) în sistemul de culoare HLS (Hue, Lightness, Saturation), se pot urma următorii pași:

Normalizarea valorilor de roșu, verde și albastru: înainte de a efectua conversia, valorile roșu (R), verde (G) și albastru (B) trebuiesc normalizate în intervalul [0, 1]. Dacă valorile sunt exprimate în intervalul [0, 255], trebuiesc împărțite fiecare la 255 pentru a le aduce în intervalul corect.

Calculul valorii de luminanță (Lightness): se identifică valoarea minimă și valoarea maximă dintre R, G și B. Lightness (L) este calculat ca medie între valorile minimă și maximă: L = (min(R, G, B) + max(R, G, B)) / 2.

Calculul valorii de saturație (Saturation): dacă L este 0 sau 1, atunci saturația este 0. Altfel, se calculează saturația (S) astfel: se calculează valoarea diferenței dintre valoarea maximă și valoarea minimă: diff = max(R, G, B) - min(R, G, B), apoi se calculează saturația folosind formula: S = diff / (1 - |2L - 1|).

Calculul valorii de nuanță (Hue): dacă diff este 0, atunci nuanța (H) este 0. În caz contrar, se calculează nuanța astfel: pentru fiecare valoare (R, G, B), se calculează diferența dintre valoarea maximă și valoarea minimă împărțită la diff. Dacă valoarea maximă este R, se calculează nuanța ca: H = (G - B) / diff. Dacă valoarea maximă este G, se calculează nuanța ca: H = 2 + (B - R) / diff. Dacă valoarea maximă este B, se calculează nuanța ca: H = 4 + (R - G) / diff.

În final, se normalizează nuanța astfel încât să fie între 0 și 360 de grade: H = H * 60 sau H = H + 360 (H este negativ).

Aplicarea măștii și transformarea într-o imagine fără fundal:

Pentru crearea măștii se aleg doi vectori: high si low. Ambii vectori reprezintă limita superioară, respectiv inferioară a culorii pixelilor din spațiul RGB ce urmează a fi selectați pentru a crea masca. Se aplică o operație NOT logic pentru a ramâne cu pixelii din interiorul siluetei obiectului țintă setați pe 1 logic (255,255,255), si cei din afara siluetei pe 0 logic (0,0,0).

Pe urma, aplicăm masca imaginii printr-o operatie de AND logic pe biții dintre imaginea initiala si masca care urmează a fi aplicată. În urma acestei operații rămân neschimbați doar pixelii care se află în interiorul siluetei ce dorim sa o mascam și background-ul de culoare neagră (0,0,0).

În final, pentru a transforma background-ul din negru în alb, iterăm prin fiecare pixel al imagii anterioare și pentru fiecare pixel care are valoarea (0,0,0), setăm fiecare canal RGB de la valoarea 0 la 255.