Nume: Sociu Daniel
Grupa: 342

Documentatie Simpson face detection si recognition

Acest proiect primeste niste imagini din serialul animat Simpsons, iar pentru task-ul unu trebuie sa detectate fetele oricaror personaje aflate in imagine, iar pentru task-ul doi trebuie identificate toate aparitiile personajelor principale: Bart, Homer, Lisa, Marge, si specificat exact numele fiecarii identificari.

Primul task foloseste un sliding window, care variaza in marime si intaltime (modific marimea imaginii pentru a obtine window-uri de marimi si proportii diferite, aceasta fiind initial 36x36, cu ajutorul a 2 variabile: params.image_initial_scale si params.image_minimize_scale care reprezinta cu cat e imaginea marita initial, si dupa cu cat la suta scade la fiecare iteratie, pana cand nu mai incape un window in aceasta imagine). Window-ul are tot timpul size-ul 36x36 pentru a putea prezice daca exista o fata sau nu, avand modelele antrenate pe window-uri de 36x36, dar pentru a obtine bounding box-ul din imaginea initala, trebuie recalculat indexii corespunzatori:

```
actual_zoom = 1 / (
        self.params.image_initial_scale * (self.params.image_minimize_scale ** power))
x_min = int((x * self.params.dim_hog_cell / fx) * actual_zoom)
y_min = int((y * self.params.dim_hog_cell / fy) * actual_zoom)
x_max = int(
    ((x * self.params.dim_hog_cell + self.params.dim_window) / fx) * actual_zoom)
y_max = int(
    ((y * self.params.dim_hog_cell + self.params.dim_window) / fy) * actual_zoom)
```

Trimit acest sliding window catre model, model selectat din parametrii, si obtin scorul acestuia.
Mai am un parametru care influenteaza daca imaginea este sau nu considerata o detectie corecta, si anume cat la suta din imagine are culoarea galbena, mai exact folosesc un interval din HSV, intre 20 si 85 hue value, care imi accepta in principal orice nuanta de galben, chiar si galben-verziu din urma umbrii zonei unde se afla personajul.

Daca imaginea mea are un scor mai mare ca threshold-ul stabilit, de asemenea mai mult galben decat minimul necesar setat din parametrii, atunci imaginea este o posibila detectie, si este adaugata listei de detectii. Aceasta lista este mai departe verificata de overlapping windows cu functia non_maximal_supression facuta in principal in laborator.
Acest fiind ultimul pas, imaginile care au trecut de ultimul filtru, sunt considerate detectii.

Ca modele de antrenare, sunt 3 care antreneaza imagini 36x36:

- SVM care primeste HOG-ul imaginilor: ~0.62

```
best_accuracy = 0
best_c = 0
best_model = None
Cs = [10 ** -5, 10 ** -4, 10 ** -3, 10 ** -2, 10 ** -1, 10 ** 0]
for c in Cs:
    print('Antrenam un clasificator pentru c=%f' % c)
    model = LinearSVC(C=c, penalty='l2', loss='squared_hinge', dual=False, tol=1e-4,
                      multi_class='ovr', fit_intercept=True, intercept_scaling=2, verbose=True)
    model.fit(training_examples, train_labels)
    acc = model.score(training_examples, train_labels)
    if acc > best_accuracy:
        best_accuracy = acc
        best_c = c
        best_model = deepcopy(model)
```

- CNN care primeste HOG cu hinge loss (SVM imitation): ~0.782

```
neural_model = keras.Sequential()
# neural_model.add(keras.Input(batch_input_shape=(64, 1296)))
neural_model.add(layers.Dense(128, input_shape=(len(training_examples[0]),), activation='relu'))
neural_model.add(layers.Dense(64, activation='relu'))
neural_model.add(layers.Dense(1, kernel_regularizer=regularizers.l2(0.01)))
neural_model.add(layers.Activation('linear'))
neural_model.summary()

neural_model.compile(loss='hinge',
                     optimizer='adam',
                     metrics=['accuracy'])

early_stopping = keras.callbacks.EarlyStopping(
    min_delta=0.01,
    patience=10,
    restore_best_weights=True
)

result = neural_model.fit(
    x_train,
    y_train,
    validation_data=(x_test, y_test),
    epochs=20,
    # shuffle=True,
    verbose=True,
    use_multiprocessing=True,
    callbacks=[early_stopping]
)
```

- CNN care primeste imagini colore 36x36x3: ~0.75

```
neural_model = keras.Sequential([
    layers.Input(shape=(36, 36, 3)),
    layers.Conv2D(16, kernel_size=5, padding="same", activation="relu"),
    layers.MaxPooling2D(pool_size=2, strides=2),
    layers.Dropout(0.2),
    layers.Conv2D(32, kernel_size=5, padding="same", activation="relu"),
    layers.MaxPooling2D(pool_size=2, strides=2),
    layers.Dropout(0.3),
    layers.Flatten(),
    layers.Dense(256, activation='relu'),
    layers.Dropout(0.3),
    layers.Dense(128, activation='relu'),
    layers.Dense(1, kernel_regularizer=regularizers.l2(0.01)),
    layers.Activation('linear')
])
neural_model.summary()

adam = keras.optimizers.Adam(
    learning_rate=0.001,
    beta_1=0.9,
    beta_2=0.999,
    epsilon=1e-7,
    amsgrad=False,
    name="Adam"
)

neural_model.compile(loss='hinge',
                     optimizer=adam,
                     metrics=['accuracy'])
```

Modelul se poate selecta cu params.model_used = SVM, CNN_with_HOG, CNN.

Am ales modelul CNN_with_HOG pentru a trimite rezultatele.

Pentru task-ul doi am luat fiecare detectie de la task-ul unu pentru a decide daca este unul dintre cele 4 caractere, sau face parte din a 5-a clasa, unknown, folosing un decoder pentru a trece de la numele clasei la un indice corespunzator.

Am facut un model CNN separat pentru recognition, cu o activare softmax la final care este antrenat pe label-urile din antrenare, de asemenea pe 36x36, predictia acestora fiind encoded inapoi in numele caracterelor, daca e diferita de clasa unknown, si scrise in fisier ca raspuns final.

```
neural_model = keras.Sequential([
    layers.Input(shape=(36, 36, 3)),
    layers.Conv2D(32, kernel_size=5, padding="same", activation="relu"),
    layers.MaxPooling2D(pool_size=2, strides=2),
    layers.Dropout(0.2),
    layers.Conv2D(64, kernel_size=5, padding="same", activation="relu"),
    layers.MaxPooling2D(pool_size=2, strides=2),
    layers.Dropout(0.3),
    layers.Flatten(),
    layers.Dense(128, activation='relu'),
    layers.Dropout(0.3),
    layers.Dense(64, activation='relu'),
    layers.Dense(len(self.params.encoding.keys()), activation="softmax")
])
```

Cel mai bun output obtinut: