Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Інститут прикладного системного аналізу
Кафедра математичних методів системного аналізу

Звіт

З дисципліни “Розпізнавання образів”

Про виконання лабораторної роботи № 3

Виконали:

Студенти 4 курсу

Групи КА-75

Осетров Антон

Федурко Микола

Перевірив:

Лисенко А. В.

Київ 2020

Постановка задачі

Енкодер та декодер (плеєр). Одним з виходів має бутм файлик у нашому форматі, що містить стиснений відеоряд . Другим виходом буде власне відтворення відео з нашого файлу.

Класифікатор. Класифікатор на основі алгоритмів отримання характеристик зображення

Набір даних. Зняти не менше сотні фото предмета, варіюючи його розміщення та ракурс в кадрі, освітлення, наявність візуальних перешкод, зашумленість зображення, фокусну віддаль та тремтіння рук. До цих фото варто додати невелику підбірку зображень, що не містять предмет, або ж містять предмет візуально подібний

Відео для енкодера

Результати роботи класифікаторів. Натренирувати обраний дескриптор для обраного предмета, після чого з його допомогою розпізнати об’єкт на всій тестовій вибірці збираючи при цьому такі дані:

1. кількість помилок 1/2 роду
2. відео з текстом класифікації
3. відносний час роботи енкодера/декодера
4. Візуальні різниці згенерованих та оригінальних зображень.

Аналіз результатів роботи . Дослідити розбіжності у роботі класифікаторів та виконати порівняльний аналіз їх поведінки, сформулювати висновки з вкладками та прикладами (в тому числі з описом виняткових особливостей, сильних та слабких сторін та обгрунтуванням чому вони поводяться саме так)

Головний репозиторій з усіми результатами та кодом: https://github.com/B1Z0N/OpenCVLabs/tree/master/lab2

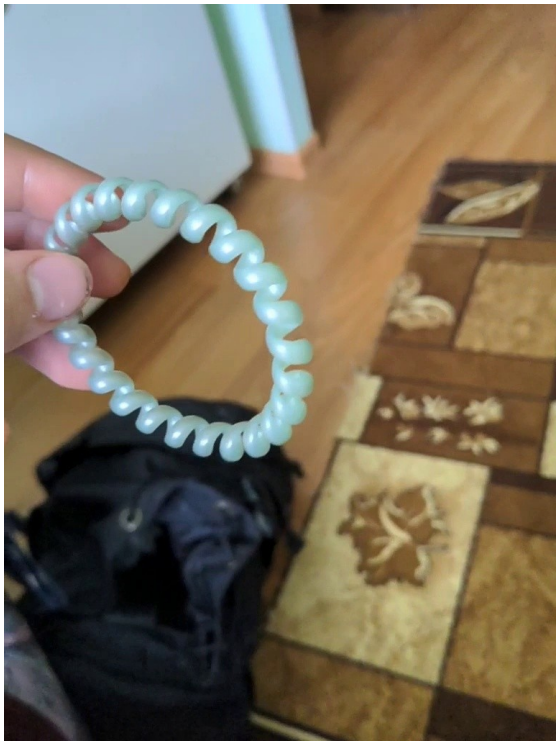
Хід роботи

1. Класифікатори, кодеки та предмет дослідження

В нашій команді з двох чоловіків ми вирішили розглянути такі два класифікатори: SVM на базі дескриптора SIFT та CNN навчену методом backpropagation. Підопитними об’єктами стали улюбені ризинка для волосся та банка пива. Загалом, для цього експерименту були взяті два сети з більш ніж 100 фотографій кожен. На деяких з цих фотографій, об’єктів навмисне немає, деякі ж з фотографій зроблені в поганій якості (розмитість або в темному приміщенні). Також самі фотографії з набору мають різні розміри. Хоча всі фотографії були заздалегідь зменшені для того, щоб програма працювала швидше, а для CNN зменшені до 32x32. Все це було зроблено для того, щоб дослідити поведінку класифікаторів в різних умовах.

Ось які об’єкти були в наших сетах.

Кодеки реалізовані на базі optical flow інтерполяції руху. Використані 2 алгоритма: Franeback та Lucas-Kanade. Відео для кодингу вибране спільне для наглядності результатів



2. Набори даних

Для кожного предмета ми зробили знімки з різними параметрами. Загалом в коді є змінна `OWNERS_DISTINCTIONS`, яка у вигляді списків зручно відображає весь зміст даних:

```
'anton' : {
  'images' : {
    # photos where object is in frame
    'frame' : {
      # ranges of samples
      'original' : ((1, 10),(30,50),(70,110), ),
      'scaled40percents' : ((10, 30), ),
      'scaled200percents' : ((50, 70), ),
    },
```

```
        # out of frame
        'noframe' : {
            # ranges
            'original' : ((111, 120), ),
        },
    },
    'descriptor' : SIFT
},
'nikolay' : {
    'images' : {
        # photos where object is in frame
        'frame' : {
            # ranges of samples
            'original' : ((1, 30), (60, 100), (150, 212), ),
            'scaled60percents' : ((30, 60), ),
            'scaled200percents' : ((100, 150), ),
        },
        # photos with object out of frame
        'noframe' : {
            # ranges
            'original' : ((1, 36), )
        },
    },
    'descriptor' : BRIEF
},
```

В директорії media в репозиторії, посилання на який було подано вище, зображення в форматі jpg мають формат імен: frame56 - зображення з об'єктом, noframe111 - зображення без об’єкту на ньому. Тому відповідно і в словнику `OWNERS_DISTINCTIONS` інтервал (1, 30), що знаходиться в категорії frame & scaled60percents, наприклад, автоматично підтяне знімки frame1-frame30 і буде їх розглядати як знімки з скейлом 0.6 та з зображенням об’єкту на них. Такий формат дуже зручний для роботи програми. І в такому форматі зручно подавати результати.

3. Результати роботи

Класифікатори:

1.CNN

Для нейронної мережі було записано 500 семплів. 350 з них об`єктом, 150 – без.

Архітектура стандартної convolutional network:

```
model = K.Sequential()
model.add(L.Conv2D(16, 3, input_shape=(64, 64, 3), padding = 'same'))
model.add(L.LeakyReLU(0.1))

model.add(L.Conv2D(32, 3, padding = 'same'))
model.add(L.LeakyReLU(0.1))

model.add(L.MaxPooling2D(pool_size=(2, 2)))
model.add(L.Dropout(0.25))

model.add(L.Conv2D(32, 3, padding = 'same'))
model.add(L.LeakyReLU(0.1))

model.add(L.Conv2D(64, 3, padding = 'same'))
```

```
model.add(L.LeakyReLU(0.1))

model.add(L.MaxPooling2D(pool_size=(2, 2)))
model.add(L.Dropout(0.25))

model.add(L.Flatten())
model.add(L.Dense(256))
model.add(L.LeakyReLU(0.1))
model.add(L.Dropout(0.5))

model.add(L.Dense(2))
model.add(L.Activation("softmax"))
```

Тренувальні дані заздалегідь зменшені до розміру 64x64, з метою зменшення кількості обрахунків. Так як при 64x64 форма майже не втрачається результати вийшли непогані.
Для тестування було відібрано 100 прикладів(70/30). На них 11 помилок першого роду т а 0 помилок другого.

2.SVM
Архітектура: SIFT -> BOW -> SVM
Sift описаний в минулий роботі.
BOW це алгоритм який з кластеризує дескриптори методом closest k-means що дозволяє для всіх зображень отримати новий дескриптор однакового розміру, який можна передати в SVM.
На тестовій вибірці в 100 правильних та 50 неправильних: 27 помилок першого роду та 13 другого.

Загалом CNN має великий потенціал при розширенні вибірки та збільшенні часу навчання без вмішання в поточний алгоритм.

Що стосужться SVM, то є сенс дослідити різні дескриптори, які гіпотетично покращать якість.

Кодеки

Обидва кодеки реалізовані з використанням optical flow.
Алгоритм Franeback обраховує зміну положення кожного пікселя, що виражається в більш довгому часі роботи декодера.
Алгоритм Lucas-Kanade знаходить граничні точки об“єктів, та рухає окіл тієї точки зображення в обрахованому напрямку. Алгоритм можна потенцційно покращити додавши рух зв“язних блоків.
Обидва енкодера видаляють кожен другий кадр з відео. А декодер відновлює по сусіднім, що залишились.

Результати

1.відео відновлене з коду
2. в папках differ візуальні різниці між зображеннями
3. відео з текстовим оверлеєм класифікації
4. дані по часу роботи, помилкам, відстанню між зображеннями

4. Висновки та аналіз результатів.

Отже, було проведено роботу з аналізу 4 алгоритмів SVM, CNN, Franeback, Lucas-Kanade. Як бачимо по результатам, ці методи досить різні, мають свої плюси та мінуси. Скоріш всього всі ці алгоритми(окрім хіба що нейронки) не є настільки універсальними, щоб використовувати їх для складних випадків на реальних проектах.

Аналізуючи результати, перш за все бачимо, що, як і очікувалося, на обробку важчих фотографій програма в усіх випадках витрачає більше часу. Порівнюючи час між двома кодаками бачимо, що Franeback спрацьовував значно(в декілька разів) повільніше ніж Lucas-Kanade. Також Franeback дає багато руху на зображені і з“являються розмитості при різких рухах. Lucas-Kanade навпаки реагує тільки на крайові рухи і загалом не звертає уваги на багато інших.

Що стосується класифікаторів CNN гарно класифікує якщо таргетні зображення на схожому фоні та під схожим кутом. І очевидно, що для покращення роботи треба більше ресурсів, різноманітних даних для тренування. Роботу SVM важко передбачити тому, що неочевидно як дескриптор підбирає ключові точки і часто знаходить їх між абсолютно різними об“єктами. Тому є сенс дослідити ринок дескрипторів.

Джерела:

1. Calonder M., Lepetit V., Strecha C., Fua P. (2010) BRIEF: Binary Robust Independent Elementary Features. In: Daniilidis K., Maragos P., Paragios N. (eds) Computer Vision – ECCV 2010. ECCV 2010. Lecture Notes in Computer Science, vol 6314. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-15561-1_56
2. https://gilscvblog.com/2013/08/26/tutorial-on-binary-descriptors-part-1/