Mihailo Vukorep (IN 40/2021)

Препознавање ручно написаног текста  
(Book Archiver)

Извештај за практично истраживање

# Увод

## Предмет истраживања

Предмет овог пројекта се заснива на испитивање шта је све потребно да би раучар препознао ручно написан текст са слике на папиру без коришћења неких већ готових библиотека које своје решење нуде у пар линија кода (нпр. *easyocr*).

## Циљеви истраживања

Сам циљ пројекта јесте архивирање ручно написаних књига у неки текстуални документ зарад презервације истих.

## Задаци истраживања

Можемо задатак за имплементирање оваквог програма поделити у две обимне целине.

Првa целина представља само тренирање довољно добре конволуционе неуронске мреже за препознавање ручно написаног тескста, док другу целину представља издвајање индивидуалних карактера са слике и прослеђивање тих слика истренираној мрежи.

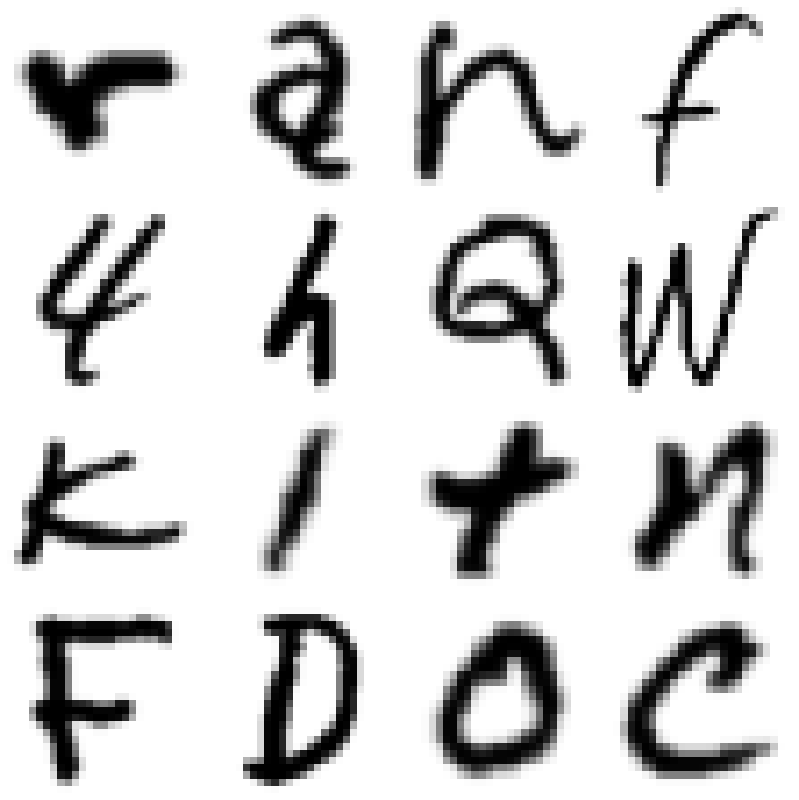
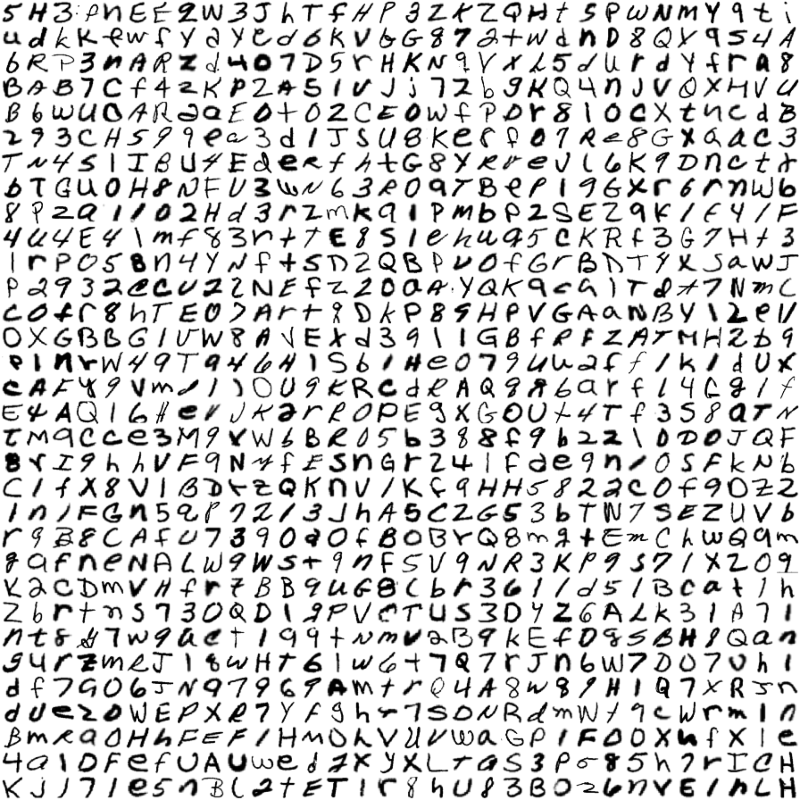
Очекивани резултати истраживања

Због недостатка рачунарске снаге које поседујемо не очекујемо да ћемо имати неке довољно успешне резултате.

# Методологија

## Коришћени подаци

Користимо један од познатијих тренинг скупова под именом EMNIST (Extended MNIST). Његов претходник MNIST је био сачињен само од ручно написаних цифара, док је EMNIST сачињен од цифара, малих и великих слова. Слике су црно-беле и у формату 28\*28 пиксела.



Сами подаци су подељени по категеоријама: ByClass, ByMerge, Balanced, Letters, Digits, MNIST

Такође је свака категорија подељена на тренинг и тест скупове.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Назив категорије** | **Укупан број слика** | **Тренинр (train)** | **Тест (test)** | **Број класа (карактера)** |
| EMNIST ByClass | 814 255 | 697 932 | 116 323 | 62 |
| EMNIST ByMerge | 814 255 | 697 932 | 116 323 | 47 |
| EMNIST Balanced | 131 600 | 112 800 | 18 800 | 47 |
| EMNIST Letters | 145 600 | 124 800 | 20 800 | 26 |
| EMNIST Digits | 280 000 | 240 000 | 40 000 | 10 |
| EMNIST MNIST | 70 000 | 60 000 | 10 000 | 10 |

Пре коришћења самог скупа проверамамо да ли постоје дупликати.

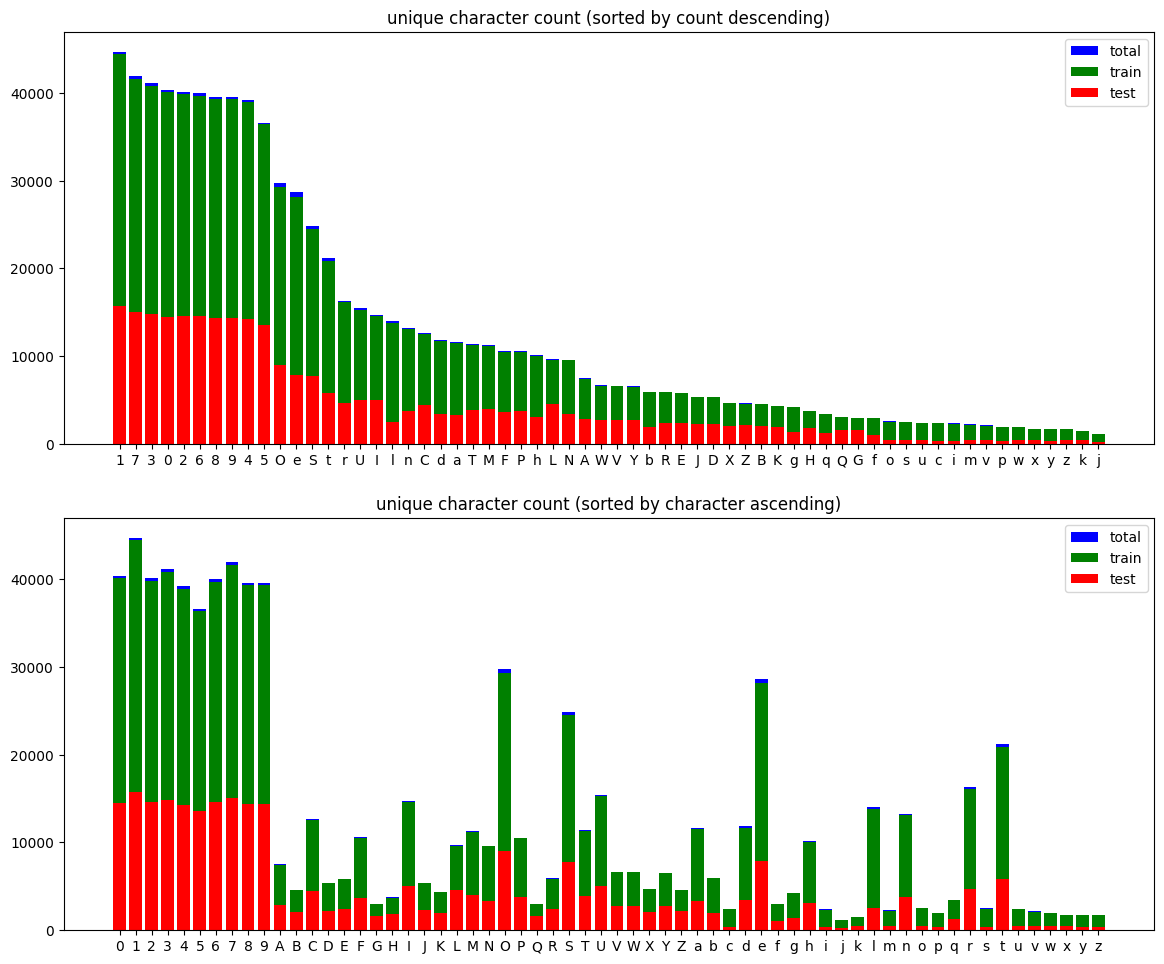
Дупликате пролазимо тако што израчунавамо MD5 хеш за сваку индивидуалну слику затим мапирамо хеш на индекс слике у сету који касније поредимо са следећим хешом који рачунамо. Ако израчунамо хеш који смо претходно израчунали наишли смо на дупликат.

Aнализом сваке од категорија, утврђено је да постоје дупликати у тренинг (*train*) сетовима:

|  |  |
| --- | --- |
| **Назив категорије** | **Број дупликата** |
| EMNIST ByClass | 9 |
| EMNIST ByMerge | 9 |
| EMNIST Letters | 5 |

Даљом аналзизом је утврђено да скоро све категорије имају неке слике која нека друга категорија има… На пример ByClass и ByMerge имају велики број слика које су исте. ByClass *train* скуп садржи слике које ByMerge користи у свом *test* скупу, и обрнуто.

Консултовати следећи графикон:

**total** – Приказује колико укупно има јединствених слика у свим скуповима (*train* и *test*).  
**train** – Приказује колико укупно има јединствених слика у *train* скупуовима.  
**test** – Приказује колико укупно има јединствених слика у *test* скупуовима.

Учитавањем и избацивањем свих дупликата утврђено је да постоји само 814 241 јединствеих слика.

Због међусобног преплитања свих категорија потребно је било самостално поделити податке на тренинг и тест скупове за сваки карактер индивидуално, за сваки карактер је узето: **80%** слика за **тренинг**, а **20%** за **тестирање**.  
(консултовати следећу скрипту: *proper\_train\_test\_split.py*).

Претходна истраживања других особа над коришћеним подацима

Анализом рада других који су користили дати скуп нису испитивали податке. Такође већински су сви тренирали мреже на смањеној верзији сета MNIST која садржи само бројеве од 0-10 што је олакшало саму класификацију за мрежу тако да су добијали веома добре резултате (прецизност до чак 90%). Али за наш скуп података EMNIST, нисмо успели да пронађемо рад који је имао успешност већи од 70% због саме комплексности класификације карактера који слично изледају  
(попут: „i“, „j“, „1“ затим „S“, „5“, такође „0“, „О“, итд.)

## Методе истраживања

Тренирање саме мреже се сводило на испитивање разних параметра и слојева која библиотека *tensorflow* нуди приликом прављења конволуционе неуронске мреже.

Напрвљено је неколико верзија мреже коју тренирамо (консултовати датотеку: *utils\_tf.py* у којој можемо видети сваку верзију).

Свака верзија је тренирана на претнходним 80% издвојеним подацима и тестирано на преосталим 20%.

Поређењем резултата сваке од верзија мреже утврђено је да нам верзија три (*v3*) даје најбоље резултате који нам највише одговарају тако да је она коришћена за сва остала испитивања.

Главне параметре које смо узимали у обзир током тренирања мреже су величина уносног тренинг беча (*batch\_size)* и број епоха (*epochs*). Испитивањем је закључено да конволуциона мрежа која има *batch\_size* 1000 или 10000 даје боље резултате од оне која има мањи. Утвеђено је да библиотека *tensorflow* за велике *batch\_size*-еве користи разне методе које смањују буку приликом израчунавања градијента што доприноси самој класификацији категорије. Такође је вођено рачуна да немамо превелики број епоха да не би дошло до превеликог прилагођава подацима (*overfitting*).

Због самог броја слика, број епоха је подешен на 4-5.

Касније је примећено да је број слика за карктере као што су „j“, „k“, „z“ врло мали. Па смо покушали да допунимо слике коришћењем *tensorflow*-овог *ImageDataGenerator*-а који дату слику ротира, зумира и транслира на разне начине и тако допуњава недостајуће слике. Нажалост због недостатка рачунарксе снаге смо избацили ту методу из оптицаја јер се мрежа јако дуго тренирала. Такође се испоставило да то допуњавање података заправо одмањује учење мреже јер превише *overfit-ује* на генерисане слике, али због довољне величине скупа података само генерисање слика заправо није било ни потребно.

Такође 1 дан је био уложен да се намести тренирање на графичкој картици (*tensorflow-cuda)*, али нажалост то није било успешно.

Након 2 дана уложених у тестирање параметара која tensorflow библиотека нуди. Одлучено је да се направи окружење које ће кориснику ручно дати да цртањем карактера тестира истренирану мрежу. Тако да смо за финалну мрежу одлучили да користим цео скуп података (и тест и тренинг податке које смо раније поделили).

# Резултати

## Приказ резултата

У датом пројекту су направљена чак два окружења за тестирање истренираних мрежа.

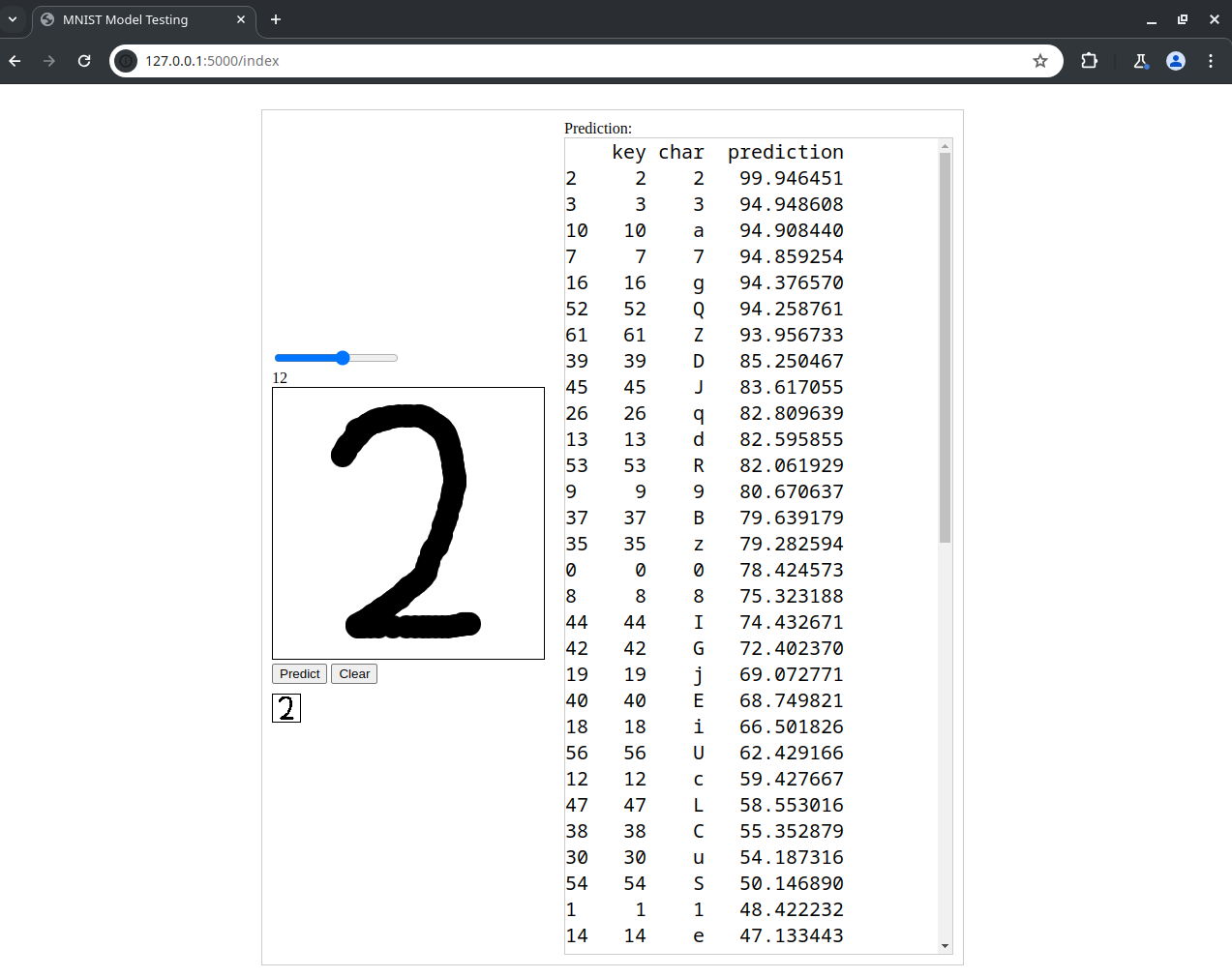
Прво окружење је за цртање (***test\_draw.sh***).

Покретањем окружења кориснику се нуди листа већ истренираних мрежа коју може изабрати да жели да тестира.

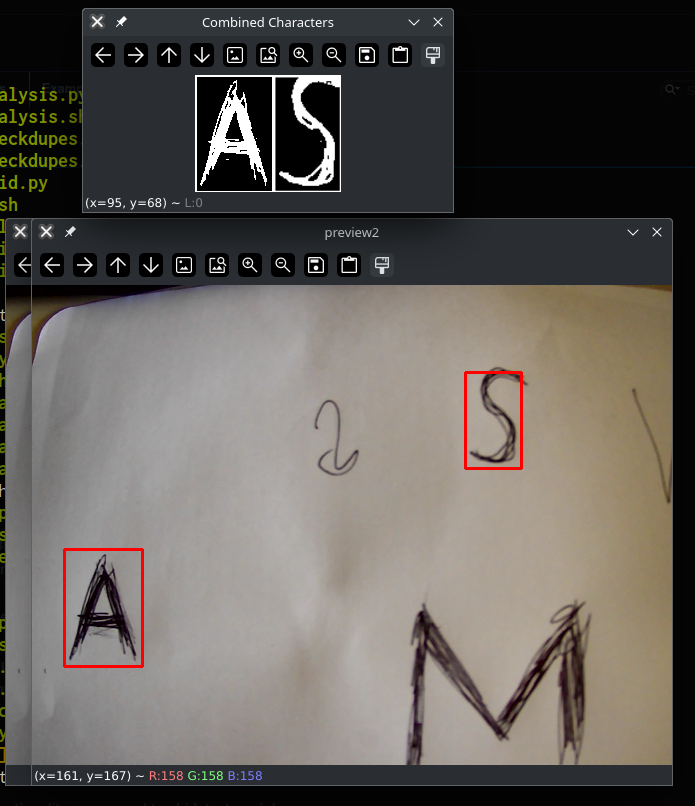
Након што корисник изабере мрежу покреће се веб сервер на адреси: [http://127.0.0.1:5000](http://127.0.0.1:5000/)

Приступ датој веб страници кориснику је омогућено да ручно тестира мрежу цртајући.

Консултовати следећу слику:

Окружење је једноставан *flask* сервер који хостује дату страницу за цртање. Страница затим шаље нацртану слику мрежи путем *POST* *request*-a која касније ту слику прослеђује изабраној неуронској мрежи.

За друго окружење је потребна камера (***test\_camera2.sh***)



## Тумачење резултата

У претнходним испитивањима утврђено је да се мрежа најчешће збуни за карактере:

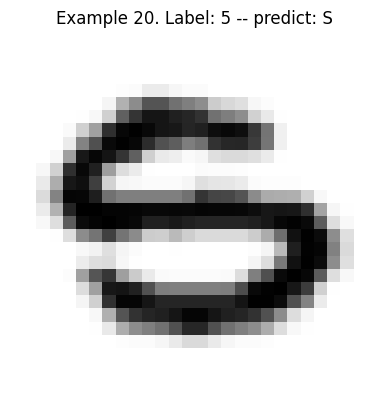
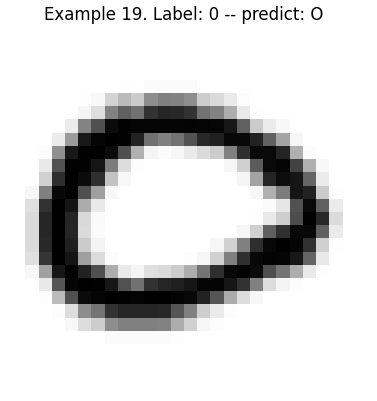
„0“, „о“, „О“

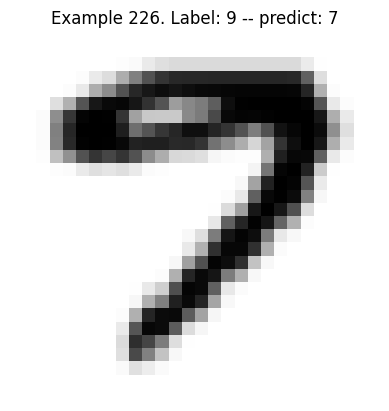
„L, „l“, „1“, „i“, „I“, „J“, „j“

„S“, „5“, „G“, „6“

„9“, „g“

Неки примери:





# Закључак

## Анализа испуњења циљева истраживања

## Анализа остварења очекиваних резултата истраживања

## Могућности за примену истраживања у пракси

## Идеје за побољшање и разраду истраживања

Резултати које смо добили су далеко од савршених. Закључујемо да је веома тешко препознати написани текст на папиру. Поготово кад смо ограничени самим преформансама рачунара којег имамо на располагању. Са јаким *hardware-ом* (рачунаром) би могли не само веће истраживање урадити већ и постићи знатно боље резултате јер би нам то омогућило тренирање са неким генератором слика.

Такође још нека идеја за побољшање би била коришћење још неких скупова података.

# Литература

## https://www.tensorflow.org/api\_docs/python/

## https://docs.opencv.org/4.x/index.html