

# Computer Vision for automate the process of Elections

---

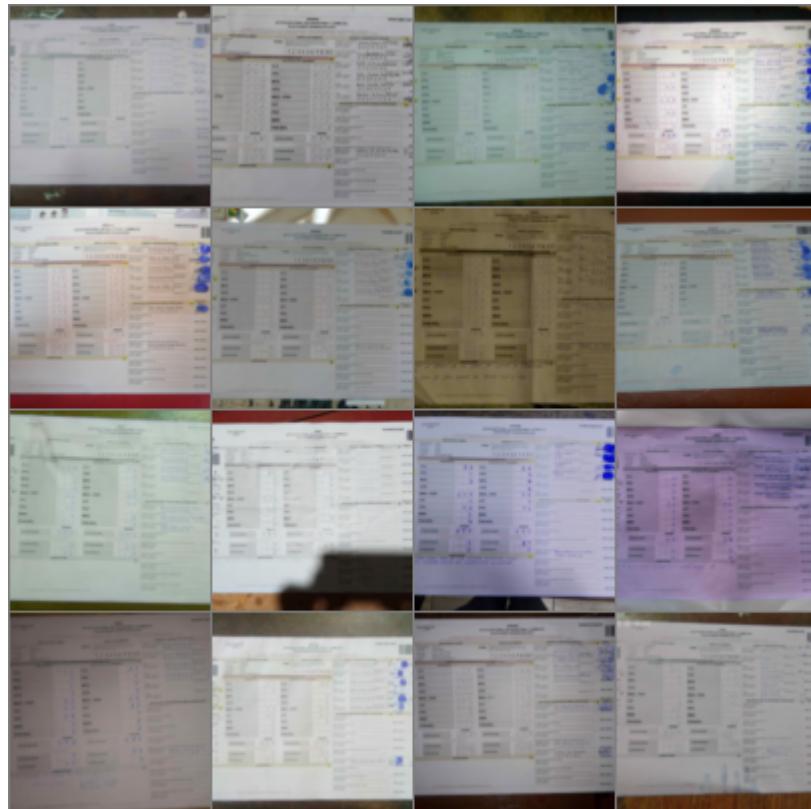
The intention of this work is to automate the counting of votes in the electoral records. Computer vision techniques are used to attempt to read the regions of interest and subsequently read the handwritten numbers within these regions of interest. The dataset that is accessed comes from the Bolivian elections 2019. Within this dataset whose size is 60 GB. We only worked on the images that were scanned since they are easier to process due to the computer vision algorithm that was developed. A work that is left as a potential research area is to use the labels generated by the computer vision algorithm, create a VOC like dataset and train an Object Detection model, the latter was completed but with poor results in a first iteration.

## Download the dataset.

To download the dataset of the images, the jupyter notebook called [DOWNLOAD\\_DATASET/Download\\_images\\_azure.ipynb](#) is used. This notebook downloads the 60 GB of information, whose categories are the following.

### uploadedimages

This images comes from a direct photo from a smartphone taked the moment where the operator in charge of the elections table must to send this report for fast counts called "TREP"



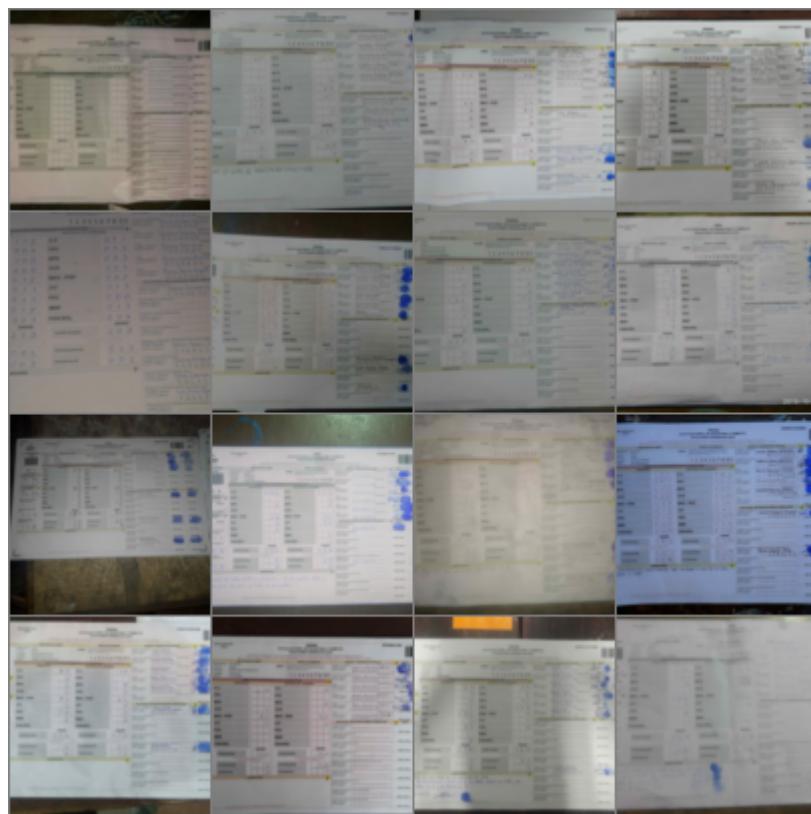
### uploadedimagescomputo

This images are the same from [uploadedimages](#)(TREP) but are been scanned. We take this set of images (~30GB) for our Computer Vision work pipeline.



### imgactastrep

This set of images are been flaged as **fraudulent** by the original team that work with this images the last year, this set of images comes from the [uploadedimages](#) set.



## Create VOC like dataset.

## Target Dataset

We take the [uploadedimagescompute](#) set of images since this are more easy to work with Computer Vision techniques. One example of this dataset set is this image.

Using Canny edge detection and other simple computer vision techniques we can start to get some detection of the areas of interest.

Into this image we are only interested in detecting this region of the image.

PRESIDENTE/A		DIPUTADO/A CIR. UNINOMINAL		
C.C.		3	0	3
FPV				3
MTS			6	3
UCS				5
MAS - IPSP	1	4	2	4
21F			2	8
PDC	1	8		2
MNR		1		0
PAN-BOL		2		
<b>VOTOS VÁLIDOS</b>		<b>2</b>	<b>0</b>	<b>1</b>
LOS VOTOS VÁLIDOS SON LA SUMA DE LOS VOTOS OBTENIDOS POR LAS CANDIDATURAS				
<b>VOTOS BLANCOS</b>			2	
<b>VOTOS NULOS</b>			4	
<b>OBSERVACIONES</b>				
<b>VOTOS VÁLIDOS</b>		<b>1</b>	<b>6</b>	<b>4</b>
LOS VOTOS VÁLIDOS SON LA SUMA DE LOS VOTOS OBTENIDOS POR LAS CANDIDATURAS				
<b>VOTOS BLANCOS</b>			3	9
<b>VOTOS NULOS</b>			4	
<b>5</b>				

Other parts of the image also can be used for another kind of applications, like "fingerprint" matching across the whole dataset or read and detect duplicate names in the region of people in charge of the election table.

This time we are only interested in counting the votes in this selected Region of Interest.

### Creation of VOC Like dataset

For create this dataset we create all the steps for this into the file

[VOC\\_CREATION/bounding\\_boxes\\_creation.py](#).

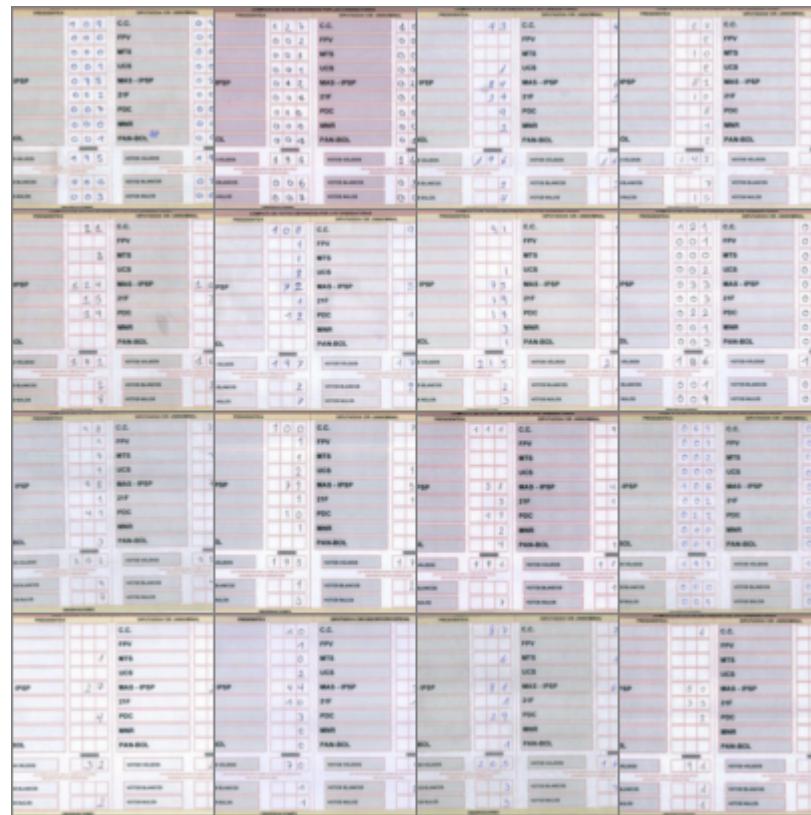
With a simple.

```
cd VOC_CREATION
python bounding_boxes_creation.py --data_path=$UPLOADED_IMAGES_COMPUTO
```

Will start to create the dataset, into a default folder called [results/](#).

We made use of an env variable [\\$UPLOADED\\_IMAGES\\_COMPUTO](#) for set the directory of the target images.

The result is a set of cropped images in `VOC_CREATION/results/Train/images` and their XML labels `VOC_CREATION/results/Train/labels`. ~21000 in total.



A sample with bounding boxes drawing this this.

COMPUTO DE VOTOS OBTENIDOS POR LAS CANDIDATURAS		DIPUTADO/A CIR. UNINOMINAL	
PRESIDENTE/A		DIPUTADO/A CIR. UNINOMINAL	
<b>C.C.</b> CC Presidente	13	<b>C.C.</b> CC Diputado	115
<b>FPV</b> FPV Presidente		<b>FPV</b> FPV Diputado	
<b>MTS</b> MTS Presidente		<b>MTS</b> MTS Diputado	
<b>UCS</b> UCS Presidente	1	<b>UCS</b> UCS Diputado	3
<b>MAS - IPSP</b> MAS IPSP Presidente	84	<b>MAS - IPSP</b> MAS IPSP Diputado	61
<b>21F</b> 21F Presidente	27	<b>21F</b> 21F Diputado	52
<b>PDC</b> PDC Presidente	9	<b>PDC</b> PDC Diputado	
<b>MNR</b> MNR Presidente	2	<b>MNR</b> MNR Diputado	7
<b>PAN-BOL</b> PAN-BOL Presidente		<b>PAN-BOL</b> PAN-BOL Diputado	
<b>VOTOS VÁLIDOS</b> Votos Validos Presidente	196	<b>VOTOS VÁLIDOS</b> Votos Validos Diputado	168
LOS VOTOS VÁLIDOS SON LA SUMA DE LOS VOTOS OBTENIDOS POR LAS CANDIDATURAS			
<b>VOTOS BLANCOS</b> Votos Blancos Presidente	9	<b>VOTOS BLANCOS</b> Votos Blancos Diputado	33
<b>VOTOS NULOS</b> Votos Nulos Presidente	4	<b>VOTOS NULOS</b> Votos Nulos Diputado	1

The total success of detected boxes with the simple computer vision algorithm is aprox.  $\sim(21000/31000) * 100 \sim= 68\%$ . We made a try to fine tune a object detection model for increase this number and read more electoral papers.

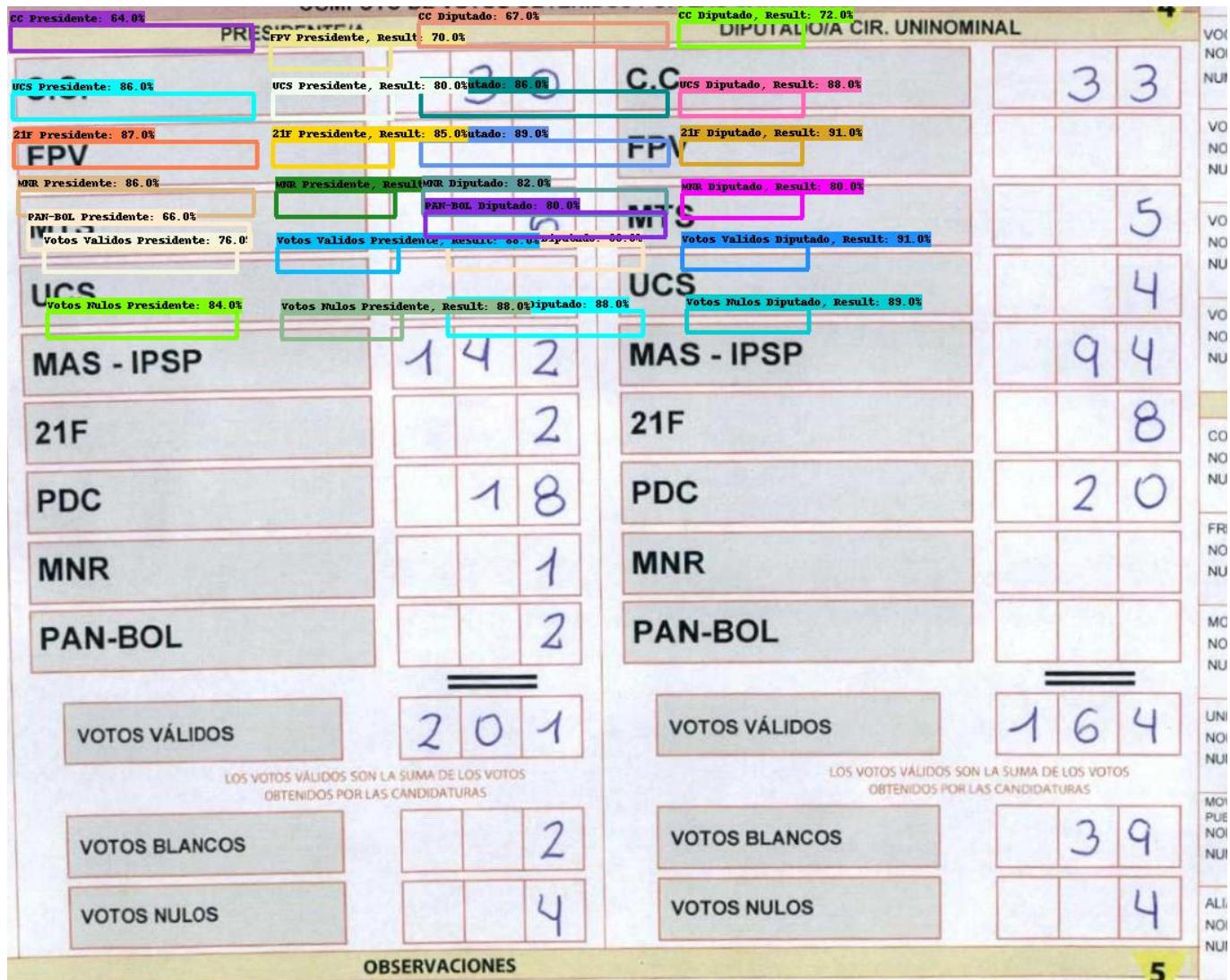
Attempt to finetune SSD MobilenetV2 with this VOC dataset.

### Create TF-RECORDS FILE

We convert to tf-records format our VOC dataset using the script in [VOC\\_CREATION/create\\_tf\\_records.py](#)

### Retrain the model

For fast iteration around this problem we use the MONK [https://github.com/Tessellate-Imaging/Monk\\_Object\\_Detection](https://github.com/Tessellate-Imaging/Monk_Object_Detection) library for retrain object detection models. The fork of his notebook called [Train Without Validation Dataset.ipynb](#) is in our folder called [OBJECT\\_DETECTION/Train\\_Without\\_Validation\\_Dataset.ipynb](#). We had a problem with our credit card and we are unable to open an azure account and train the model inside azure compute instances. For this reason we made use of google colab, we use a lot of GPU VRAM for finetune the mobilenetv2 model. The result of this finetuning is .



We think that the MONK library is not resizing correctly our image in this high level interface

We have not enough time for test the native tensorflow object detection library or another models, by this reason we leave this approach for get to work the remaining 32% of the images.

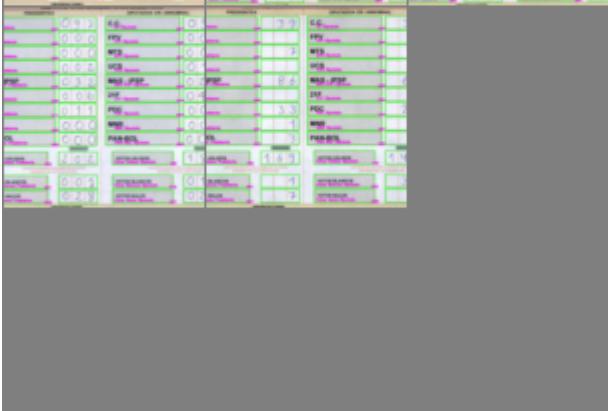
## Count of votes.

Since we labeled our Regions of Interest, we can start to count the numbers inside the boxes with another bit of work of computer vision.

We only take a sample of 5 images for proof the concept.

We create the instance `VotesCounter(ImageHandler)` inside of the `VOTES_COUNTER/votes_counter.py` script, this instance expects some MNIST model for read the digits. We deploy a custom MNIST in our local server, we also can use any other model for read digits inside this images.

We explore the possibility to create other datasets by the side only of the digits inside this images. We will explore this alternative later. For now we obtain the following results with this simple MNIST.



CÁMARA DE DIPUTADOS		CÁMARA DE DIPUTADOS		CÁMARA DE DIPUTADOS	
001	C.C.	002	FPV	003	MTS
004	UCS	005	MNR	006	PDC
007	PAN-BOL	008	21F	009	MAS - IPSP
010		011		012	
013		014		015	
016		017		018	
019		020		021	
022		023		024	
025		026		027	
028		029		030	
031		032		033	
034		035		036	
037		038		039	
039		040		041	
040		041		042	
041		042		043	
042		043		044	
043		044		045	
044		045		046	
045		046		047	
046		047		048	
047		048		049	
048		049		050	
049		050		051	
050		051		052	
051		052		053	
052		053		054	
053		054		055	
054		055		056	
055		056		057	
056		057		058	
057		058		059	
058		059		060	
059		060		061	
060		061		062	
061		062		063	
062		063		064	
063		064		065	
064		065		066	
065		066		067	
066		067		068	
067		068		069	
068		069		070	
069		070		071	
070		071		072	
071		072		073	
072		073		074	
073		074		075	
074		075		076	
075		076		077	
076		077		078	
077		078		079	
078		079		080	
079		080		081	
080		081		082	
081		082		083	
082		083		084	
083		084		085	
084		085		086	
085		086		087	
086		087		088	
087		088		089	
088		089		090	
089		090		091	
090		091		092	
091		092		093	
092		093		094	
093		094		095	
094		095		096	
095		096		097	
096		097		098	
097		098		099	
098		099		100	
099		100		101	
100		101		102	
101		102		103	
102		103		104	
103		104		105	
104		105		106	
105		106		107	
106		107		108	
107		108		109	
108		109		110	
109		110		111	
110		111		112	
111		112		113	
112		113		114	
113		114		115	
114		115		116	
115		116		117	
116		117		118	
117		118		119	
118		119		120	
119		120		121	
120		121		122	
121		122		123	
122		123		124	
123		124		125	
124		125		126	
125		126		127	
126		127		128	
127		128		129	
128		129		130	
129		130		131	
130		131		132	
131		132		133	
132		133		134	
133		134		135	
134		135		136	
135		136		137	
136		137		138	
137		138		139	
138		139		140	
139		140		141	
140		141		142	
141		142		143	
142		143		144	
143		144		145	
144		145		146	
145		146		147	
146		147		148	
147		148		149	
148		149		150	
149		150		151	
150		151		152	
151		152		153	
152		153		154	
153		154		155	
154		155		156	
155		156		157	
156		157		158	
157		158		159	
158		159		160	
159		160		161	
160		161		162	
161		162		163	
162		163		164	
163		164		165	
164		165		166	
165		166		167	
166		167		168	
167		168		169	
168		169		170	
169		170		171	
170		171		172	
171		172		173	
172		173		174	
173		174		175	
174		175		176	
175		176		177	
176		177		178	
177		178		179	
178		179		180	
179		180		181	
180		181		182	
181		182		183	
182		183		184	
183		184		185	
184		185		186	
185		186		187	
186		187		188	
187		188		189	
188		189		190	
189		190		191	
190		191		192	
191		192		193	
192		193		194	
193		194		195	
194		195		196	
195		196		197	
196		197		198	
197		198		199	
198		199		200	
199		200		201	
200		201		202	
201		202		203	
202		203		204	
203		204		205	
204		205		206	
205		206		207	
206		207		208	
207		208		209	
208		209		210	
209		210		211	
210		211		212	
211		212		213	
212		213		214	
213		214		215	
214		215		216	
215		216		217	
216		217		218	
217		218		219	
218		219		220	
219		220		221	
220		221		222	
221		222		223	
222		223		224	
223		224		225	
224		225		226	
225		226		227	
226		227		228	
227		228		229	
228		229		230	
229		230		231	
230		231		232	
231		232		233	
232		233		234	
233		234		235	
234		235		236	
235		236		237	
236		237		238	
237		238		239	
238		239		240	
239		240		241	
240		241		242	
241		242		243	
242		243		244	
243		244		245	
244		245		246	
245		246		247	
246		247		248	
247		248		249	
248		249		250	
249		250		251	
250		251		252	
251		252		253	
252		253		254	
253		254		255	
254		255		256	
255		256		257	
256		257		258	
257		258		259	
258		259		260	
259		260		261	
260		261		262	
261		262		263	
262		263		264	
263		264		265	
264		265		266	
265		266		267	
266		267		268	
267		268		269	
268		269		270	
269		270		271	
270		271		272	
271		272		273	
272		273		274	
273		274		275	
274		275		276	
275		276		277	
276		277		278	
277		278		279	
278		279		280	
279		280		281	
280		281		282	
281		282		283	
282		283		284	
283		284		285	
284		285		286	
285		286		287	
286		287		288	
287		288		289	
288		289		290	
289		290		291	
290		291		292	
291		292		293	
292		293		294	

00c354f2-f7ce-11e9-95e8-c8ff28027534.jpg,023,4  
00c354f2-f7ce-11e9-95e8-c8ff28027534.jpg,204,5  
00c354f2-f7ce-11e9-95e8-c8ff28027534.jpg,181,6  
00c354f2-f7ce-11e9-95e8-c8ff28027534.jpg,000,7  
00c354f2-f7ce-11e9-95e8-c8ff28027534.jpg,000,8  
00c354f2-f7ce-11e9-95e8-c8ff28027534.jpg,000,9  
00c354f2-f7ce-11e9-95e8-c8ff28027534.jpg,004,10  
00c354f2-f7ce-11e9-95e8-c8ff28027534.jpg,032,11  
00c354f2-f7ce-11e9-95e8-c8ff28027534.jpg,026,12  
00c354f2-f7ce-11e9-95e8-c8ff28027534.jpg,001,13  
00c354f2-f7ce-11e9-95e8-c8ff28027534.jpg,007,14  
00c354f2-f7ce-11e9-95e8-c8ff28027534.jpg,102,15  
00c354f2-f7ce-11e9-95e8-c8ff28027534.jpg,078,16  
00c354f2-f7ce-11e9-95e8-c8ff28027534.jpg,002,17  
00c354f2-f7ce-11e9-95e8-c8ff28027534.jpg,008,18  
00c354f2-f7ce-11e9-95e8-c8ff28027534.jpg,009,19  
00c354f2-f7ce-11e9-95e8-c8ff28027534.jpg,011,20  
00c354f2-f7ce-11e9-95e8-c8ff28027534.jpg,001,21  
00c354f2-f7ce-11e9-95e8-c8ff28027534.jpg,003,22  
00c354f2-f7ce-11e9-95e8-c8ff28027534.jpg,057,23  
00c354f2-f7ce-11e9-95e8-c8ff28027534.jpg,044,24  
00151e8d-f7dd-11e9-b71f-c8ff28027534.jpg,004,1  
00151e8d-f7dd-11e9-b71f-c8ff28027534.jpg,004,2  
00151e8d-f7dd-11e9-b71f-c8ff28027534.jpg,000,3  
00151e8d-f7dd-11e9-b71f-c8ff28027534.jpg,013,4  
00151e8d-f7dd-11e9-b71f-c8ff28027534.jpg,199,5  
00151e8d-f7dd-11e9-b71f-c8ff28027534.jpg,186,6  
00151e8d-f7dd-11e9-b71f-c8ff28027534.jpg,000,7  
00151e8d-f7dd-11e9-b71f-c8ff28027534.jpg,000,8  
00151e8d-f7dd-11e9-b71f-c8ff28027534.jpg,003,9  
00151e8d-f7dd-11e9-b71f-c8ff28027534.jpg,007,10  
00151e8d-f7dd-11e9-b71f-c8ff28027534.jpg,005,11  
00151e8d-f7dd-11e9-b71f-c8ff28027534.jpg,004,12  
00151e8d-f7dd-11e9-b71f-c8ff28027534.jpg,014,13  
00151e8d-f7dd-11e9-b71f-c8ff28027534.jpg,073,14  
00151e8d-f7dd-11e9-b71f-c8ff28027534.jpg,027,15  
00151e8d-f7dd-11e9-b71f-c8ff28027534.jpg,024,16  
00151e8d-f7dd-11e9-b71f-c8ff28027534.jpg,001,17  
00151e8d-f7dd-11e9-b71f-c8ff28027534.jpg,004,18  
00151e8d-f7dd-11e9-b71f-c8ff28027534.jpg,000,19  
00151e8d-f7dd-11e9-b71f-c8ff28027534.jpg,002,20  
00151e8d-f7dd-11e9-b71f-c8ff28027534.jpg,000,21  
00151e8d-f7dd-11e9-b71f-c8ff28027534.jpg,000,22  
00151e8d-f7dd-11e9-b71f-c8ff28027534.jpg,149,23  
00151e8d-f7dd-11e9-b71f-c8ff28027534.jpg,072,24  
011e2371-f7e3-11e9-8f53-c8ff28027534.jpg,028,1  
011e2371-f7e3-11e9-8f53-c8ff28027534.jpg,027,2  
011e2371-f7e3-11e9-8f53-c8ff28027534.jpg,001,3  
011e2371-f7e3-11e9-8f53-c8ff28027534.jpg,019,4  
011e2371-f7e3-11e9-8f53-c8ff28027534.jpg,202,5  
011e2371-f7e3-11e9-8f53-c8ff28027534.jpg,185,6  
011e2371-f7e3-11e9-8f53-c8ff28027534.jpg,000,7  
011e2371-f7e3-11e9-8f53-c8ff28027534.jpg,000,8  
011e2371-f7e3-11e9-8f53-c8ff28027534.jpg,000,9

```
011e2371-f7e3-11e9-8f53-c8ff28027534.jpg,000,10
011e2371-f7e3-11e9-8f53-c8ff28027534.jpg,011,11
011e2371-f7e3-11e9-8f53-c8ff28027534.jpg,004,12
011e2371-f7e3-11e9-8f53-c8ff28027534.jpg,006,13
011e2371-f7e3-11e9-8f53-c8ff28027534.jpg,044,14
011e2371-f7e3-11e9-8f53-c8ff28027534.jpg,032,15
011e2371-f7e3-11e9-8f53-c8ff28027534.jpg,025,16
011e2371-f7e3-11e9-8f53-c8ff28027534.jpg,002,17
011e2371-f7e3-11e9-8f53-c8ff28027534.jpg,014,18
011e2371-f7e3-11e9-8f53-c8ff28027534.jpg,000,19
011e2371-f7e3-11e9-8f53-c8ff28027534.jpg,001,20
011e2371-f7e3-11e9-8f53-c8ff28027534.jpg,000,21
011e2371-f7e3-11e9-8f53-c8ff28027534.jpg,000,22
011e2371-f7e3-11e9-8f53-c8ff28027534.jpg,097,23
011e2371-f7e3-11e9-8f53-c8ff28027534.jpg,097,24
012c5403-f7ce-11e9-b2ea-c8ff28027534.jpg,007,1
012c5403-f7ce-11e9-b2ea-c8ff28027534.jpg,005,2
012c5403-f7ce-11e9-b2ea-c8ff28027534.jpg,001,3
012c5403-f7ce-11e9-b2ea-c8ff28027534.jpg,032,4
012c5403-f7ce-11e9-b2ea-c8ff28027534.jpg,169,5
012c5403-f7ce-11e9-b2ea-c8ff28027534.jpg,140,6
012c5403-f7ce-11e9-b2ea-c8ff28027534.jpg,003,7
012c5403-f7ce-11e9-b2ea-c8ff28027534.jpg,000,8
012c5403-f7ce-11e9-b2ea-c8ff28027534.jpg,001,9
012c5403-f7ce-11e9-b2ea-c8ff28027534.jpg,001,10
012c5403-f7ce-11e9-b2ea-c8ff28027534.jpg,033,11
012c5403-f7ce-11e9-b2ea-c8ff28027534.jpg,024,12
012c5403-f7ce-11e9-b2ea-c8ff28027534.jpg,000,13
012c5403-f7ce-11e9-b2ea-c8ff28027534.jpg,004,14
012c5403-f7ce-11e9-b2ea-c8ff28027534.jpg,086,15
012c5403-f7ce-11e9-b2ea-c8ff28027534.jpg,065,16
012c5403-f7ce-11e9-b2ea-c8ff28027534.jpg,000,17
012c5403-f7ce-11e9-b2ea-c8ff28027534.jpg,009,18
012c5403-f7ce-11e9-b2ea-c8ff28027534.jpg,007,19
012c5403-f7ce-11e9-b2ea-c8ff28027534.jpg,003,20
012c5403-f7ce-11e9-b2ea-c8ff28027534.jpg,000,21
012c5403-f7ce-11e9-b2ea-c8ff28027534.jpg,004,22
012c5403-f7ce-11e9-b2ea-c8ff28027534.jpg,039,23
012c5403-f7ce-11e9-b2ea-c8ff28027534.jpg,030,24
```

which later can be processed with pandas for further analysis.

## Conclusions

With this work we wanted to demonstrate the proof of concept that it is possible to automatically read the digits written by hand within physical electoral records. The procedure ranges from using simple computer vision techniques to exploring the possibility of creating an object detector using deep learning. The latter needs more work and it is hoped that this can be achieved. We also leave open the possibility of using this dataset for other types of work that can be done on top of the images that are available.