Distant reading of the Engineering and Mining Journal

Application of detectron2, layoutparser and labelstudio

August 6, 2023

This notebook summarizes the process of creating an Engineering a Mining Journal database using computer vision and labeling using the free tools of detectron2, layoutparser, and label studio. Primarily, this exercise follows the steps developed by lolipopshock

The process is composed of six steps: 1. Preparation of the training datasets 2. Labeling of the datasets 3. Training and evaluation 4. Prediction using the model 5. Construction of the database 6. Evaluation of the database

1 Preparation of the database

This step allowed the researchers, Edward Beatty and Israel G. Solares, to provide an example of labeling using labelstudio for Jack Wang, who labeled a representative sample of the dataset, composed of 99 volumes of the Engineering and Mining Journal, downloaded from Hathitrust from 1872 to 1923. #### Construction of the training dataset

[5]:
```
pip install detectron2 https://dl.fbaipublicfiles.com/detectron2/wheels/cu113/
↪torch1.10/index.html
```

```
Defaulting to user installation because normal site-packages is not writeable
Collecting
https://dl.fbaipublicfiles.com/detectron2/wheels/cu113/torch1.10/index.html
  Using cached
https://dl.fbaipublicfiles.com/detectron2/wheels/cu113/torch1.10/index.html (468
bytes)
  ERROR: Cannot unpack file /tmp/pip-unpack-6g2f3tqa/index.html (downloaded
from /tmp/pip-req-build-33t0u1p0, content-type: text/html); cannot detect
archive format
ERROR: Cannot determine archive format of /tmp/pip-req-
build-33t0u1p0
Note: you may need to restart the kernel to use updated packages.
```

[1]:
```
import requests
import numpy as np
```

```
import layoutparser as lp
import glob
from PIL import Image as im
import PIL
import pandas as pd
import random
import cv2
import pdf2image
from pdf2image import convert_from_path
import matplotlib.pyplot as plt
```

Loading the list of files and creating a random list of items that will be labeled

[ ]:
```
train = pd.read_csv('pag2.csv')
todo = pd.read_csv('pag1.csv')
filelist= train['name']
path1  = '/home/.../ensayo/emj/'
path2 = '/home/.../ensayo/img/'
path3 = '/home/..../ensayo/entrenamiento/'
train['path']= path1 + train['name']
todo['path']= path1 + todo['name']
todo['pageind']=todo['pageind']+1
imagenes= []
prueba =[]
```

[18]:
```
for i in range(0, len(train)):
    imagen= convert_from_path(train['path'][i], first_page=train['page1'][i],␣
 ↪last_page=train['page2'][i])
    k = int(round(len(imagen)*0.69, 0))
    muestra= random.sample(imagen, k)
    for j in range(0, len(muestra)):
        filename= "".join([path3, str(train['volume'][i]),"_", str(j) ,".jpg"])
        picture = muestra[j]
        picture = picture.save(filename)
```

After the first dataset was labeled, we created a second random set of items to be labeled.

[4]:
```
todo = pd.read_csv('pag1.csv')
filelist= todo['name']
path1  = '/home/.../ensayo/emj/'
path2 = '/home/.../ensayo/img/'
path3 = '/home/.../ensayo/entrenamiento2/'
todo['path']= path1 + todo['name']
todo['page1']=todo['pageind']+300
todo['page2']=todo['pageind']+400
imagenes= []
prueba =[]
```

[ ]:
```
for i in range(0, len(todo)):
    try:
        imagen= convert_from_path(todo['path'][i], first_page=todo['page1'][i],␣
↪last_page=todo['page2'][i])
        k = 3
        muestra= random.sample(imagen, k)
    except ValueError:
        imagen= convert_from_path(todo['path'][i], first_page=todo['pageind'][i])
        k = 3
        muestra= random.sample(imagen, k)
    for j in range(0, len(muestra)):
        filename= "".join([path3, str(todo['volume'][i]),"_", str(j) ,".jpg"])
        picture = muestra[j]
        picture = picture.save(filename)
```

[5]:
```
for i in range(56, len(todo)):
    try:
        imagen= convert_from_path(todo['path'][i], first_page=todo['page1'][i],␣
↪last_page=todo['page2'][i])
        k = 3
        muestra= random.sample(imagen, k)
    except ValueError:
        imagen= convert_from_path(todo['path'][i], first_page=todo['pageind'][i])
        k = 3
        muestra= random.sample(imagen, k)
    for j in range(0, len(muestra)):
        filename= "".join([path3, str(todo['volume'][i]),"_", str(j) ,".jpg"])
        picture = muestra[j]
        picture = picture.save(filename)
```

2 Labeling

The two datasets were labeled by following the examples of a first test and using the process proposed by label studio. As the labeling was made on multiple computers, the ids of the coco annotations are inconsistent, so the consolidation requires homogenizing the labeling ids. Furthermore, the PI also double-checked the labeling tasks, eliminating ambiguities in the process.

[ ]:
```
pip install detectron2 https://dl.fbaipublicfiles.com/detectron2/wheels/cu113/
↪torch1.10/index.html
pip install -U scikit-learn scipy matplotlib
```

[1]:
```
import requests
import numpy as np
import layoutparser as lp
import glob
```

```
from PIL import Image as img
import re
import pandas as pd
import json
from pycocotools.coco import COCO
import layoutparser as lp
import random
import cv2
from numpy import asarray
```

[2]:
```
def load_coco_annotations(annotations, coco=None):
    """
    Args:
        annotations (List):
            a list of coco annotaions for the current image
        coco (`optional`, defaults to `False`):
            COCO annotation object instance. If set, this function will
            convert the loaded annotation category ids to category names
            set in COCO.categories
    """
    layout = lp.Layout()

    for ele in annotations:

        x, y, w, h = ele['bbox']

        layout.append(
            lp.TextBlock(
                block = lp.Rectangle(x, y, w+x, h+y),
                type  = ele['category_id'] if coco is None else coco.
↪cats[ele['category_id']]['name'],
                id = ele['id']
            )
        )

    return layout
```

Uploading the three files coming from the human labeling of the random samples

[ ]:
```
listarchivos=glob.glob('/origin/*/result.json')
listarchivos
```

Evaluating the categories of the files

[ ]:
```
f1 = open(listarchivos[0])
image_ann1 = json.load(f1)
f2 = open(listarchivos[1])
image_ann2 = json.load(f2)
```

```
f3 = open(listarchivos[2])
image_ann3 = json.load(f3)
image_ann1['categories']
```

[ ]:
```
image_ann2['categories']
```

[ ]:
```
image_ann3['categories']
```

Compiling the files into a single one, and correcting the connections

[ ]:
```
rep= 'images/(\d)/'
image_list={'images':[],
            'categories':[{'id': 0, 'name': 'Author'},
                          {'id': 1, 'name': 'Date'},
                          {'id': 2, 'name': 'Page'},
                          {'id': 3, 'name': 'Text'},
                          {'id': 4, 'name': 'Title'}],
            'annotations':[],
            'info': {'year': 2023, 'version': '1.0', 'description': '',␣
 ↪'contributor': 'Label Studio', 'url': '', 'date_created': '2023-03-07 17:43:40.
 ↪748757'}
           }
```

[ ]:
```
for j in range(0, len(listarchivos)):
    f = open(listarchivos[j])
    image_ann = json.load(f)
    for i in range(0, len(image_ann['images'])):
        x= image_ann['images'][i]['id']+len(image_list['images'])
        y=re.findall(r'/.*|\\.*', image_ann['images'][i]['file_name'])[0]
        y=y.replace('\\', '/')
        y=y.replace('/4/', '/')
        y=y.replace('///', '/')
        y="images/"+y
        image_ann['images'][i]['id']= x
        image_ann['images'][i]['file_name']= y
    for i in range(0, len(image_ann['annotations'])):
        x= image_ann['annotations'][i]['id']+len(image_list['annotations'])
        y= image_ann['annotations'][i]['image_id']+len(image_list['images'])
        image_ann['annotations'][i]['id']= x
        image_ann['annotations'][i]['image_id']= y
    image_list['images'].extend(image_ann['images'])
    image_list['annotations'].extend(image_ann['annotations'])
```

[ ]:
```
with open('/destination/resultbeta.json', 'w') as outfile:
    json.dump(image_list, outfile)
```

After visually looking at the labeling, the PI eliminated the inconsistencies in the images.

[ ]:
```
COCO_ANNO_PATH = '/destination/resultbeta.json'
COCO_IMG_PATH  = '/destination'
COCO_IMG_TEST = '/corroboration'
coco = COCO(COCO_ANNO_PATH)
```

[ ]:
```
for image_id in coco.imgs:
    image_info = coco.imgs[image_id]
    annotations = coco.loadAnns(coco.getAnnIds([image_id]))
    image = cv2.imread(f'{COCO_IMG_PATH}/{image_info["file_name"]}')
    layout = load_coco_annotations(annotations, coco)
    viz = lp.draw_box(image, layout, show_element_type=True)
    path = f'{COCO_IMG_TEST}/{image_info["file_name"]}'
    viz.save(path)
```

Date
January 6, 1923

Engineering and Mining Journal-Press

Page
21

Text
For the treatment of ores containing both sulphide and oxide minerals, the processes might be made to supplement each other. For instance, if lead anodes were used in a diaphragm cell, the current efficiency would be high, the percentage of copper removed from the solution for each pass would be quite high, and still, in the anode compartments, there would be some formation of ferric salts, enough for an ore largely but not entirely oxidized. Another and probably better system for the treatment of partly oxidized ores would be to use the diaphragm process, apply the ferric-sulphate solution to the ores, when all soluble copper minerals would dissolve, the oxides dissolving on account of the free acid present, and the sulphides metallic copper, and cuprous oxide, being attacked and dissolved by the ferric sulphate. After the leaching the solution would still contain some ferric sulphate and this could then be reduced with sulphur dioxide so that the solution passing to the cells could be practically free from ferric iron. At the same time the sulphur dioxide, by being converted into sulphuric acid, or sulphuric-acid intermediates, would build up the solution in acid, so that portions of the electrolyte could be bled off continually, thus serving for purification and control of iron content and making up for the loss of acid being discharged as iron sulphates.

All electrolytic copper processes, in one particular, appear to possess one disadvantage in comparison with the cyanide process wherein a solvent solution is applied to the ore and the solution is then made to yield the great bulk of the gold. In the electrolytic copper processes, the cells will not extract a very large percentage of the copper advantageously, and it is therefore necessary to keep a lot of copper sulphate in circulation, from ore to cells and from cells to ore. This is a disadvantage in some respects, although by no means sufficiently so as to render the process entirely inapplicable in the connection noted.

It may justly be said that the accomplishments up to the present in electrolytic recovery of copper from ore are due largely to financial and engineering skill rather than to development and application of electrochemical progress. In consequence there is much room for improvement in the chemical and electrochemical features.

Title
Metallurgical Practice in the Porcupine Gold District

Text
In one point there is uniformity in milling practice in the three large plants of Porcupine, says J. C. Murray in the *Canadian Mining Journal*. Precipitation of gold from cyanide solution is effected in all three plants with zinc dust. At the McIntyre mill, zinc boxes were given a trial and were quickly abandoned. In all three mills lead acetate is used to aid precipitation.

There is wide divergence in refining methods at the three plants. At the Hollinger, the raw precipitate is smelted in blast furnaces; the bullion is brought down with lead; the lead is driven off in standard cupellation furnaces, and a bullion of 950 total fineness is obtained. At the McIntyre the raw precipitate is treated with sulphuric acid and melted in a carborundum-lined, tilting, reverberatory furnace. The bullion here is brought to a total fineness of 900. The practice at the Dome consists in treating the precipitate with sulphuric acid, calcining, and melting in small clay-lined graphite pots in a reverberatory furnace. The fineness obtained is from 990 to 995.

It is likely that the roar of the stamps, will soon be a forgotten sound in Porcupine. The stamp is an illogical and power-wasting contrivance at best. It is a strange and anomalous survival. In the last ten years it has lost much face. Experience at Porcupine has contributed not a little to the stamp's approaching relegation to oblivion.

Regarding Porcupine milling practice, some incidental changes in the flow sheets are extraordinarily interesting. For instance, the use of Hudson Bay blankets on the lower ends of amalgamating plates at the Dome mill was a distinct improvement. In due time these costly items were superseded by a cheaper and more suitable material, wide strips of "silent felt" the stuff in common use between the linen tablecloth of civilization and the mahogany table.

Another instance is afforded by the supersession of pebble grinding at the McIntyre by the use of steel slugs. A slight increase in cost was compensated for by greater tonnage and a much more uniform product.

Another change in practice at the McIntyre resulted in reducing the consumption of zinc from 0.20 lb. per ton of solution to 0.05 lb. per ton of solution.

Title
Mine Telephoning by Radio

Text
The use of radio telephone instruments has now become so general that inquiries have been made as to the feasibility of the use of radio instruments underground, with the idea advanced that they would be especially valuable when accident interfered with the ordinary means of communication. The U. S. Bureau of Mines has investigated the matter at the Bureau experimental mine at Bruceton, Pa., in co-operation with radio engineers from the Westinghouse Electric & Manufacturing Co. The experiments are described in *Reports of Investigations* No. 2,407.

The experiments consisted first in receiving signals from without the mine by means of a receiver located inside the mine, and, second, both sending and receiving messages underground through the strata. It was found that with a receiving instrument set at a point 100 ft. underground, signals from KDKA station of the Westinghouse Electric & Manufacturing Co., East Pittsburgh, Pa., could be heard distinctly. Station KDKA is at a distance of about 18 miles from the experimental mine. About 50 ft. from the receiving station used in this test was a 6-in. bore hole from the surface, lined with iron pipe and containing electric-light wires which extended therefrom throughout the mine. The presence of these wires evidently assisted greatly in the reception, for when the receiving set was carried to another point in the mine removed from wires and tracks the signals were barely audible through 50 ft. of cover. The fact that signals were detected, however, even though faintly, is sufficient evidence of transmission through the ground to encourage experimenting.

The preliminary experiments, although unsuccessful in indicating any practical method of using wireless waves for underground communication, nevertheless indicate clearly that electromagnetic waves may be made to travel through solid strata. The "absorption" or loss of intensity with distance is very great for the short wave lengths (200 to 360 meters) used in these experiments. Longer wave lengths are known to undergo less absorption and may possibly be found practically effective under certain conditions.

Example of a labeled image

After eliminating the ambiguous labeling, we proceed to redo the JSON file

[25]:
```
listimgs=glob.glob('/corroboration/images/*.jpg')
listimgs2 = pd.DataFrame(listimgs)
listimgs2='images//'+listimgs2[0].str.extract(r'((?<=images[/]).*.jpg)')
listimgs2=listimgs2[0].values.tolist()
len(listimgs2)
```

[25]:
```
514
```

[26]:
```
f3 = open('/home/.../ensayoigs/ensayo3/destination/resultbeta.json')
image_ann = json.load(f3)
image_list={'images':[],
            'categories':[{'id': 0, 'name': 'Author'},
                          {'id': 1, 'name': 'Date'},
                          {'id': 2, 'name': 'Page'},
                          {'id': 3, 'name': 'Text'},
                          {'id': 4, 'name': 'Title'}],
            'annotations':[],
            'info': {'year': 2023, 'version': '1.0', 'description': '',␣
 ↪'contributor': 'Label Studio', 'url': '', 'date_created': '2023-03-07 17:43:40.
 ↪748757'}
           }
ids = []
for i in range(0, len(image_ann['images'])):
    if image_ann['images'][i]['file_name'] in listimgs2:
        ids.append(image_ann['images'][i]['id'])
for i in range(0, len(image_ann['images'])):
    if image_ann['images'][i]['file_name'] in listimgs2:
        image_list['images'].append(image_ann['images'][i])
    else :
        pass
for i in range(0, len(image_ann['annotations'])):
    if image_ann['annotations'][i]['image_id'] in ids:
        image_list['annotations'].append(image_ann['annotations'][i])
    else :
        pass
```

Comparing the two files

[27]:
```
len(image_ann['images'])-len(image_list['images'])
```

[27]:
```
36
```

[28]:
```
len(image_ann['annotations'])-len(image_list['annotations'])
```

[28]:
```
142
```

Selecting a random sample for training and evaluation sets.

[29]:
```
k = round(len(image_list['images'])*.15)
lista1 = random.sample(image_list['images'], k)
lista = []
for i in range(0, len(lista1)):
    x= lista1[i]['id']
    lista.append(x)
```

[30]:
```
result={'images':[],
            'categories':[{'id': 0, 'name': 'Author'},
                          {'id': 1, 'name': 'Date'},
                          {'id': 2, 'name': 'Page'},
                          {'id': 3, 'name': 'Text'},
                          {'id': 4, 'name': 'Title'}],
            'annotations':[],
            'info': {'year': 2023, 'version': '1.0', 'description': '',␣
 ↪'contributor': 'Label Studio', 'url': '', 'date_created': '2023-03-07 17:43:40.
 ↪748757'}
           }
control={'images':[],
            'categories':[{'id': 0, 'name': 'Author'},
                          {'id': 1, 'name': 'Date'},
                          {'id': 2, 'name': 'Page'},
                          {'id': 3, 'name': 'Text'},
                          {'id': 4, 'name': 'Title'}],
            'annotations':[],
            'info': {'year': 2023, 'version': '1.0', 'description': '',␣
 ↪'contributor': 'Label Studio', 'url': '', 'date_created': '2023-03-07 17:43:40.
 ↪748757'}
           }
for i in range(0, len(image_list['images'])):
    if image_list['images'][i]['id'] in lista:
        control['images'].append(image_list['images'][i])
    else:
        result['images'].append(image_list['images'][i])
for i in range(0, len(image_list['annotations'])):
    if image_list['annotations'][i]['image_id'] in lista:
        control['annotations'].append(image_list['annotations'][i])
    else :
        result['annotations'].append(image_list['annotations'][i])
```

Saving the control and the training group

[33]:
```
with open('/destination/result.json', 'w') as outfile:
    json.dump(result, outfile)
with open('/destination/control.json', 'w') as outfile:
    json.dump(control, outfile)
```

Alternatively, the splitting can be made using the code proposed by lolipopshock

python cocosplit.py
–annotation-path ../destination/result.json
–split-ratio 0.83
–train ../destination/train.json
–test ../destination/test.json

3 Training and evaluation

We used the GPUs provided by Google Cloud. These are the steps followed.

1. Choose a 100GB disk with the Debian image for ML
2. Be sure that Nvidia drivers are in. If not, look for the Nvidia drivers
   https://la.nvidia.com/content/DriverDownloads/confirmation.php?url=/tesla/460.106.00/NVIDIA-Linux-x86_64-460.106.00.run&lang=us&type=Tesla
   sudo chmod +x filename.run
3. Be sure to have the latest cuda 11.3 wget
   https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_64/cuda-keyring_1.0-1_all.deb
   sudo dpkg -i cuda-keyring_1.0-1_all.deb sudo apt-get update sudo
   apt-get -y install cuda
4. Install torch. pip install torch==1.10.1 torchvision==0.11.2 torchaudio==0.10.1 conda install pytorch==1.10.1 torchvision==0.11.2 torchaudio==0.10.1 cudatoolkit=11.3 -c pytorch -c conda-forge
5. Install detectron2 python3 -m pip install detectron2 -f https://dl.fbaipublicfiles.com/detectron2/wheels/cu113/torch1.10/index.html
6. Use the interaction with bucket gsutil -m cp -r gs://... /home/.../
7. There might be a mistake in setup utils; remove it. AttributeError: module 'distutils' has no attribute 'version' https://stackoverflow.com/questions/70520120/attributeerror-module-setuptools-distutils-has-no-attribute-version
8. If there is this mistake ImportError: libtorch_cuda_cu.so: cannot open shared object file: No such file or directory Then check the installation of torch (step 4)
9. check if Imagesize and Funcy are present, if not pip install imagesize pip install funcy
10. run train_fourth.sh Note that the file train_net.py merges weights from "model_init.pth", the result of a previous iteration of this method.
11. See the results of the model in tensorboard

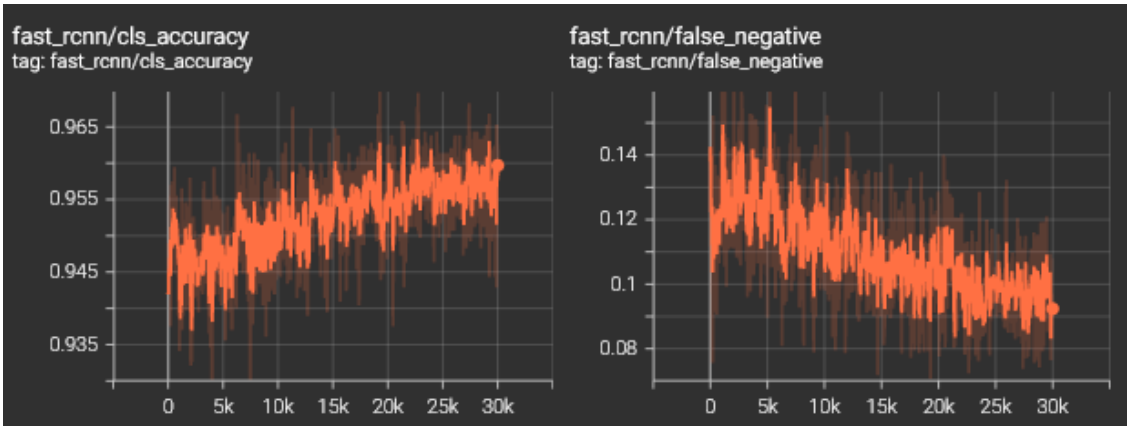
[ ]:
```
!tensorboard --logdir=../outputs/fourthtest/
```

```
TensorFlow installation not found - running with reduced feature set.

NOTE: Using experimental fast data loading logic. To disable, pass
    "--load_fast=false" and report issues on GitHub. More details:
    https://github.com/tensorflow/tensorboard/issues/4784

Serving TensorBoard on localhost; to expose to the network, use a proxy or pass
--bind_all
TensorBoard 2.11.0 at http://localhost:6006/ (Press CTRL+C to quit)
```

Some metrics



AP	AP50	AP75	APs	APm	APl
30.969	73.927	22.484	0.000	21.381	30.098

Category	AP
Author	22.473
Date	29.107
Page	13.324
Text	63.920
Title	26.023

4 Using the model to create csvs by volume.

This step can be done using a CPU instead of a GPU

run ocr_fourth.sh

This creates a list of .csv files containing the information for each volume.

Example of a layout by the model

312 THE ENGINEERING AND MINING JOURNAL. AUGUST 29, 1903.

VOLATILIZATION OF METALS AS CHLORIDES

BY STUART CROASDALE.

It is a well known fact that many of the metals can be volatilized as chlorides when heated in a current of muriatic acid or chlorine gas. This paper relates only to the volatilization of metals as chlorides from their ores during what is termed a chloridizing roast, or the roasting of ores with salt and sulphur in an oxidizing atmosphere.

The chloridizing roast has been in use for a long time for the treatment of silver ores preparatory to leaching with "hyposulphite" or salt solutions, or to pan amalgamation. It has also been in use a long time on low grade copper ores preparatory to leaching with water to extract the copper. In connection with these operations it has been noticed that the loss of gold by volatilization is so great that the use of this method is made prohibitory on ores containing appreciable quantities of that metal. The loss of silver has varied from 2 to 25 per cent, depending upon the temperature and method of roasting and the presence of other volatile chlorides. The volatilization of the base metals has been but little studied. Their presence in small quantities in dry silver ores is accepted as a necessary evil, and in chloridizing the ores of copper the loss of that metal is reduced to a minimum by careful roasting. In short, all the energy expended on this branch of metallurgy has been in efforts to prevent losses by volatilization.

During my connection with the Holden Lixiviation Works at Aspen, Colo., from 1891 to 1893, I had opportunity to study the volatilization of silver and lead in the chloridizing roast with a Stetefeldt furnace. It was also my fortune while there to conduct the original experiments on barrel chlorination for Cripple Creek ores. During the course of these experiments I had occasion to notice the volatilization of gold when subjected to a chloridizing roast, the details of which will be mentioned later. Circumstances prevented further work in this line until 1896, when, in conjunction with Mr. Edwin C. Pohle, who had also done some preliminary work in this line, we conducted a systematic series of experiments to determine the extent to which metals could be volatilized by a chloridizing roast. These experiments have since been continued on a large scale by myself, together with Mr. E. N. Hawkins and others interested in the commercial development of these reactions. The ease with which the metals referred to seemed to take a gaseous form indicated that this branch of metallurgy along the old lines was in opposition to the laws of nature, and that the proper course would be to let nature have her own way and volatilize all the metals as chlorides and then collect them from the fumes, thereby eliminating the cost of treating the ores after roasting.

These early experiments were conducted in the muffle of an assay furnace. The charges consisted of 50 to 100 grams of ore with the requisite amounts of salt and sulphur which were charged at once into a temperature of about 1000° C., either in a roasting dish or on the floor of the muffle. Chemical action began almost immediately and fumes began to come off. During the first part of the roast the ore was frequently rabbled to break up the loose crust that formed. After that there was seldom any tendency to form lumps unless the character of the ore made it easily fusible.

In all experiments the ore was carefully weighed before and after roasting and the necessary corrections were made in the assays to render the results comparable.

The experiments on gold ores are given somewhat in detail, since they were of a preliminary nature and formed a basis for subsequent work on other metals.

Gold.—Of all the metals tested this seemed to be the most readily volatilized, with a possible exception of lead. This is best illustrated by the original experiment to which reference has already been made. For roasting Cripple Creek ores I had a furnace that would hold a charge of 200 lb. During the course of the chlorination experiments it was desired to do some experimental work on chloridizing silver ores by roasting with salt, which was done in this furnace.

On resuming the chlorination experiments, the furnace was carefully scraped out and a charge of Cripple Creek ore put in. After roasting it was found to have lost over 50 per cent of the gold. The next charge lost over 25 per cent of the gold. This was accomplished by the salt remaining in the hearth of the furnace, or perhaps not more than a quarter of a pound for the 200 pounds of ore. When the furnace was re-lined there was no loss of gold on roasting subsequent charges of ore.

While this confirms the experience of Aaron and others, I mention it to show the ease with which gold is volatilized. The following experiments were made on a Cripple Creek ore having the following value and approximate composition:

Gold	2.14 oz. per ton.
Sulphur	1.30 per cent.
Silica	55 per cent. approx.
Alumina	30 per cent. approx.

Experiment A—Salt Variation.—Additional sulphur was used in the form of pyrite, making the total sulphur 3.2 per cent.

Per cent. salt used.	Assay roasted ore, oz.	Per cent. volatilization.
5	0.10	95.1
10	0.06	97.2
15	0.05	97.7
20	0.04	98.1
25	0.02	99.0

Experiment B—Sulphur Variation.—Ten per cent salt giving good results, that amount was used as a common quantity for the sulphur variation, which was as follows:

Per cent. total sulphur.	Assay roasted ore, oz.	Per cent. volatilization.
1.3	0.04	98.1
2.3	0.026	98.8
3.2	0.06	97.2
5.2	0.72	66.3
7.2	0.84	60.7
9.1	1.20	43.9

These two series of experiments show very clearly the necessary relation of salt to sulphur, which is 3 per cent salt to 2.3 per cent (or 2.5 per cent) sulphur, or the following chemical reaction:

$$2NaCl + S + 2O_2 = Na_2SO_4 + Cl_2$$

It will be shown later that these compounds also bear a direct relation to the volatile base metals in the ore. With gold and silver ores the theoretical amounts of salt and sulphur necessary to combine with these metals to form chlorides would be so small that it would be impossible, in a roasting operation; to secure physical contact of all the particles with each other, in order to obtain a complete chemical reaction. For this reason it was found necessary to use an excess of the theoretical amounts of both salt and sulphur where gold and silver are the only volatile metals in the ore.

Experiment C—Time Variation.—Using the above ratio for the salt and sulphur, the rate of volatilization for gold was determined as follows:

Time of roast, minutes.	Assay roasted ore, oz.	Per cent. volatilization.
10	0.97	54.7
20	0.34	84.1
30	0.21	90.0
40	0.08	96.2
50	0.06	97.2
60	0.05	97.7

It is well known that chloride of gold decomposes at a temperature far below the temperature at which it was volatilized in these experiments (about 1000° C.), yet there can be but little doubt that it was volatilized as a chloride or possibly as a double chloride with the sodium. At what point decomposition takes place is uncertain, but subsequent experiments on a larger scale (in plants treating ore by the ton) have revealed the fact that both in the dry fumes deposited in the flues near the furnace and in the fumes condensed by water 100 ft. from the furnace the gold occurs in metallic form, totally insoluble in water or in the acidified condensing solutions.

The ore used in these experiments varied from 80 to 20 mesh in size.

The following table shows the volatilization of gold from different ores. Those marked with a * were lots of a ton or more treated in a reverberatory or White-Howell furnace.

Locality.	Character of Ore.	Oz. Gold per Ton Raw Ore.	Oz. Gold per Ton Roasted Ore.	Per cent. Volatilization.
Cripple Creek, Colo.	Silica and alumina.	0.72	0.04	94.4
*Cripple Creek	Silica and alumina.	1.25	0.08	93.6
Georgia	Iron pyrite.	1.28	0.06	95.3
Nevada	Oxidized arsenical.	0.68	0.01	98.7
British Columbia	Sulphide arsenical.	3.34	0.02	99.4
New Mexico	Iron pyrite.	0.40	0.08	80.0
Utah	Iron pyrite.	0.60	0.10	83.3
*Boulder Co., Colo.	Iron pyrite.	0.40	0.04	90.0
Montana	Mixture of ores.	1.02	0.04	96.1
[illegible]	Siliceous lime.	[illegible]	[illegible]	[illegible]
Utah	Siliceous lime.	0.48	0.01	97.9
Mexico	Siliceous lime.	1.00	0.06	94.0

Silver.—The chloridization of silver in ores by means of salt is, perhaps, one of the oldest of metallurgical processes. It is sometimes done in the wet way on the raw ore and at other times it is done by the chloridizing roast. The latter method is treated extensively in Stetefeldt's "Lixiviation of Silver Ores," second edition, which is largely a record of the work done at the Holden mill, at Aspen, Colo., during my connection with that company. Hoffman has also published[1] a record of most careful experiments on this subject. Both of these contributions treat of the efforts to effect chloridization with the least possible volatilization of the silver.

At the Holden mill a Stetefeldt furnace was used, in which not more than 40 per cent of the silver was chloridized, owing to the short exposure to the heat, but the oxidation of the pyrite was effected so that when the ore was drawn from the furnace and piled on the cooling floor the chemical action continued for several days, chloridizing an additional 30 to 40 per cent of the silver.

Hoffman used reverberatory and Bruckner furnaces and found that when insufficient air was admitted to the furnace the chloridization was not good and the loss of silver was increased.

My experiments were confined to higher temperatures and an effort to secure a high volatilization of the silver. This was best accomplished by admitting an *excess* of air to the roasting ore, and increasing its effect by constantly rabbling the ore, or otherwise exposing a fresh surface to the atmosphere.

My theory of this is that when a particle of salt comes in contact with a particle of sulphide of silver under these conditions (plenty of air and high temperature), the chemical action is not only very rapid but the temperature is sufficiently high to volatilize the chloride of silver as soon as it is formed, instead of allowing it to melt, thus leaving a fresh surface of the sulphide continually exposed until it is all volatilized.

Other chlorides, such as those of the base metals, or extra salt assist in the volatilization of silver chloride.

Silver occurs in the fume as the chloride or subchloride, which is totally insoluble in water.

To secure good results, silver ores require finer crushing than any of the ores of the other metals. Even with the same general character of ores the results are sometimes irregular under the same treatment. The cause of this has not been thoroughly investigated, but since antimony was found to exert a notable action against the volatilization of silver, I attribute most of the trouble in such ores to antimonial silver compounds.

The following list shows the effect of the chloridizing roast on the so-called "dry" silver ores. The lead-silver, copper-silver and gold-silver ores are given under "Complex Ores":

Locality.	Character of ore.	Oz. silver per ton raw ore.	Oz. silver per ton roasted ore.	Per cent. volatilization.
Montana	Siliceous	10.0	1.9	81.0
Mexico	Siliceous	26.0	2.1	91.9
Mexico	Siliceous mixture.	27.5	1.0	96.4
Mexico	Siliceous	27.0	3.1	88.5
Mexico	Siliceous	101.44	18.3	81.9
Nevada	Siliceous	25.9	1.6	93.8
Nevada	Siliceous	136.0	10.6	93.2
Colorado	Lime and baryta.	234.0	2.3	99.0
Colorado	Silica and talc.	19.0	0.9	95.3
Utah	Siliceous	186.0	30.9	83.9

[1] "Mineral Industry," 1896.

5 Creating a database.

The file pag.R compiles the database on a single file. The main objective is to use regular expressions to find text breaks that the layout model does not identify. As for our purposes, we prefer stricter definitions of contextual information; we introduce a variable named subtitle to compile extracts in the same general title. In any case, this code can be modified in cases where the journals use more

extensive articles and fewer small vignettes with news.

6 Testing final result.

As a result of the database, we created a randomized sample at 95% significance (385 observations) that was tested by a human eye against the original document's layout. Here are the results

Item	Errors	Mistakes	Accuracy
Type	7	1.82%	98.18%
Page	73	18.96%	81.04%
Date	2	0.52%	99.48%
Author	12	3.12%	96.88%
Title	35	9.09%	90.91%
Subtitle	55	14.29%	85.71%
Text (under)	12	3.12%	94.81%
Text (over)	8	2.08%	
Overall	204	6.62%	93.38%

The results show that the categories with the most mistakes are Page and Subtitle, while the ones with fewer errors are Type and Date. We divided the mistakes on the Text into two categories, one counting the times when the cell did not capture the whole piece of information (under) and a second one measuring the observation when the cell captured data from a different title. As shown in the results, the code privileged the reduction of errors that mixed information. The accuracy of the Text category is 94.81%, and the overall accuracy of the dataset is 93.38%.