AVANTIX

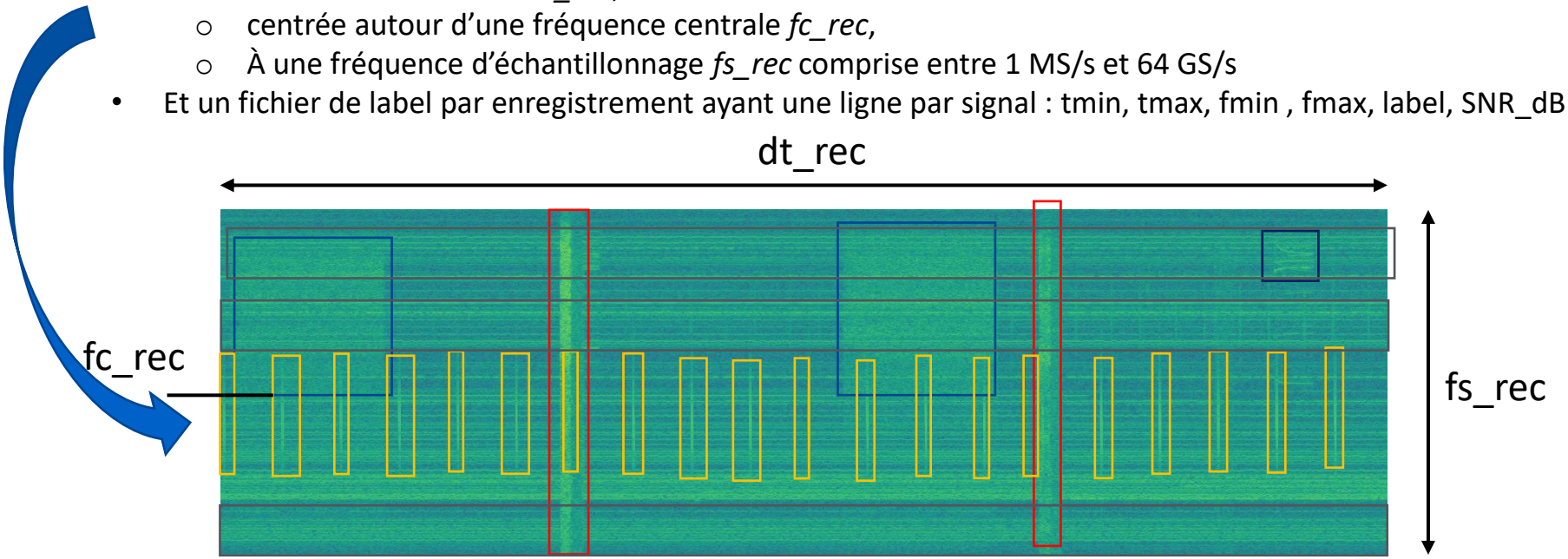
Avancement Simulateur IQ

20/12/2023

Besoin

Avoir un programme qui génère en sortie :

- un enregistrement IQ
  - d'une certaine durée *dt_rec*,
  - centrée autour d'une fréquence centrale *fc_rec*,
  - À une fréquence d'échantillonnage *fs_rec* comprise entre 1 MS/s et 64 GS/s
- Et un fichier de label par enregistrement ayant une ligne par signal : tmin, tmax, fmin , fmax, label, SNR_dB



Simulateur

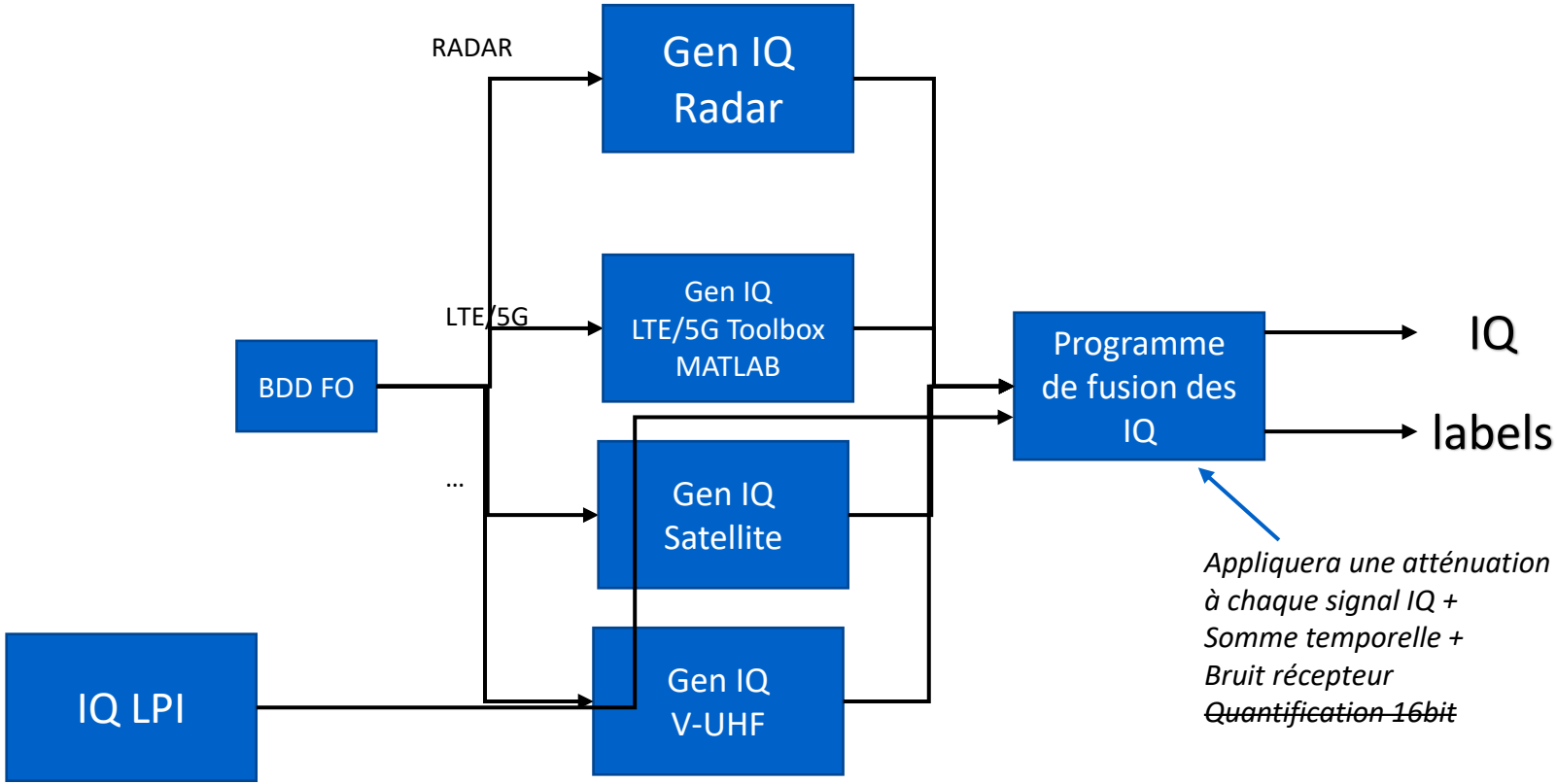
Ce que le programme fera :

- Capable de générer un signal IQ des FO suivantes :
  - RADAR
  - LTE/5G
  - V-UHF
  - LoRA bande ISM
  - Satellite :
    - THURAYA
    - INMARSAT
    - IRIDUM
    - OneWeb
    - Starlink

Ce que le programme ne fera pas :

- Gérer les cas spécifiques à chaque protocole de communication
- Préfiltrer des FO pour des raisons d'occupation spectrale irréaliste
- Pondérer de manière réaliste les puissances de réception

Spécifications fonctionnelles



Spécifications fonctionnelles

```
import pickle
from iq_sigint_simulator import iq_sigint_simulator

sig_types=['cellular', 'radar']
iq_rec, labels, detailed_labels = iq_sigint_simulator(fc_rec, fs_rec, dt_rec,
sigma_noise, sig_types)

pickle.dump((record, labels), open('record.pkl', 'wb'))
```

- `fc_rec` : Fréquence centrale d'écoute
- `fs_rec` : Fréquence d'échantillonnage de la bande d'écoute
- `dt_rec`: Durée en secondes de l'écoute
- `sigma_noise`: Ecart-type de l'amplitude crête du bruit en Volt
- `sig_types` : Familles des formes d'ondes à simuler

Spécification en cours de rédaction :
\\srv-filer01.global.ad\AFFAIRES_TELECOM\1-10-R&D-2200\TECHNIQUE\DOCUMENTS\2_Spécification\DP078293STB004_00-Specification Simulateur R&D ELINT 2023.docx

Spécifications fonctionnelles

L'utilisateur renseigne les familles de formes d'ondes (cellulaire ou RADAR) qu'il souhaite générer sur une fenêtre temps-fréquence.
La fonction *iq_sigint_generator* va ensuite piocher aléatoirement parmi les FO éligibles et générer la fusion de tous les IQ.

Contraintes d'éligibilités :

- RADAR: Génère entre 200k et 1M d'impulsions par seconde pour une bande d'écoute de 500 MHz

- Cellulaire : Un seuil d'occupation spectrale est renseigné par l'utilisateur en ratio

SNR : Cas taille fft optimal

Pour chaque signal, un SNR est choisi aléatoirement entre 3dB et 40dB en apprentissage. On peut envisager sur la base de test entre -10dB et 40dB.
A partir de la puissance de bruit et du SNR, l'amplitude crête du signal est calculé.

$$SNR_{dB} = DSP_{x_{dB}} - DSP_{N_{dB}}$$

$$\Longrightarrow DSP_{x_{dB}} = SNR_{dB} + DSP_{N_{dB}}$$

$$\Longrightarrow DSP_x = 10^{\frac{SNR_{dB}}{10}} * \frac{\sigma_N^2}{fs}$$

$$\Longrightarrow \frac{A^2}{2} * k_{type} = 10^{\frac{SNR_{dB}}{10}} * \frac{\sigma_N^2}{fs}$$

$$\Longrightarrow \boldsymbol{A = \sqrt{\frac{2}{fs * k_{type}} 10^{\frac{SNR_{dB}}{10}}} * \sigma_N}$$

Avec $k_{type} = \begin{cases} \sqrt{\frac{pw}{\Delta f}} \ si\ chirp\ radar \\ pw_{moment}\ si\ pmop\ radar \\ pw \ \ si\ radar\ no\ mod\ ou\ cw \\ \frac{1}{\Delta f} \ \ sinon \end{cases}$

SNR effectif

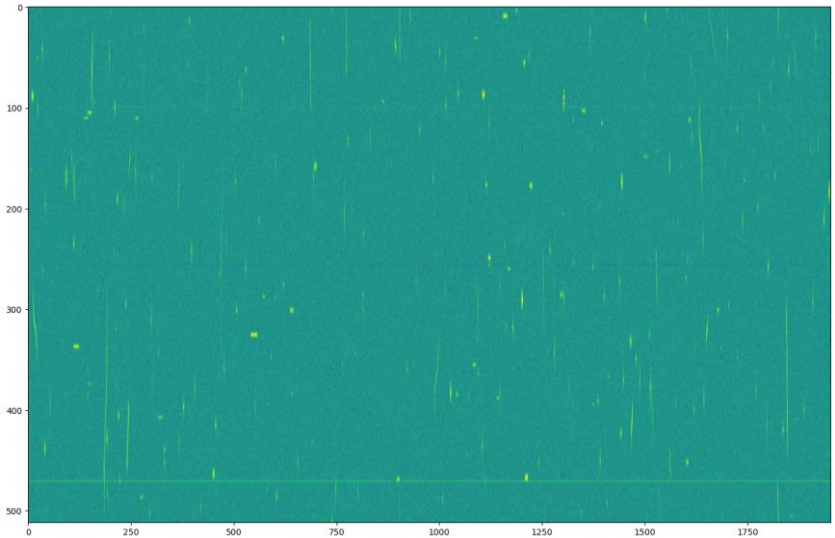
Dans la fenêtre de la bounding box, on récupère la valeur max en dB du spectrogramme (Sxx) que l'on soustrait à la DSP du bruit.

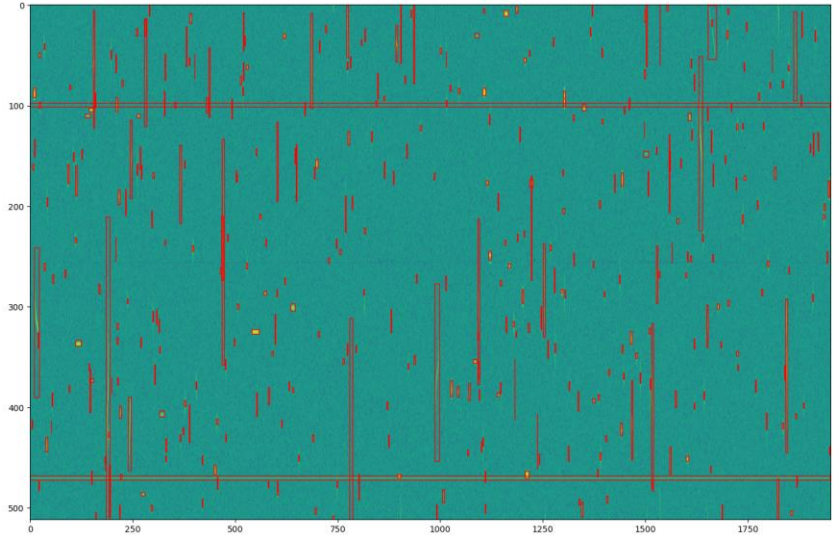
$$SNR_{eff\,dB} = \max(Sxx_{dB}[x_{min}:x_{\max}; y_{min}:y_{\max}]) - DSP_{N\,dB}$$

On ne conserve que les SNR supérieur à 1dB.

Labelizer

Implémentation d'une fonction permettant de générer des bounding box quelques soit les paramètres du spectrogramme.





Exemple : nfft = 512, Overlap = 50%, SNR_eff_min = 1dB

Génération BT 0.1.0 : en cours

Paramètres :

fc_rec = 1GHz
fs_rec = [1, 5, 25, 100, 500, 1e3, 2e3, 4e3] MS/s
dt_rec = 1ms
dt_total = 1min → 60k enregistrements
sigma_noise = 1mV
sig_types = ‘radar’
snr_db = 3 à 40dB

Durée simulation = 60h

Taille de fft optimal

No mod

fs_rec en MS/s	nfft min	pw min en ns	nfft max	pw max en ms
1	2	2000	1024	1,024
5	2	400	4096	0,8192
25	2	80	16384	0,65536
100	2	20	65536	0,65536
500	4	8	524288	1,048576
1000	8	8	1048576	1,048576
2000	16	8	2097152	1,048576
4000	32	8	4194304	1,048576

f mod

fs_rec en MS/s	nfft min	sqrt(pw_/df)_min en ns	nfft max	sqrt(pw_/df)_max en µs
1	2	2000	256	256
5	2	400	1024	204,8
25	2	80	4096	163,84
100	8	80	16384	163,84
500	64	128	131072	262,144
1000	128	128	262144	262,144
2000	256	128	524288	262,144
4000	512	128	1048576	262,144

p mod

fs_rec en MS/s	nfft min	pw_moment min en ns	nfft max	pw_moment max en µs
1	2	2000	128	128
5	2	400	512	102,4
25	2	80	4096	163,84
100	8	80	16384	163,84
500	64	128	65536	131,072
1000	128	128	131072	131,072
2000	256	128	262144	131,072
4000	512	128	524288	131,072

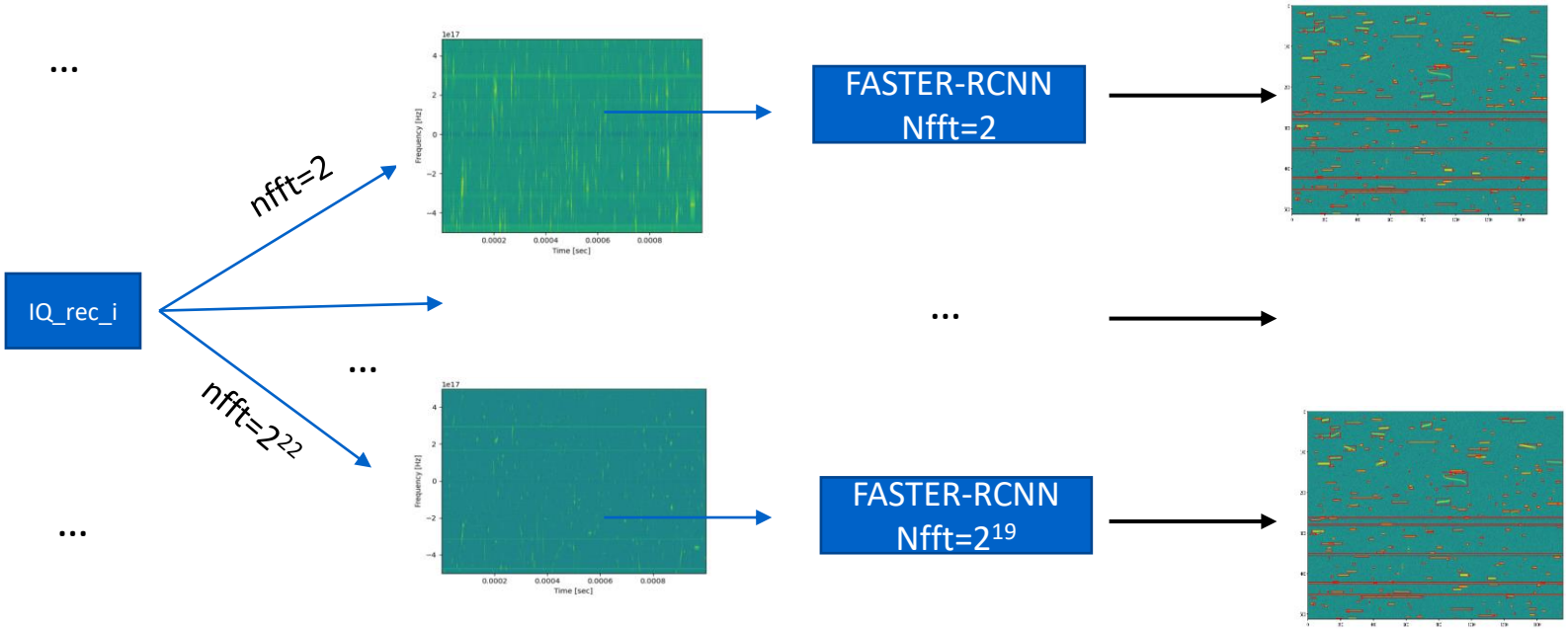
Résumé

fs_rec en MS/s	nfft_min	log2(nfft_min)	nfft_max	log2(nfft_max)
1	2	1	1024	10
5	2	1	4096	12
25	2	1	16384	14
100	2	1	65536	16
500	4	2	524288	19
1000	8	3	1048576	20
2000	16	4	2097152	21
4000	32	5	4194304	22

FFT qui seront réalisées :
2, 4, 8, 16, 32, 64, 128, 256, 512, 2048, 8192, 32768, 131072, 524288

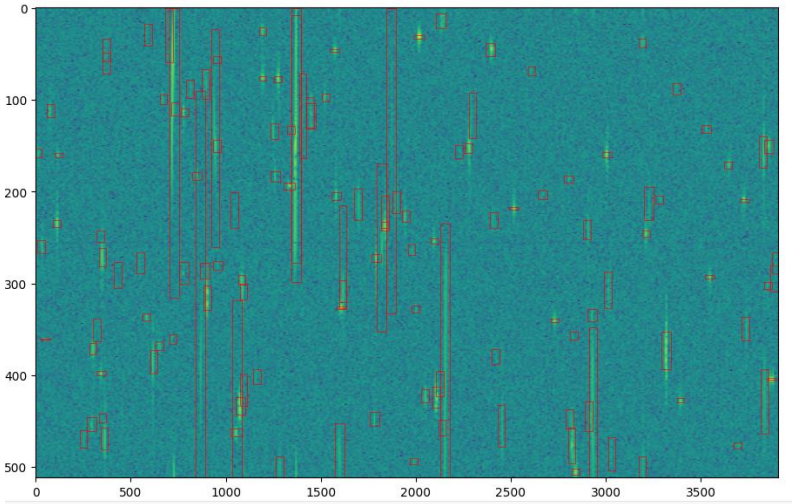
Détection (Etat de l'art en cours)

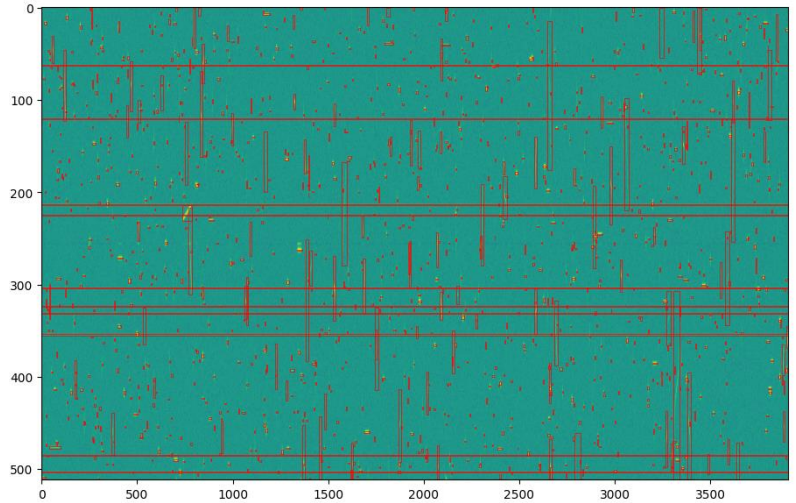
Un modèle par taille de FFT



Extrait BT0.1.0 | NFFT=512

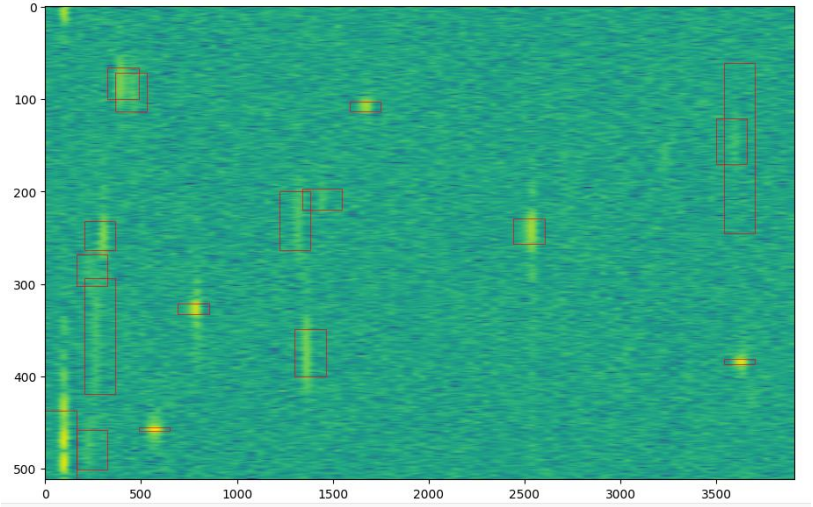
- FS_REC = {25; 100; 500; 1000} MS/s
- NFFT = 512 | NOVERLAP = 256
- Format image : (512, 3905) UINT8-RGB ou UINT16-GRAY
- Train : 6172 | Val : 1508

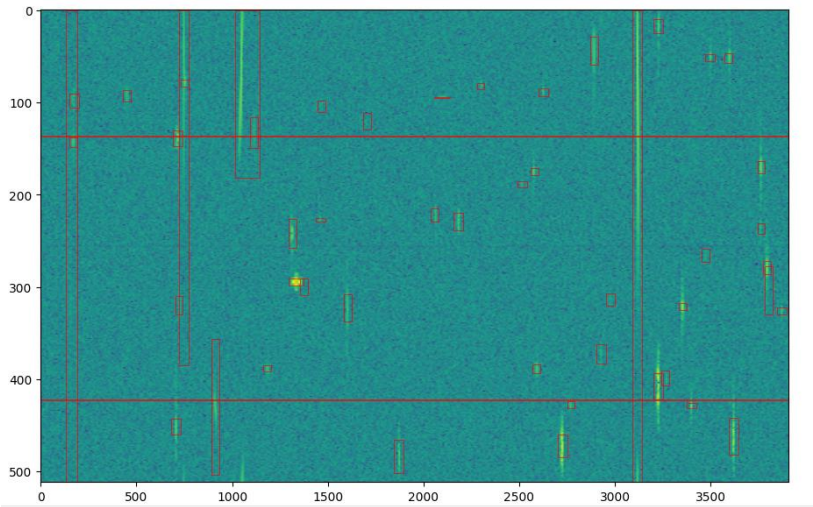




Extrait BT0.1.0 | NFFT=512

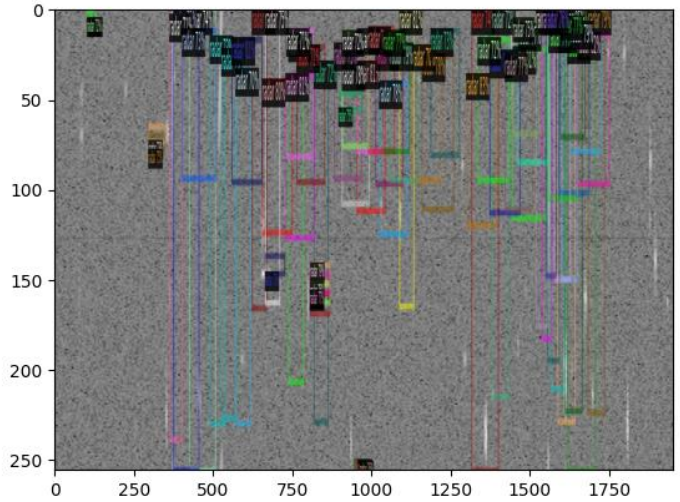
- FS_REC = {25; 100; 500; 1000} MS/s
- NFFT = 512 | NOVERLAP = 256
- Format image : (512, 3905) UINT8-RGB ou UINT16-GRAY
- Train : 6172 | Val0 : 1508





Detrectron2: Faster-RCNN

```
[12/19 10:23:20 d2.data.common]: Serialized dataset takes 272.93 MiB
[12/19 10:23:20 d2.data.build]: Making batched data loader with batch_size=2
[12/19 10:23:20 d2.checkpoint.detection_checkpoint]: [DetectionCheckpointer] Loading from https://dl.fbaipublicfiles.com/detectron2/COCO-Detection/faster_rcnn_R_50_FPN_3x/137849458/model_final_280758
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
Skip loading parameter 'roi_heads.box_predictor.cls_score.weight' to the model due to incompatible shapes: (81, 1024) in the checkpoint but (2, 1024) in the model! You might want to double check if
Skip loading parameter 'roi_heads.box_predictor.cls_score.bias' to the model due to incompatible shapes: (81,) in the checkpoint but (2,) in the model! You might want to double check if this is expe
Skip loading parameter 'roi_heads.box_predictor.bbox_pred.weight' to the model due to incompatible shapes: (320, 1024) in the checkpoint but (4, 1024) in the model! You might want to double check if
Skip loading parameter 'roi_heads.box_predictor.bbox_pred.bias' to the model due to incompatible shapes: (320,) in the checkpoint but (4,) in the model! You might want to double check if this is exp
Some model parameters or buffers are not found in the checkpoint:
roi_heads.box_predictor.bbox_pred.{bias, weight}
roi_heads.box_predictor.cls_score.{bias, weight}
[12/19 10:23:21 d2.engine.train_loop]: Starting training from iteration 0
/home/chakib/miniconda3/envs/sigint/lib/python3.9/site-packages/torch/functional.py:504: UserWarning: torch.meshgrid: in an upcoming release, it will be required to pass the indexing argument. (Tri
  return _VF.meshgrid(tensors, **kwargs)  # type: ignore[attr-defined]
[12/19 10:23:25 d2.utils.events]:  eta: 0:02:17  iter: 19  total_loss: 11.84  loss_cls: 0.654  loss_box_reg: 0.02148  loss_rpn_cls: 8.764  loss_rpn_loc: 2.388    time: 0.1787  last_time: 0.1117  dat
[12/19 10:23:29 d2.utils.events]:  eta: 0:02:12  iter: 39  total_loss: 6.515  loss_cls: 0.6022  loss_box_reg: 0.005338  loss_rpn_cls: 3.825  loss_rpn_loc: 2.346    time: 0.1772  last_time: 0.1864  d
[12/19 10:23:32 d2.utils.events]:  eta: 0:02:30  iter: 59  total_loss: 3.983  loss_cls: 0.5866  loss_box_reg: 0.003032  loss_rpn_cls: 0.8431  loss_rpn_loc: 2.68    time: 0.1786  last_time: 0.2511  d
[12/19 10:23:36 d2.utils.events]:  eta: 0:02:31  iter: 79  total_loss: 3.704  loss_cls: 0.54  loss_box_reg: 0.002461  loss_rpn_cls: 0.6375  loss_rpn_loc: 2.447    time: 0.1821  last_time: 0.1859  da
[12/19 10:23:40 d2.utils.events]:  eta: 0:02:28  iter: 99  total_loss: 3.333  loss_cls: 0.5257  loss_box_reg: 0.001912  loss_rpn_cls: 0.5566  loss_rpn_loc: 2.195    time: 0.1844  last_time: 0.2088
[12/19 10:23:44 d2.utils.events]:  eta: 0:02:21  iter: 119  total_loss: 3.609  loss_cls: 0.5374  loss_box_reg: 0.001718  loss_rpn_cls: 0.5114  loss_rpn_loc: 2.549    time: 0.1846  last_time: 0.2500
[12/19 10:23:47 d2.utils.events]:  eta: 0:02:12  iter: 139  total_loss: 3.327  loss_cls: 0.5054  loss_box_reg: 0.0015  loss_rpn_cls: 0.5214  loss_rpn_loc: 2.285    time: 0.1812  last_time: 0.2147  d
[12/19 10:23:50 d2.utils.events]:  eta: 0:02:08  iter: 159  total_loss: 3.07  loss_cls: 0.4981  loss_box_reg: 0.001254  loss_rpn_cls: 0.5371  loss_rpn_loc: 2.081    time: 0.1796  last_time: 0.3042
[12/19 10:23:54 d2.utils.events]:  eta: 0:02:05  iter: 179  total_loss: 3.856  loss_cls: 0.511  loss_box_reg: 0.001384  loss_rpn_cls: 0.5009  loss_rpn_loc: 2.824    time: 0.1805  last_time: 0.0829
[12/19 10:23:58 d2.utils.events]:  eta: 0:02:02  iter: 199  total_loss: 3.392  loss_cls: 0.4963  loss_box_reg: 0.001284  loss_rpn_cls: 0.4857  loss_rpn_loc: 2.409    time: 0.1826  last_time: 0.1804
[12/19 10:24:02 d2.utils.events]:  eta: 0:02:00  iter: 219  total_loss: 3.848  loss_cls: 0.4718  loss_box_reg: 0.001435  loss_rpn_cls: 0.4597  loss_rpn_loc: 2.914    time: 0.1821  last_time: 0.2157
[12/19 10:24:05 d2.utils.events]:  eta: 0:01:57  iter: 239  total_loss: 3.344  loss_cls: 0.4738  loss_box_reg: 0.005334  loss_rpn_cls: 0.516  loss_rpn_loc: 2.337    time: 0.1805  last_time: 0.2591
[12/19 10:24:09 d2.utils.events]:  eta: 0:01:56  iter: 259  total_loss: 3.785  loss_cls: 0.448  loss_box_reg: 0.00197  loss_rpn_cls: 0.4643  loss_rpn_loc: 2.832    time: 0.1810  last_time: 0.2686  d
[12/19 10:24:12 d2.utils.events]:  eta: 0:01:53  iter: 279  total_loss: 3.412  loss_cls: 0.4438  loss_box_reg: 0.002529  loss_rpn_cls: 0.4845  loss_rpn_loc: 2.487    time: 0.1815  last_time: 0.3687
[12/19 10:24:15 d2.utils.events]:  eta: 0:01:47  iter: 299  total_loss: 3.546  loss_cls: 0.4261  loss_box_reg: 0.02888  loss_rpn_cls: 0.4935  loss_rpn_loc: 2.596    time: 0.1788  last_time: 0.1442
[12/19 10:24:19 d2.utils.events]:  eta: 0:01:44  iter: 319  total_loss: 3.782  loss_cls: 0.44  loss_box_reg: 0.002813  loss_rpn_cls: 0.4511  loss_rpn_loc: 2.845    time: 0.1777  last_time: 0.3566  d
[12/19 10:24:22 d2.utils.events]:  eta: 0:01:41  iter: 339  total_loss: 4.171  loss_cls: 0.3974  loss_box_reg: 0.003872  loss_rpn_cls: 0.4378  loss_rpn_loc: 3.388    time: 0.1775  last_time: 0.3677
[12/19 10:24:26 d2.utils.events]:  eta: 0:01:38  iter: 359  total_loss: 3.75  loss_cls: 0.4193  loss_box_reg: 0.01197  loss_rpn_cls: 0.4575  loss_rpn_loc: 2.834    time: 0.1776  last_time: 0.1101  d
[12/19 10:24:29 d2.utils.events]:  eta: 0:01:34  iter: 379  total_loss: 3.485  loss_cls: 0.4088  loss_box_reg: 0.006607  loss_rpn_cls: 0.4815  loss_rpn_loc: 2.545    time: 0.1767  last_time: 0.1505
[12/19 10:24:33 d2.utils.events]:  eta: 0:01:31  iter: 399  total_loss: 3.616  loss_cls: 0.3829  loss_box_reg: 0.003764  loss_rpn_cls: 0.4681  loss_rpn_loc: 2.707    time: 0.1768  last_time: 0.3362
[12/19 10:24:36 d2.utils.events]:  eta: 0:01:28  iter: 419  total_loss: 4.064  loss_cls: 0.3381  loss_box_reg: 0.004002  loss_rpn_cls: 0.445  loss_rpn_loc: 3.331    time: 0.1765  last_time: 0.1609
[12/19 10:24:40 d2.utils.events]:  eta: 0:01:25  iter: 439  total_loss: 3.216  loss_cls: 0.3436  loss_box_reg: 0.01485  loss_rpn_cls: 0.4596  loss_rpn_loc: 2.367    time: 0.1770  last_time: 0.1901
[12/19 10:24:43 d2.utils.events]:  eta: 0:01:22  iter: 459  total_loss: 3.35  loss_cls: 0.3238  loss_box_reg: 0.01904  loss_rpn_cls: 0.5013  loss_rpn_loc: 2.542    time: 0.1770  last_time: 0.1861  d
[12/19 10:24:47 d2.utils.events]:  eta: 0:01:19  iter: 479  total_loss: 2.999  loss_cls: 0.3503  loss_box_reg: 0.004455  loss_rpn_cls: 0.4761  loss_rpn_loc: 2.135    time: 0.1774  last_time: 0.0944
[12/19 10:24:51 d2.utils.events]:  eta: 0:01:16  iter: 499  total_loss: 3.184  loss_cls: 0.3062  loss_box_reg: 0.004265  loss_rpn_cls: 0.4478  loss_rpn_loc: 2.436    time: 0.1773  last_time: 0.2203
...
[12/19 10:26:21 d2.data.common]: Serializing the dataset using: <class 'detectron2.data.common._TorchSerializedList'>
[12/19 10:26:21 d2.data.common]: Serializing 1505 elements to byte tensors and concatenating them all ...
[12/19 10:26:25 d2.data.common]: Serialized dataset takes 69.37 MiB
WARNING [12/19 10:26:26 d2.engine.defaults]: No evaluator found. Use `DefaultTrainer.test(evaluators=)`, or implement its `build_evaluator` method.
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

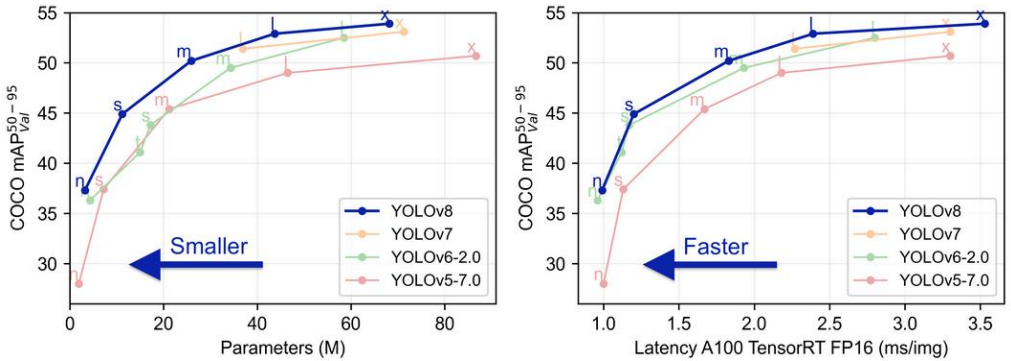


- Bias error : Courbe d’apprentissage qui ne converge pas
- Pas de maîtrise sur le format d’image. Possible que le modèle redimensionne et/ou cast

ultralytics: YOLOv8

https://github.com/ultralytics/ultralytics

Framework implémentant YOLOv8 en pytorch



Stockage

Etat stockage sur serveur avx-ai-server

```
Last login: Tue Dec 12 11:38:38 2023 from 10.106.42.4
(base) ➜  ~ df -h
Filesystem      Size  Used Avail Use% Mounted on
tmpfs            38G  3,6M   38G   1% /run
/dev/sda2       219G  166G   42G  80% /
tmpfs           189G   60M  189G   1% /dev/shm
tmpfs           5,0M     0  5,0M   0% /run/lock
/dev/sda1       511M  6,1M  505M   2% /boot/efi
tmpfs            38G   84K   38G   1% /run/user/128
tmpfs            38G  112K   38G   1% /run/user/1000
/dev/sdb1       3,6T  2,4T  1,1T  70% /data
tmpfs            38G   64K   38G   1% /run/user/1003
tmpfs            38G   64K   38G   1% /run/user/1004
(base) ➜  ~
```

Samsung 870 QVO MZ-77Q8T0BW | Disque SSD Interne 8 To, SATA III, 2,5'' - Technologie QLC seconde génération

AVANTIX

MERCI