
Content
Style
Pastiche

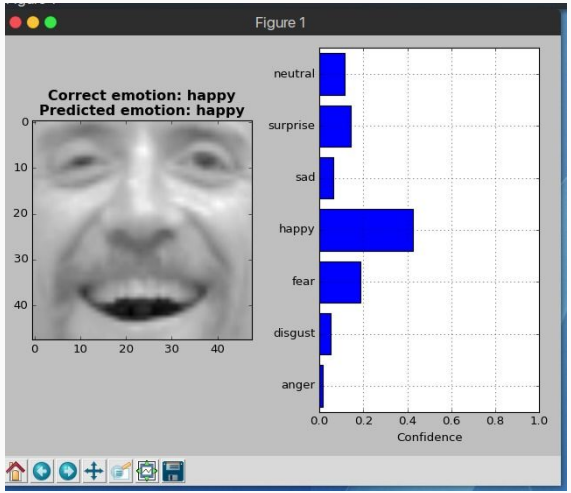


Figure 1
Correct emotion: happy
Predicted emotion: happy
neutral
surprise
sad
happy
fear
disgust
anger
Confidence

Exemples de projets avec du Machine Learning

Du Machine Learning dans Android

Partie I : Sans Datascientist - Maxime FERRIER

Maxime FERRIER

Ingénieur d’études

Développeur mobile Android & Hybride

Clients

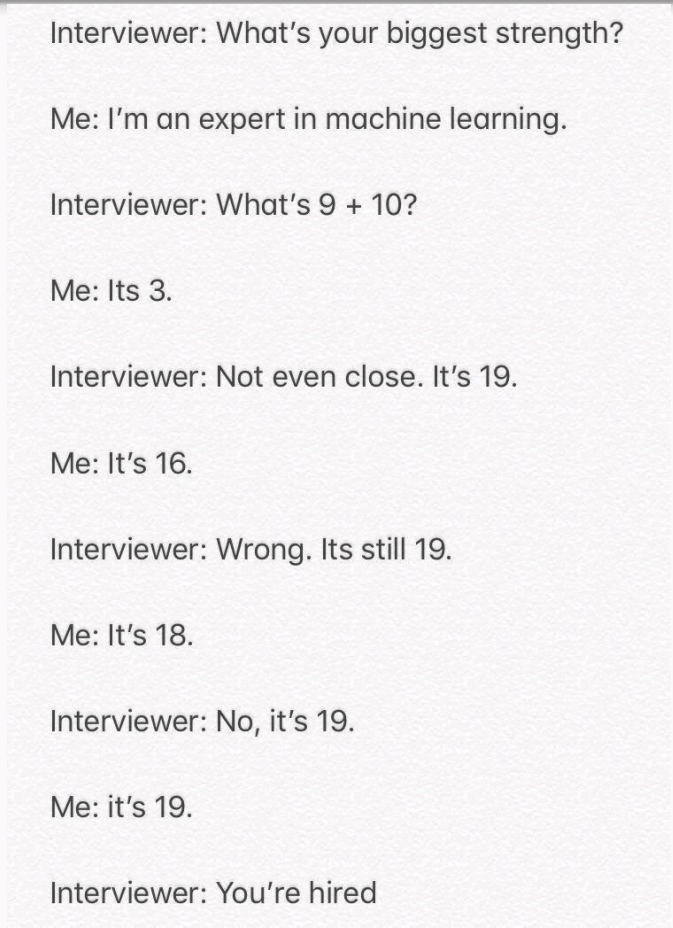
Contexte Socram Banque
Macif
Maif

"Mon conseil est que lorsque vous voulez vous lancer dans le machine learning, la 1ère chose que vous devez faire est de regarder s'il n'y a pas déjà des modèles pré-entraînés."

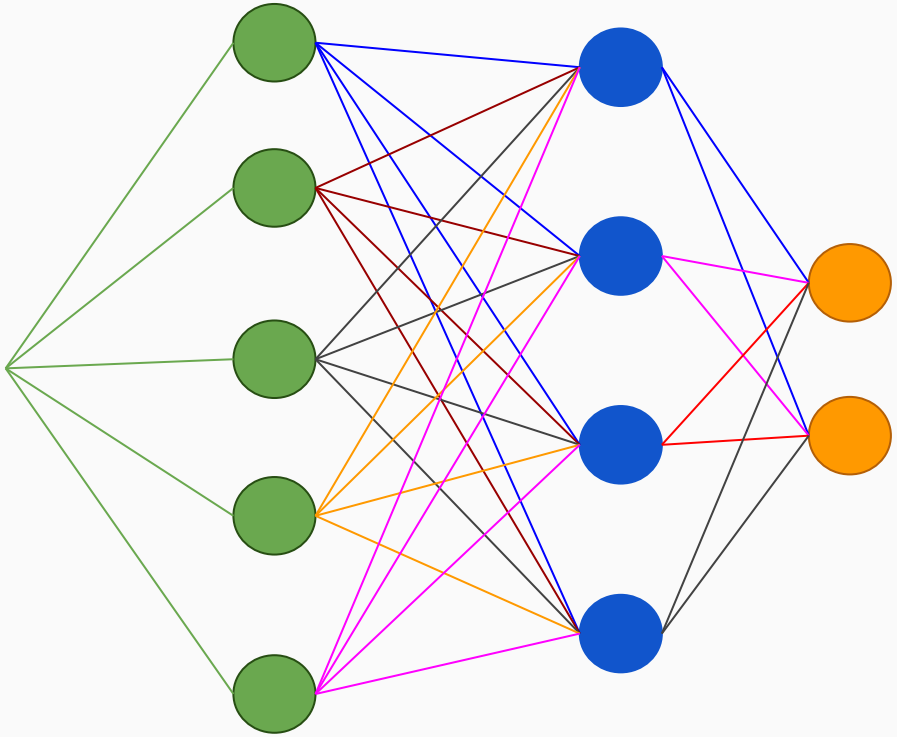
Définitions

Le Machine Learning est une branche de l'intelligence artificielle qui s'occupe de la construction et l'étude de systèmes qui apprennent à partir de données.

ML != IA



Définitions



Droid (0.96)

Pomme (0.04)

Outils

Kotlin

Outils

Tensorflow - Architecture

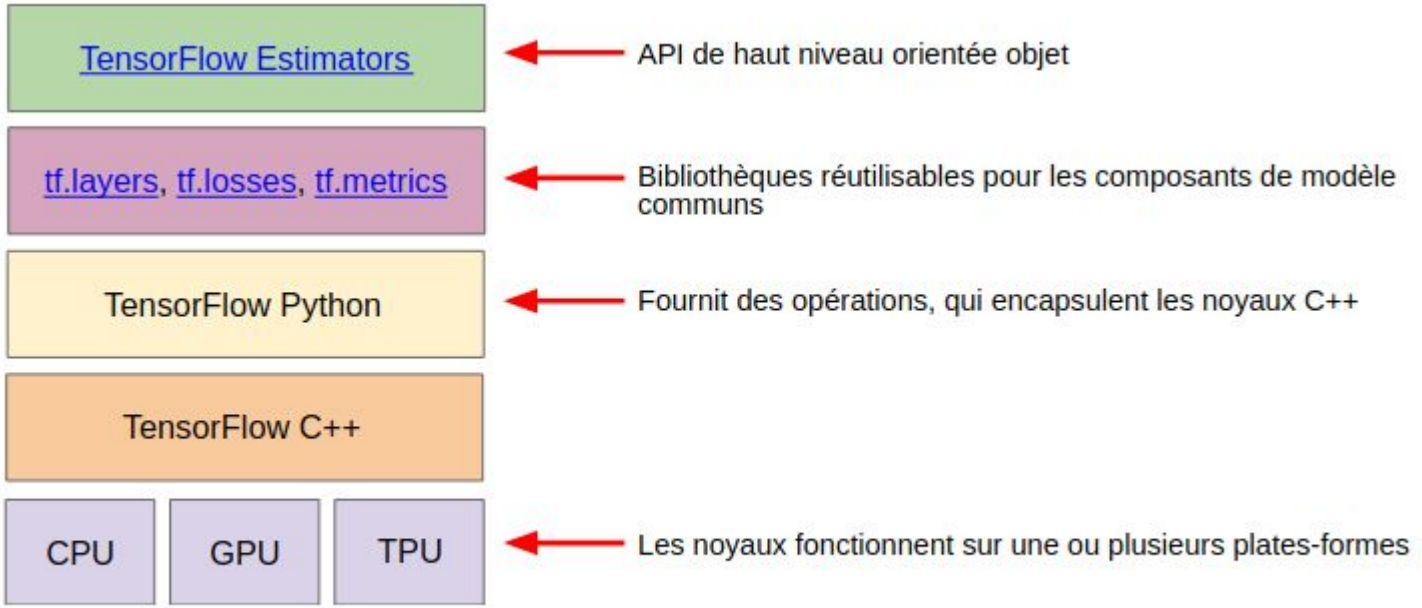
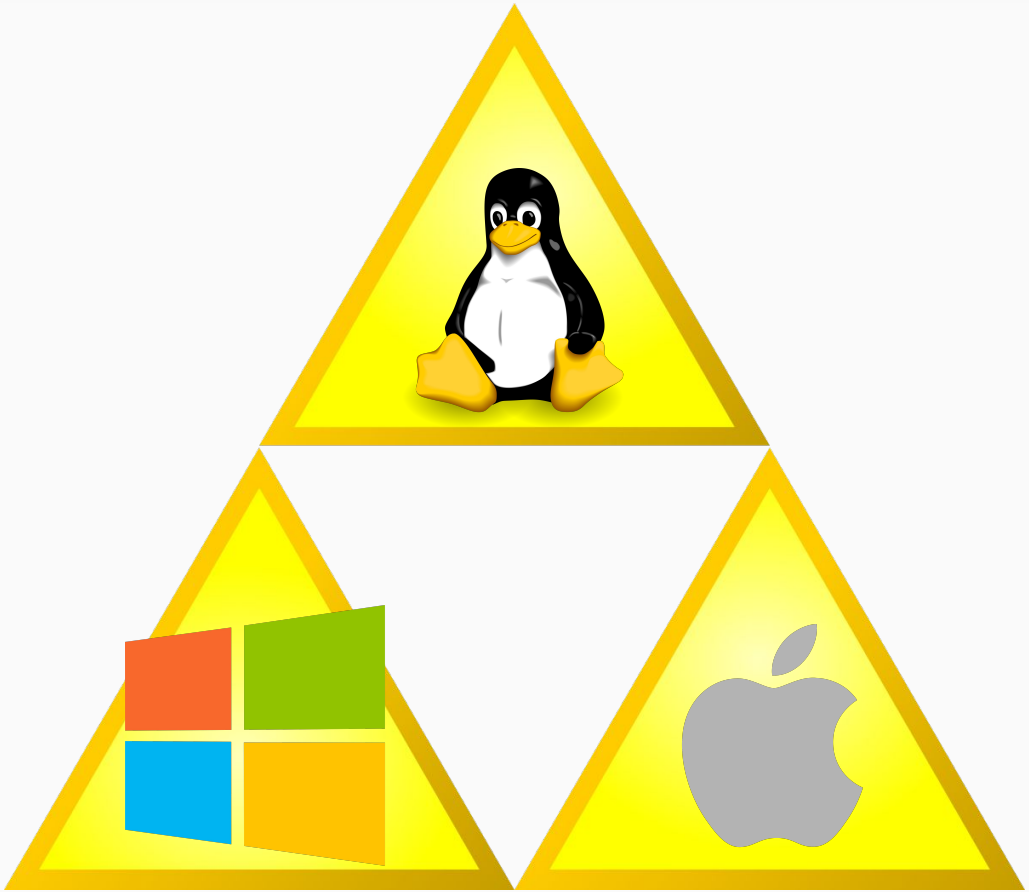


Figure 1 : Hiérarchie des kits de TensorFlow

Tensorflow - Installation



NVIDIA®

Tensorflow - Installation

```
$ pip install tensorflow
```

OU

```
$ pip install tensorflow-gpu
```

Tensorflow - Vérification

```
#!/usr/bin/python

import tensorflow as tf
hello = tf.constant('Hello, TensorFlow!')
sess = tf.Session()
print(sess.run(hello))
```

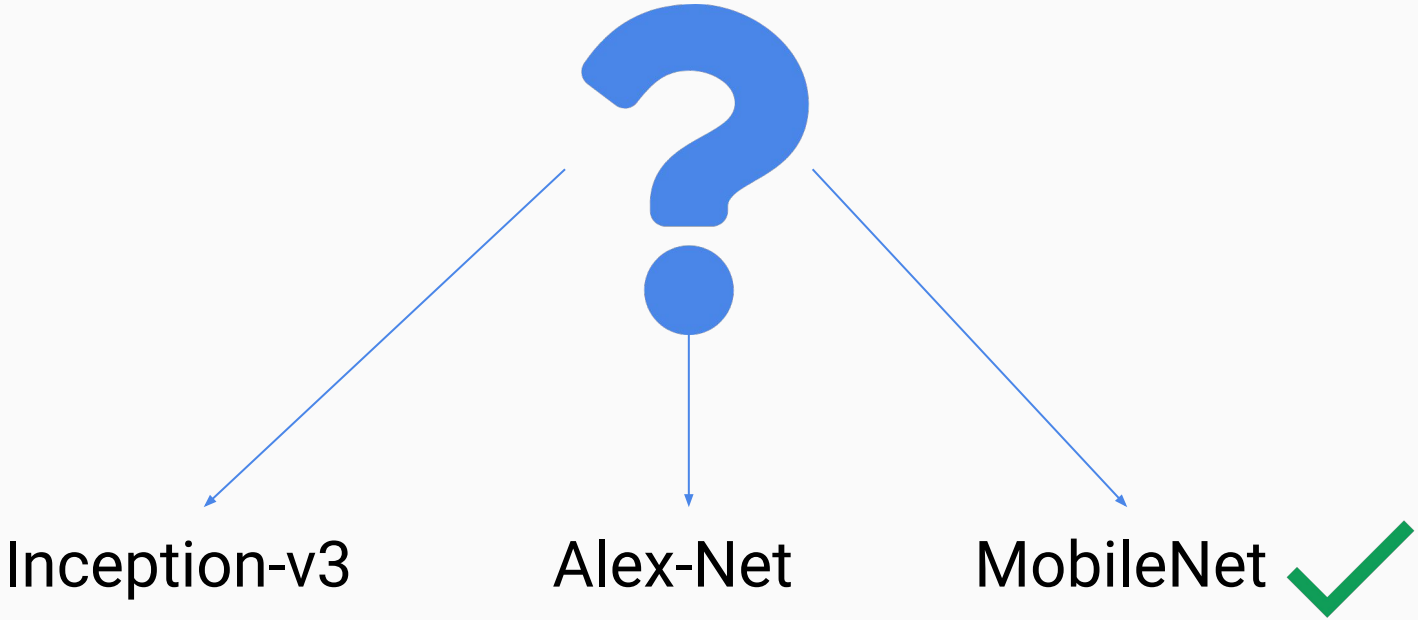
Cueillette de champignons

Application cible

Cueillette de champignons - Procédure

1 - Trouver un modèle
2 - Récupération de données
3 - Ré-entraîner le modèle
4 - Test
5 - Création de l’application

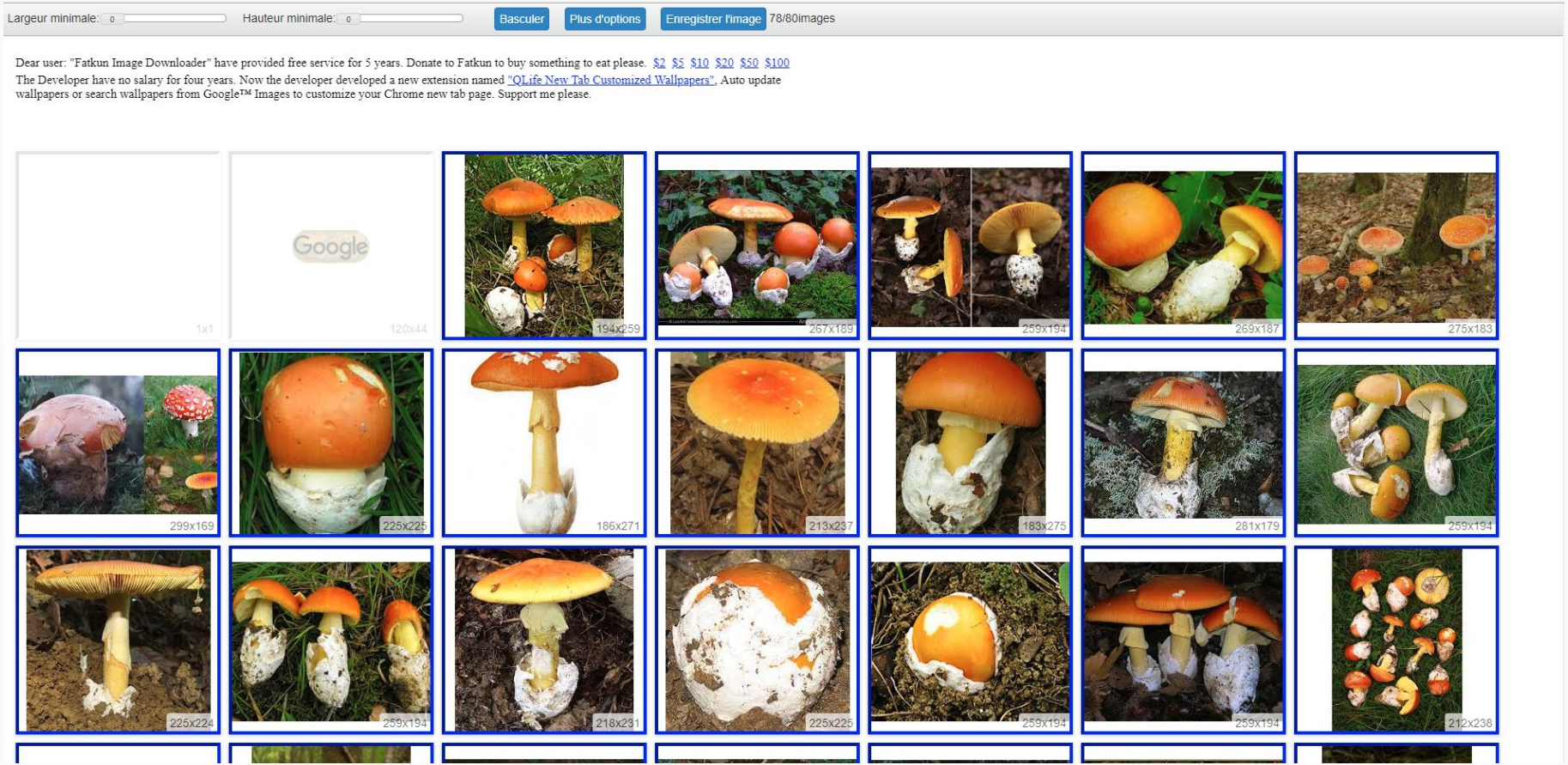
Cueillette de champignons - Trouver un modèle



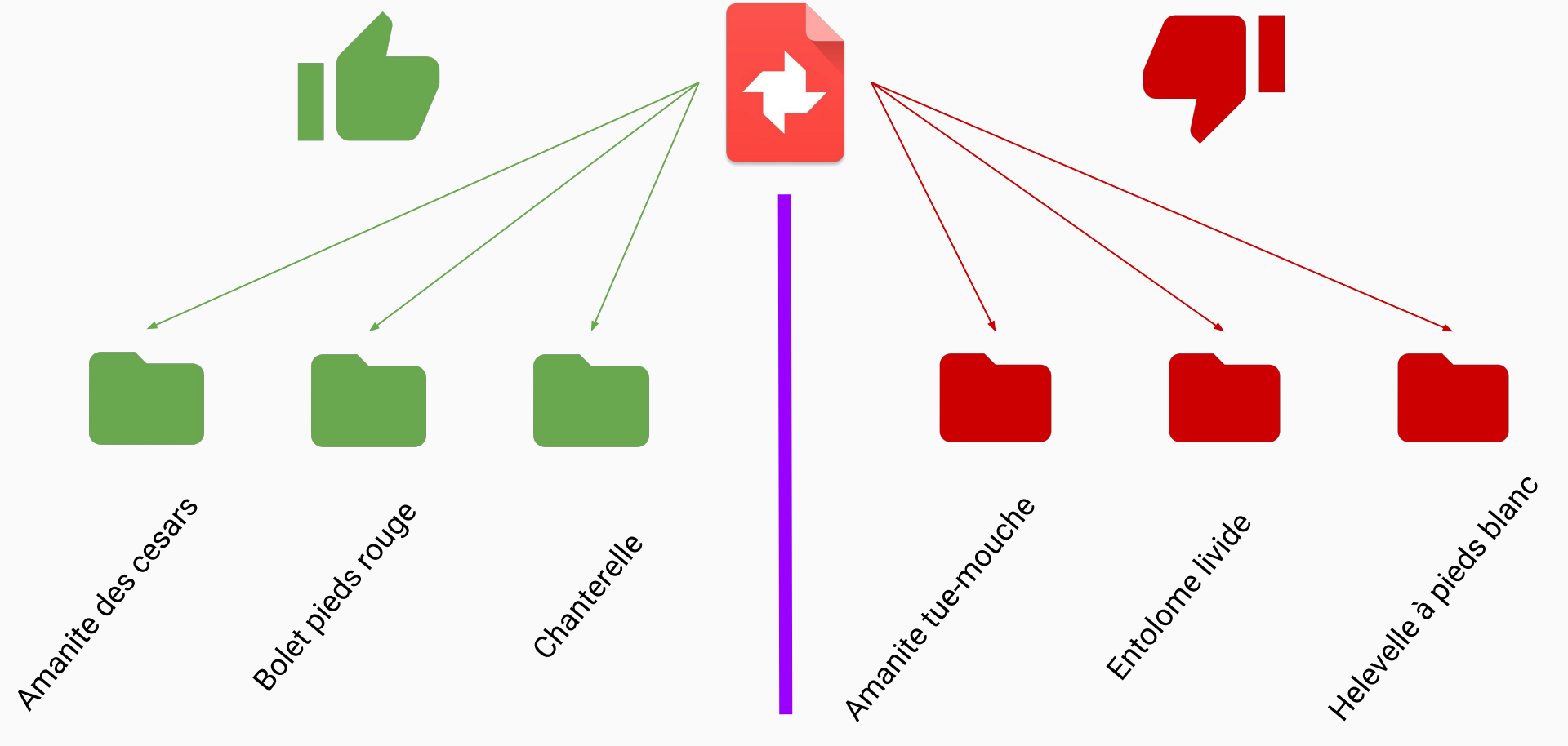
Cueillette de champignons - Récupération des images

Fatkun Batch

Cueillette de champignons - Récupération des images



Cueillette de champignons - Récupération des images



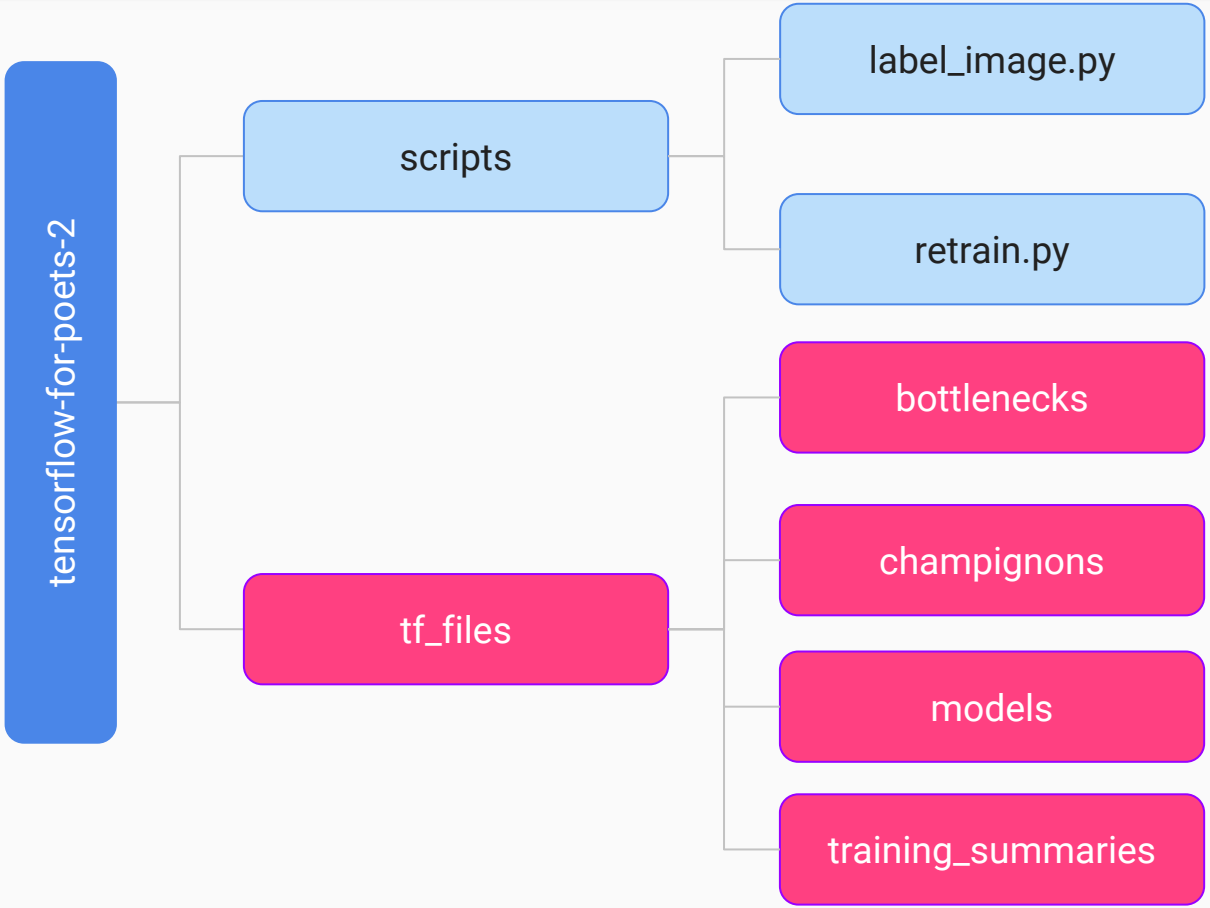
Cueillette de champignons - Ré-entraîner le modèle

```
$ git clone
https://github.com/googlecodelabs/tensorflow-for-poets-2
```

```
$ cd tensorflow-for-poets-2
```

```
$ mkdir tf_files tf_files/champignons
```

Cueillette de champignons - Ré-entraîner le modèle



Cueillette de champignons - Ré-entraîner le modèle

```
$ python -m scripts.retrain
         --bottleneck_dir=tf_files/bottlenecks
         --how_many_training_steps=500
         --model_dir=tf_files/models/
         --summaries_dir=tf_files/training_summaries/mobilenet_0.50_224
         --output_graph=tf_files/retrained_graph.pb
         --output_labels=tf_files/retrained_labels.txt
         --architecture=mobilenet_0.50_224
         --image_dir=tf_files/champignons
```

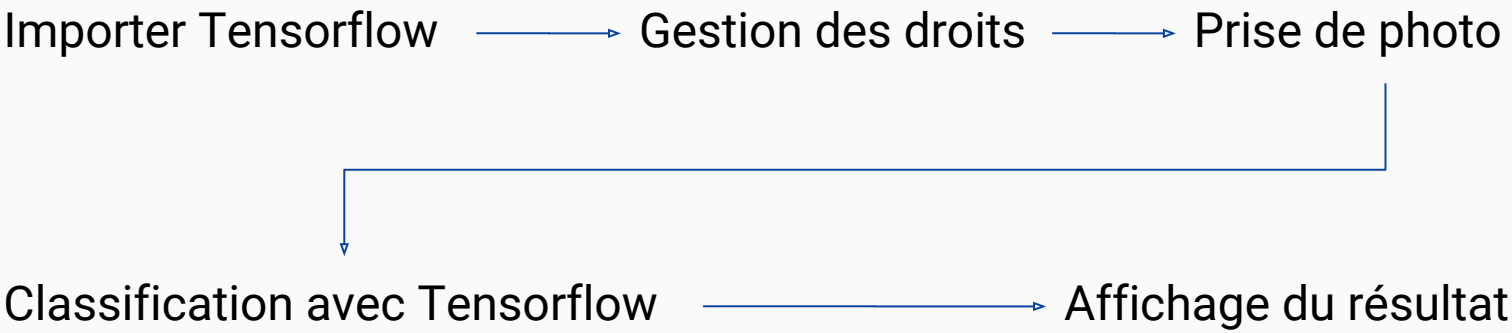
Cueillette de champignons - Test

```
$ python -m scripts.label_image
          --graph=tf_files/retrained_graph.pb
          --image=tf_files/mon_image.jpg
```

```
Evaluation time (1-image): 0.217s

veneneux amanite tue mouche 0.9999999
commestible amanite des cesar 6.541308e-08
veneneux entolome livide 2.396536e-13
veneneux helevelle a pieds blanc 1.8298406e-15
commestible chanterelle 2.8183814e-16
```

Cueillette de champignons - Création de l'application



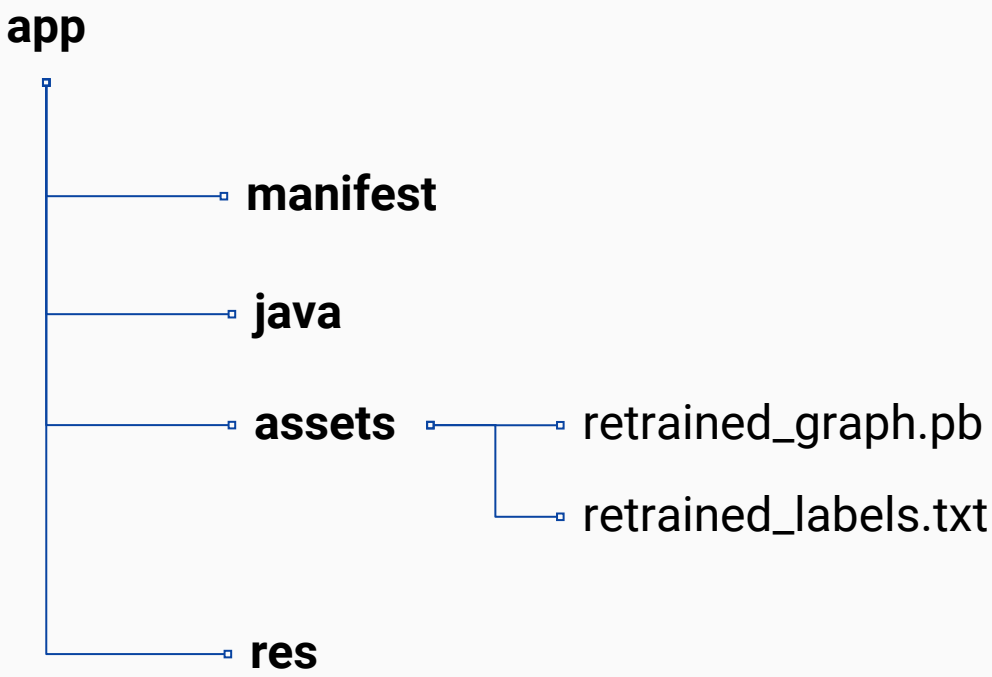
Cueillette de champignons - Création de l'application

build.gradle :

```
dependencies {
    ...

    //Tensorflow
    implementation "org.tensorflow:tensorflow-android:1.7.0"
}
```

Cueillette de champignons - Création de l'application

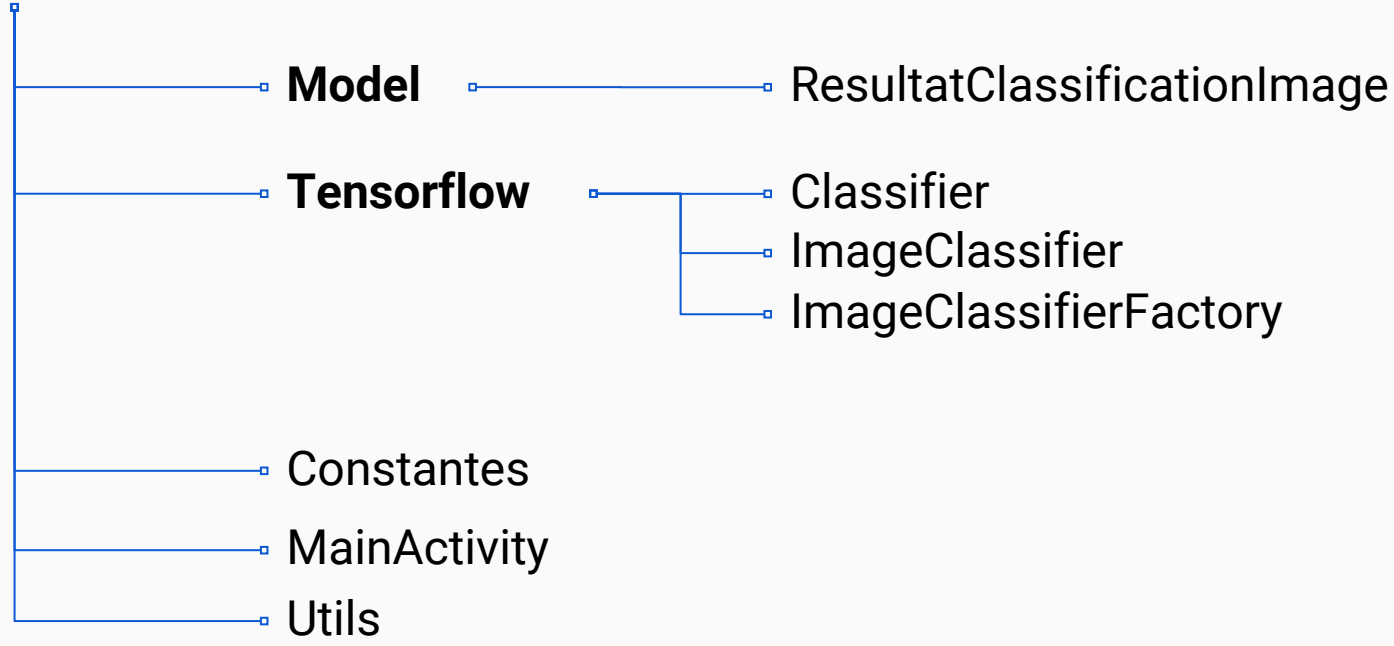


Manifest

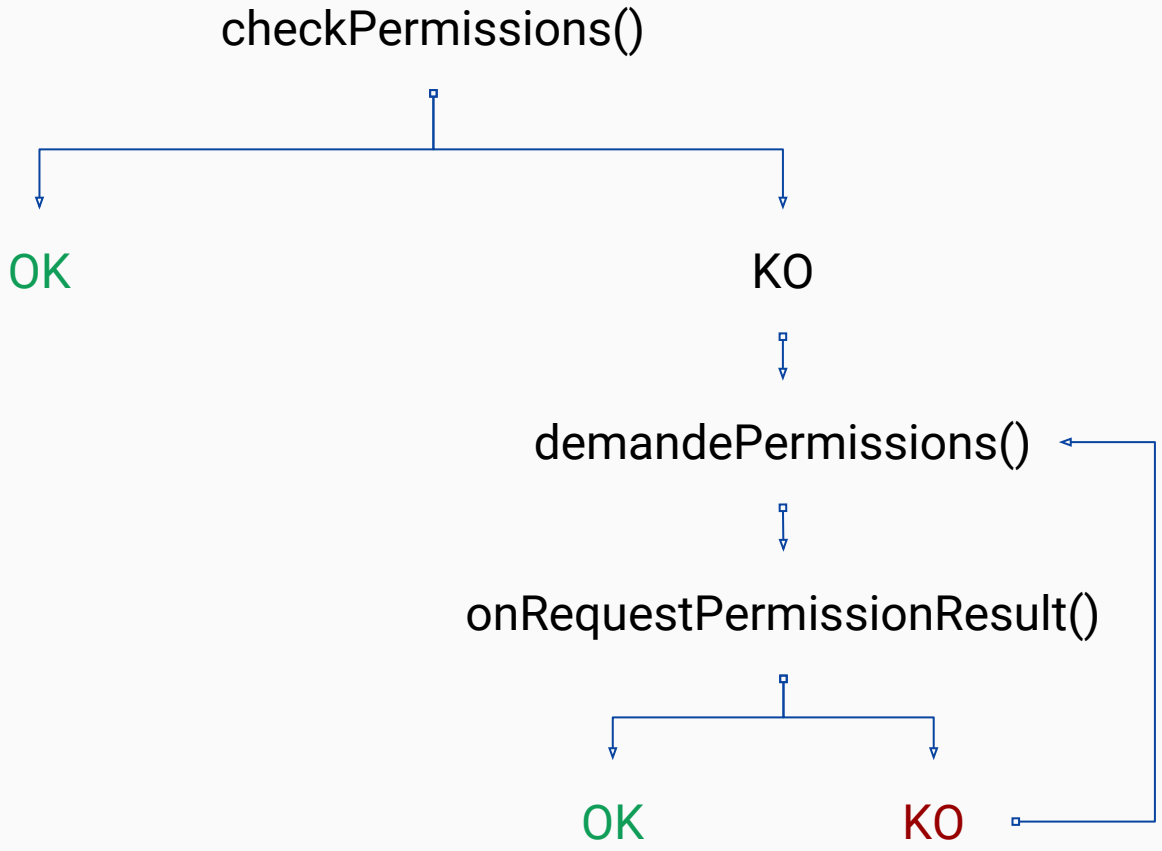
```
<uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
```

Cueillette de champignons - Création de l'application

java/Main



MainActivity



Cueillette de champignons - Création de l'application

```
private fun checkToutesPermissionsNecessairesDejaAccordees() : Boolean {
   return ContextCompat.checkSelfPermission(this,
Manifest.permission.WRITE_EXTERNAL_STORAGE) == PackageManager.PERMISSION_GRANTED
}
```

```
private fun demandePermissions() {
   ActivityCompat.requestPermissions(this,
           arrayOf(Manifest.permission.WRITE_EXTERNAL_STORAGE),
           DEMANDE_PERMISSIONS)
}
```

```
override fun onRequestPermissionsResult(requestCode: Int, permissions: Array<out
String>, grantResults: IntArray) {
    if (requestCode == DEMANDE_PERMISSIONS &&
    checkToutesPermissionsAccordees(grantResults)) {
       init()
   } else {
       demandePermissions()
   }
}
```

Cueillette de champignons - Création de l'application

```
private fun prisePhoto() {
    photoPath = Environment.getExternalStoragePublicDirectory(
     Environment.DIRECTORY_PICTURES).absolutePath
     + "/${System.currentTimeMillis()}.jpg"

    val photoUri = extractionUriDepuisPathFichier(this, photoPath)

    val intent = Intent(MediaStore.ACTION_IMAGE_CAPTURE)
    intent.putExtra(MediaStore.EXTRA_OUTPUT, photoUri)
    intent.flags = Intent.FLAG_GRANT_READ_URI_PERMISSION

    if (intent.resolveActivity(packageManager) != null) {
        startActivityForResult(intent, DEMANDE_PRISE_PHOTO)
    }
}
```

ResultatClassificationImage
categorie : String
indiceConfiance : Float

```
data class ResultatClassificationImage(
    val categorie : String,
    val indiceConfiance : Float)
```

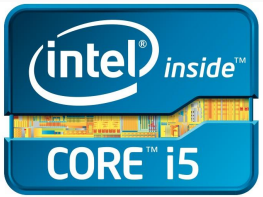
Classifier

```
interface Classifier {
            fun reconnaissanceImage(bitmap: Bitmap):
        ResultatClassificationImage
}
```

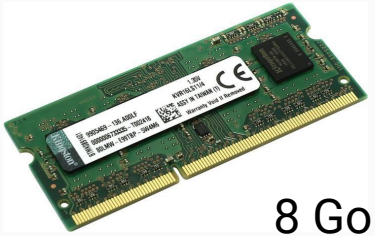
ImageClassifier

```
override fun reconnaissanceImage(bitmap: Bitmap):
ResultatClassificationImage {
    preprocessImageToNormalizedFloats(bitmap)
    classifyImageToOutputs()
    val outputQueue = getResults()
    return outputQueue.poll()
}
```

Cueillette de champignons



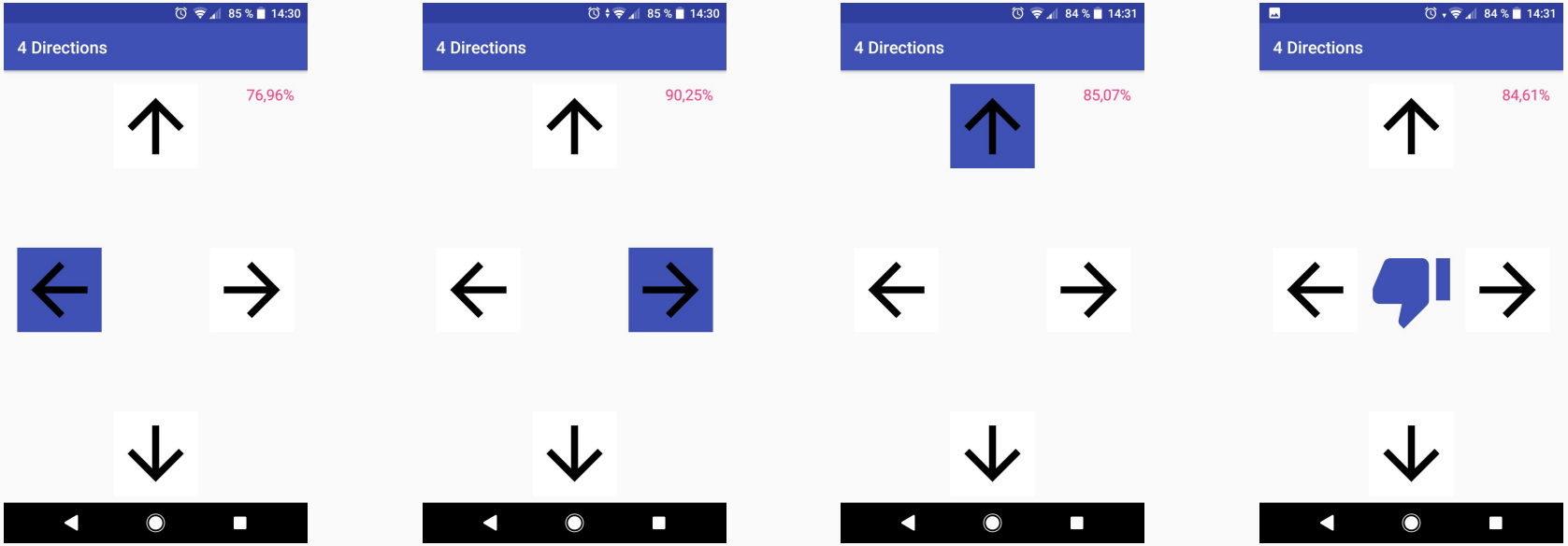
2.6 GHz



8 Go

	Inception-v3	MobileNet
Temps 500 IT (mn)	~15	~15
Taille .apk (Mo)	105	26

4 Directions


85 % 14:30
4 Directions
76,96%
85 % 14:30
4 Directions
90,25%
84 % 14:31
4 Directions
85,07%
84 % 14:31
4 Directions
84,61%

4 Directions - Procédure

1 - Trouver un modèle
2 - Récupération de données
3 - Ré-entraîner le modèle
4 - Test
5 - Création de l’application

4 Directions - Trouver un modèle

Simple Audio Recognition

↓

~100 enregistrements / mot-clef

4 Directions - Récupération de données

```
$ git clone https://github.com/tensorflow/tensorflow

$ cd tensorflow

$ python tensorflow/examples/speech_commands/train.py
                                        --data_dir=tf_files/dataset
                                        --summaries_dir=tf_files/summaries
                                        --train_dir=tf_files/train
```

4 Directions - Ré-entraîner le modèle

train

checkpoint
conv.ckpt-400.data-00000-of-00001
conv.ckpt-400.index
conv.ckpt-400.meta
conv.ckpt-500.data-00000-of-00001
…
conv.pbtxt
conv_labels.txt

4 Directions - Ré-entraîner le modèle

```
$ python tensorflow/examples/speech_commands/freeze.py
                    --start_checkpoint=tf_files/train/conv.ckpt-18000
                    --output_file=tf_files/my_frozen_graph.pb
```

4 Directions - Test

$ **python tensorflow/examples/speech_commands/label_wav.py**

--graph=tf_files/my_frozen_graph.pb

--labels=tf_files/train/conv_labels.txt

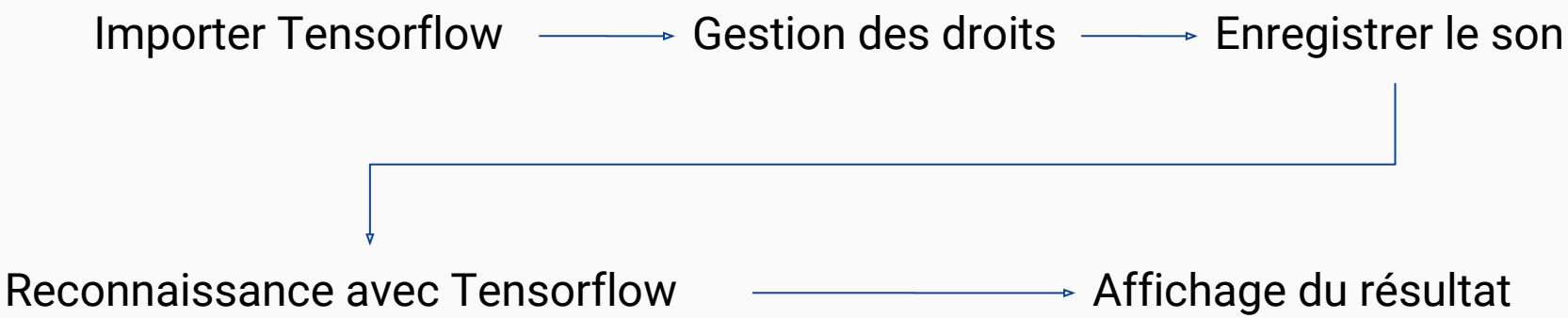
--wav=enregistrement_perso.wav

left (score = 0.81477)

right (score = 0.14139)

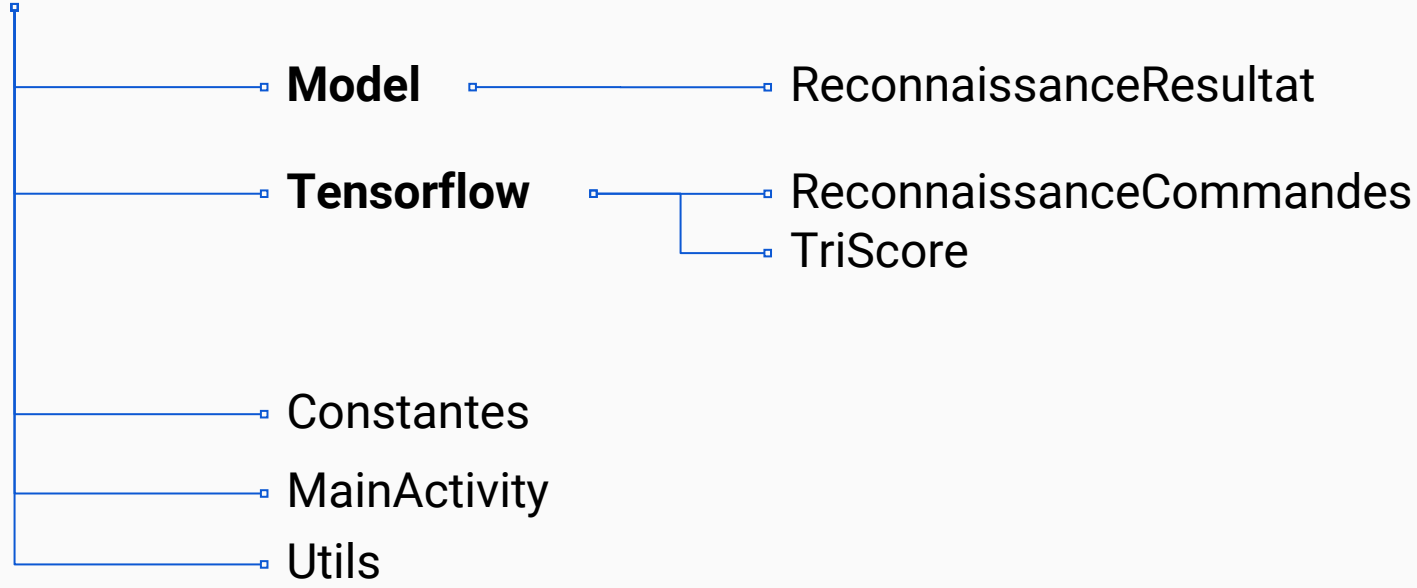
unknown (score = 0.03808)

4Directions - Création de l’application



4Directions - Création de l'application

java/Main



Manifest

```
<uses-permission
android:name="android.permission.RECORD_AUDIO"/>
```

4Directions - Création de l'application

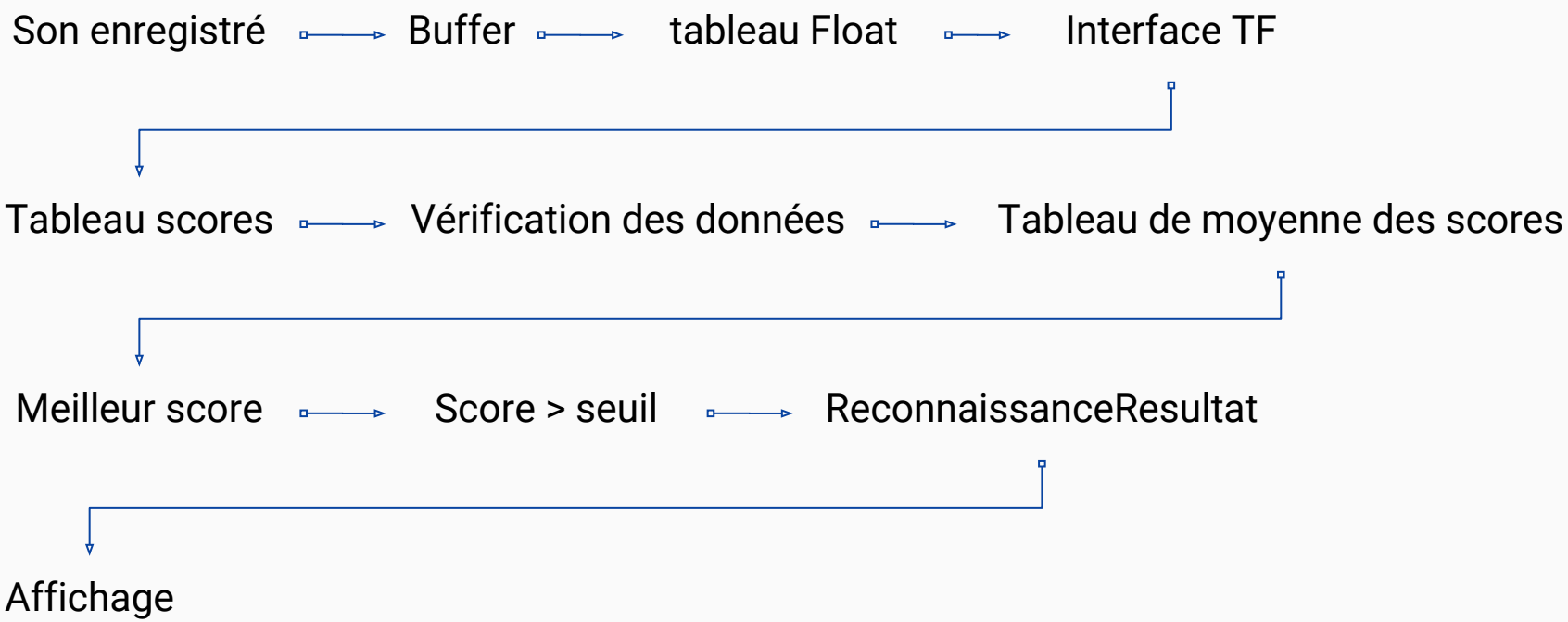
```
private fun checkToutesPermissionsNecessairesDejaAccordees() : Boolean {
    return ContextCompat.checkSelfPermission(this,
Manifest.permission.RECORD_AUDIO) == PackageManager.PERMISSION_GRANTED
}
```

```
private fun demandePermissions() {
    ActivityCompat.requestPermissions(this,
            arrayOf(Manifest.permission.RECORD_AUDIO),
            DEMANDE_PERMISSIONS)
}
```

ReconnaissanceResultat
commandeDetectee : String
score : Float
nouvelleCommande : Boolean

```
data class ReconnaissanceResultat(
    val commandeDetectee: String,
    val score: Float,
    val nouvelleCommande: Boolean)
```

4Directions - Création de l’application



MainActivity - Interface TF

```
interfaceTensorflow!!.feed(NOM_ECHANTILLON, listSampleRate)
interfaceTensorflow!!.feed(NOM_ENTREE_DONNEES,
bufferEntreeFloat, DUREE_ENREGISTREMENT.toLong(), 1)
interfaceTensorflow!!.run(scoresIntitules)
interfaceTensorflow!!.fetch(NOM_SCORE, scores)

tempsActuelMS = System.currentTimeMillis()
resultat =
reconnaissanceCommande!!.executionDerniersResultats(scores,
tempsActuelMS)
```

ReconnaissanceCommandes - Moyenne des scores

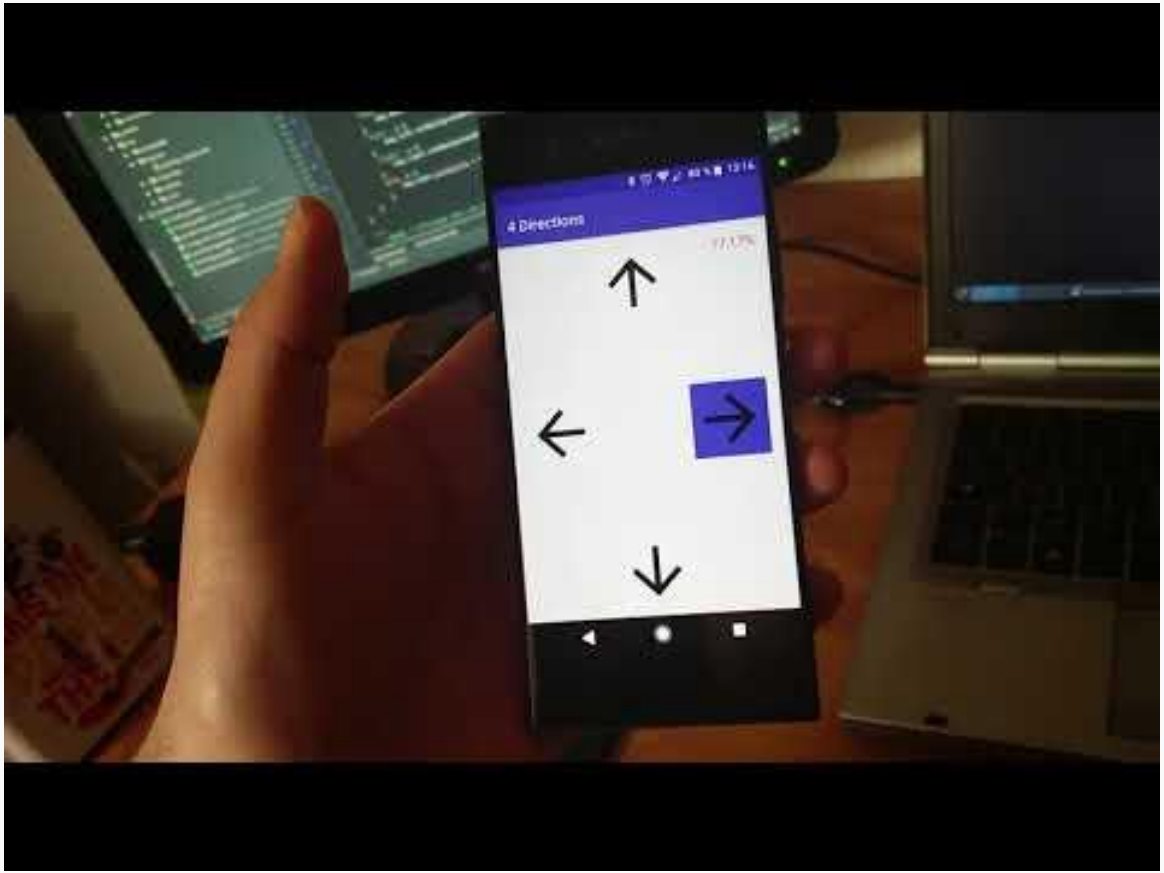
```
resultatsPrecedents.addLast(Pair<Long, FloatArray>
(tempsActuelMS, resultatsActuels))
[...]
for (resultatPrecedent in resultatsPrecedents) {
    val score : FloatArray = resultatPrecedent.second

    for(i in 0 until score.size) {
        scoresMoyens[i] += score[i] / nombreResultats
    }
}
```

MainActivity - affichage des résultats

```
runOnUiThread(
{
    if(!resultat.commandeDetectee.startsWith("_") &&
resultat.nouvelleCommande) {
        changeBgDirection(resultat.commandeDetectee)
        tv_indiceConfiance.text =
    "%.2f".format(resultat.score * 100) + "%"
    }
})
```

4Directions



4 Directions

	Simple Audio Recognition
Temps 18K IT (H)	45
Temps freeze (mn)	1
Taille .apk (Mo)	24.5

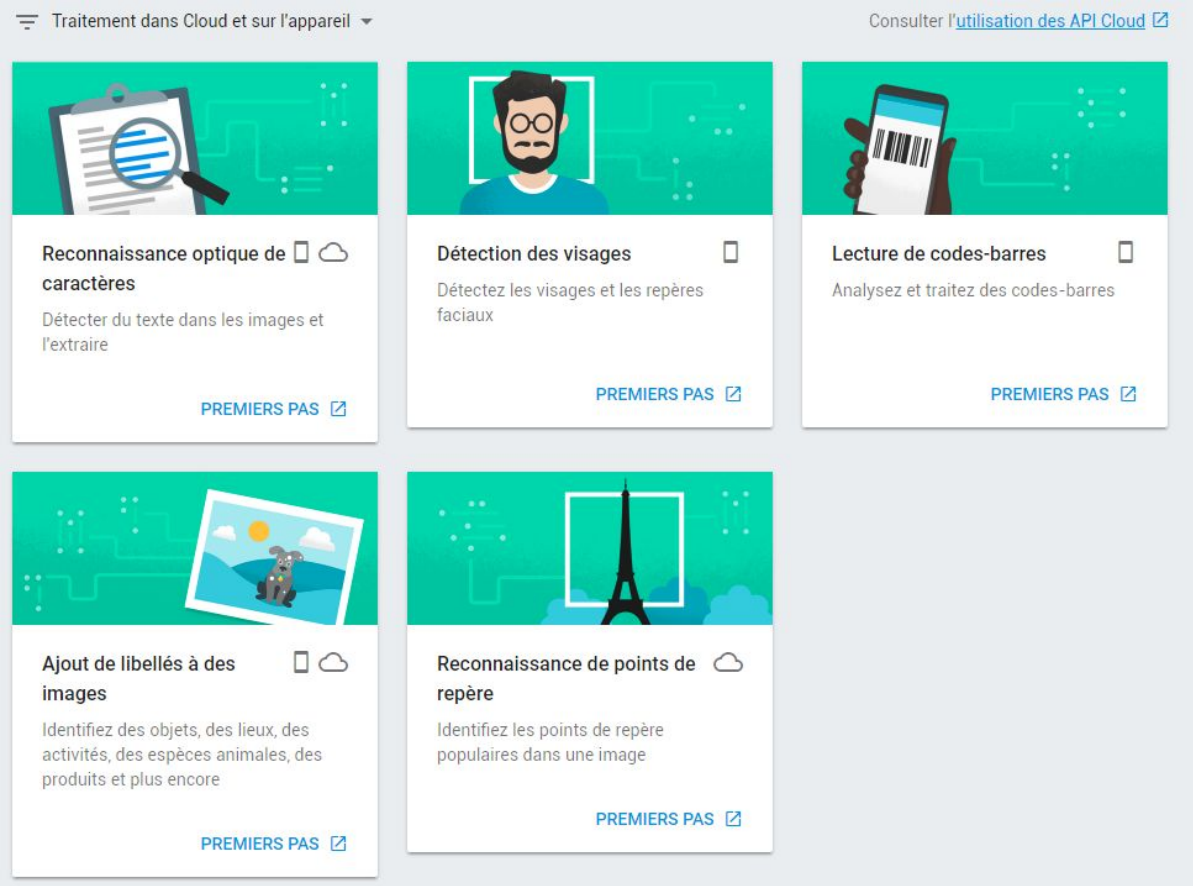
Et ensuite ?

Et ensuite ?

Et ensuite ?

Et ensuite ?

Et ensuite ?


Traitement dans Cloud et sur l'appareil
Consulter l'utilisation des API Cloud
Reconnaissance optique de caractères
Détecter du texte dans les images et l'extraire
PREMIERS PAS
Détection des visages
Détectez les visages et les repères faciaux
PREMIERS PAS
Lecture de codes-barres
Analysez et traitez des codes-barres
PREMIERS PAS
Ajout de libellés à des images
Identifiez des objets, des lieux, des activités, des espèces animales, des produits et plus encore
PREMIERS PAS
Reconnaissance de points de repère
Identifiez les points de repère populaires dans une image
PREMIERS PAS

Et ensuite ?

SUMMER 2018