Problema1: Necesitamos la librería de simpleITK para hacer correr la segmentación

Sub-problema1: Android utiliza Bionic para compilar C/C++, en cambio linux utiliza libc, haciendo imposible la llamada de SimpleITK desde android nativo(chaquopy).

Sub-problema2: Aunque se cross-compile simpleITK, es posible que halla problemas de en tiempo de enlace y runtime.

ideas:

Usar Kivy (compilador de android en python)

Difícil implementación por el motor de C necesario sería necesario recompilar todo ITK(Cmake/ DNS), se le llama receta.

Correr servicio Linux nativo con script python

Requiere hasta 10 GB (rootfs) de usuario, además de que levantar un servicio gastaría mucha batería al dispositivo. Mucha latencia entre chroot de linux y android

API REST(servidor) gratis

Difícil obtención de servidor gratis, demora en peticiones, conexión a internet, se necesita vinculación de tarjeta

Caquopy con Android studio

Se necesita compilar ITK, tarea titánica y difícil.

Re-compilar ITK, y utilizar solo las funciones de no-rigid segmentation

Se necesita mucho tiempo.

Usar los binarios de Java de SimpleITK

Saber configurar el proyecto. Uso de Pyjnius con chaquopy.
Puede generar incompatibilidades al ser binarios de escritorio.

Usar Docker

Problema2: Buscar alternativas para iOS, vamos a utilizar la librería opencv.

Oficial Apple: Xcode & macOS

Oficialmente Apple no ofrece soporte a desarrollo de apps, si no es en Mac. Hay varias alternativas, no oficiales.

Usar máquina virtual

Viable, poca estabilidad al compilar proyectos pesados.

IDEs no oficiales

Imposible compilación (solo se puede desarrollar, pero al hacer la build, se debe hacer en un mac).

Usar briefcase Python-Apple-support + PythonKit/C API

Usar una vm, instalar XCode, utilizar los framework Python-Apple-support

Difícil manejo de versiones e incompatibilidades

Usar OpenCV y TFLite iOS nativo en Swift

Modificar el código del proyecto inicial en java/kotlin

Pasar a react-native el código de python(Lo más viable)

Requisitos:
opencv: cvtColor, threshold, bitwise_not, findContours, drawContours, contourArea, fillConvexPoly, getAffineTransform, wrapAffline, flip, resize, morphologyEx, imread, imwrite

tesseract: image_to_string

amig_3ch: astype

numpy: expand_dims, arg_max, squeeze, ones

tflite: set_tensor, get_tensor

Encontrados para react-native:

- react-native-fast-opencv
- @react-native-ml-kit/text-recognition

Pasos usados

crear un proyecto expo:

```
npx create-expo-app@latest my_app
npx expo install expo-camera expo-image-picker
npx prebuild

npm install react-native-fast-opencv@latest @react-native-ml-kit/text-recognition@latest
./run-android-with-emulator.sh
```

Links relacionados al proyecto:

1. github-App-Mamitas
2. kaggle-modelo y funciones