SIRT – SRS

Software Requirements Specification

Descripción breve

SIRT es una aplicación para la asignatura de *Sistemas Inteligentes* implementada en el ecosistema Android que permite la detección en tiempo real de formas y objetos.

Javier Alonso Silva, Roberto Álvarez Garrido
https://github.com/SIRTDetection

1. Introducción

Esta guía (*SRS: **S**oftware **R**equirements **S**pecification*) establece los requisitos a nivel *software* que ha de satisfacer la aplicación SIRT (*SIRT: **S**istemas **I**nteligentes **R**eal **T**ime Software*).

1.1. Propósito

En la SRS, vamos a disponer los requisitos fundamentales que debe cumplir esta aplicación, para poder satisfacer todas las necesidades que requiera.

Esta SRS está dirigida a todo aquél que esté interesado en investigar cómo funciona y está planteado este proyecto, así como a los profesores de la asignatura de Sistemas Inteligentes que tengan que evaluar dicha práctica.

1.2. Alcance

El producto *software* a desarrollar (SIRT y dependencias necesarias) pretende detectar y analizar objetos y formas en tiempo real, bien mediante emisión de la captura de la cámara o usando una imagen estática dispuesta por la misma, aprovechando el ecosistema Android y las ventajas que éste ofrece para poder trabajar con distintos lenguajes de programación y las librerías que lo conforman.

Esta aplicación aprenderá por sí misma qué objetos son los que identifica, pudiendo un usuario entrenar su propio modelo para detectar nuevas imágenes que el modelo, por sí mismo, no es capaz de detectar en una primera instancia. Para ello, dispondremos de un servidor privado en donde se almacenarán y entrenarán modelos genéticos y basados en redes de neuronas, si fuera posible, por cada usuario, evitando así un uso excesivo del espacio y de los recursos del propio terminal donde se ejecute la susodicha aplicación, debido a las evidentes limitaciones *software* y *hardware*.

Si el desarrollo del *software* avanza como se espera en un principio, se creará un ecosistema de aplicaciones y recursos bastante potente que podrá facilitar la detección de objetos en otras aplicaciones y/o sistemas inteligentes que lo necesiten.

1.3. Definiciones, siglas y abreviaturas

1.3.1. Definiciones

- **Android**: “Android es un sistema operativo basado en el núcleo Linux. Fue diseñado principalmente para dispositivos móviles con pantalla táctil, como teléfonos inteligentes, tabletas y también para relojes inteligentes, televisores y automóviles” [1].
- **Sistema Inteligente**: “Un sistema inteligente es un programa de computación que reúne características y comportamientos asimilables al de la inteligencia humana o animal. La expresión "sistema inteligente" se usa a veces para sistemas inteligentes incompletos, por ejemplo para una casa inteligente o un sistema experto” [2].
- ***Software***: “Se conoce como software al soporte lógico de un sistema informático, que comprende el conjunto de los componentes lógicos necesarios que hacen posible la realización de tareas específicas, en contraposición a los componentes físicos que son llamados hardware” [3].
- ***Hardware***: “La palabra hardware en informática se refiere a las partes físicas tangibles de un sistema informático; sus componentes eléctricos, electrónicos, electromecánicos y mecánicos” [4].
- **Algoritmo genético**: “Estos algoritmos hacen evolucionar una población de individuos sometiéndola a acciones aleatorias semejantes a las que actúan en la evolución biológica (mutaciones y recombinaciones genéticas), así como también a una selección de acuerdo con algún criterio, en función del cual se decide cuáles son los individuos más adaptados, que sobreviven, y cuáles los menos aptos, que son descartados” [5].
- **Red de neuronas**: “Las redes neuronales (también conocidas como sistemas conexionistas) son un modelo computacional basado en un gran conjunto de unidades neuronales simples (neuronas artificiales) de forma aproximadamente análoga al comportamiento observado en los axones de las neuronas en los cerebros biológicos1. La información de entrada atraviesa la red neuronal (donde se somete a diversas operaciones) produciendo unos valores de salida” [6].
- **Java**: “Java es un lenguaje de programación de propósito general, concurrente, orientado a objetos, que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible” [7].

- **Python**: “Python es un lenguaje de programación interpretado cuya filosofía hace hincapié en una sintaxis que favorezca un código legible. Se trata de un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional” [8].

1.3.2. Siglas

- **SIRT**: ***S**istemas **I**nteligentes **R**eal **T**ime Software*.
- **SRS**: ***S**oftware **R**equirements **S**pecification*.
- **API**: ***A**pplication **P**rogramming **I**nterface*.

1.3.3. Referencias

[1] Comunidad, «Android,» Wikipedia ORG, Noviembre 2018. [En línea]. Available: https://es.wikipedia.org/wiki/Android. [Último acceso: 16 Noviembre 2018].

[2] Comunidad, «Sistema inteligente,» Wikipedia ORG, Julio 2017. [En línea]. Available: https://es.wikipedia.org/wiki/Sistema_inteligente. [Último acceso: 16 Noviembre 2018].

[3] Comunidad, «Software,» Wikipedia ORG, Octubre 2018. [En línea]. Available: https://es.wikipedia.org/wiki/Software. [Último acceso: 16 Noviembre 2018].

[4] Comunidad, «Hardware,» Wikipedia ORG, Noviembre 2018. [En línea]. Available: https://es.wikipedia.org/wiki/Hardware. [Último acceso: 16 Noviembre 2018].

[5] Comunidad, «Algoritmo genético,» Wikipedia ORG, Octubre 2018. [En línea]. Available: https://es.wikipedia.org/wiki/Algoritmo_gen%C3%A9tico. [Último acceso: 16 Noviembre 2018].

[6] Comunidad, «Red neuronal artificial,» Wikipedia ORG, Octubre 2018. [En línea]. Available: https://es.wikipedia.org/wiki/Red_neuronal_artificial. [Último acceso: 16 Noviembre 2018].

[7] Comunidad, «Java,» Wikipedia ORG, Noviembre 2018. [En línea]. Available: https://es.wikipedia.org/wiki/Java_(lenguaje_de_programaci%C3%B3n). [Último acceso: 16 Noviembre 2018].

[8] Comunidad, «Python,» Wikipedia ORG, Octubre 2018. [En línea]. Available: https://es.wikipedia.org/wiki/Python. [Último acceso: 16 Noviembre 2018].

1.4. Visión global

El resto de la SRS explica detalladamente los requisitos funcionales de la aplicación, así como lo que necesitamos a nivel de servidor, estructurados en describir y especificar el producto y luego, en más detalle, cómo debe funcionar a nivel interno.

2. Descripción general

2.1. Perspectiva del producto

SIRT es un producto *software*, creada en el ecosistema Android, cuya utilidad será la detección en tiempo real de imágenes y formas, bien mediante vídeo o mediante imágenes tomadas en el momento.

Se conectará a un servidor privado donde se hará la gestión del procesado de la imagen, así como la generación del correspondiente árbol de decisiones utilizando algoritmos genéticos para ello.

La aplicación se hará completamente en Java, y se usará la JNI (*Java Native Interface*) cuando sea necesario, para utilizar toda la potencia de procesado del terminal Android.

2.2. Funciones del producto

Las funciones de SIRT se pueden clasificar en funciones que se detallan a continuación:

I. Detección de contornos

La aplicación Android tendrá que ser capaz de detectar todos los contornos posibles y viables en la imagen capturada. Si este reconocimiento se hace en vídeo, la aplicación tendrá que estar altamente optimizada para no provocar una ralentización en el dispositivo en general, por generar contornos por cada fotograma.

II. Detección de formas

Con los contornos obtenidos por la aplicación, un servidor privado se encargará de la gestión de dicho contorno, identificando a qué imagen se corresponde (o se aproxima más).

Cada instalación de la aplicación tendrá su propio árbol de decisiones en el servidor, siendo cada modelo genético único para cada usuario. De esta manera, la interacción de un dispositivo con el servidor no afectará a los otros usuarios, ni para bien ni para mal.

III. Entrenamiento del modelo

Debido a que cada usuario tiene su propio modelo y árbol de decisión (aunque se parta de un mismo origen), cada dispositivo tiene la opción de entrenar el modelo. Por ejemplo, si una imagen se detecta incorrectamente, el usuario puede informar de dicho error y aportar el nombre correcto de la imagen, provocando así que el modelo se reentrene.

Del mismo modo, se puede asegurar al servidor que la imagen detectada es correcta, provocando así un refuerzo de esa parte del árbol de decisiones.

2.3. Características del usuario

Cualquier usuario podrá instalar y usar esta aplicación, pero debido a las restricciones obvias del servidor, en cuanto a que no se dispone ni de espacio ni de capacidad de cómputo ilimitados, se limitará la cantidad máxima de usuarios a 10.

2.4. Restricciones

Es imprescindible que el dispositivo disponga tanto de cámara como de conexión a Internet, así como una versión de Android al menos de KitKat (Android 4.4).

El dispositivo, a su vez, tendrá que tener al menos 1 GB. de memoria RAM, así como un procesador de, como poco, dos núcleos con un reloj de 1 GHz.

2.5. Supuestos y dependencias

El *software* dependerá directamente del sistema operativo Android, ya que ha de ser ejecutado en dicho ecosistema. A su vez, la aplicación en el servidor todavía está por determinar su lenguaje de programación, pero será necesario que sea compatible con sistemas Linux (en particular sistemas *deb*, es decir, basados en Debian).

Posiblemente usaremos *Python* para desarrollar la parte del servidor, o bien aprovechar la infraestructura que ofrece Java.

3. Requisitos específicos

3.1. Requisitos de la interfaz externa

Interfaz con el usuario

La interfaz del usuario tendrá que ser lo más amigable posible, siguiendo los últimos patrones de diseño de *Material Design*.

De igual manera, esta aplicación lo que pretende es identificar y detectar imágenes, por lo que la parte de interfaz no es la más relevante.

Interfaz con el *hardware*

La aplicación únicamente necesitará de cámara, así como de los permisos necesarios para utilizarla.

Interfaz con el *software*

La aplicación, en principio, no usará ninguna API externa. En cualquier caso, el servicio de *Firebase* para detección de errores y analíticas.

Interfaz de comunicaciones

La aplicación requerirá de una conexión constante a Internet para poder analizar de forma eficiente las imágenes, no existiendo un modo "sin conexión".

3.2. Requisitos Funcionales

3.2.1.1. Detección de contornos

- Descripción: La aplicación debe tomar una imagen usando la cámara del dispositivo Android y sacar sus contornos.
- Entradas: Imagen capturadas por la cámara.
- Procesamiento: Usar *OpenCV* para calcular los contornos y/u obtener la imagen en escala de grises.
- Salidas: Contorno de dicha imagen.

3.2.1.2. Evaluación del contorno

- Descripción: El servidor deberá evaluar el contorno de la imagen que se ha obtenido en Android con las que tiene en su árbol de decisiones
- Entradas: Contorno que le ha enviado la aplicación Android.
- Procesamiento: Usar un árbol de decisiones para identificar el contorno.
- Salidas: Un texto con el resultado del reconocimiento.

3.2.1.3. Entrenamiento

- Descripción: El usuario, si ve un error, puede entrenar el modelo de la aplicación.
- Entradas: Un texto con la corrección.
- Procesamiento: El árbol del modelo se entrena con distintos algoritmos (entre ellos genéticos) para adaptar el árbol.
- Salidas: La modificación del arbol

3.2.1.4. Obtención de la información

- Descripción: tras procesar la imagen y entrenar el árbol, se obtiene una descripción de la imagen.
- Entradas: una imagen en escala de grises, por ejemplo.
- Procesamiento: evaluando el árbol de decisiones, se decide qué imagen es así como su descripción.
- Salidas: un *string* con la información correspondiente a la imagen.

3.3. Requisitos de rendimiento

La primera vez que se genere el árbol de decisiones, es normal que tarde bastante (unos 30 minutos aproximadamente). Para evitar esa carga el resto de las veces, el primer entrenamiento del árbol se copiará y se guardará el resultado, para que se parta de una misma base con cada dispositivo y así no sea necesario esperar.

Del mismo modo, y debido a las limitaciones hardware de los dispositivos Android, la aplicación deberá ser lo más ligera y estable posible, y que el envío, procesado y obtención de la información de la imagen use los menos recursos posibles.

3.4. Requisitos de diseño

Si bien esto no es relevante, es interesante que la aplicación siga las últimas descripciones de *Material Design*, para que resulte atractiva e intuitiva para el usuario que la use.

3.5. Atributos del sistema *software*

SIRT será una aplicación que gozará de:

- Seguridad: aunque, en principio, los datos no son sensibles, serán tratados con la máxima discreción y seguridad, para evitar posibles problemas o robos de datos al usar las conexiones de red.
- Disponibilidad: la aplicación deberá poder ser utilizable, o al menos intentarlo, las 24 horas del día.
- Escalabilidad y fiabilidad: se ha de intentar que, por muchas imágenes que se añadan, el sistema deberá funcionar lo más rápido posible y que no falle.

Apéndices

Índice