2. OBJETIVOS

En este capítulo se enumeran las diferentes metas a conseguir durante el desarrollo de este TFG, además de los medios software y hardware que han sido utilizados para su desarrollo.

2.1 OBJETIVO PRINCIPAL

El objetivo principal de este TFG es la creación de una aplicación que permita generar fuentes y tipos de letra originales y usarlas para transcribir texto.

Ademas el usuario sera capaz de cambiar los parámetros de la Variable Oculta de los VAE para poder generar fuentes propias. El usuario también podrá visualizar las fuentes creadas y podrá borrarlas en caso de que así quiera.

2.2 OBJETIVOS ESPECIFICOS

Para cumplir con el objetivo principal debemos cumplir una serie de objetivos secundarios para con este TFG. Estos objetivos específicos son:

2.2.1 ESTUDIO

El desarrollo de este proyecto conlleva conocimientos de Machine Learning, Deep Learning, Autoencoders y los Autoencoders Variacionales (VAE) como generadores de objetos. Por lo que se estudiaran documentos científicos y libros divulgativos en estas materias.

El estudio se divide en varios subobjetivos.

2.2.1.1 Redes neuronales y Deep Learning

Para comenzar el estudio primero debemos tener claro los conceptos de Red Neuronal y Aprendizaje profundo (Deep Learning).

2.2.1.2 Autoencoders

Profundizando en el estudio estudiaremos una clase de Red Neuronal dentro del ámbito del Deep Learning llamadas Autoencoders que explicaremos mas adelante.

2.2.1.3 Autoencoders Variacionales

Como ultima etapa del estudio tendremos que comprender este tipo de Autoencoder que nos permiten mediante su entrenamiento generar datos originales como veremos mas adelante.

2.2.2 DESARROLLO

Como segundo subobjetivo está el desarrollo del sistema generador de fuentes. Este objetivo se subdivide en varios subobjetivos agrupados en las dos partes del sistema (backend y frontend).

<u>BAKCEND</u>

2.2.2.1 Construir un VAE

Para poder generar datos primero necesitamos un VAE entrenado que sea capaz de generarlos, para entrenar dicho VAE primero necesitamos programarlo/construirlo. Para ello se elegirá un VAE ya construido el cual modificaremos/adaptaremos al problema.

2.2.2.2 Elegir o crear un conjunto de datos de entrenamiento valido y entrenar los VAE

Para que los VAE sean capaz de generar datos deben ser entrenados con un conjunto de datos (dataset) valido, por lo que se busca un conjunto valido. Un conjunto valido de datos en este caso es un conjunto de imágenes de letras escritas en distintas fuentes y que no contenga ruido (fuentes que son símbolos, fuentes ilegibles, etc.) para el correcto entrenamiento de los VAE.

Para ello se crea un conjunto de datos con las fuentes propias del sistema operativo Windows y eliminamos de ellas las fuentes simbólicas (basadas en símbolos) y se entrenan los VAE con dicho conjunto, de forma que sean capaces de generar letras legibles.

2.2.2.3 Desarrollar un sistema generador de fuentes y transcriptor de texto

Con los VAES entrenados debemos exportar sus pesos y usarlos en un sistema que conecte con una BBDD en MongoDB y cuyas funciones nos permitan:

- Transcribir texto con una fuente ya generada
- Transcribir texto con una fuente generada con los parámetros de entrada
- Generar una fuente
- Guardar una fuente
- Borrar una fuente
- Ver una del sistema

2.2.2.4 Desarrollo de un API REST

Una API REST es una interfaz con el servidor, unas funcionalidades que se pueden usar del servidor sin estado en este, es decir, sin que el servidor cambie de forma interna para dicha petición. Este termino tiene dos partes:

1. API: Es un conjunto de funciones que sirven de interfaz con un sistema
2. REST: El servidor que contiene la API y el sistema que usa dicha API es carente de estado, es decir, su estado siempre es el mismo y solo se encarga de responder a las peticiones sin modificarse internamente.

Con el fin de poder utilizar la aplicación desarrollada, se implementará una API REST. A partir de esta API REST, se podrá usar el sistema generador de fuentes y transcriptor de texto y sus funciones.

Esta API funcionara como backend, es decir, funcionara como servidor sin estado (REST), conectara con MongoDB e implementara sus propias funciones:

- Crear un usuario
- Borrar un usuario
- Cambiar la contraseña de un usuario
- Hacer login en el sistema (comprobar si ese usuario existe)
- Ver todas las fuentes del sistema

FRONTEND

2.2.2.5 Desarrollo de una aplicación Android

Para poder validar el backend del sistema se necesita un cliente que haga uso de este, en nuestro caso se decidió usar una aplicación móvil desarrollada en Android.

Este cliente en Android implementa las funcionalidades del API REST, permite usarlas y visualizarlas desde un móvil. Este cliente nos permitirá usar todas las funciones del servidor y funcionara como ejemplo/demo para el uso del generador automático de fuentes.

2.3 HERRAMEINTAS Y MEDIOS UTILIZADOS

2.3.1 MEDIOS HARDWARE

Ordenador portátil de uso profesional

- Memoria RAM: 8.00 GB
- Procesador: Intel Core i7-4510U CPU 2.00 GHz 3.10 GHz
- Disco Duro: 1TB
- Tarjeta Gráfica: Nvidia GeForce 840M 4GB DRR3

Ordenador sobremesa de uso personal

- Memoria RAM: 8.00 GB DRR4
- Procesador: AMD Ryzen 5 1600 3.2GHz
- Disco Duro: 2TB
- Tarjeta Gráfica: Gygabyte GTX 1060 Windforce oc 3GB GDDR5

2.3.2 MEDIOS SOFTWARE

Sistema Operativo Windows 10 Educational Pro

El sistema Operativo (SO) usado para desarrollar todo el sistema y la documentación del presente TFG ha sido Windows 10 en su edición educacional pro.

Flask

Flask es un Framework minimalista escrito en Python que permite crear aplicaciones web. Está basado en el kit de herramientas Werkzeug y el motor de plantillas Jinja2. Es el principal partícipe para la implementación de la API REST que se comentara mas adelante en el capitulo 3.

PIL/Pillow

La Python Imaging Library (PIL) otorga al interprete Python la capacidad de procesar imagenes.

Esta librería soporta una gran cantidad de extensiones de archivo, ademas permite una representación

interna eficiente y una gran capacidad de procesamiento de imagenes.

Ademas el núcleo de esta librería esta diseñado para acceder de manera rápida a los datos guardados con un formato básico de pixeles.

TensorFlow

TensorFlow ™ es una biblioteca de software de código abierto para el cálculo numérico de alto rendimiento. Su arquitectura flexible permite una fácil implementación de computación en una variedad de plataformas (CPU, GPU, TPU) y desde escritorios hasta clústeres de servidores y dispositivos móviles y periféricos.
Ademas viene con un fuerte soporte para Machine Learning y Deep Learning y el núcleo de computación numérica flexible se utiliza en muchos otros dominios científicos.

Keras

Keras es una API de redes neuronales de alto nivel, escrita en Python y capaz de ejecutarse sobre TensorFlow, CNTK o Theano. Fue desarrollado para permitir la experimentación rápida. Poder pasar de la idea al resultado con la menor demora posible es la clave para hacer una buena investigación.

Python

Python es un lenguaje de programación interpretado. Se trata de un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y programación funcional. En este TFG se ha usado para el desarrollo de la API REST con Flask, así como para la implementación, entrenamiento y desarrollo del generador de fuentes y del sistema transcriptor de texto.

Git

Git es un sistema de control de versiones de código abierto y gratuito para manejar desde proyectos pequeños a muy grandes, con velocidad y eficiencia.

GitHub

GitHub es una plataforma de desarrollo inspirada en la forma en que se trabaja. Desde el código abierto hasta el negocio, se puede alojar y revisar códigos, administrar proyectos y crear software.

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. Es un SO basado en java para el computo y las operaciones y XML para la interfaz y su diseño.

Android Studio

Android Studio es el entorno de desarrollo integrado oficial para la plataforma Android.

Está basado en el software IntelliJ IDEA de JetBrains y ha sido publicado de forma gratuita a través de la Licencia Apache 2.0. Está disponible para las plataformas Microsoft Windows, macOS y GNU/Linux. Ha sido diseñado específicamente para el desarrollo de Android.

Conda

Conda es un sistema de gestión de paquetes de código abierto y un sistema de gestión del entorno que se ejecuta en Windows, macOS y Linux. Conda crea, guarda, carga y cambia fácilmente entre entornos en su computadora local. Fue creado para programas de Python, pero puede empaquetar y distribuir software para cualquier idioma. En el presente TFG ha sido utilizado para programar y desarrollar Python en Windows.

Spyder

Spyder es el Entorno de desarrollo Científico de Python, cuanta con un entorno de desarrollo Python muy interactivo con propiedades avanzadas de edición, diseño, debuggin e introspección.

Spyder también trae un entorno de computación numérica gracias al soporte de Ipython (un interprete interactivo de Python) y librerias populares de Python para computación numérica, tales como NumPy (Algebra linear), SciPy (Señales y procesamiento de imagenes) y matplotlib (Trazado interactivo para 2D/3D).

OpenOffice

Apache OpenOffice es una suite de oficina de código abierto líder para el procesamiento de palabras, hojas de cálculo, presentaciones, gráficos, bases de datos y más. Se ha usado en este TFG para el desarrollo de la presente documentación.

Latex

LaTeX es un sistema de composición tipográfica de altas prestaciones con características diseñadas para la producción de documentación técnica y científica. Se ha usado en este TFG para el desarrollo de la presente documentación.