ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

MÁSTER UNIVERSITARIO EN INGENIERÍA DEL SOFTWARE E INTELIGENCIA ARTIFICIAL

ASISTENTE DE TOMA DE NOTAS PARA INTÉRPRETES

NOTE-TAKING ASSISTANT FOR INTERPRETERS

Realizado por
Francisco Javier Lima Florido
Tutorizado por
Enrique Alba Torres, Gloria Corpas Pastor
Departamento
Lenguajes y Ciencias para la Computación

UNIVERSIDAD DE MÁLAGA
MÁLAGA, Septiembre, 2019

Resumen La toma de notas es una de las actividades más importantes en la interpretación consecutiva y requiere de mucha práctica y concentración por parte del intérprete para llegar a dominarla. A pesar de los avances e innovaciones tecnológicas de los últimos años, las tecnologías de la interpretación no se han visto beneficiadas por dichos avances, haciéndose necesario un impulso en la investigación e innovación en este ámbito. En este trabajo se persigue el objetivo de sacar provecho de los avances actuales de la inteligencia artificial para desarrollar un sistema inteligente que apoye al intérprete humano en la toma de notas. Actualmente no existen herramientas basadas en la inteligencia artificial que traten de mejorar el trabajo de los intérpretes durante la toma de notas, lo cual empeora por la falta de conjuntos de datos públicos de símbolos o notas de intérpretes que poder utilizar en métodos de aprendizaje. Tras investigar las posibles técnicas y métodos que poder utilizar para cumplir el mencionado objetivo, se ha desarrollado un sistema basado en aprendizaje computacional el cual ha sido entrenado con un conjunto de datos de símbolos creado durante el propio desarrollo del trabajo. El sistema resultante consiste en una aplicación web que permite a los usuarios dibujar símbolos y averiguar que concepto representan, así como recopilar nuevos datos para ampliar el conjunto de entrenamiento. La aplicación ofrece buenos resultados en el reconocimiento de un conjunto de símbolos que se utilizan en la toma de notas, sirviendo de proyecto base para el desarrollo efectivo de herramientas que apoyen al intérprete tanto durante el transcurso de la interpretación y la preparación previa, como en el trabajo posterior a la misma.

Abstract Note-taking is one of the most important activities in consecutive interpreting and requires a lot of practice and concentration to master it. In spite of the innovations and the advances in technolgy in recent years, interpretation technologies have not been benefitted by such innovations, becoming necessary a boost in research and innovation in this area. The main goal of this project is to take advantage of the current advances in artificial intelligence to develope a smart system that supports the interpreter in note-taking process. Currently, there are no tools based on artificial intelligence that seek to improve the work of interpreters during note-taking process, which gets worse by the lack of public datasets of symbols or interpreter's notes that can be used in machine learning. After seeking for possible techniques and methods that can be used to achive the mentioned goal, a system based on deep learning has been developed and trained with a dataset of symbols created during the development of this project. The final system is a web app that allows users to draw symbols and find out what term they represent, as well as collect new data to expand the training dataset. The application offers good results in the recognition of a set of symbols that are used in note-taking process, and serves as a basic project for the future development of tools that support the interpreter both during the course of the interpretation and the previous preparation, as well as in the subsequent work.

Keywords: interpreting technologies · machine learning · translation and interpretating · consecutive interpreting · note-taking symbols · image classification · image comparing

Índice general

1. Introducción

En los últimos años los traductores se han visto beneficiados respecto al desarrollo de su trabajo por la multitud de herramientas que han surgido (y siguen creándose) a raíz de los avances en inteligencia artificial (Bowker and Corpas Pastor, 2015). Es diferente en el caso de la interpretación, donde existe una laguna tecnológica en la que no existen herramientas actuales diseñadas específicamente para el trabajo de la interpretación, que se han quedado obsoletas, o que no aportan la suficiente utilidad (Corpas Pastor, 2018). La interpretación es un trabajo estresante cuyas condiciones implican mayores exigencias para poder innovar que para las herramientas de la traducción (tiempo de respuesta, tolerancia a fallos, etc.), provocando que haya muchos aspectos del trabajo de los intérpretes para los que aún se utilizan recursos rudimentarios (Costa et al., 2014). Uno de esos aspectos es el de la toma de notas.

La toma de notas en interpretación consecutiva consiste en anotar el discurso del hablante para la posterior traducción al idioma objetivo. El problema de este proceso reside en la dificultad que supone anotar todo el contenido del discurso a la vez que trata de asimilarse, lo que hace que esta técnica requiera de mucha práctica (Ulloa and Navarrete, 2014). Para facilitar este trabajo, los intérpretes utilizan símbolos que sintetizan las ideas o conceptos del discurso. Con el presente trabajo se busca optimizar la toma de notas durante la interpretación.

Actualmente, las escasas innovaciones en la toma de notas se encuentran principalmente en herramientas para intérpretes en formación, como la aplicación Cleopatra, que permiten practicar los símbolos con el fin de ayudar a su memorización (Torre Salceda, 2017). Son herramientas con una funcionalidad muy simple en la que el usuario añade sus símbolos a partir de los cuales se generan ejercicios de memorización en los que se muestra una imagen y el usuario debe acertar a que concepto corresponde, o se muestra un concepto y el usuario debe dibujar el símbolo correspondiente. Incluso en el segundo caso, estas herramientas no son capaces de reconocer el dibujo del símbolo, si no que solo muestran posteriormente el símbolo previamente almacenado para que el usuario compruebe si lo ha dibujado correctamente.

Este trabajo tiene como principal objetivo mejorar lo existente aplicando técnicas de inteligencia artificial que permitan reconocer los símbolos dibujados a mano al estilo de las herramientas OCR (Singh, 2013), permitiendo ampliar las posibilidades de innovación en el desarrollo de herramientas para la toma de notas. Para cumplir dicho objetivo, se establecen otros secundarios: estudiar el estado actual en cuanto a técnicas y métodos de reconocimiento de símbolos o dibujos manuscritos; buscar el método que ofrezca los mejores resultados e investigar las posibilidades de aplicabilidad en casos reales.

El desarrollo de este trabajo mejora las funcionalidades de aplicaciones como Cleopatra, mencionada anteriormente, ofreciendo al usuario una mayor capacidad de autoevaluación en la práctica de símbolos. También amplia las posibilidades de las herramientas genéricas para toma de notas que suelen utilizar los intérpretes, y que son capaces de reconocer el texto escrito, ofreciendo la posibilidad de reconocer los símbolos. Por otro lado, existen múltiples conjuntos de datos de caracteres y dígitos manuscritos como IAM (Marti and Bunke, 2002) o MNIST (Deng, 2012) que permiten entrenar sistemas de aprendizaje computacional para el reconocimiento de la escritura manual. En este trabajo se ha desarrollado un conjunto de datos de símbolos que pueden utilizarse para dicho propósito.

En las secciones de este trabajo se habla en varios puntos de diferentes fases de ejecución. En cada fase se han ido aplicando diferentes métodos y realizado diversas tareas que han llevado al resultado final. Para aclarar que corresponde a cada fase durante la lectura de este trabajo, se indican en la figura 1 las fases y sus correspondientes métodos.

Este trabajo se estructura en las siguientes secciones: en las secciones 2 y 3 se tratan los antecedentes del trabajo divididos respectivamente en el estado actual de las tecnologías de

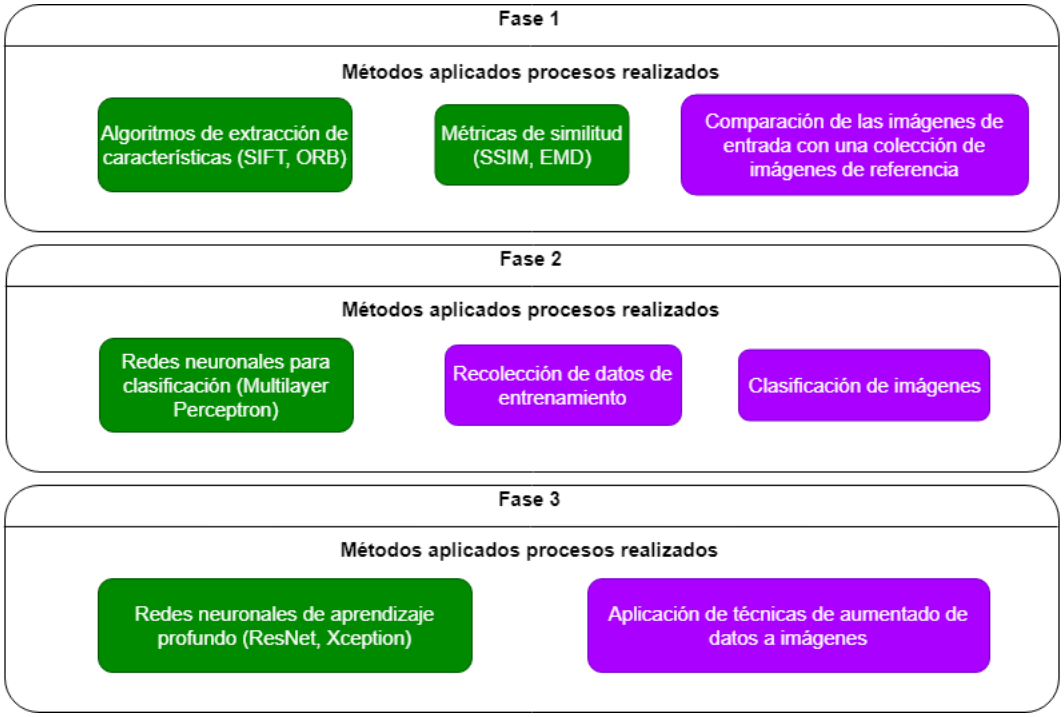

Figura 1: Clasificación de métodos y procesos realizados según las fases de desarrollo del trabajo

la interpretación, y los antecedentes en el procesamiento y comparación de dibujos o símbolos manuscritos. En la sección 4 se da una descripción del problema a resolver. En la sección 5 se expone una descripción técnica del sistema desarrollado. En la sección 6 se exponen los métodos y técnicas investigados. En la sección 7 se describen las características del conjunto de datos creado y el método que se ha seguido para crearlo. En la sección 8 se muestran las pruebas y los resultados obtenidos. Por último, en la sección 9, se exponen las conclusiones extraídas y las líneas de trabajo futuro previstas.

2. Antecedentes: la toma de notas y las tecnologías de la interpretación

Los antecedentes a la solución propuesta en el presente TFM pueden analizarse desde diferentes perspectivas. Por un lado, desde una perspectiva genérica, se aporta una solución a la laguna tecnológica que existe en el trabajo de los intérpretes, concretamente, para la tarea de la toma de notas. Por otro lado, para encontrar la solución final se han investigado y probado diferentes técnicas y métodos. En este apartado se van a presentar las soluciones y herramientas existentes que tratan de facilitar la toma de notas en cuanto al primer enfoque, y también las técnicas y métodos existentes que han dado lugar a la metodología de la solución propuesta.

2.1. La técnica de la toma de notas

El presente trabajo se centra en una de las técnicas que mayor importancia tiene en modalidades de interpretación como la consecutiva o la simultánea. Se trata de la toma de notas. En interpretación, la toma de notas consiste en la escritura de símbolos o formas abreviadas que representen palabras, expresiones o frases (entre otras) para ayudar a la memorización y optimizar el tiempo (Lung, 1999). La importancia de la toma de notas reside en la representación de las ideas y conceptos que se expongan durante el discurso (Gillies, 2017). Aunque el conjunto

de símbolos y la metodología escogida de la toma de notas es personal de cada intérprete, es esencial preparar a los estudiantes en esta disciplina para sus futuros trabajos (Ulloa and Navarrete, 2014). Existen múltiples métodos y recomendaciones para la toma de notas, aunque una de las más trascendentes es la metodología de Jean-François Rozan. La metodología de Rozan propone 7 principios y 20 símbolos que proporcionan una base sólida en el entrenamiento en la interpretación (Rozan, 1956). En base a estos principios y las carencias tecnológicas que sufren los intérpretes, la técnica de la toma de notas resulta un campo abierto en el que aportar innovaciones que faciliten notablemente los trabajos de interpretación consecutiva.

2.2. La tecnología actual en la interpretación

Varios estudios recientes muestran el estado actual en cuanto a la inclusión de las nuevas tecnologías en el trabajo de los intérpretes. En (Annalisa, 2015) se repasan las tendencias de los últimos años en cuanto a la inclusión de las nuevas tecnologías en la preparación de nuevos intérpretes. Por otro lado, tanto en (Costa et al., 2014) como en (Corpas Pastor, 2018) se enumeran las herramientas que más se utilizan en interpretación actualmente. A pesar del creciente interés que existe en la integración de las nuevas tecnologías en el trabajo de los intérpretes, estos dos últimos estudios confirman que los avances en inteligencia artificial y procesamiento de lenguaje natural no están siendo aprovechados de la misma forma que en otros ámbitos.

En cuanto a la toma de notas, existen tres categorías de herramientas que pueden resultar de utilidad para la toma de notas en la interpretación consecutiva. Podemos encontrar las aplicaciones de toma de notas (Evernote[1], OneNote[2], etc.), los dispositivos de toma de notas (Livescribe[3], Neo Smartpen[4], etc.) y las herramientas de grabación/reconocimiento de voz .

Las primeras son herramientas de notas multipropósito (tomar apuntes, hacer listas, apuntar recordatorios, etc.) que permiten una gestión de las notas versátil, ofreciendo a los intérpretes muchas más opciones que el soporte en papel. Este tipo de aplicaciones han conseguido que las *tablets* se conviertan en un dispositivo muy extendido para la toma de notas (Goldsmith, 2018).

Con respecto al segundo tipo de herramientas, se trata de dispositivos que se conectan (generalmente por *bluetooth*) a nuestro equipo de trabajo, ya sea un *smartphone*, una *tablet* o incluso un ordenador. Estos dispositivos consisten en bolígrafos o soportes sobre los que escribir que digitalizan la escritura de forma automática, permitiendo al usuario mantener el soporte tradicional en papel a la vez que obtiene las ventajas del soporte electrónico, pudiendo almacenar las notas escritas o incluso notas de voz asociadas a las escritas.

Una de las grandes ventajas de estas dos categorías de herramientas es que ambos tipos suelen incluir entre sus funciones la posibilidad de transcribir lo escrito a texto digital mediante técnicas OCR. En el caso que nos ocupa, esta funcionalidad puede resultar de gran utilidad para los intérpretes en la tarea de transcribir las notas. El gran inconveniente reside en que, como se ha explicado en el apartado anterior, en la interpretación consecutiva las notas están formadas por símbolos o abreviaturas que resultan poco útiles a pesar de que pudieran transcribirse, o que incluso no es posible hacerlo.

En cuanto a la tercera categoría, las herramientas de grabación de voz pueden resultar útiles como asistente durante la interpretación, pero sobre todo suelen utilizarse como herramientas para practicar, dado que es común que los intérpretes utilicen grabaciones para entrenarse en la captación del discurso. También pueden utilizarse herramientas de reconocimiento de voz para

[1] `https://evernote.com/intl/es`
[2] `https://products.office.com/es-es/onenote/digital-note-taking-app`
[3] `https://www.livescribe.com/site/`
[4] `https://www.neosmartpen.com/en/`

tomar notas rápidamente. De hecho, muchas de las herramientas de las otras dos categorías incluyen grabación de voz asociada a las notas escritas o incluso reconocimiento de voz para la ejecución de comandos o para convertir la voz en texto.

2.3. La formación de intérpretes y la toma de notas

A pesar de ser una técnica tan importante en la interpretación, los intérpretes no suelen optar por seguir una metodología concreta de toma de notas, si no que cada profesional establece sus pautas y recursos propios. Esto tiene sentido dado que en interpretación el propósito de las notas se reduce solo al contexto del momento de la interpretación. De hecho, establecer una metodología y técnica personal suele ser lo más conveniente para conseguir la mayor eficiencia posible en la captación de un discurso. El inconveniente reside en que llegar a diseñar, ejecutar y pulir la toma de notas requiere de mucha práctica y de una base técnica y metódica. Existen metodologías como el manual de Rozan (Rozan, 1956) que lleva aplicándose con éxito en la docencia desde que fue creado; y a pesar de ello, los alumnos de interpretación echan en falta una formación más completa en la técnica (Ulloa and Navarrete, 2014).

En cuanto a recursos, existen plataformas para practicar la interpretación, como *Black Box* (Sandrelli, 2005), que ofrecen ejercicios variados y colecciones de archivos de audio y vídeo para realizar simulaciones de interpretación supervisadas directamente por un docente. También existen plataformas de recursos lingüísticos, entornos virtuales o repositorios de discursos (Corpas Pastor, 2018) que resultan muy útiles para preparar un trabajo de interpretación o para la formación de alumnos. En cuanto a la toma de notas, existen iniciativas muy interesantes como la aplicación *Cleopatra* (Torre Salceda, 2017) que plantean el aprendizaje de símbolos como un juego de memoria en ambos sentidos. Ofrece un ejercicio en el que te muestra un símbolo aleatorio y el usuario debe adivinar que concepto representa; o la alternativa contraria, en la que se muestra un concepto, el usuario debe dibujar el símbolo correspondiente en un tiempo determinado y la aplicación muestra el símbolo correcto para poder comprobar el resultado. La idea de está aplicación se basa en otra herramienta multiplataforma creada por Jenny Ackerman, de la Universidad de Leipzig. Estas dos ideas destacan por su exclusividad, ya que no existe ninguna otra herramienta que ofrezca las mismas funcionalidades.

Todas estás herramientas poseen un inconveniente común: necesitan de una evaluación externa o de una autoevaluación por parte del usuario. En los casos en los que se necesita una evaluación externa, se dificulta la independencia del alumno a pesar de que se reduzca la intervención y el trabajo del docente. Por otro lado, en las herramientas que permiten la total independencia del alumno, existe una falta de automatización de la evaluación de los ejercicios, ya que en muchos casos deben autoevaluarse ellos mismos. La efectividad de estas herramientas aumentaría si se integran en ellas funcionalidades automáticas que evalúen las prácticas realizadas, consiguiendo una mayor efectividad del aprendizaje en menos tiempo.

3. Antecedentes: análisis de dibujos o escritura a mano

Ya sea en formato electrónico o en papel, las notas de los intérpretes consisten en una colección de símbolos dibujados a mano. Por ello, el problema a resolver se presenta como una clasificación o reconocimiento de los símbolos en forma de imágenes. En esta línea, se han planteado varias formas de abordar una posible solución:

1. Con una colección de símbolos de referencia, que pueden ser símbolos genéricos o personales del usuario, se comparan las imágenes de entrada (símbolo dibujado) con cada una de las imágenes de referencia para encontrar la que más se asemeje.

2. Utilizar un modelo de aprendizaje computacional entrenado para clasificar imágenes con un conjunto de símbolos de intérpretes. De esta forma, se podrían tener diferentes símbolos para cada concepto representado.

Para resolver el problema mediante el primer enfoque, se han planteado dos paradigmas: las métricas de similitud y los algoritmos de comparación. Tanto dentro de estas dos posibilidades como en el caso del aprendizaje computacional, existen diferentes posibilidades que se pueden aplicar. En las siguientes secciones se exponen diferentes métodos de cada caso, indicando las ventajas e inconvenientes de cada una.

3.1. Métricas de similitud

Las métricas de similitud son métodos de comparación de imágenes basadas en técnicas estadísticas que realizan una valoración cuantitativa entre dos imágenes. Existen muchas y cada una valora diferentes aspectos de las imágenes, por lo que en base al problema, es más conveniente una que otra. Estas métricas se utilizan en problemas de visión por computador de ámbitos muy diferentes. Algunas de las más utilizadas en la literatura son las siguientes:

- **Índice de Similitud Estructural:** Esta métrica trata de detectar la diferencia estructural entre dos imágenes. Se basa en percibir la degradación de una imagen en base a la relación entre los píxeles de la misma con el fin de medir la calidad. Existen diferentes variantes de esta métrica con aplicabilidad y propósito diferentes, pudiendo utilizarse no solo para medir la calidad, si no para otros propósitos de la comparación de imágenes, como la detección de patrones entre imágenes diferentes (Wang et al., 2010).
- **Distancia de Wasserstein o EMD:** La Distancia de Wasserstein o *Earth Mover's Distance (EMD)* mide la distancia entre dos distribuciones de probabilidad de una misma región. En la comparación de imágenes, la EMD se utiliza sobre colecciones de imágenes en escala de grises (Huang et al., 2014) para detectar patrones o defectos. El problema de esta métrica se encuentra en el alto coste computacional que supone, por lo que suele utilizarse junto a otros métodos o algoritmos que optimicen su cálculo.

Para el caso presentado en este trabajo, las métricas presentan la ventaja de no necesitar una gran cantidad de datos para poder ser aplicadas. Sin embargo, el problema de estos métodos reside en el alto volumen de consumo de recursos que supone su cálculo. Por este motivo, a día de hoy se utilizan métodos más ligeros, o se coordina el uso de métricas con otros métodos que optimicen su cálculo (Courty et al., 2017).

3.2. Algoritmos de extracción de características

Otro método común en la visión por computador son los algoritmos de detección o extracción de características. Estos algoritmos realizan una serie de operaciones y procesos sobre una imagen o una colección de ellas para extraer características que permitan obtener información útil. Estos algoritmos poseen muchas aplicaciones, como la clasificación de imágenes, detección de objetos, reconocimiento de escenarios, etc.

Como en el caso de las métricas, no existe un algoritmo perfecto, si no que lo mejor es encontrar el adecuado al problema que se intenta resolver. En este caso, nos interesa aplicar estos algoritmos en el primer enfoque que se describió al principio de esta sección, en la comparación de imágenes. Siguiendo esta línea, se ha utilizado como referencia el trabajo de (Karami et al., 2017) donde se comprueba el rendimiento de tres de estos algoritmos (SIFT, Lowe, 2004; SURF, Herber et al., 2008; ORB, Ethan et al., 2011) en la comparación de imágenes. Estos algoritmos

funcionan de manera similar identificando una serie de puntos clave en cada una de las imágenes para compararlas entre sí determinando la diferencia entre pares de puntos utilizando diferentes métodos matemáticos. Los resultados de las pruebas revelan que SIFT ofrece un mayor índice de acierto, sin embargo el tiempo de respuesta es mucho mayor.

Al igual que en el caso de las métricas, los algoritmos de extracción de características presentan la ventaja de necesitar tan solo una imagen de referencia para poder procesar la comparación. Además, estos algoritmos ofrecen buenos resultados y siguen utilizándose y evaluándose las posibilidades que ofrecen. A pesar de esto, presentan un problema de rendimiento: para obtener buenos tiempos de respuesta hay que sacrificar precisión en los aciertos. Es por ello que algoritmos como SIFT suelen utilizarse a día de hoy en conjunto a otros métodos como los algoritmos genéticos (Taqdir and Dhir, 2017) o las redes neuronales de aprendizaje profundo (Zhang et al., 2016).

3.3. Aprendizaje computacional

Tras los avances en los últimos años, es muy común hoy día la aplicación de algoritmos y métodos de aprendizaje en tareas con imágenes, sobre todo desde el aumento de popularidad del *Deep Learning* cuyos resultados en problemas con imágenes sobresalen con respecto a otros métodos.

En la clasificación de imágenes se han creado y probado diversos modelos de aprendizaje utilizando diferentes conjuntos de datos. El conjunto de datos MNIST (Deng, 2012), formado por un gran conjunto de imágenes de dígitos escritos a mano, es uno de los conjuntos de datos más utilizados en pruebas de rendimiento de modelos de aprendizaje. En la literatura pueden encontrarse pruebas con este conjunto de datos en diferentes métodos como las máquinas de soporte vectorial (Keysers, 2007) o las redes neuronales (El Kessab et al., 2013).

Otro conjunto de datos de imágenes más actual y de gran popularidad utilizados en pruebas de modelos de aprendizaje es ImageNet (Deng et al., 2009); el cual es ampliamente utilizado en pruebas con redes neuronales de aprendizaje profundo. En los últimos años han surgido arquitecturas de redes de aprendizaje profundo como ResNet (He et al., 2016) de *Microsoft* o Xception (Chollet, 2017) de *Google* que han conseguido ratios de acierto superiores al 90 % en pruebas con este conjunto de datos. ResNet fue la arquitectura ganadora del ILSVRC del año 2015, mientras que Xception es una mejora de una red anterior, Inception, pero superó a ResNet en los resultados que obtuvo en el reto.

En este trabajo se ha planteado la integración de un modelo de aprendizaje profundo para resolver el problema desde el segundo enfoque descrito. El inconveniente de este enfoque reside en la necesidad de conjuntos de datos de gran volumen para conseguir buenos resultados de predicción. A pesar de ello, existen métodos de generación automática de datos que ofrecen buenos resultados. En (Perez and Wang, 2017) se explora la efectividad de diferentes técnicas de manipulación de las imágenes que forman un conjunto de datos para mejorar los resultados de una red neuronal.

4. Descripción del problema

En interpretación, la toma de notas consiste en anotar rápidamente y de forma memotécnica el discurso del hablante, realizando una codificación intermedia del mensaje en el idioma origen para la posterior decodificación al idioma objetivo. En la mayoría de las situaciones resulta prácticamente imposible escribir exactamente todo lo que dice el hablante, por lo que dicha codificación consiste en anotar símbolos que representen las ideas o conceptos que se tratan durante el discurso. Debido al estrés al que se encuentra sometido el intérprete, es importante

que las anotaciones resulten precisas a la vez que concisas de forma que no se pierda ninguna información del mensaje durante el proceso. Por este motivo, la toma de notas requiere de muchísima práctica y una metodología bien definida para poder llegar a dominarse.

Existen diversas guías y metodologías que tratan de definir una técnica de toma de notas precisa y eficiente (Chen et al., 2016). Una de las más extendidas es la metodología de Rozan, que define un conjunto de reglas y una base de 20 símbolos. Sin embargo, dichas metodologías se centran sobre todo en la formación ya que cada intérprete acaba definiendo sus propios símbolos y técnicas personales a fin de crear una metodología lo más adaptada posible a su manera de trabajar. Esto se debe a que el objetivo de las notas es solo el de ayudar a traducir el mensaje que se transmite, perdiendo su valor de forma posterior. A pesar de ello, puede ser interesante archivar las notas como histórico del trabajo realizado.

La principal idea del presente trabajo es la de ayudar a los intérpretes en los procesos que involucren la toma de notas. Para saber como proceder, se han analizado los problemas mencionados en anteriores secciones sobre la toma de notas: la ausencia de herramientas específicas, la necesidad de mejorar las actividades en la práctica de la técnica y la idoneidad de que cada intérprete cree su propio método para tomar las notas. En base a esto, se han identificado tres situaciones que se podrían semiautomatizar:

1. Durante un trabajo de interpretación se puede ayudar al intérprete tanto ofreciendo sugerencias de símbolos o abreviaciones como, en sentido contrario, transcribiendo a texto los símbolos que dibuja el intérprete.
2. De forma posterior a un trabajo de interpretación puede resultar útil archivar las notas en forma de texto. Debido a que las notas son personales, es necesario "traducir"los símbolos y abreviaturas de las notas personales a texto completo. Una herramienta capaz de transcribir las notas a texto ayudaría al intérprete a ahorrar tiempo del archivado.
3. En cuanto a la práctica de símbolos, se pueden diseñar ejercicios en los que se supervisen automáticamente la corrección de las notas tomadas o ejercicios de memorización de los símbolos en los que el alumno podría comprobar de forma directa si ha dibujado el símbolo correcto asociado a un concepto.

El primer caso tampoco resultaría especialmente útil dado que actualmente existen tecnologías e investigación activa en las aplicaciones del reconocimiento de voz como asistencia en la interpretación (Fantinuoli, 2017). Por este motivo, el trabajo se ha centrado sobre todo en el tercer punto.

Tratamos entonces con un problema basado en el reconocimiento de los símbolos dibujados. Los símbolos suelen ser dibujos simples y muy representativos de los conceptos a los que son asociados. El problema principal es el de reconocer que concepto se está representando con cada símbolo dibujado. Es, entonces, un problema similar al de reconocimiento de letras o números manuscritos, con la diferencia de que pueden existir dos o más representaciones distintas de cada concepto.

La idea es implementar un sistema que dado una imagen de entrada (símbolo) sea capaz de ofrecer el término o concepto asociado a dicha imagen. En la figura 2 se muestra un diagrama simple del proceso que se va a implementar. En el diagrama puede verse como una imagen de entrada es procesada por un sistema de reconocimiento que da como salida una etiqueta (término asociado). Por otro lado, en la figura 3 se muestra un diagrama prácticamente idéntico. La diferencia la encontramos en que a pesar de mostrarse una imagen de entrada diferente, el sistema devuelve la misma etiqueta. Esto es así debido al problema, anteriormente mencionado, de la diferencia de representación de conceptos entre diferentes intérpretes.

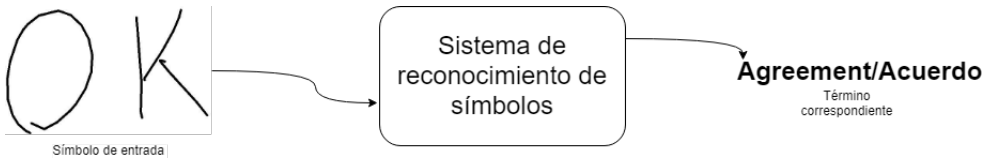

Figura 2: Diagrama del proceso deseado

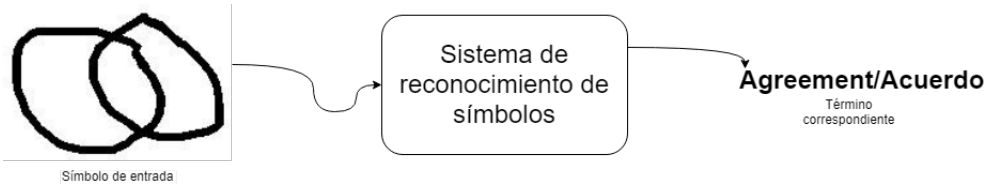

Figura 3: Situación alternativa del proceso

Se va a tratar entonces de implementar el *sistema de reconocimiento de símbolos* que a partir de un conjunto de términos sea capaz de establecer una relación entre las imágenes de entrada y uno de esos términos.

Debido también a la baja estandarización de los símbolos, otro gran problema encontrado a la hora de dar una solución es que no existen datos publicados. Para poder crear un sistema de aprendizaje es esencial contar con un conjunto de datos con un volumen suficiente. En este caso, no existe ninguna base de datos o conjunto de imágenes de símbolos que pueda usarse para entrenar un sistema.

Por último, y estrechamente relacionado con el problema de la escasez de datos, un asistente para la toma de notas debería poderse adaptar al usuario, pudiendo ser entrenador por el mismo sin que suponga un gran esfuerzo.

5. Herramienta de reconocimiento de símbolos

La solución propuesta se ha basado en el desarrollo de una herramienta que fuera capaz de reconocer símbolos. Para eliminar problemas de compatibilidad y facilitar la difusión, la herramienta consiste en una aplicación web sencilla que se comunica con un servidor en la que se pide al usuario de dibuje un símbolo y compruebe si el sistema es capaz de reconocerlo. Se puede acceder a la web en el siguiente enlace: `javilimatfm.ddns.net`

En la figura 4 se muestra un diagrama con los elementos que componen el sistema y las relaciones que se establecen entre ellos. En los siguientes subapartados se van a enumerar dichos elementos, las tecnologías utilizadas en cada uno y el comportamiento conjunto de todos ellos.

5.1. La aplicación web

Debido a la simplicidad del propósito de la herramienta, no se ha optado por utilizar ningún framework de desarrollo. Se ha utilizado HTML y CSS para la vista, y *Javascript* para el controlador. En la figura 5 se muestra la vista inicial, que se compone de un canvas, programado para permitir al usuario dibujar sobre él, y dos botones: uno permite limpiar el canvas y otro que inicia el reconocimiento del símbolo dibujado.

Cuando el usuario inicia el reconocimiento del símbolo, el controlador se encarga de realizar las comunicaciones con el servidor, enviando la imagen del símbolo y quedando a la espera de respuesta. Una vez recibido el resultado desde el servidor, se encarga de enviar a la vista el resultado obtenido. Para finalizar el proceso, se pide al usuario que compruebe si el resultado

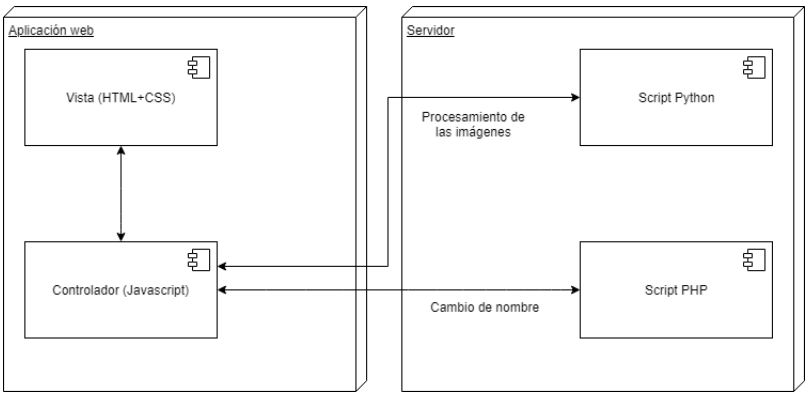

Figura 4: Componentes y relaciones del sistema

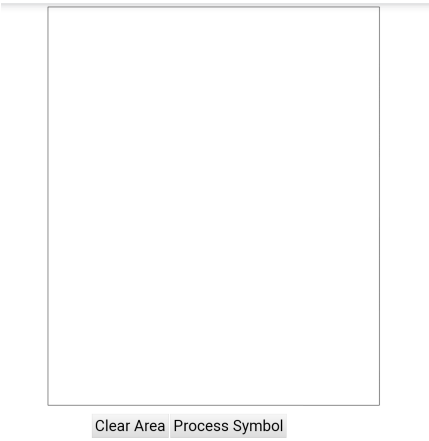

Figura 5: Interfaz inicial de la herramienta

es correcto y que proporcione el correcto en caso negativo. En la figura 6 se puede ver como la herramienta ofrece el resultado obtenido y pregunta al usuario si es el resultado correcto.

La web se encuentra alojada en un servidor accesible desde Internet para eliminar procesos de instalación y facilitar su uso. En este mismo sentido, se ha optado por diseñar una interfaz simple con elementos de tamaño suficiente para hacerla apta al uso desde dispostivos móviles. De esta manera los usuarios pueden acceder desde una tablet o smartphone a la herramienta para poder utilizar su propio dedo o un lápiz táctil para que la experiencia sea similar a la del papel.

5.2. Los *scripts* del servidor

La web se comunica con dos *scripts* escritos en Python y PHP que realizan el reconocimiento de los símbolos y el almacenamiento de las imágenes recibidas. El *script* Python recibe la imagen y la procesa en varios pasos:

1. Recibe la imagen en forma de buffer y la convierte en un archivo de imagen que almacena en el servidor.
2. Preprocesa la imagen convirtiéndola a una matriz de píxeles y recortando el espacio en blanco para que el contenido se limite al símbolo dibujado.
3. Procesa la imagen y devuelve el concepto que estima que se corresponde al símbolo recibido.

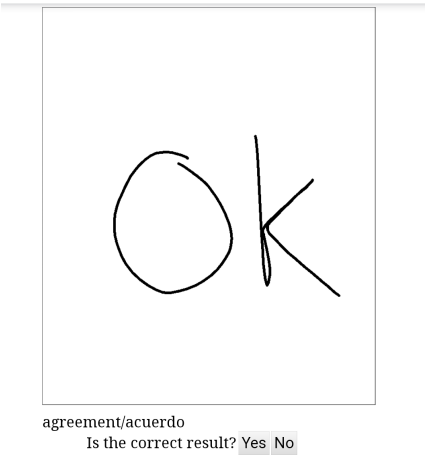

Figura 6: Se solicita al usuario que compruebe el resultado

Este *script* almacena las imágenes procesadas y recortadas con la finalidad de recolectar datos. Dichas imágenes se almacenan con el nombre del concepto correspondiente al símbolo. En caso de que el sistema no haya identificado correctamente el símbolo, el controlador del cliente web comunica el concepto real proporcionado por el usuario al *script* PHP, que es el encargado de renombrar el archivo de la imagen del símbolo que el usuario ha enviado.

6. Métodos y técnicas aplicadas

Son varios los métodos y técnicas que se han planteado utilizar durante el desarrollo del trabajo. Se van a exponer en este apartado que técnicas y como se han aplicado, y que observaciones se han obtenido al hacerlo, justificando que técnica ha sido la finalmente elegida. En apartados posteriores se mostrarán los resultados exactos de las pruebas que se han realizado con cada método, comparándolos entre sí.

6.1. Comparación de imágenes

En la primera fase del proyecto se tomó un enfoque del problema que consiste en comparar las imágenes de los usuarios con una base de datos de símbolos de referencia. Dicha base de datos está formada en una colección de veintitrés imágenes de símbolos asociadas al concepto que representan. Las imágenes y sus conceptos asociados fueron extraídas de la web *Interpreter Training Resources*[5] cuyo creador, Andrew Gillies, es autor de diversos libros y cursos sobre interpretación consecutiva.

Las imágenes, como la que podemos ver en la figura 7, consisten en ejemplos de símbolos manuscritos que se utilizan en la toma de notas. Dichas imágenes funcionan como referencia para la aplicación de dos métodos de comparación: el índice de similitud estructural y el algoritmo ORB de extracción de características.

Para calcular el índice de similitud estructural, se ha utilizado el módulo *measure* del paquete *scikit-image* de *python* (Van der Walt et al., 2014) que implementa una función que calcula dicho índice entre dos imágenes. Las imágenes se preprocesan obteniendo dos matrices de dos dimensiones (escala de grises) del mismo tamaño. La función procesa ambas matrices y devuelve un valor entre 0 y 1 correspondiente al índice de similitud estructural entre las dos imágenes.

[5] `http://interpreters.free.fr`

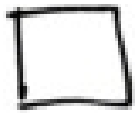

Figura 7: Símbolo correspondiente al concepto *país*

En el caso de la comparación mediante el algoritmo ORB, se ha utilizado la implementación del mismo incluida en el paquete *open-cv* de *python*[6]. En este caso las imágenes son procesadas normalmente y se les aplica el detector de características ORB. Una vez obtenidas las características de cada imagen, se comparan los puntos en común y se obtiene un conjunto de regiones similares. Finalmente, se devuelve la división del número de elementos en las regiones similares entre el número de puntos coincidentes totales. De esta manera, la similitud se corresponde a un valor entre 0 y 1.

En la herramienta ambos métodos son aplicados de forma que se calcula la similitud entre el símbolo del usuario y cada uno de los símbolos de la base de datos, obteniendo una lista de ratios de similitud, uno por cada concepto de la base de datos. De entre esa lista se busca el mayor valor y se devuelve el concepto correspondiente al símbolo más parecido.

La ventaja del método de comparación es que solo necesita una imagen de referencia por cada símbolo, motivo por el cual se abordó el problema de está forma en las primeras fases. El segundo planteamiento, basado en aprendizaje computacional, presentaba el problema de necesitar conjuntos de datos de mayor volumen para ofrecer buenos resultados. Mientras que obtener las imágenes de referencia ha resultado sencillo, obtener un conjunto de datos de imágenes para un modelo de aprendizaje automático supone una tarea mayor que decidió evitarse en esta fase del trabajo. Además, el método de comparación permitiría adaptar el sistema a cualquier usuario solicitando tan solo una imagen de referencia para cada símbolo que se quiera introducir. Sin embargo, los resultados obtenidos de estos métodos no eran fiables ni ofrecían un buen rendimiento. Como añadido, para casos genéricos, se encontró el problema de la diferencia de representación entre diferentes intérpretes, que daba lugar a que un mismo concepto pudiera representarse de formas muy diferentes para cada usuario. Por este motivo, en las fases posteriores, se optó por crear un conjuntos de datos que sirviera para entrenar un sistema de aprendizaje. En el siguiente subapartado se exponen los métodos de aprendizaje automático que se se han utilizado y en el apartado 7 del documento se darán detalles sobre las características y el proceso de compilación del conjunto de datos de entrenamiento.

6.2. Clasificación de imágenes con redes neuronales

Tras las primeras pruebas con las técnicas de comparación de imágenes, se optó por intentar solucionar el problema de la falta de datos para entrenar un modelo de aprendizaje mediante la recopilación de las imágenes utilizando la herramienta. De esta manera, se ha conseguido un aporte extra mediante este trabajo, que es la creación de un conjunto de datos inexistente actualmente de forma libre. Este conjunto de datos no se encuentra publicado actualmente, aunque se entregará como parte de este trabajo y se planteará su publicación en el futuro. En la sección 7 se darán detalles sobre dicho conjunto de datos y como ha sido creado.

Una vez obtenido los datos de entrenamiento se realizaron pruebas entrenando redes neuronales artificiales para clasificar las imágenes. De esta forma, las imágenes del conjunto de datos son procesadas como los datos fuente, mientras que las etiquetas son los conceptos representados por cada imagen. En total se han probado tres redes: Un percentron multicapa en la segunda fase del desarrollo, y dos arquitecturas de aprendizaje profundo en la tercera fase. Se van a dar detalles de las características de cada red y como se han aplicado para tratar de resolver el problema.

En el caso del perceptrón multicapa se ha utilizado la biblioteca de aprendizaje computacional *scikit-learn* de *python* (Pedregosa et al., 2011). La red está formada por dos capas ocultas

[6] `https://opencv-python-tutroals.readthedocs.io/en/latest/`

con tres neuronas cada una que utiliza ReLU como función de activación y *Adam* como función de optimización ($\alpha = 1e-3$; $\beta 1 = 0,9$; $\beta 2 = 0,999$; $\epsilon = 1e-8$). Para el entrenamiento se ha aplicado una tolerancia de parada de $1e-3$. Para preparar los datos, las imágenes son preprocesadas transformando primero cada imagen a una matriz de 299×299 (escala de grises) que luego es transformada a un vector concatenando sus filas. Por otro lado, las etiquetas se proporcionan como cadenas de caracteres. El inconveniente de este método es que presenta un problema de pérdida de información producido por la necesidad de vectorizar las imágenes, dando lugar a que las pruebas mostraran resultados de precisión muy bajos.

Como alternativa al perceptrón multicapa, se optó por realizar pruebas de aprendizaje profundo basado en redes neuronales convolucionales. En este caso se han probado dos arquitecturas de red que cuentan con cierta popularidad. Se trata de la red ResNet (He et al., 2016) introducida por *Microsoft* y la red Xception (Chollet, 2017) de *Google*. Ambas arquitecturas se encuentran implementadas en la biblioteca *Keras* (Chollet et al., 2015) de aprendizaje profundo para python. Al igual que en el caso del perceptrón multicapa, se ha tratado el problema como un problema de clasificación de imágenes, solo que existen dos diferencias significativas en el preprocesado del conjunto de datos. Al contar con la posibilidad de utilizar tensores, las imágenes se han transformado a una estructura multidimensional de $224 \times 224 \times 3$ en el caso de ResNet y $299 \times 299 \times 3$ en el caso de Xception. En la figura 8 puede observarse la arquitectura y los parámetros de la red ResNet, mientras que en la figura 9 se muestran los de la red Xception. En dichos diagramas se pueden ver las capas que las componen y los parámetros correspondientes a cada una. Las redes utilizadas en este trabajo poseen esa misma arquitectura, exceptuando la capa de salida, que se ha adaptado a las condiciones del problema. Para el caso de este trabajo, ha optado por preprocesar las etiquetas utilizando la técnica de codificación one-hot. De esta forma, cada etiqueta se ha transformado a un vector de 24 elementos de los cuales 23 son ceros y el correspondiente a la posición del concepto en una lista ordenada alfabéticamente es un uno. Por ello, la capa de salida de las redes ResNet y Xception se ha modificado a un tamaño de 24, correspondiente a las 24 posibles clases del sistema. Para los entrenamientos, se han realizado cinco fases y en cada fase se ha procesado el conjunto de entrenamiento completo.

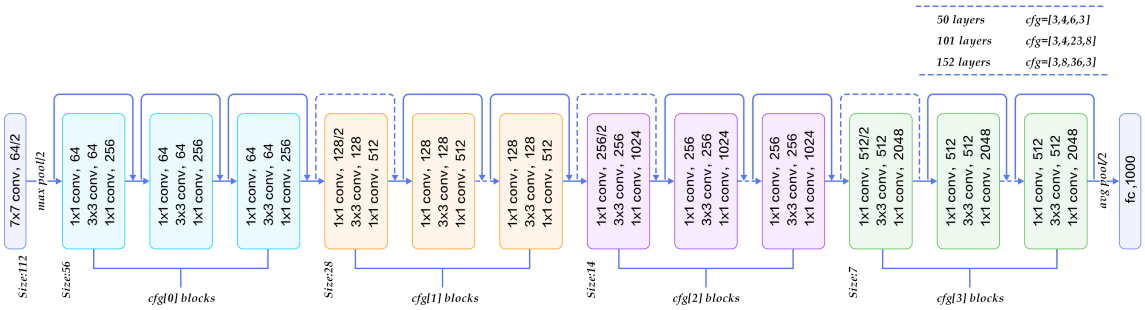

Figura 8: Arquitectura de la red ResNet

Debido al reducido tamaño del conjunto de datos, se han aplicado técnicas de generación automática de datos para conseguir mejores resultados en todos los entrenamientos. Concretamente, se ha aplicado distorsión elástica utilizando la biblioteca *Augmentor* de python (Bloice et al., 2019). A su vez, se ha duplicado el número de imágenes (originales más deformadas) girando en un ángulo de 90º hacia la derecha cada una de ellas. Este método ha permitido obtener una mejora en los resultados, obteniendo ratios de precisión cercanos al 100 % con las redes ResNet y Xception.

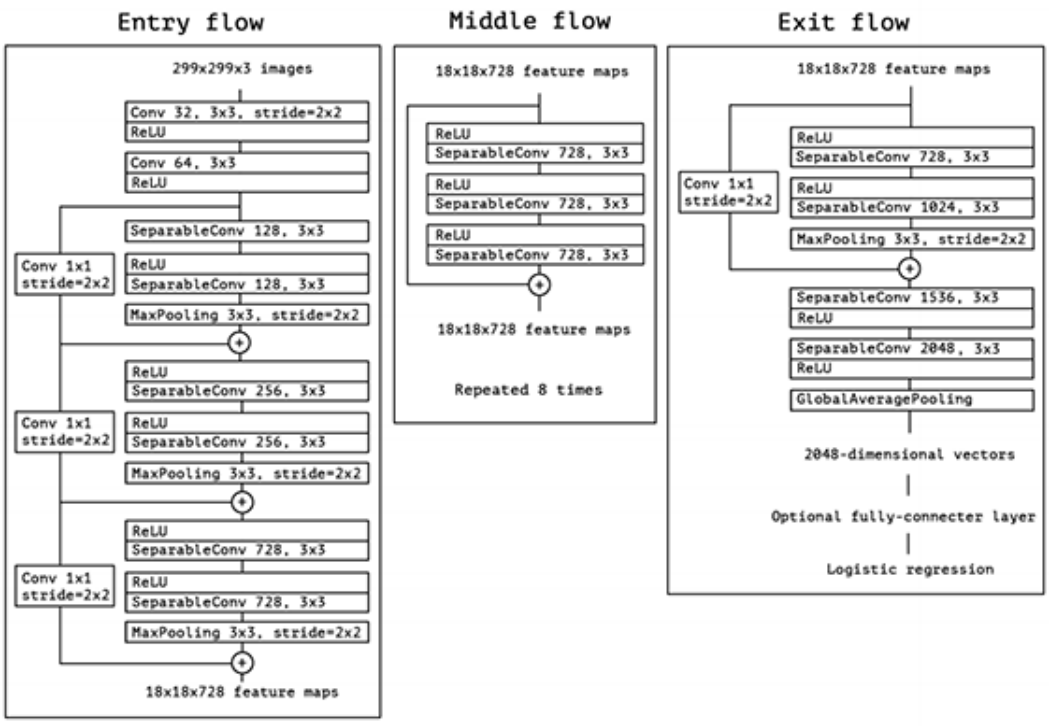

Figura 9: Arquitectura de la red Xception

7. Creación de un conjunto de datos de símbolos

En la primera fase del trabajo se creó una primera versión de la herramienta que utilizaba comparación de imágenes para el reconocimiento de los símbolos. Tras optar por compilar un conjunto de datos para entrenamiento de modelos de aprendizaje, se adaptó la herramienta, permitiendo recopilar las imágenes de los símbolos dibujados por los usuarios.

Las imágenes se almacenan en el mismo servidor donde se encuentra alojada la web. La forma en la que son almacenadas es sencilla: cada imagen se guarda en un archivo jpeg nombrado con el concepto que representa el símbolo seguido de un guión y un número. El número es asignado según el número de imágenes que haya en la carpeta destino, sumando uno al número total de imágenes que haya en ese momento. Los archivos se nombran a priori con el concepto que el sistema determina dado el símbolo de entrada y posteriormente se solicita al usuario que indique si el sistema a reconocido el símbolo correctamente. En la figura 6 del apartado 5 se muestra el resultado ofrecido por la herramienta, que es el nombre actual del archivo correspondiente a dicho símbolo, y los botones que permiten al usuario responder si el resultado es correcto. En caso afirmativo, el archivo se queda con el nombre dado, mientras que en caso negativo, se inicia un segundo proceso.

En el caso negativo, la interfaz cambia a un estado en el que se solicita al usuario que seleccione el término correcto en un desplegable. En la figura 10 se muestra la interfaz de este proceso. En ella puede verse el botón que extiende el desplegable que permite seleccionar el concepto correcto de entre todos las posibilidades que contempla el sistema. En la figura 11 se muestran las opciones que el usuario puede escoger en el desplegable. Una vez el usuario ha seleccionado el término correcto, puede pulsar el botón “Ok” para que el sistema cambie el nombre del archivo con el concepto correcto.

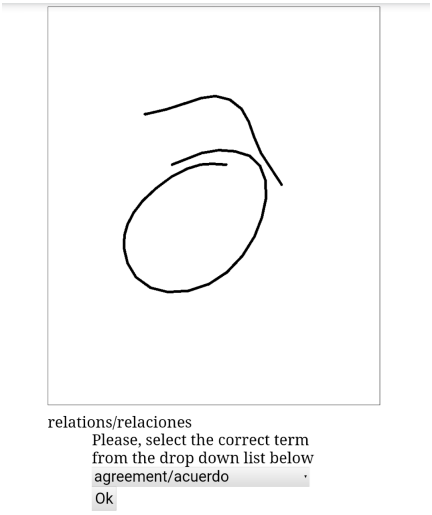

Figura 10: Captura del caso de error, en el que el usuario puede utilizar el desplegable para seleccionar el término correcto

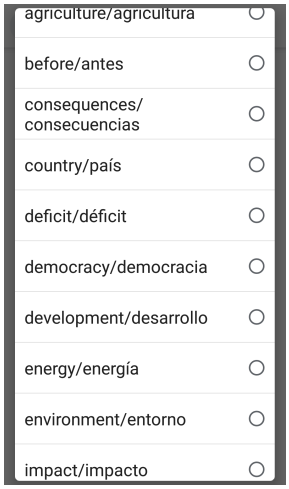

Figura 11: El desplegable mostrado con los términos contemplados por el sistema

Con este método se realizaron pruebas de la herramienta durante la primera fase en las que se aplicaba comparación de imágenes, con el único propósito de recopilar datos. Para conseguir imágenes de símbolos lo más cercanos posibles a los símbolos que utilizan los intérpretes en casos reales, se ofreció la herramienta a cuatro estudiantes de último año del Grado de Traducción e Interpretación de la Universidad de Málaga. También se difundió la herramienta entre algunos intérpretes profesionales para que pudieran probarla. La herramienta no ofrecía un buen funcionamiento en esta fase, por lo que tan solo se pidió a los usuarios que proporcionaran sus símbolos utilizando la herramienta y así también poder comprobar su funcionamiento de cara a pruebas futuras. Tras el período que duraron estas pruebas se obtuvo un conjunto de 800 imágenes de los 23 términos del sistema. En la siguiente lista se muestran los 23 términos incluidos en el conjunto de datos:

1. Agreement/acuerdo
2. Agriculture/agricultura
3. Consequences/consecuencias

4. Country/país
5. Déficit/déficit
6. Democracy/democracia
7. Development/desarrollo
8. Energy/energía
9. Environment/entorno
10. Impact/impacto
11. Industry/industria
12. Inflation/inflación
13. Meeting/encuentro
14. Money/dinero
15. Politics/política
16. Problem/problema
17. Relations/relaciones
18. Repression/represión
19. Role/rol
20. Success/éxito
21. Surplus/excedente
22. Trade/comercio
23. Work/trabajo

El conjunto de datos obtenido posee un tamaño reducido y es desequilibrado en cuanto a la cantidad de imágenes de cada concepto. Para los conceptos más populares se han podido obtener más de treinta imágenes, mientras que existen conceptos de los cuales han podido obtenerse apenas tres imágenes. Por otro lado, en las imágenes obtenidas se aprecia la falta de estandarización entre los símbolos que se ha comentado en apartados anteriores, existiendo casos en los que las imágenes de un mismo concepto difieren entre sí. A pesar de esto, se ve una clara tendencia para cada símbolo. En la figura 12 se muestran los símbolos que predominan de cada concepto incluido en el sistema numerados según el término de la lista anterior. Dentro de las imágenes obtenidas de cada concepto, encontramos que más de la mitad coinciden o son muy similares al símbolo mostrado en la figura, incluyendo aquellos casos en los que se han obtenido una cantidad de imágenes escasa.

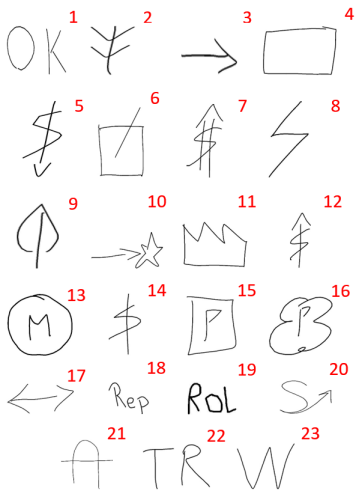

Figura 12: Imágenes de los símbolos que predominan en el conjunto de datos para cada concepto

A pesar de estos problemas, el conjunto de datos presenta la gran ventaja de tratarse de imágenes sencillas, en blanco y negro y con formas simples. Esto permite aplicar técnicas de generación automática de los datos que pueden ofrecer resultados satisfactorios.

Uno de los problemas resueltos es el de la disyuntiva entre símbolos del mismo concepto. Aunque no en todos los casos ha mostrado buenos resultados, hay casos en los que dos símbolos muy diferentes han sido reconocidos por la herramienta con el concepto correcto.

Se puede ver un ejemplo de este hecho en las figuras 13 y 14 en las que vemos como la herramienta ha sido capaz de reconocer con el mismo concepto los dos símbolos mostrados. El denominado símbolo principal se corresponde con el símbolo que predomina en el conjunto de datos como el correspondiente a *consequences*, existiendo un total de 32 imágenes correspondientes al mismo. Sin embargo, encontramos con que del denominado símbolo alternativo tan solo se cuenta con tres imágenes.

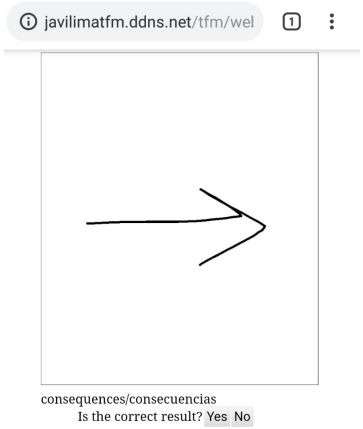

Figura 13: Símbolo principal del concepto *consequences*

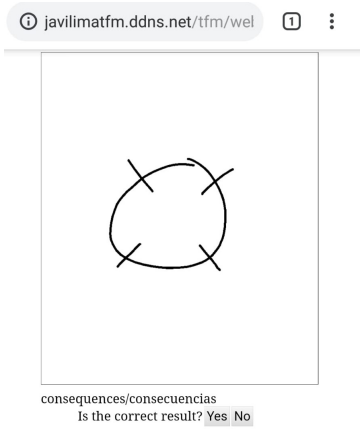

Figura 14: Símbolo alternativo al de la figura anterior del concepto *consequences*

Para solucionar el problema de la escasez de imágenes, se ha utilizado la biblioteca *Augmentor* (Bloice et al., 2019) que ha permitido aplicar la técnica de la distorsión aleatoria para generar nuevas imágenes. En la figura 15 se ilustra un ejemplo de como se genera de forma au-

tomática una nueva imagen utilizando esta técnica. De esta forma se han generado un total de 5000 imágenes aplicando un factor de distorsión de 0,8 sobre 1. Posteriormente, el conjunto de imágenes se duplica aplicando un giro de 90º hacia la derecha a cada una de las 5000 imágenes. Este conjunto de datos generado automáticamente es el utilizado en las pruebas de la sección 8.

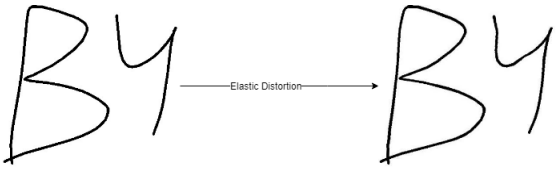

Figura 15: Ejemplo de aplicación de la técnica de distorsión aleatoria para la generación automática de nuevas imágenes

Cabe destacar que uno de los símbolos del conjunto de datos, el correspondiente al concepto *before*, ha sido sido añadido íntegramente utilizando generación automática de imágenes a partir de una imagen original con el objetivo de resolver el problema de la personalización del sistema. De esta manera, se podrían crear modelos personales de cada intérprete sin que se tenga que aportar una cantidad alta de imágenes, lo cual puede resultar incómodo para el usuario. En la sección 8 se muestra la prueba realizada con imágenes de este símbolo tras entrenar el sistema.

Junto con esta memoria, el trabajo incluye el conjunto de datos original con los símbolos proporcionados por el usuario además del conjunto total de imágenes generadas con *Augmentor*.

8. Pruebas y resultados

En el presente trabajo se han realizado diversas pruebas hasta dar con el método que mejor se adaptara al problema presentado. En esta sección se van a mostrar los resultados de las pruebas realizadas con las tres redes neuronales que se han planteado utilizar. En la sección 6 pueden verse diagramas detallados de las arquitecturas de las redes Xception (Chollet, 2017) y ResNet (He et al., 2016), además de toda la información necesaria sobre como se han aplicado a este problema concreto. Por otro lado, en el caso del perceptrón multicapa, se muestran los parámetros de la red y el número de neuronas de la capa oculta en esa misma sección.

8.1. Pruebas de entrenamiento

Para entrenar las diferentes redes neuronales que se han planteado para resolver el problema se ha realizado una división del conjunto de datos del que se habla en la sección 7 en datos de entrenamiento y datos de test. Concretamente, se ha separado, aleatoriamente, un 20 % del conjunto de datos total para probar el resultado del entrenamiento una vez finalizado. El problema de este método es que la división del conjunto de datos puede no ser la óptima. Por este motivo, se han hecho un total de 32 pruebas de entrenamiento con cada red y calculado la precisión de cada prueba. Debido a las limitaciones de recursos hardware, cada prueba de entrenamiento ha supuesto una cantidad de tiempo considerable. Este problema ha llevado a limitar la cantidad de las pruebas.

En la tabla 1 se muestran los resultados obtenidos. Los resultados sacan a relucir la alta efectividad de las redes de aprendizaje profundo frente al perceptrón multicapa, cuya media del ACC no alcanza ni 1 %. Por otro lado, entre ResNet y Xception la mejor arquitectura resulta ser Xception, con una precisión más que aceptable para poder ser integrada a la herramienta.

Con el fin de evitar el problema de la división aleatoria del conjunto de datos, se almacenaron los modelos cuyas pruebas ofrecieron un mayor resultado en los tests.

	Media del ACC (%)
Xception	**96,02 %**
ResNet	84,37 %
MLPerceptron	0,11 %

Tabla 1: Resultados de las pruebas de entrenamiento

8.2. Pruebas de rendimiento

Tras los resultados entrenando las redes, entre ResNet y Xception, la segunda sería preferible dado que es casi un 12 % mejor. Sin embargo, es importante tener en cuenta también el rendimiento de cada una. Por este motivo se realizó una segunda prueba midiendo la velocidad media de respuesta en predicciones utilizando los modelos almacenados; los cuales, como se ha explicado en el apartado anterior, fueron los correspondientes a la mejor prueba de precisión. Para la prueba de rendimiento, cada red procesó 32 imágenes y se midió el tiempo transcurrido en el reconocimiento de cada una. Después se calculó la media de tiempo que cada red tarda en procesar una imagen.

La diferencia en los resultados mostrados en la tabla 2 revela que el perceptrón consigue un rendimiento muy superior a las redes de aprendizaje profundo. Sin embargo, su índice de error cercano al 99 % obliga a descartar la red frente a las otras dos. Por otro lado, Xception no solo consigue mayor índice de acierto que ResNet, si no también un mayor rendimiento. Finalmente, tras los resultados obtenidos, el modelo generado con la arquitectura Xception es el que se integró en la herramienta para el reconocimiento de los símbolos.

	Media de tiempo (segundos)
Xception	3,73
ResNet	5,07
MLPerceptron	**0,02**

Tabla 2: Resultados de las pruebas de rendimiento

8.3. Prueba del símbolo añadido a partir de una imagen

En la sección 7 se destacaba la posibilidad de añadir al conjunto de datos un nuevo símbolo generando las imágenes automáticamente a partir de una sola imagen original, llevando a la hipótesis de poder capacitar al sistema con modelos de aprendizaje personalizados para cada usuario sin necesidad de que tenga que aportar un número excesivo de imágenes.

Tras dar con el mejor modelo, el cual es el que se encuentra integrado en la herramienta, se realizó una prueba con 31 imágenes sin etiquetar del símbolo. El resultado fue una precisión de reconocimiento con un **margen de error del 45 %**. Aunque el resultado no es admisible, aporta el suficiente aliciente como para mejorar el método y que permita incluir los modelos personalizados. En este trabajo tan solo se ha probado una vez este método utilizando la distorsión aleatoria y el giro horizontal. Además, el conjunto de datos contiene símbolos de diferentes

usuarios, y con este método se plantea la posibilidad de crear conjuntos de datos personales de cada intérprete. Por tanto, para llegar a una conclusión válida se requiere de un conjunto de pruebas mayor en el que se contemplen las siguientes posibilidades:

1. Generar nuevas imágenes para el conjunto de datos mediante la técnica de distorsión aleatoria.
2. Integrar otras técnicas de generación automática de imágenes además de las ya aplicadas.
3. Solicitar al usuario más imágenes del símbolo para obtener más posibles datos desde los que partir en la generación automática de imágenes.
4. En este caso los símbolos corresponden a varios usuarios, por lo que habría que comprobar los resultados de un conjunto de símbolos perteneciente a un solo usuario.

Si tras estas pruebas se obtienen resultados satisfactorios se eliminaría el problema del conflicto de presentación de imágenes, ya que serían los símbolos personales de un solo intérprete. Y por otro lado, tampoco sería un problema la diferencia de estilo de trazado de cada símbolo entre diferentes personas, ya que cada modelo sería utilizado tan solo por el usuario al que pertenece.

9. Conclusiones y líneas futuras

Los avances de los últimos años en inteligencia artificial han permitido crear herramientas de gran utilidad que facilitan el trabajo de profesionales en muchísimos campos, tanto automatizando tareas como mejorando la calidad del trabajo. A pesar de ello, el ámbito de la interpretación ha resultado ser uno de los sectores que se quedan atrás. Esto denota una necesidad en la innovación del trabajo de los intérpretes debido a la existencia de una laguna tecnológica que aún debe cubrirse.

El presente trabajo trata de producir un aporte para solucionar este problema aplicando métodos que automaticen o semiautomaticen procesos que giran entorno a las notas que los intérpretes toman durante su trabajo. El trabajo desarrollado ha dado lugar a varios descubrimientos y aportes entorno a la herramienta creada, que resulta ser más una prueba de concepto que una herramienta funcional para los intérpretes. Sin embargo ha permitido realizar un estudio base del cual extraer varias conclusiones.

En la primera fase se integraron métodos de comparación de imágenes en los que se utilizaron imágenes de referencia para dichas comparaciones. Estos métodos han resultado ser inviables debido principalmente a los problemas de rendimiento que acarrean. Por otro lado, para poder resolver el problema de asociar diferentes símbolos aun mismo concepto, sería necesario aumentar la cantidad de imágenes de referencia; lo cual supone una mayor carga de trabajo.

El conjunto de datos creado para el entrenamiento de las redes neuronales ha resultado ser efectivo gracias a la generación automática de datos explicada en la sección 7. Siendo Xception la red neuronal que ha ofrecido mejores resultados, las pruebas realizadas antes de la generación de imágenes mostraban índices de error superiores al 50 %. Tras obtener las imágenes autogeneradas, el error disminuyó por debajo del 5 %. A su vez, el conjunto de datos sirve de punto de partida al que añadir nuevos símbolos y aumentar su utilidad. En el futuro se planteará publicar este conjunto de datos junto a los resultados obtenidos mediante las técnicas de generación automática.

De los resultados de las pruebas mostrados en la sección 8 puede destacarse la alta efectividad del *deep learning* frente a otros métodos en problemas de clasificación de imágenes, y más concretamente, para resolver el problema que se ha planteado en este trabajo.

A su vez, con las conclusiones obtenidas pueden establecerse líneas de trabajo futuras que permitan extender el trabajo desarrollado. En primer lugar, pueden realizarse pruebas más

extensas con intérpretes profesionales que valoren diferentes puntos: los símbolos incluidos actualmente, el comportamiento del sistema, las posibilidades de ampliación, etc. Para apoyar dicho estudio, se ha planteado difundir una encuesta de pruebas y evaluación de la herramienta. En el anexo C se muestra una versión inicial de las preguntas que pueden formar parte de dicha encuesta, que no ha podido desarrollarse y difundirse por limitaciones de tiempo. Tras este estudio, se pueden extraer posibilidades de desarrollo de herramientas más avanzadas y aplicables a problemas reales, que incluyan integraciones con otras posibilidades, como el reconocimiento de voz.

Por otro lado, se puede realizar un estudio más amplio para la creación de modelos de aprendizaje personales para cada intérprete. En este sentido se pueden valorar técnicas alternativas de generación de datos, u otras técnicas que aún no se hayan probado como el *transfered learning*. En esta misma línea, pueden explorarse otras posibilidades de mejora del sistema que ofrezcan mejores resultados que los encontrados en el desarrollo del trabajo.

Por último, el conjunto de datos creado puede extenderse aumentando la cantidad de datos para mejorar los resultados de los entrenamientos. También pueden explorarse usos alternativos del mismo, por ejemplo, entre las posibilidades de los modelos personales podría integrarse un sistema que sugiera al intérprete nuevos símbolos que añadir a su colección.

Agradecimientos

El presente TFM se enmarca en el seno del proyecto 'VIP: sistema integrado Voz-texto para IntérPretes' (ref. FFI2016-75831-P, 2017-2020. MINECO) y parcialmente en el proyecto 'TERMITUR: Diccionario inteligente TERMInológico para el sector TURístico (alemán-inglés-español)' (ref. HUM2754, 2014-2019. Junta de Andalucía).

Bibliografía

Annalisa, S. (2015). Becoming an interpreter: the role of computer technology. *MonTI. Monografías de Traducción e Interpretación*, pages 111–138.

Bay, H., Ess, A., Tuytelaars, T., and Van Gool, L. (2008). Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359.

Bloice, M. D., Roth, P. M., and Holzinger, A. (2019). Biomedical image augmentation using augmentor. *Bioinformatics*.

Bowker, L. and Corpas Pastor, G. (2015). Translation technology. In *The Oxford Handbook of Computational Linguistics 2nd edition*.

Chen, S. et al. (2016). Note taking in consecutive interpreting: A review with special focus on chinese and english literature. *The Journal of Specialised Translation*, 26(1):151–171.

Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258.

Chollet, F. et al. (2015). Keras. `https://keras.io`.

Corpas Pastor, G. (2018). Tools for interpreters: the challenges that lie ahead.

Costa, H., Corpas Pastor, G., and Durán Muñoz, I. (2014). Technology-assisted interpreting. *MultiLingual*, 143(25):3.

Courty, N., Flamary, R., and Ducoffe, M. (2017). Learning wasserstein embeddings. *arXiv preprint arXiv:1710.07457*.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.

Deng, L. (2012). The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, 29(6):141–142.

El Kessab, B., Daoui, C., Bouikhalene, B., Fakir, M., and Moro, K. (2013). Extraction method of handwritten digit recognition tested on the mnist database. *International Journal of Advanced Science & Technology*, 50:99–110.

Fantinuoli, C. (2017). Speech recognition in the interpreter workstation. *Proceedings of the Translating and the Computer*, 39.

Gillies, A. (2017). *Note-taking for consecutive interpreting: A short course*. Routledge.

Goldsmith, J. (2018). Tablet interpreting. *Translation and Interpreting Studies. The Journal of the American Translation and Interpreting Studies Association*, 13(3):342–365.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Huang, J., Zhang, R., Buyya, R., and Chen, J. (2014). Melody-join: Efficient earth mover's distance similarity joins using mapreduce. In *2014 IEEE 30th International Conference on Data Engineering*, pages 808–819. IEEE.

Karami, E., Prasad, S., and Shehata, M. (2017). Image matching using sift, surf, brief and orb: performance comparison for distorted images. *arXiv preprint arXiv:1710.02726*.

Keysers, D. (2007). Comparison and combination of state-of-the-art techniques for handwritten character recognition: topping the mnist benchmark. *arXiv preprint arXiv:0710.2231*.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110.

Lung, R. (1999). Note-taking skills and comprehension in consecutive interpretation. *Babel*, 45(4):311–317.

Marti, U.-V. and Bunke, H. (2002). The iam-database: an english sentence database for offline handwriting recognition. *International Journal on Document Analysis and Recognition*, 5(1):39–46.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Perez, L. and Wang, J. (2017). The effectiveness of data augmentation in image classification using deep learning. *arXiv preprint arXiv:1712.04621*.

Rozan, J.-F. (1956). 1973. *La Prise de notes en interprétation consecutive*.

Rublee, E., Rabaud, V., Konolige, K., and Bradski, G. R. (2011). Orb: An efficient alternative to sift or surf. In *ICCV*, volume 11, page 2. Citeseer.

Sandrelli, A. (2005). Designing cait (computer-assisted interpreter training) tools: Black box. In *MuTra–Challenges of Multidimensional Translation, Saarbrücken 2-6 May 2005. Conference Proceedings–EU High Level Scientific Conference Series. Proceedings of the Marie Curie Euroconferences*.

Singh, S. (2013). Optical character recognition techniques: a survey. *Journal of emerging Trends in Computing and information Sciences*, 4(6):545–550.

Taqdir and Dhir, R. (2017). Face recognition using sift key with optimal features selection model. *INTERNATIONAL JOURNAL OF ADVANCED COMPUTER SCIENCE AND APPLICATIONS*, 8(2):403–409.

Torre Salceda, M. (2017). La aplicación de herramientas tecnológicas a la interpretación consecutiva. cleopatra: la aplicación para el entrenamiento en automatización de símbolos.

Ulloa, I. J. F. and Navarrete, M. T. H. (2014). La importancia de las técnicas de toma de notas para los estudiantes de interpretación. *Saber, ciencia y libertad*, 9(2):205–216.

Van der Walt, S., Schönberger, J. L., Nunez-Iglesias, J., Boulogne, F., Warner, J. D., Yager, N., Gouillart, E., and Yu, T. (2014). scikit-image: image processing in python. *PeerJ*, 2:e453.

Wang, J., Fan, G., and Wang, Z. (2010). A simple cw-ssim kernel-based nearest neighbor method for handwritten digit classification. *arXiv preprint arXiv:1008.3951*.

Zhang, T., Zheng, W., Cui, Z., Zong, Y., Yan, J., and Yan, K. (2016). A deep neural network-driven feature learning method for multi-view facial expression recognition. *IEEE Transactions on Multimedia*, 18(12):2528–2536.

A. Guía de la Herramienta

A fin de facilitar el uso de la aplicación desarrollada, se ha redactado una guía rápida de uso de la misma que sirva de apoyo al usuario en caso de dudas. En esta guía se muestran todas las interfaces y componentes que forman la aplicación, que pueden verse en las capturas. También se incluyen en esta guía la lista de términos aceptados, un apartado de consideraciones para el usuario, y el contacto del autor para poder consultar aspectos ajenos a esta guía.

A.1. Acceso a la aplicación

Se trata de una aplicación web simple que abarca una sola funcionalidad. No requiere instalación ni descargas y puede accederse a ella desde la siguiente dirección: `www.javilimatfm.ddns.net`

A.2. Interfaz inicial

Dado que solo abarca una función, la aplicación web no posee ningún tipo de menú, ni accesos; esta formada por una sola interfaz que va cambiando según las acciones que se realicen. Inicialmente, la aplicación está compuesta con un área de dibujo, un botón "Clear Area" y un segundo botón "Process Symbol". En la figura 16 podemos ver la interfaz con los elementos mencionados. A continuación se dan detalles de cada uno de ellos:

1. **Área de dibujo:** Aquí se pueden dibujar los símbolos para que la red neuronal artificial los reconozca y devuelva la representación del mismo. Por cuestiones de homogeneidad, esta área está configurada para mantener el mismo color y grueso del trazo para todos los símbolos.
2. **Botón "Clear Area":** Este botón sirve para dejar en blanco de nuevo el área de dibujo en caso de cometer un error al intentar dibujar un símbolo.
3. **Botón "Process Symbol":** Este botón inicia el proceso de reconocimiento del símbolo enviando la imagen del mismo a la red neuronal. El proceso posterior se explica en las siguientes secciones.

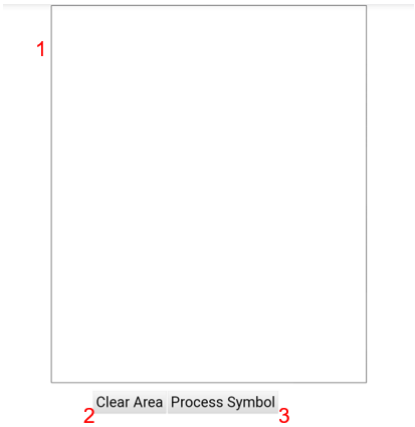

Figura 16: Interfaz inicial

A.3. Procesado de los símbolos

Cuando el botón “Process Symbol” es pulsado se inicia el proceso de reconocimiento de la imagen. Tras unos segundos en los que se muestra el mensaje de carga (Figura 17), aparece en la pantalla el concepto que el sistema ha determinado que representa el símbolo procesado (Figura 18).

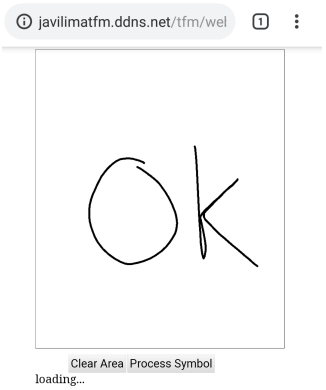

Figura 17: Procesado de los símbolos

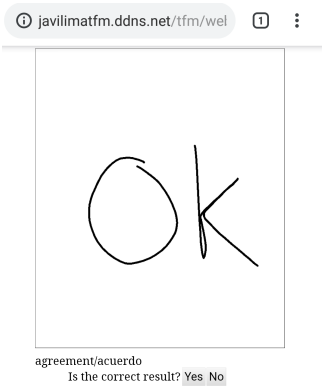

Figura 18: Resultado del reconocimiento

Una vez mostrado el resultado, se pregunta si el resultado es correcto o no. El sistema propone 3 posibilidades (incluyendo la primera) de forma que en caso de que no sea correcto, muestra la siguiente. En caso de que ninguna de las 3 posibilidades sea la correcta, la aplicación solicita que se le proporcione cual es la respuesta correcta.

En la figura 19 se muestra como la aplicación solicita el término correcto correspondiente al símbolo. Para ello, debe seleccionarse la opción correcta pulsando en el selector y posteriormente pulsar el botón “Ok”. En el selector aparece la lista de símbolos que la red neuronal del sistema es capaz de reconocer. Una vez pulsado “Ok”, el sistema almacena la imagen del símbolo asociada a su correspondiente término y la aplicación vuelve a su interfaz inicial.

A.4. Lista de términos que el sistema es capaz de reconocer

En este apartado se listan los términos admitidos en el sistema. Cualquier símbolo correspondiente a un término no incluido en esta lista no será reconocido correctamente, si no que el sistema intentará encuadrarlo en uno de los conceptos de la lista.

1. Agreement/acuerdo
2. Agriculture/agricultura
3. Before/antes
4. Consequences/consecuencias
5. Country/país
6. Déficit/déficit
7. Democracy/democracia
8. Development/desarrollo
9. Energy/energía
10. Environment/entorno
11. Impact/impacto
12. Industry/industria
13. Inflation/inflación
14. Meeting/encuentro
15. Money/dinero
16. Politics/política
17. Problem/problema
18. Relations/relaciones
19. Repression/represión
20. Role/rol
21. Success/éxito
22. Surplus/excedente
23. Trade/comercio
24. Work/trabajo

A.5. Consideraciones que tener en cuenta

Con fines de recolección de datos, todo símbolo que se dibuje es almacenado en cuanto se comienza el procesamiento del mismo. Por este motivo se recomienda evitar dibujar cualquier

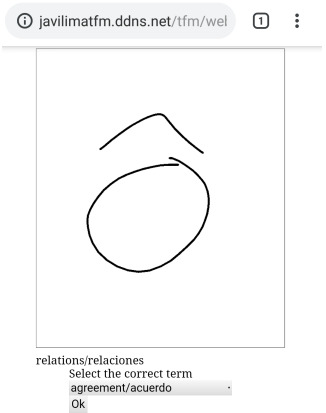

Figura 19: Selección del término correcto

cosa que no tenga relación con el propósito de la aplicación y se insta a no dibujar/escribir nada que pueda considerarse un dato sensible bajo ningún concepto.

La aplicación se encuentra en una fase prototipo o de prueba de concepto, por lo que no se trata de un sistema plenamente funcional. Es posible que surjan errores o comportamientos inesperados.

A.6. Contacto

Autor: Francisco Javier Lima Florido.
e-mail: `fco.javier.lima@uma.es`

B. Guía de instalación y uso

Sin entrar en profundidad a nivel técnico sobre todo el código desarrollado, en este apéndice se exponen los requisitos, consideraciones, pasos y toda la información necesaria para poder utilizar la herramienta en cualquier máquina.

B.1. Estructura y ficheros

La herramienta esta compuesta por varias carpetas y ficheros que podemos ver en la figura 20. En la carpeta *web* se encuentran los ficheros correspondientes a la aplicación web desarrollada en este trabajo. En ella se encuentran los archivos del cliente web y el *script* PHP para cambiar los nombres de los archivos. La carpeta *python* se estructura a su ven en varias carpetas y ficheros. Podemos encontrar dos carpetas llamadas *dataset* y *dataset_augmented* correspondientes al conjunto de datos de símbolos de usuarios y el conjunto de datos generado con *Augmentor*, respectivamente. Existe otra carpeta llamada *img* en la que se encuentran las imágenes de referencia utilizadas en la primera fase del trabajo. Entre los ficheros encontramos dos archivos *h5* y un archivo *joblib* que se corresponden a los modelos entrenados de las redes utilizadas en la herramienta y en las pruebas ejecutadas. En el *script measure_img_similarity* encontramos los métodos de comparación utilizados en la primera fase. El resto de archivos son los *scripts* utilizados para los entrenamientos, las predicciones y resto de utilidades del sistema.

B.2. Despliegue y acceso a la aplicación

Para poder desplegar y utilizar la aplicación web en cualquier equipo, basta con poner las carpetas *web* y *python* en una carpeta e instalar un servidor web que permita acceder en modo local. Para que todo funcione correctamente, es esencial que el servidor instalado permita ejecutar scripts CGI. Esto es necesario para que funcionen los scripts de lenguaje python.

Además de poder ser instalada de forma local, la aplicación web se encuentra instalada actualmente en un servidor web al que puede accederse desde internet[7] a través del siguiente enlace: `javilimatfm.ddns.net`

C. Encuesta sobre la aplicación

Una de las líneas de trabajo futuro previstas es la de realizar una evaluación en profundidad de la herramienta mediante una encuesta para intérpretes profesionales. La encuesta posee una doble finalidad: por un lado se busca extraer información sobre el estado actual de la herramienta y su utilidad, y por otro averiguar las posibilidades reales que pueda ofrecer desde el punto de vista de un intérprete. En este anexo aparecen divididas en secciones las preguntas de la encuesta diseñada a tal fin.

[7] El acceso a la aplicación puede verse afectado por motivos de actualizaciones o movimientos a otros alojamientos que puedan hacerse en el futuro

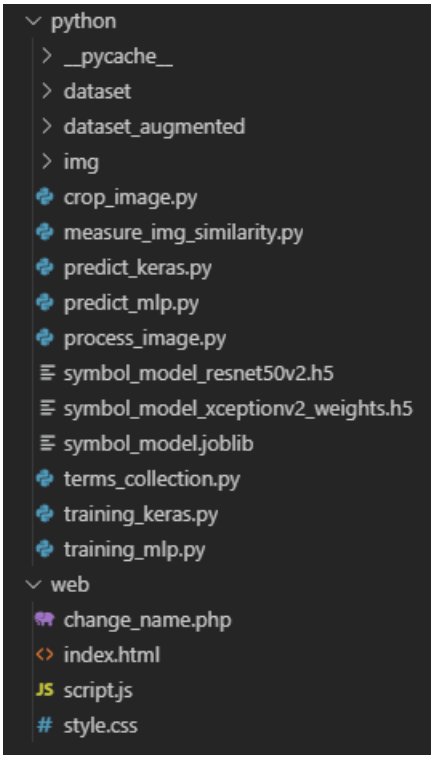

Figura 20: Carpetas y ficheros del proyecto

C.1. Pruebas de la herramienta

En esta sección se solicitan pruebas de los conceptos que la herramienta es capaz de reconocer. No es necesario probar cada uno de los símbolos, solo proporcionar la información que se pide para aquellos que se han probado.

Pregunta 1. Nº de veces que se ha probado cada símbolo (Marca solo una opción por fila).

Pregunta 2. Nº de veces que la herramienta ha reconocido correctamente cada símbolo en el primer intento (Marca solo una opción por fila).

Tanto la pregunta 1 como la pregunta 2 se responden sobre la tabla 3.

Pregunta 3. Tras las pruebas realizadas, ¿qué indice de acierto has observado o calculado? (Marcar una opción).

1. 0 %
2. <30 %
3. 30-50 %
4. 50-70 %
5. >70 %

C.2. Evaluación de la herramienta

Pregunta 4. ¿Te parecen útiles los símbolos que reconoce la herramienta actualmente? (Marcar una opción)

1. Sí, todos.
2. No todos, la mitad aproximadamente.
3. La mayoría no.

	1	2	3	4	5	6	7	8
Agreement								
Agriculture								
Before								
Consequences								
Country								
Deficit								
Democracy								
Development								
Energy								
Environment								
Impact								
Industry								
Inflation								
Meeting								
Money								
Politics								
Problems								
Relations								
Repression								
Role								
Success								
Surplus								
Trade								
Work								

Tabla 3: Tabla de las preguntas 1 y 2

4. No, ninguno.

Pregunta 5. ¿Qué símbolos quitarías? (Marcar uno o varios)

1. Agreement/acuerdo
2. Agriculture/agricultura
3. Before/antes
4. Consequences/consecuencias
5. Country/país
6. Déficit/déficit
7. Democracy/democracia
8. Development/desarrollo
9. Energy/energía
10. Environment/entorno
11. Impact/impacto
12. Industry/industria
13. Inflation/inflación
14. Meeting/encuentro
15. Money/dinero
16. Politics/política
17. Problem/problema
18. Relations/relaciones
19. Repression/represión
20. Role/rol
21. Success/éxito

22. Surplus/excedente
23. Trade/comercio
24. Work/trabajo

Pregunta 6. ¿Qué símbolos añadirías? (Escribir los términos o conceptos)

Pregunta 7. ¿En qué aspecto del trabajo de interpretación ves más útil el uso de una herramienta que reconoce símbolos? (Marcar una opción)

1. En la transcripción a tiempo real durante la toma de notas.
2. Para pasar notas a lenguaje escrito.
3. En ejercicios para practicar.
4. Otro:

Pregunta 8. ¿Utilizarías una herramienta como esta si estuviese específicamente entrenada en tus propios símbolos? (Marcar una opción).

1. Sí.
2. Prefiero que ofrezca también otros símbolos.
3. No.

Pregunta 9. ¿Y si tuvieras que entrenarla proporcionando tus símbolos previamente? (Marcar una opción)

1. Sí, aunque el proceso requiera algo de tiempo
2. Solo si es sencillo de hacer
3. No

Pregunta 10. ¿Cómo usarías la herramienta y en que aspectos de tu trabajo piensas que te ahorraría tiempo?

Pregunta 11. ¿En que otras situaciones del trabajo de los intérpretes aplicarías la Inteligencia Artificial para automatizar tareas y ahorrar tiempo?

Índice de figuras

Índice de tablas