Generación sintética de secuencias temporales a través de redes neuronales profundas

Paula Delgado de Santos

Máster en Ingeniería de Telecomunicación

MÁSTERES DE LA UAM

2019 - 2020

Escuela Politécnica Superior

UAM Universidad Autónoma de Madrid

UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

TRABAJO FIN DE MÁSTER

Generación Sintética de Secuencias Temporales a través de Redes Neuronales Profundas

Máster Universitario en Ingeniería de Telecomunicación

Autora: Paula Delgado de Santos
Tutor: Rubén Tolosana Moranchel
Co-Tutor: Andrés Pérez Uribe

FECHA: JUNIO 2020

GENERACIÓN SINTÉTICA DE SECUENCIAS TEMPORALES A TRAVÉS DE REDES NEURONALES PROFUNDAS

Autora: Paula Delgado de Santos

Tutor: Rubén Tolosana Moranchel
Co-Tutor: Andrés Pérez Uribe
Ponente: Julián Fiérrez Aguilar

BiDA Lab

Biometrics and Data Pattern Analytics Lab

Departamento de Tecnología Electrónica y de las Comunicaciones

JUNIO 2020

Resumen

Resumen

Gracias a los algoritmos de Aprendinzaje Automático, como son las redes neuronales profundas, se ha producido un gran avance en el estudio de secuencias temporales. Dentro de la biometría, uno de los campos más importantes para verificar la identidad de un usuario es su firma y escritura manuscrita. Estos rasgos son difíciles de estudiar debido a la escasez de sus datos y variabilidad. Un individuo nunca realiza su firma dos veces de igual manera, aun así existen diferentes características que permiten distinguir a cada individuo. Uno de los grandes inconvenientes que muestra este ámbito, especialmente la firma manuscrita, es la escasez de datos que existe debido a los problemas legales y al elevado coste del proceso de captura de los mismos.

En este trabajo se propone el estudio y desarrollo de sistemas para la generación sintética de firma y escritura dinámica, con el objetivo de solventar la escasez de datos en este campo. El método propuesto está basado en la última tecnología disponible en el ámbito de las redes neuronales profundas y consta de dos módulos principales: el módulo de segmentación a nivel de trazos y el módulo de síntesis. En primer lugar, el módulo de segmentación a nivel de trazos particiona las secuencias temporales de la firma y escritura en tramos de menor número de muestras, adecuando así los datos para el siguiente módulo. En segundo lugar, el módulo de síntesis consta de un *Variational Autoencoder (VAE)* que realiza la tarea de síntesis de firma y escritura manuscrita *on-line*. Se trata de un sistema entrenado a nivel de trazos. Una de las mayores ventajas que presenta el modelo es su impersonalidad, es decir, el modelo entrenado es adaptable para cualquier usuario y cualquier situación de firma y escritura manuscrita *on-line*. A su vez, al ser entrenado a nivel de trazos permite la síntesis de nuevas muestras con un gran nivel de detalle y variabilidad.

El modelo de síntesis ha sido probado en dos escenarios de escritura manuscrita: contraseñas y firmas *on-line* obteniendo muy buenos resultados tanto a nivel visual como de funciones temporales. Por último, se ha demostrado la gran utilidad del método propuesto en escenarios con escasez de datos, mejorando las tasas de rendimiento de los sistemas automáticos de firma manuscrita *on-line*.

Palabras Clave

Escritura, Firma, Biometría, Síntesis, Secuencias Temporales, *Variational Autoencoder*, *Autoencoder*

Abstract

Machine Learning algorithms, suck as Neural Networks, have achieved great results in the study of temporal sequences. Within biometrics, one of the most important research lines to verify the identity of a user is his/her signature and handwritting. These traits are difficult to study due to their data' paucity and variability. An individual never makes the same signature twice, nonetheless there are many characteristics that allow us to distinguish each individual. One of the key disadvantages in this area, especially in the handwritten signature, is the scarcity of data, due to legal problems and also the high cost existed in the capturing process.

The study and development of systems for the generation of synthetic dynamic signature and writing is proposed in this M.Sc. Thesis with the aim of solving the scarcity of data in this field.

The proposed method is based on the latest technology available in the field of deep neural networks. It consists of two main modules: the stroke-level segmentation module and the synthesis module. First, the Stroke-Segmentation Module partitions the temporal sequences in sections with fewer samples, adapting the data for the next module. Then, the synthesis module consists of a Variatonal Autoencoder that performs the synthesis of on-line signature and handwriting.

One of the key advantages of the proposed approach impersonality, that is to say, the trained model is adaptable for any user and any on-line signature and handwriting situation. At the same time, since this module is trained at stroke level, it allows the synthesis of new samples with a higher level of detail and variability.

The proposed synthesis approach has been tested in two handwriting scenarios: on-line passwords and signatures, in which it has achieved good visual and temporal results. Finally, we have performed some experiments using our proposed synthesis approach on few-shot scenarios, generating synthetic signatures and using them to train more robust on-line signature verification systems. Our proposed approach has proven to be a good method under few-shot scenarios.

Key words

Handwriting, Signature, Biometrics, Synthesis, Deep Learning, Variational Autoencoder, Autoencoder

Agradecimientos

Me gustaría agradecer este trabajo a mi tutor, Rubén Tolosana, por ayudarme y estar siempre pendiente. Aconsejarme y guiarme en todo, consiguiendo ser un tutor inmejorable.

También a mi co-tutor, Andrés Pérez-Uribe, por la ayuda prestada durante mi estancia en Suiza.

A mi familia, por apoyarme en todas mis decisiones y aguantar mis momentos difíciles.

Por último, a todos aquellos amigos que nunca han dejado de confiar en mi, que comparten mis penas y alegrías. Mi gente de teleco, Segovia, Ayllón... ¡Gracias! Una verdadera amistad es aquella en la que no importa el tiempo ni la distancia, si no la grandeza de los momentos compartidos.

Índice general

Índice de Figuras

Índice de Tablas

1 Introducción

1.1. Motivación

La disponibilidad de grandes volúmenes de datos resulta crucial para el correcto entrenamiento de los sistemas automáticos, especialmente tras los recientes avances producidos en el ámbito de las redes neuronales profundas. Una forma de mejorar el rendimiento de dichos sistemas es por medio del incremento de la cantidad y variabilidad de los datos de los que se dispone, mediante técnicas de *Data Augmentation* [9]. Dichas técnicas son ampliamente utilizadas hoy en día en el mundo de la imagen junto con el uso de *Convolutional Neural Networks (CNNs)*. Sin embargo, pocos avances se han producido en los últimos años cuando los datos a utilizar son secuencias temporales, no imágenes. [5].

Tradicionalmente, la generación sintética de secuencias temporales se ha estado realizando a partir de técnicas propias de imagen [10], de secuencias temporales como *Dynamic Time Warping (DTW)* y *k-Nearest Neighbors (k-NN)* [11] [12]. Sin embargo, pocas mejoras se han propuesto dentro del ámbito de las redes neuronales profundas [13] [14]. En concreto, en [13] se propuso un sistema denominado *Multi-Channels Deep Convolution Neural Networks (MC-DCNN)*, donde cada canal corresponde a una única dimensión de series temporales multivariadas como entrada y aprende las características individualmente. En [14], se propuso *Multi-scale Convolutional Neural Network (MCNN)*, que incorpora la extracción de características y clasificación en una sola etapa.

En este Trabajo de Fin de Máster (TFM) se pretende avanzar en esta línea de investigación por medio del uso y análisis de técnicas de redes neuronales profundas en el estado del arte.

Los resultados obtenidos en este Trabajo Final de Máster han generado un artículo de investigación que será enviado al congreso internacional *35th AAAI Conference on Artificial Intelligence, 2021.*

1.2. Objetivos

Con el objetivo de generar nuevas secuencias temporales sintéticas, se pretende analizar en detalle las siguientes técnicas en el estado del arte: *Autoencoder (AE)* y *Variational Autoencoder (VAE)*. Ambos sistemas se componen de una red neuronal recurrente bidireccional en la parte

del encoder (transformación de las secuencias temporales al espacio de características) y una red neuronal recurrente autorregresiva en la parte del decoder (transformación del espacio de características a nuevas secuencias temporales sintéticas) [2].

En primer lugar, las distintas técnicas propuestas serán estudiadas haciendo uso del código libre proporcionado por Google dentro del proyecto denominado *Quick Draw* [1]. En dicho proyecto, los investigadores implementaron y entrenaron un *VAE* con la finalidad de generar nuevos dibujos (en forma de secuencias temporales). La primera tarea a realizar consistirá en replicar el correcto funcionamiento de dicho sistema (en código Python con el entorno Magenta), entrenando y evaluando su correcto funcionamiento con la base de datos masiva también disponible por Google (Quick Draw).

Tras esta primera etapa, se adaptará el sistema inicial a otras líneas de investigación. En concreto, se trabajará en la generación sintética de firma y escritura manuscrita a través del uso de bases de datos disponibles por el grupo (*MobileTouchDB* [4] y *DeepSignDB* [15]). Se observará a nivel de imagen y secuencia temporal cómo afectan los distintos parámetros del *VAE* en la generación de nuevas muestras, tanto durante el proceso de entrenamiento (variando los parámetros de la función de coste), como de su posterior generación. Posteriormente, se realizará una evaluación del sistema de síntesis propuesto en escenarios con escasez de datos. En concreto, se va a emplear la base de datos *DeepSignDB* [4] evaluando el *Equal Error Rate (EER)* en un escenario con firmas originales en comparación con escenarios que utilizan tanto firmas originales como firmas generadas sintéticamente con el método propuesto en este trabajo.

Por último, se remarcará el impacto y futuras aplicaciones de la tecnología desarrollada en este TFM dentro del ámbito de la firma y la escritura manuscrita.

1.3. Metodología y Plan de Trabajo

Se ha ideado un plan de trabajo para poder cumplir con los objetivos propuestos. Para ello se han ido cumpliendo por orden las siguientes fases:

- Investigación de los métodos propuestos en el Estado del Arte sobre los sistemas de aprendizaje automático en escenarios con escasez de datos.
- Investigación más a fondo sobre métodos de *Data Augmentation* para secuencias temporales.
- Estudio y puesta en práctica de la técnica *SketchRNN* [1] con el propósito de utilizarla como una técnica de *Data Augmentation*.
- Análisis de las diferentes bases de datos a utilizar.
- Adecuación del sistema propuesto a las diferentes bases de datos a utilizar, consiguiendo universalizar el sistema lo máximo posible. Estudio a nivel de imagen y secuencia temporal tanto las muestras originales como las muestras generadas sintéticamente observando cómo afectan los distintos parámetros del sistema propuesto.
- Evaluación del sistema de síntesis propuesto en escenarios con escasez de datos.
- Análisis de los resultados obtenidos.
- Redacción de este documento y preparación de la defensa.

1.4. Estructura de la Memoria

- *Capítulo 1*: Plantea la motivación, los objetivos y la estructura de este trabajo.
- *Capítulo 2*: Resume el estado del arte. Ofrece una introducción a la generación sintética de secuencias temporales así como un resumen de artículos de interés relacionados con este TFM.
- *Capítulo 3*: Describe las bases de datos utilizadas en este TFM.
- *Capítulo 4*: Describe los métodos propuestos para la realización de este TFM.
- *Capítulo 5*: Detalla los experimentos llevados a cabo, así como los resultados obtenidos en cada uno de ellos.
- *Capítulo 6*: Expone las conclusiones obtenidas y el trabajo futuro.

2

Estado del Arte

En este capítulo se describe el estado del arte en el ámbito de la generación sintética de secuencias temporales. En primer lugar, se estudian los sistemas de aprendizaje automático en escenarios con escasez de datos mostrando una pequeña introducción, retos y aproximaciones actuales dentro de la sección 2.1. Posteriormente se presenta la técnica *Data Augmentation* y su aplicación a secuencias temporales en la sección 2.2. Por último, en la sección 2.3, se realiza una introducción a los sistemas biométricos centrándose en los campos de aplicación abordados en este estudio, la firma y la escritura biométrica.

2.1. Sistemas de Aprendizaje Automático en Escenarios con Escasez de Datos

2.1.1. Introducción

El Aprendizaje Automático o *Machine Learning (ML)* hace alusión a la rama de la Inteligencia Artificial que tiene como objetivo desarrollar técnicas que permitan a las computadoras aprender. Esta tarea se realiza directamente de los datos observados, sin predeterminar las relaciones mecanicistas que gobiernan el sistema. Los algoritmos de *ML* pueden mejorar de manera adaptativa su rendimiento con cada nueva muestra de datos y descubrir patrones en datos complejos y de alta dimensión. Este tipo de aprendizaje se ha convertido en la tecnología central para numerosas aplicaciones del mundo real. Estas labores van desde el pronóstico del tiempo, el estudio de secuencias de ADN, los motores de búsqueda en internet, el reconocimiento de imágenes, la clasificación de texto binario, el reconocimiento de objetos hasta el reconocimiento de rasgos biométricos [16, 17, 18, 19]. Dentro de los modelos de *ML* se encuentran los de *Deep Learning (DL)* los cuales requieren de grandes volúmenes de datos para su entrenamiento. Por ello, estos modelos de *DL* rara vez se ven en el contexto de pequeños conjuntos de datos, donde un número insuficiente de ejemplos de entrenamiento puede comprometer el éxito del aprendizaje [20, 21]. Esto es debido a que los modelos se adaptan en exceso a la información aportada [22]. Algunas de las áreas donde apenas se encuentra información puede ser la medicina, con la predicción de trasplante de órganos debido a la gravedad de las enfermedades y la complejidad de las operaciones clínicas [23], ciencia de los materiales [24], reconocimiento facial de infrarrojo cercano [22] y reconocimiento de firma [25] entre otros. *ML* enfrenta los desafíos de

la volatilidad de los resultados entre modelos del mismo diseño debido a los datos insuficientes. Hay algoritmos como *Decision Trees (DT)*, *Random Forest (RF)* o redes neuronales que muestran una variación significativa en el rendimiento debido a la cantidad de datos con los que se entrena [26]. Los resultados varían mucho debido a que los sistemas de aprendizaje automático han observado menos variabilidad de la existente en todo el estudio de población durante la etapa de entrenamiento. De ahí que en la etapa de evaluación, cuando tiene ejemplos nuevos, no generaliza bien.

2.1.2. Retos

Los sistemas automáticos basados en el uso de volúmenes de datos reducidos han sido ampliamente estudiados durante décadas. Aun así, hoy en día sigue siendo un campo muy complejo incluso con el avance de la minería de datos y el *ML*. Algunos de los factores que contribuyen a su dificultad son:

- Los diferentes conjuntos de datos aplicados a las diversas áreas pueden requerir la representación de características en distintas escalas de tiempo. Difícilmente la información se adapta a las escalas correctas.
- Los datos poseen distorsiones por medio de ruidos aleatorios, como son las variaciones casuales del brillo o el color en las imágenes digitales, perturbaciones de alta frecuencia en los sonidos, o la precisión del bolígrafo en la escritura. Por ello, suelen necesitar procesamientos automáticos de suavizado de la señal, procurando ser poco sensibles a los *outliers*.
- Estos algoritmos de *ML* y sobretodo de *DL* como son las redes neuronales profundas, requieren de grandes volúmenes de datos para su entrenamiento. Aun así, para muchas aplicaciones, como *passwords* manuscritos, firmas o detección de animales solo existen pequeños *datasets*. En estos casos, los sistemas son incapaces de aprender la variabilidad existente en los datos, proporcionando altas tasas de error en su posterior evaluación con nuevos datos [27].

2.1.3. Aproximaciones Actuales

El procedimiento para solventar la falta de datos ha sido resuelto desde tres perspectivas, la primera visión es a nivel de sistemas, promoviendo el desarrollo de algoritmos de extracción de las características discriminatorias. Tienen que estudiar el problema incluso con un conjunto de datos extremadamente limitado al tiempo que adoptan el aprendizaje de transferencia como es el ***Few/Low-Shot Learning*** [28]. La segunda perspectiva trata del ***Transfer Learning*** donde se transmite información de un dominio a otro relacionado [29]. Por último, la tercera perspectiva se basa en la generación sintética de nuevos datos con técnicas como ***Data Augmentation*** [27].

- Por un lado, una de las técnicas que se está llevando a cabo es el ***Low-Shot Learning*** o ***Few-Shot Learning***. Con ella, se optimizan los sistemas con funciones de coste adaptadas a la escasez de datos. Los algoritmos se preparan a partir de un pequeño *dataset* con datos de entrenamiento y test que representan a un gran universo de datos etiquetados. A su vez, existen las técnicas de *Zero-Shot Learning* donde solamente se tiene una muestra por clase en forma de descriptor de clase. *Low-Shot Learning* y *Zero-Shot Learning* están siendo estudiados activamente para abordar este problema imitando el proceso del cerebro humano, y, por lo tanto, más cerca de los escenarios de aplicaciones del mundo real [28, 30].

- ***Transfer Learning*** es un método por el cual se crean técnicas de alto rendimiento entrenadas con gran cantidad de datos obtenidos de diferentes dominios. Tras ello, estas técnicas ya entrenadas se adaptan al mundo de la escasez de datos. En numerosos estudios, el proceso de adaptación del dominio se centra en corregir las diferencias de distribución marginal o las diferencias de distribución condicional entre los dominios del origen y el destino [6, 29].
- ***Data Augmentation*** consiste en la generación de datos sintéticos utilizando un método específico, consiguiendo reducir los errores del clasificador gracias a un mayor volumen de datos. El aumento de dichos datos también se utiliza para minimizar el desequilibrio existente en algunas clases en el momento del entrenamiento de sistemas de aprendizaje automático, es decir, cuando las diferentes clases que se utilizan a la hora de entrenar un algoritmo no tienen un número igual o parecido de ejemplos [27, 31].

2.1.3.1. Few-Shot y Low-Shot Learning

El ser humano es capaz de aprender a identificar nuevas categorías a partir de unos pocos ejemplos, incluso de uno solo y a veces hasta puede entender perceptivamente sin aprendizaje. Por ejemplo, los humanos pueden reconocer la cara de una persona a raíz de simplemente haberla observado en una fotografía. Las técnicas del *Few-Shot Learning* o *Zero-Shot Learning* se basan en ello teniendo como objetivo producir modelos que puedan generalizar a partir de pequeñas cantidades de datos etiquetados [28, 32].

Zero-Shot Learning consiste en aprender una clase de los datos bajo estudio a partir de un solo ejemplo etiquetado. La principal diferencia con el *Few-Shot Learning* es que con un solo ejemplo por clase se describe la categoría, y el algoritmo aprende a comparar esta única entrada con los descriptores de las demás categorías. En *Zero-Shot Learning* se aporta un solo dato para definir cada clase a reconocer. Por ello, en lugar de recibir un conjunto de características con una única entrada para cada una de las clases de entrenamiento, contiene un vector de características de clase semántica para cada una [33].

El *Few-Shot Learning* y el *Zero-Shot Learning* se han vuelto esenciales para crear modelos que pluralicen a partir de pocos o solo un ejemplo por clase. Se puede dividir estas técnicas en dos tipos de enfoques, generalistas y transductivos. Por un lado, los métodos generalistas son aquellos que abordan este problema creando un clasificador genérico en una gran cantidad de tareas multiclase y adaptan el modelo a una nueva tarea. Por otro lado, los métodos transductivos realizan una inferencia transductiva que clasifica todo el conjunto de pruebas a la vez.

Uno de los enfoques con más fuerza en este ámbito, el cual se utiliza tanto en enfoques generalistas como transductivos es el *Meta-Learning*. Este método entrena a funciones parametrizadas o *Meta-Learners*. Estas funciones se testean mediante el muestreo de pequeños conjuntos de entrenamiento y prueba pertenecientes un gran universo de ejemplos etiquetados. El *Meta-Learning* enmarca el *Few-Shot Learning* como un problema de optimización. Su meta es entrenar un modelo en una variedad de tareas de aprendizaje, de modo que se pueda resolver utilizando solo un pequeño número de ejemplos de captación [30, 7].

Dentro de los **enfoques generalistas** podemos encontrar tres métodos diferentes: métodos que aprenden la métrica, métodos con redes de memoria y métodos basados en descenso por gradiente.

- **Métodos que aprenden la métrica**: estudian un espacio de similitud donde aprender es eficiente para pocos ejemplos. Se centran en el aprendizaje de una inserción transferible. Algunas aproximaciones predefinen una métrica fija para obtener una buena representación y predecir clases calculando la distancia euclídea con respecto a los prototipos de clase [34]. Otras aproximaciones optan por aprender una métrica profunda transferible para comparar

la relación entre datos de distintas clases (*Few-Shot Learning*) o entre datos y descriptores de clases (*Zero-Shot Learning*) [35].

- **Métodos con redes de memoria**: aprenden a generalizar la "experiencia" partir de tareas previamente observadas. Tras ello, pluralizan las nuevas tareas a raíz de la información anterior. *TADAM (TAsk Dependent Adaptive Metric)* fue creada para realizar un entrenamiento de un conjunto de tareas auxiliares aprendiendo un espacio métrico dependiente de la tarea [36]. *SNAIL (Simple Neural AttentIve Learner)* se propuso como funciones parámetricas simples y genéricas que usan una combinación de circunvoluciones temporales. Se basaba en el *Meta-Learning*. Con ello, agregan la información de experiencias pasadas e identifican las piezas específicas de la información [37].

- **Métodos basados en descenso por gradiente**: entrenan funciones parametrizadas específicas que aprenden a inicializarse correctamente para nuevas tareas de aprendizaje. Ravi y Larochelle propusieron un enfoque de *Meta-LSTM*. Su encuadre implicaba la formación de un *Long Short-Term Memory (LSTM)* que aprendía a entrenar un modelo personalizado para cada episodio [38]. A su vez, *Model-Agnostic Meta-Learning (MAML)* fue creado con la particularidad de ser compatible con cualquier modelo entrenado con descenso por gradiente y aplicable a una gran variedad de problemas de aprendizaje. A esto se incluye la clasificación, la regresión y el aprendizaje por refuerzo. Los parámetros del modelo están entrenados explícitamente de modo que con pocos pasos de gradiente y con un pequeño *dataset* de entrenamiento de una nueva tarea, el algoritmo producirá un buen rendimiento en este reciente cometido [7].

En cuanto a los **enfoques transductivos** tienen como característica ser sistemas entrenados conjuntamente desde cero, donde se transfiere información entre los datos. *Reptile* fue creada con ese fin utilizando *Meta-Learning*. Esta técnica se basa en compartir información entre ejemplos de prueba mediante la normalización por lotes. Aun así, el algoritmo no realizaba una transferencia propiamente dicha entre ejemplos, si no que se realizaba por lotes [39]. Por ello, se creó *Transductive Propagation Network (TPN)*. Clasificaba todo el conjunto de pruebas a la vez para aliviar el problema de pocos datos. El algoritmo aprende a propagar etiquetas de instancias ya clasificadas a instancias de pruebas sin catalogar. De esta forma crea un módulo de construcción de gráficos que explota la estructura múltiple en los datos. *TPN* aprende conjuntamente los parámetros de una estructura de características y la construcción de gráficos de manera íntegra [8].

Matching Nets es otra técnica con un enfoque transductivo la cual se propuso como un algoritmo de redes de correspondencia. Este modelo utilizaba mini-lotes de ejemplos llamados episodios durante el entrenamiento, donde cada episodio estaba diseñado para imitar la tarea de pocos disparos mediante clases de submuestreo y puntos de datos. Los mini-lotes de ejemplos los unió en el *dataset* ***miniImageNet*** el cual se usa como referencia para comprobar la precisión de los sistemas. El *dataset* proviene de uno más grande *ImageNet*, siendo este un enorme conjunto de datos que puede ser de gran ayuda para generar experimentos. Por lo tanto, se diseña un nuevo conjunto de datos, ***miniImageNet***, que consta de 60, 000 imágenes en color de tamaño 84 x 84 con 100 clases, cada una con 600 ejemplos [40].

A continuación se muestra la tabla 2.1 con los estudios que presentan mejores resultados en el ámbito. Según la terminología convencional, las tareas de clasificación de *K-shot* utilizan K pares de entrada / salida de cada clase, obteniendo un total de puntos de datos NK para la clasificación *N-way*. En la tabla se expone la precisión de clasificación de *5-way* con *1-shot* y *5-shot* en el conjunto de datos ***miniImageNet***. Los resultados se muestran en intervalos de confianza del 95 % sobre las tareas [6].

Artículo	Técnica	*5-way accurancy*	
		1-shot	5-shot
Métodos Generalistas			
Métodos que aprenden la métrica			
[40]	*Matching Nets*	43.44	55.31
[34]	*ProtoNets*	49.42	68.20
[41]	*CompareNets*	50.44	65.32
[35]	*RelationNet*	51.38	67.07
Métodos de redes con memoria			
[37]	*SNAIL*	55.71	68.88
[36]	*TADAM*	58.5	**76.7**
Métodos basados en descenso por gradiente			
[38]	*Meta-LSTM*	43.56	60.60
[7]	*MAML*	48.70	63.11
[42]	*adaResNet*	56.88	71.94
Métodos Transductivos			
[39]	*Reptile*	47.07	62.74
[8]	*TPN*	55.51	69.86

Tabla 2.1: Diferentes técnicas de *Low-Shot Learning* y *Few-Shot Learning* y su rendimiento con la base de datos *miniImageNet* [6, 7, 8]

2.1.3.2. Transfer Learning

Una persona puede tomar información de una tarea previamente aprendida y usarla de manera beneficiosa para aprender una tarea relacionada. Ocurre lo mismo con la técnica de *Transfer Learning*. Dentro del mundo del *DL*, esta técnica es utilizada mediante redes neuronales pre-entrenadas en varias tareas que proporcionan características genéricas. Luego, se utilizan para construir modelos para nuevas tareas de destino sin entrenar redes neuronales específicas de cero para la tarea a tratar [43].

Transfer-Learning realiza un re-entrenamiento de las últimas capas de una red neuronal profunda, extrayendo las características relevantes para estas nuevas clases con pocos ejemplos. Con ello se creó el *Meta-Transfer Learning (MTL)*. "*Meta*" se refiere al entrenamiento de múltiples tareas, y "*Transfer*" se logra aprendiendo las funciones de escala y desplazamiento de los pesos de las redes neuronales profundas para cada tarea [6].

En *Transfer Learning for Clinical Time Series Analysis using Recurrent Neural Networks* [43] investigaron en qué medida el *Transfer-Learning* puede modelar series de tiempo clínicas multivariadas cuando se utilizan *RNN* profundas. Consideraron transferir el conocimiento capturado en una *RNN* entrenada en varias tareas simultáneamente usando un gran conjunto de datos etiquetados para construir el modelo para una tarea objetivo con datos etiquetados limitados.

2.1.3.3. Data Augmentation

Data Augmentation se creó como alternativa a los datos reales, consiguiendo reducir la varianza de los resultados de los algoritmos con la meta de disminuir el número de errores. Los datos generados conservan solo las características más importantes de los datos reales en los que se basan para ser extraídos. Las muestras sintéticas se utilizan como sustitutivo de muestras auténticas que son sensibles a la clase, protegiendo así la privacidad de los datos en los ámbitos necesarios. Sin embargo, las distintas situaciones que existen a la hora de tratar dichos datos,

obligan a que las muestras generadas se asemejen lo máximo a las originales, siendo complicada la tarea de diferenciarlas, preservando entre ellas ciertas similitudes [44, 27, 45].

Data Augmentation es un método por el cual se incrementa el volumen de los datos a través de diversas técnicas de transformación e invarianza al ruido modificando diferentes parámetros. Fue creado en el mundo de la imagen para reducir el *overfitting* o sobre-ajuste en el entrenamiento. Para modelos de clasificación de imágenes, los datos pueden aumentarse a partir de métodos que no requieran el entrenamiento de los sistemas, se basan en la experiencia o con métodos que requieren del entrenamiento de los sistemas para generar las nuevas muestras sintéticas.

Por una parte, en los métodos que no requieran del entrenamiento de los sistemas, se produce un aumento a partir de transformaciones. Este método consiste en alteraciones geométricas, fotométricas e invariantes al ruido [19, 46, 47]:

- **Transformaciones geométricas**: alteran la geometría de la imagen. Teniendo como objetivo que el sistema de *DL* sea invariante a la posición y orientación de la imagen. A este campo pertenecen las técnicas como rotación, volteo, asimetría, escalado y recorte.
- **Transformaciones fotométricas**: modifican los canales de color para hacer el sistema de *DL* invariante al color y a la iluminación de la imagen. Algunos ejemplos son modificar la saturación o el brillo de la imagen.
- **Técnicas invariantes al ruido**: se utilizan métodos como el desenfoque Gaussiano, nitidez, detección de bordes y relieve.

Por otro lado, se encuentran los métodos que requieren del entrenamiento de sistemas para generar nuevas muestras mediante un algoritmo que construya las imágenes sintéticas desde cero. Estos pueden ser técnicas de *DL* como *Generative Adversarial Networks (GAN)*, *AE* y *VAE* entre otras.

- ***GAN***: Estas redes son capaces de generar nuevas muestras después de haber recibido alimentación de otras reales junto con ruido aleatorio. Están formadas por dos partes, Generador (G), el cual se encarga de crear nuevos datos sintéticos a partir de un ruido Gaussiano, y el Discriminador (D), evalúa la calidad de los ejemplos generados por G [48]. Un ejemplo de ello es *SenseGen*, red neuronal que sintetiza los datos que provienen de los sensores [45]. Asimismo, Zhang *et al.* utilizaron una construcción de pila de *GAN* para generar imágenes realistas de pájaros y flores a partir de descripciones de texto [49]. A su vez, se creó SmartAugmentation como una red neuronal profunda que genera nuevos datos al fusionar dos o más ejemplos de la misma clase. Otra forma de utilizar las *GANs* fue bajo un enfoque bayesiano. Este planteamiento se basó en la distribución aprendida del conjunto de entrenamiento [46].

  En cuanto a este tipo de redes, hay diferentes aproximaciones destacando:

  Conditional GAN (CGAN): una versión condicional del marco GAN. Agrega información adicional tanto al G como al D. Dicha información podría ser la etiqueta de clase o cualquier otro tipo de información auxiliar. Además, el procedimiento de capacitación para *CGAN* es idéntico al de *GAN*, utilizando la divergencia de *Jensen-Shannon (JS)* como una medida de muestras generativas [49].

  Wasserstein GAN (WGAN): reemplaza la divergencia *JS* utilizada tanto en *GAN* como en *CGAN* por la distancia *Earth-Mover (EM)*. A su vez, puede resolver el problema de la desaparición de gradientes en las anteriores redes generativas. Un estudio introduce la información de la etiqueta en la red para ahorrar el costo del etiquetado manual de las imágenes generadas. Tras ello se introduce la pérdida del clasificador para mejorar aún más

la calidad de las imágenes generadas, y la precisión de la clasificación se mejora después de expandir el conjunto de datos [50].

StyleGAN: fue creada para generar rostros de personas inexistentes. Realiza una separación automática y sin supervisión de los atributos de alto nivel. Su principal contribución a las *GANs* fue una variación estocástica en las imágenes generadas a partir de un *Style-based generator*. Dentro del generador se crean detalles estocásticos mediante la introducción de entradas de ruido explícitas. A su vez, permite una escala intuitiva [51].

- ***AE***: Dentro de un *AE* se tienen dos partes, el *encoder* y el *decoder*. Entre ambos se encuentra un *Latent Space* donde los datos se transforman. Terrance DeVries y Graham W. Taylor propusieron aplicar transformaciones simples, como agregar ruido, interpolar o extrapolar, en este nuevo espacio transformado [52].
- ***VAE***: tiene la misma estructura que un *AE* con la particularidad de introducir cierta aleatoriedad en las muestras reconstruidas. Formularon el problema con modelos gráficos probabilísticos cuyo objetivo era maximizar el límite inferior de la probabilidad de los datos [53]. Se propuso un *VAE* con una mezcla de Gaussianas antes en el *Latent Space*, con un componente por ejemplo introducido [54].
- ***Generative Stochastic Networks (GSN)***: se basan en el aprendizaje del operador de transición de una cadena de Markov cuya distribución estacionaria estima la distribución de datos [55].
- ***Autoregressive Models***: utilizaban redes neuronales para modelar la distribución condicional del espacio. PixelRNN fue creado como una *Recurrent Neural Networks (RNN)* que secuencialmente predice los píxeles en una imagen a lo largo de los dos dimensiones espaciales [56].

Hasta la fecha, los parámetros y técnicas de *Data Augmentation* son específicas para cada campo y se basan en la experiencia. Sin embargo, con el objetivo de solventar dicha configuración manual de los parámetros óptimos para nuestra tarea, los investigadores de Google publicaron recientemente AutoAugment [57]. Este sistema permite ajustar automáticamente la mejor configuración para cada problema. Este trabajo buscaba las políticas mejoradas de aumento de datos diseñando un espacio donde una política consta de muchas subpolíticas. Cada una de ellas se elige al azar para una nueva imagen perteneciente a un mini-lote con una *RNN*. Cada subpolítica consta de dos operaciones, tratándose de procedimientos de transformación de la imagen.

2.2. Data Augmentation: Aplicación a Secuencias Temporales

Las técnicas de *Data Augmentation* son más populares en el campo de la imagen. Se basan en transformaciones lineales que pueden, hasta cierto punto, garantizar que la imagen modificada aún tenga parte de la información de la imagen original [58, 59]. No ocurre lo mismo en lo relativo a las secuencias temporales, ya que no se puede asegurar que la información discriminatoria no se pierda al modificar la serie de tiempo. Esto se debe a las frecuentes transformaciones no lineales en la escala temporal entre otras. Por ello, existe la enorme necesidad de inventar métodos que creen nuevos ejemplos conservando información suficiente para identificar la clase [60, 61].

El estudio de secuencias temporales es un campo difícil por lo que normalmente requiere tener en cuenta tres aspectos: las características multidimensionales, su relación con sub-secuencias de características que pertenecen a la misma serie temporal y las posibles distorsiones no lineales en la escala de tiempo [62]. Otro gran problema son las secuencias de diferentes longitudes además de ocurrir a diferentes velocidades. Por ejemplo, los gestos del lenguaje de señas donde

el mismo significado se puede realizar de manera diferente dependiendo del estado de ánimo o las intenciones del firmante [63].

Por una parte, uno de los campos a destacar dentro de las secuencias temporales es el reconocimiento de audio. Para este ámbito en concreto se pueden encontrar numerosos estudios debido a la enorme cantidad de datos que se posee. Por ejemplo, los datos se aumentaron artificialmente para tareas de reconocimiento de voz de bajos recursos en [64, 65]. La normalización de la longitud del tracto vocal se ha adaptado para el aumento de datos en [66]. El audio ruidoso se ha sintetizado mediante la superposición de audio limpio con una señal de audio ruidosa en [67]. Además, el uso de un simulador de sala acústica ha sido explorado en [68]. Así mismo, el aumento de datos para la detección de palabras clave se ha estudiado en [69, 70]. También, uno de los estudios más novedosos en reconocimiento de voz es SpecAugment [71]. Se aplica a las características de entrada deformándolas con bloques de enmascaramiento de la señal. Dicho estudio consta de tres tipos de deformaciones del espectrograma *log mel*: deformación del tiempo, enmascaramiento del tiempo y de la frecuencia.

Por otra parte, se va a realizar un estudio de aquellos métodos en los que no se poseen grandes volúmenes de datos. Dentro de este estudio, se puede hacer dos grandes distinciones como en imagen. Por un lado, se encuentran los métodos que no requieren entrenamiento de sistemas básandose en la experiencia. Por otro lado, se encuentran los métodos que requieren del entrenamiento de sistemas para generar nuevas muestras sintéticas.

Dentro del primer grupo de métodos, aquellos que no requieren entrenamiento de sistemas basándose en la experiencia, se pueden encontrar varios tipos:

- **Métodos basados en deformaciones y perturbaciones**: se generan nuevos ejemplos al deformarlas aleatoriamente [44] o perturbarlas [10]. Además, se utilizan promedios alineados ponderados [27, 31, 61] o modelos generativos [72].

  Existe un estudio que alteraba la ubicación temporal del evento dentro de la ventana en una serie temporal de datos de sensores portátiles para monitorear la enfermedad de Parkinson. Realizaban transformaciones de rotación, permutación, escalado y recorte entre otras [10].

  Otro campo estudiado son los datos electroencefalográficos aumentados utilizando distorsiones temporales y espaciales o rotacionales [73].

- **Promedios alineados ponderados**: se pueden encontrar dos técnicas principales: *Window Slicing (WS)* o corte de ventana y *Manual Wraping* o deformación manual [44].

  Un estudio introdujo cambios en el ruido y la magnitud, segmentos de tiempo deformados o recortados. Esta propuesta aparte de realizar un *WS*, aplica una deformación a cada una de las ventanas creando así secuencias temporales sintéticas (*Window Warping (WW)*), mejorando la posterior clasificación con una *CNN* [44].

  A su vez, se formaron nuevas series temporales deformando el espacio con una alineación subóptima llamándose *SuboPtimAl Warped time-series geNEratoR (SPAWNER)*. Dicha alineación es causada por restringir el camino de deformación. La deformación se realiza con *DTW* [61].

Dentro del segundo grupo de métodos, aquellos que requieren entrenamiento de sistemas para generar nuevas secuencias sintéticas, se pueden encontrar varios tipos, como *GAN*, *AE* y *VAE* entre otros. Algunos ejemplos de estudios en este ámbito son:

- **GAN**: SenseGAN fue creada para la generación de datos provenientes de sensores. En la parte del generador cuenta con varios *LSTM* junto con *Mixture Density Network (MDN)*

mientras que el discriminador esta formado por *LSTM* para distinguir entre datos sintéticos y originales [45]. También son utilizadas en bocetos basados en trazos como *SkeGAN* [74] o para generación de firma *on-line* [75].

- **AE**: se realizó un estudio de diversas enfermedades y las diferentes etapas de progresión con los diversos tipos de intervención terapéutica con *AE*. Dentro del mismo crearon un nuevo tipo de celdas *Time-Aware LSTM (T-LSTM)* para manejar intervalos de tiempo irregulares en registros longitudinales de pacientes [76].

- **VAE**: SketchRNN se creó para la generación sintética de bocetos a partir de dibujos secuenciales, mediante un *VAE*. En la parte del *encoder* cuenta con una *bidirectional RNN*, mientras que el *decoder* cuenta con un *autorregresive RNN* [1]. Tras ello, hubo muchos estudios que aplicaron la misma técnica [77, 78, 79], o alteraciones de la misma como incluir una red discriminativa en el codificador [80] o agregar un *Conditional-AE* para capturar la información posicional de cada trazo [81]. A su vez, recientemente ha sido empleado en un estudio para caracteres manuscritos, *Cross-VAE*. Este estudio se utiliza simplemente para cambiar el dominio de los datos de secuencias temporales a imágenes y viceversa [82].

- ***VAE-GAN***: *VASkeGAN* se creó para la generación de bocetos en formato vectorial. La parte del *VAE* aporta una representación eficiente de los datos y la *GAN* contribuye con capacidades de generación para producir los bocetos visualmente atractivos [74].

En la tabla 2.2 se muestran algunas de las técnicas más relevantes que se han encontrado en la literatura sobre el *Data Augmentation* en secuencias temporales y los tipos de secuencias que usan. Todas ellas aportan una mejoría al rendimiento de los sistemas de clasificación y verificación entre otros.

Como se puede observar en la tabla 2.2 existen numerosas técnicas para la generación sintética de secuencias temporales. Todas ellas han sido creadas para solventar la cuestión de la falta de datos en los distintos campos. Se tienen desde métodos más sencillos que no requieren entrenamiento como rotaciones, permutaciones, *WS* y *WW* o *DTW*, hasta métodos que requieren entrenamiento como son las redes neuronales profundas. Dentro del primer grupo, nos encontramos con un estudio que utiliza *WS* con el *UCR dataset*. Con ello genera nuevos datos consiguiendo una tasa de error de 12.9 % [44]. *DBA* también es utilizado con el mismo *dataset*. En este caso, se demuestra que con el conjunto *DiatomSizeReduction* aumentando con muestras sintéticas el *accurancy* incrementa de un 30 % a un 96 % [27]. Dentro del segundo grupo, principalmente se tienen *GANs* y *RNNs* ya que son las redes más completas para la generación sintética de muestras. En un estudio se demuestra como *RGAN* y *RCGAN*, siendo redes generadoras de secuencias sintéticas, pueden conseguir un buen rendimiento de los sistemas. Se llega a obtener un *accurancy* del 96 % solo con datos generados en el conjunto de entrenamiento cuando con datos reales es del 99 % [85]. En *GAN Text-to-Speech* se genera audio a partir de texto, consiguiendo en la escala de MOS un 4.213 mientras que el audio normal tiene un 4.55. En SenseGan el *accurancy* para detectar muestras generadas es del 50 % [45]. Por último, un estudio para generar secuencias tanto de texto como de firma manuscrita a partir de una *RNN* con un error cuadrático medio por punto de datos del 23 % [87].

Artículo	Técnica	Campo de Aplicación	Base de Datos
Métodos que no requieren entrenamiento			
[10]	*Rotation, Permutation, Time Warping, Scaling, Magnitude-Warping, Cropping*	Datos obtenidos por sensores que monitorizan la enfermedad del Parkinson	Dataset de 30 pacientes recolectado mediante Microsoft Band 2
[73]	*Temporal and spatial/rotational distortions*	Datos encefalográficos	P300 dataset y MRCP dataset
[44]	*Window Slicing (WS)*	Varios campos: sensor, encefalogramas, espectros y movimiento entre otros	UCR datasets
[44]	*Window Warping (WW)*	Varios campos: sensor, encefalogramas, espectros y movimiento entre otros	UCR datasets
[83]	*WS variation*	Datos de impago de hipotecas	2000 clientes aleatorios de financial group DNB
[27]	*ResNet (DTW Barycentric)*	Varios campos: diatomeas y carne	UCR datasets
[31]	*DTW Barycentric Averaging (DBA)*	Varios campos	UCR datasets
[84]	*SynSys*	Datos de casa inteligente	Dataset recolectado de *CASAS smart home*
[61]	*SPAWNER*	Varios campos: lenguajes de señas gestos, ocupación y movimiento entre otros	Benchmark datasets
Métodos que requieren entrenamiento			
[85]	*Recurrent GAN (RGAN)*	Datos de UCI	Philips eICU database
[85]	*Recurrent Conditional GAN (RCGAN)*	Datos de UCI	Philips eICU database
[86]	*GAN Text-to-Speech (GAN-TTS)*	Texto a audio	Audio-libro en inglés norteamericano
[45]	*SenseGen (Generative Model with RNN)*	Datos provenientes de sensores (Acelerómetro, barómetro, ...)	HAR dataset
[76]	*AE* con *Time-Aware LSTM (T-LSTM)*	Datos médicos para diagnosticar Diabetes Mellitus	EHR dataset
[87]	*RNN(LSTM)*	Texto	Penn Treebank portion of the Wall Street Journal corpus
[87]	*RNN(LSTM)*	Escritura manuscrita	IAM Online Handwriting Database
[1]	*VAE (RNN)*	Bocetos	Quick Draw

Tabla 2.2: Diferentes técnicas de *Data Augmentation* para secuencias temporales

2.3. Sistemas Biométricos

2.3.1. Introducción

Comúnmente, los sistemas de reconocimiento biométrico se centran en la investigación de patrones específicos de una persona siendo los rasgos fisiológicos o de comportamiento. Se puede definir como el reconocimiento automatizado de individuos basado en su fisiología ya sea con rasgos morfológicos (huella digital, cara, iris, geometría de la mano y de la cara entre otros) o biológicos (ADN y encefalograma entre otros) y características de comportamiento (firma, voz, forma de andar o forma de teclear entre otros) [88].

Un sistema biométrico debe tener las siguientes propiedades:

- **Universalidad**: toda persona debería poseer esta característica pero hay algunos casos, como la huella dactilar en trabajadores manuales, que no conservan esta particularidad.
- **Singularidad**: no existen dos individuos con las mismas características. Esta propiedad es difícil de aprobar analíticamente.
- **Permanencia**: los rasgos biométricos son invariantes al tiempo. Esto es una aproximación, pero se prevee que al menos sea duradero a lo largo del tiempo.
- **Colectividad**: características que permitan su medición cuantitativamente. En la práctica deben ser sistemas no intrusivos, robustos y rentables para una aplicación determinada.

Para la evaluación de una actividad específica un sistema debe cumplir unos mínimos de rendimiento (precisión de identificación, velocidad... etc), aceptación por el usuario y resistencia a ataques y falsificaciones.

Algunas de las aplicaciones más importantes del reconocimiento biométrico son el análisis forense, las transacciones de finanzas, el cruce de fronteras internaciones o la seguridad informática.

Hay dos escenarios de aplicación de los sistemas biométricos, los de verificación, que comprueban si la persona es quien dice ser, y los de identificación, que determinan la identidad de la persona. Debido a su alto impacto en la sociedad, la precisión de estos procedimientos es muy relevante. Tal es su importancia, que el estudio de la verificación de firma ha atraído a la industria y a la investigación en las últimas décadas [89, 90].

2.3.2. Firma y Escritura Manuscrita

La firma es un rasgo biométrico ampliamente utilizado para verificar la identidad de un individuo. La aceptación a lo largo de la historia de la firma como forma de autenticación personal y su presencia en documentos de gran validez, ha conseguido que su estudio tenga gran reconocimiento [90, 91].

La firma y escritura manuscrita son rasgos difíciles de estudiar por su comportamiento. Un individuo nunca realiza dos veces su firma o un escrito de forma idéntica, pero siempre se preserva una "marca de agua", es decir, su seña de identidad. Surgen muchos cambios en la representación a lo largo de la vida, desde la primera que se realiza siendo pequeños hasta que se va adaptando según pasa el tiempo. Otros aspectos externos que también influyen son la superficie donde escribimos, las herramientas con las que realizamos el boceto, el tiempo que poseemos para realizarlo e infinitas más. Por ello, muchos sistemas poseen diferente efectividad, ya que depende enormemente de las condiciones del entrenamiento. Uno de los principales campos de actuación es el análisis forense. En él se examina la autenticidad de la firma mediante una inspección cuidadosa de características, consiguiendo detectar falsificadores y cambios legítimos. Otro gran campo es la salud, donde la escritura puede ayudar a la identificación de problemas en el aprendizaje infantil o problemas cognitivos y motores degenerativos a edad temprana. Además, la grafología utiliza la escritura a mano para estimar la personalidad, inteligencia, habilidades sociales, emociones y actitudes sociales de un individuo [25, 92].

Un rasgo muy importante a la hora del estudio es la variabilidad intraclase, en otras palabras, las ejemplos pertenecientes a la misma clase, y variablidad interclase, correspondientes a diferentes. Los métodos implantados hoy en día requieren de un número considerable de ejemplos del mismo conjunto para estudiar las características que comparten los datos intraclase e

interclase. Una gran incógnita a la hora de investigar la firma y escritura manuscrita es la falta de ejemplos. Coleccionar una gran base de datos es algo costoso, sobre todo cuando se requieren numerosos ejemplos de un mismo usuario [25, 92].

Es necesario realizar una distinción entre los dos tipos de firma y escritura manuscrita según el método de adquisición, teniendo en cuenta que estos datos se convierten siempre a digital para su posterior análisis: *off-line* y *on-line*. Por un lado, la firma y escritura ***off-line*** o estática es aquella en la que el boceto se trata como una imagen. Puede ser tanto un manuscrito escrito previamente en papel y escaneado o realizado sobre una superficie electrónica y guardado como imagen. Por otro lado, la firma y escritura ***on-line*** o dinámica tiene la particularidad de ser adquirida mediante un dispositivo electrónico. Consta de los sucesivos puntos escritos representados en dos coordenadas x e y como una función de tiempo. Estos puntos están ordenados según el momento de su representación sobre la superficie. Aparte de obtener las posiciones del lápiz, nos suele aportar información extra como la presión ejercida, el tiempo de realización, la posición espacial en cada instante de tiempo, la velocidad de la firma... etc. A la hora de guardar la información, ocupa más un manuscrito *off-line* que uno *on-line*. A su vez, una firma o escrito *off-line* aporta menos información característica y posee menos precisión que una *on-line*. [91]

La concepción de firmas y bocetos sintéticos es un problema a la orden del día. Surgió con la finalidad de resolver el problema de la falta de ejemplos. Diseñar una nueva base de datos es algo costoso ya que se suele enfocar a un problema concreto, limitando su aplicación a otras áreas. De igual manera, los problemas legales con respecto a la protección de datos no ayudan a la distribución de la información biométrica [25].

La generación sintética de firmas y escritura manuscrita ha nacido para poder solventar esta cuestión. Se pueden encontrar algunas técnicas en la literatura para originarlas.

Por un lado, algunas de las técnicas utilizadas en la generación de estas secuencias temporales no requieren de entrenamiento como son la distorsión afín o distorsión de trazo [93] o métodos basados en glifos, los cuales registran letras o palabras separadas de un usuario, aplican una deformación geométrica y las vuelven a unir [94, 95]. El modelo más conocido es el modelo ***Sigma-LogNormal*** donde la información dinámica de los datos se capta mediante un muestro *LogNormal* de la trayectoria contínua. La teoría cinemática sugiere que la velocidad de los golpes rápidos se puede moldear mediante una *delta LogNormal*. Este método fue utilizado en firma basándose en el comportamiento humano [25], o imitando el movimiento motor de una persona a partir de descriptores estadísticos [92]. Otro enfoque de esta técnica es variando los trazos de los parámetros lognormales o modificando sus puntos de destino virtuales.

En general, el rendimiento de un sistema mejora con el número de ejemplos de entrenamiento que se posee. Este hecho se ha demostrado en numerosos estudios tanto de escritura manuscrita como de firma.

En cuanto a escritura manuscrita, *BioTouchPass* utilizó modelos basados en *DTW* que lograron una mejora absoluta del 4.8 % de *Equal Error Rate (EER)* al pasar de entrenar el sistema con un ejemplo del dígito 5 manuscrito a 4 ejemplos manuscritos del mismo dígito por usuario. De igual manera, el rendimiento mejoró al emplear los mismos experimentos con *Bidirectional Long Short-Term Memory (BLSTM)* en un 2.2 % de ERR. Con este aumento pudieron analizar la variabilidad intrausuario y realizar un análisis completo de como cambia el rendimiento del sistema biométrico dependiendo del número de ejemplos adquirido [96]. Otro estudio que muestra una mejora al aumentar el número de ejemplos es *BioTouchPass2*. Expone que aplicar *DTW* con 5 veces más de datos manuscritos disminuye el *ERR* en un 5.41 %, mientras que utilizarlo con su algoritmo propuesto, *Time-Aligned Recurrent Neural Networks (TA-RNNs)*, reduce el *ERR* en un 7.44 % [97].

En en ámbito de firma *on-line*, en [98] se realizaron experimentos generando firmas duplicadas con el modelo *Sigma LogNormal* que son indistinguibles de las originales, como lo demuestra

una prueba visual de *Turing*. De igual manera [99] utiliza el modelo *Sigma LogNormal* para generar firmas sintéticas. Con ello mejoran los resultados de la autenticación de firmas. Otro estudio de firma manifiesta que para los sistemas basados en *Hidden Markov Model (HMM)* y *Gaussian Mixture Model (GMM)* sería conveniente tener en cuenta todas las firmas de capacitación disponibles o al menos algunas firmas de entrenamiento pero de varias sesiones para generar un plantilla de usuario más robusta. Con ello, también verifica la premisa de que con más firmas se obtienen mejores resultados [100]. Por último, *DeepSign* es un estudio que abarca numerosas bases de datos de firma. En él se demuestra junto con el algoritmo *TA-RNN* que el *ERR* pasa de ser de un 9.6 % con una firma por usuario a un 8,0 % con 4 firmas por usuario [101].

Desde el punto de vista de la seguridad se ha demostrado que el caso ideal es tener la mayor cantidad de información del usuario posible. Por todo ello, en este TFM se quiere aplicar técnicas de *Data Augmentation* para incrementar el número de ejemplos de un *dataset* y estudiar la variabilidad intrapersonal, siendo el primer trabajo que propone el uso de un *VAE* para firma manuscrita *on-line*.

3
Bases de Datos

En este capítulo se van a describir las bases de datos utilizadas en este TFM: *Quick Draw* dentro de la sección 3.1, *eBioDigitDB* en la sección 3.2 y por último en la sección 3.3, *DeepSignDB*. Se detalla la información que posee cada una de ellas y como se ha tratado para el problema a resolver.

3.1. Quick Draw

Quick Draw [1] es una colección de 50 millones de dibujos separados en 3.345 categorías diferentes. Se pueden encontrar desde animales como jirafa, cerdo, gato, hasta aviones, la Torre Eiffel o la Mona Lisa. Son capturas *on-line* o dinámicas y están etiquetadas con metadatos de la categoría del esbozo y el país donde se ha realizado.

Figura 3.1: Ejemplo de *Quick Draw* [1].

El formato utilizado en ***Quick Draw*** es el siguiente: se representa cada secuencia vectorial como una lista de desplazamientos en las coordenadas (Δx, Δy) y el valor binario del *Pen State* ('0' cuando el bolígrafo está dibujando y '1' cuando está levantado del papel, para cada *stroke*). Esto es lo que se denomina como *Stroke-3*, como se observa en la figura 3.2. Al utilizar esta base de datos con la arquitectura *SketchRNN*, los datos se transforman a *Stroke-5*, es decir, (Δx, Δy, *p1*, *p2*, *p3*). Los dos primeros datos se refieren a la distancia espacial entre muestras temporales

consecutivas. El primer estado, *p1*, representa que el lápiz está en contacto con el papel y por consiguiente en el instante posterior va a seguir dibujando. El estado *p2* señala que el lápiz se ha levantado del papel inmediatamente después del punto en el que se encontraba, y que a continuación no se dibujará. Por último, en el estado *p3*, se indica que el dibujo ha terminado de realizarse.

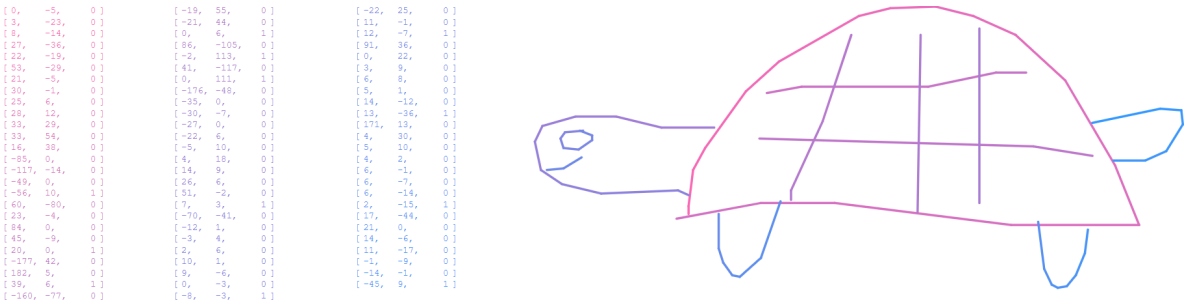

Figura 3.2: Ejemplo de dibujo en formato *Stroke-3* [2].

3.2. eBioDigitDB

La base de datos ***eBioDigitDB*** [3] ha sido creada por *BiDAlab*. Contiene los dígitos del 0 al 9 adquiridos mediante escritura *on-line* con una tableta Samsung Galaxy Note 10.1. El *dataset* cuenta con 93 usuarios. Los datos adquiridos hacen referencia a las posiciones (x,y) donde se encuentra el dedo a la hora de dibujar sobre la pantalla.

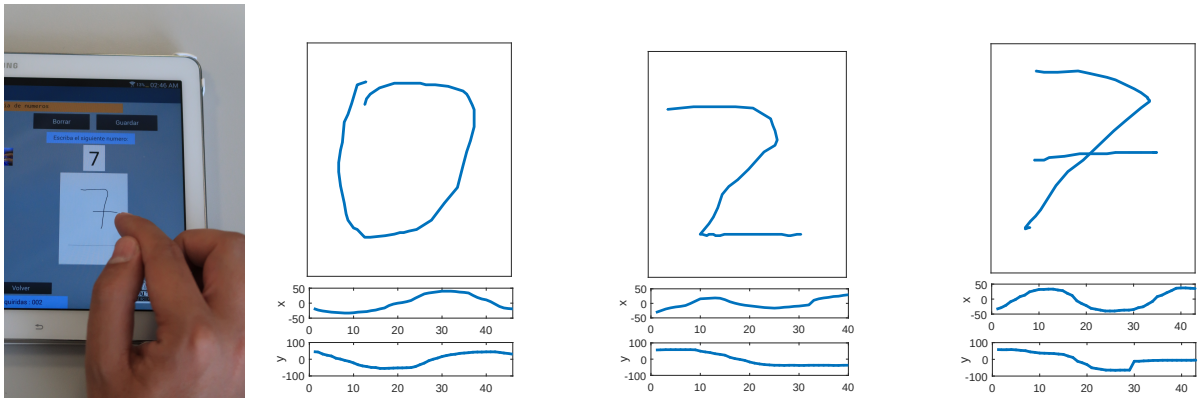

Figura 3.3: Configuración de adquisición de dígitos mediante *eBioDigitDB* [3].

En la figura 3.3 se muestra un ejemplo de la adquisición. Los patrones fueron obtenidos en dos sesiones diferentes con una diferencia de tiempo de al menos tres semanas entre sesiones, para poder contar con la variabilidad inter-sesión. Dentro de cada una de las sesiones el usuario repetía cuatro veces una serie de dibujas las cifras del 0 al 9. Con ello la base de datos cuenta con 80 ejemplos por usuario, 8 de las cuales pertenecientes a cada dígito. El *software* desarrollado pretende minimizar la variabilidad intra-usuario en el proceso de adquisición. La superficie de la tableta es de 5 pulgadas y permite al usuario repetir el dibujo siempre y cuando considere que no esté correcto.

Cada muestra cuenta con cuatro datos, entre los que se encuentran la posición (x,y) del boceto, su *timestamp* y por último la presión. Dentro de esta última característica, al estar las cifras esbozadas con el dedo, cuenta con los valores 255, si se ejerce presión, y 0, en caso contrario.

3.3. DeepSignDB

La base de datos ***DeepSignDB*** [4] también fue creada por *BiDAlab*. Su principal objetivo es la recopilación de firma manuscrita *on-line* para la verificación de identidad. Está compuesta por 1526 usuarios de cinco *datasets* diferentes, entre los que se incluyen *MCYT* con 330 usuarios y una sola sesión por usuario, *BiosecurID* con 400 usuarios y 4 sesiones, *Biosecure DS2* con 650 usuarios y 2 sesiones, *e-BioSign DS1* con 65 usuarios y 2 sesiones y *e-BioSign DS2* con 81 usuarios y 2 sesiones, siendo esta última adquirida al crear *DeepSignDB*. Se utilizan 8 dispositivos diferentes para la adquisición y dos entradas diferentes, mediante lápiz y con el dedo del usuario, aportando variabilidad.

Dentro de *DeepSignDB* han sido utilizados los usuarios pertenecientes a *Stylus*. Estos usuarios se distinguen debido a su forma de adquisición, el lápiz. La base de datos ha sido dividida en dos conjuntos diferentes, uno para el desarrollo del sistema que cuenta con el 70 % de los datos y otro para validar con el 30 % de los usuarios. Es de vital importancia remarcar que los usuarios pertenecientes a cada uno de los conjuntos son diferentes, evitando resultados sesgados. Dentro del conjunto de desarrollo se poseen 1084 usuarios. Dicho grupo se fracciona en dos partes, el 80 % de los usuarios son utilizados para entrenar el sistema, mientras que el 20 % restante se utiliza para la validación del desarrollo. La agrupación de datos para validación cuenta con 442 usuarios. Con el propósito de realizar un análisis completo y honesto de los sistemas de verificación de firmas, y observar su funcionamiento en diferentes escenarios, ya que aportan variabilidad entre sesiones y diferentes forma de adquisición (Lápiz o dedo del usuario) entre otras. A continuación se muestra la figura 3.4 con todos los detalles sobre la distribución de los datos.

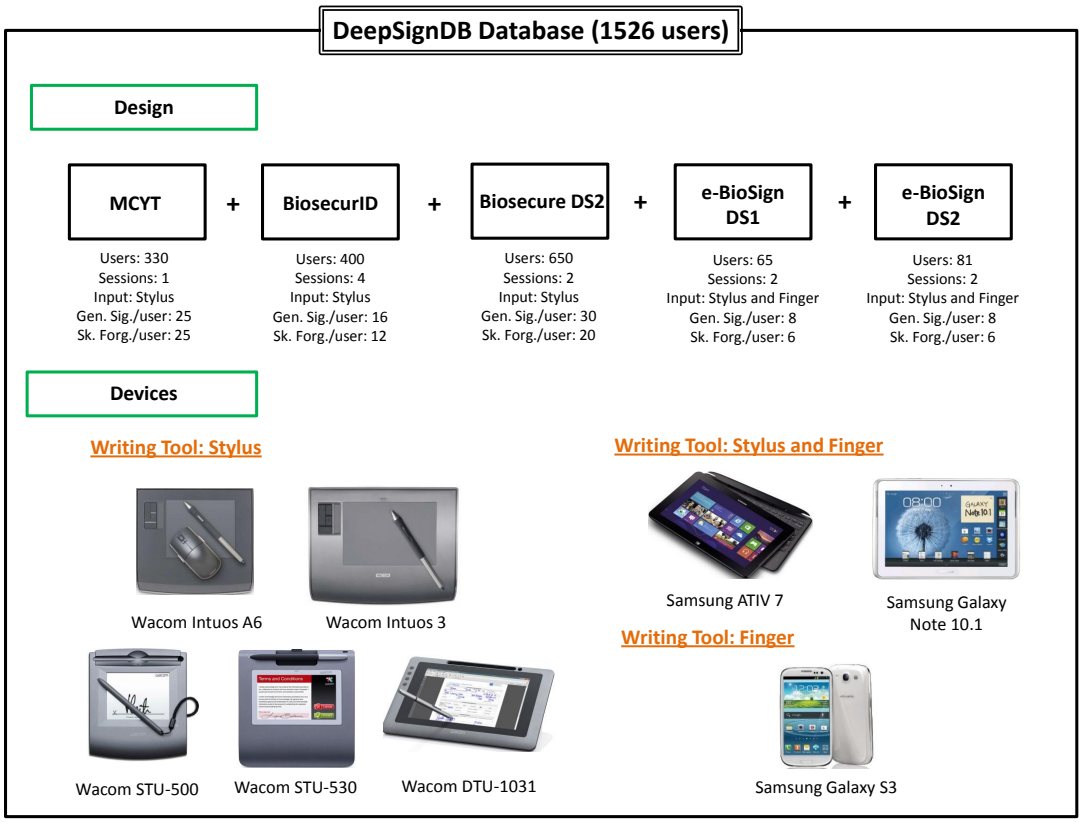

Figura 3.4: Descripción del diseño, dispositivos de adquisición y herramientas de escritura en *DeepSignDB* [4].

4

Método Propuesto

En este capítulo, se describe el método propuesto para la generación de secuencias temporales de firma y escritura sintéticas. El sistema inicial de partida es *SketchRNN* [1] creado por Google. En primer lugar, se explican los inconvenientes encontrados en el sistema de *SketchRNN* [1] en la sección 4.1. Seguidamente, en la sección 4.2, se desarrolla el sistema propuesto el cual consta de dos módulos: el primero se encarga de la segmentación en trazos y el segundo es el módulo de síntesis.

4.1. Inconvientes del Sistema Inicial

El algoritmo *SketchRNN* [1] cuenta con una serie de inconvenientes que impiden su aplicación directa en los escenarios de estudio del presente TFM:

- El algoritmo solo acepta como entrada secuencias temporales de dimensión *Stroke-3*, es decir, de cada muestra temporal 3 datos: (Δx, Δy, *pen-state*) convirtiéndolo posteriormente en *Stroke-5*, extiende el *pen-state* a tres valores según la posición del lápiz con respecto del papel. Por esta razón, hay que adecuar el *dataset* de entrada a esta dimensión.
- El código proporcionado esta adecuado solamente a una clase de *Quick Draw*, por ejemplo la clase oveja. Todas las secuencias temporales del *dataset* se completan con ceros para presentar la misma longitud. El principal problema llega al estar N_{max}, la longitud más larga del *dataset* de entrada, predeterminada al conjunto de datos “aaron-sheep.npz". Teniendo en cuenta que cada *dataset* tiene una longitud máxima, este parámetro es necesario ajustarlo cada vez que se utilice un nuevo conjunto de datos de entrada.
- Sólo funciona con secuencias temporales de un tamaño máximo de 300 muestras para aportar una buena reconstrucción. Esto supone un gran problema para las aplicaciones de escritura a mano, ya que la frecuencia de muestreo de los dispositivos es de alrededor de 200 Hz. Por ello, se han analizado las secuencias temporales utilizadas en este estudio. En el caso de *eBioDigitDB* las secuencias temporales no superan esta longitud, mientras que en *DeepSignDB* sí. Así pues, para el caso de *DeepSignDB*, se propuso dividir las firmas que conforman la base de datos en trazos, basándose en la velocidad de las mismas y consiguiendo ejemplos más pequeños. Estos trazos son los utilizados en el *VAE* consiguiendo

un modelo que sintetice a nivel de trazos, no a nivel de firma completa ni de usuario como lo realizan otro enfoques de *DL*.

- Por último, esta técnica no es capaz de generalizar correctamente cuando el entrenamiento se realiza con más de una clase a la vez, por ejemplo la clase oveja y cactus. Es necesario entrenar un modelo para cada clase de datos y así poder obtener buenos resultados. En este TFM se propone entrenar un único sistema que permita su correcta generalización a nuevas muestras no vistas durante el entrenamiento. Para ello, las secuencias temporales introducidas se dividen en trazos como se ha explicado anteriormente.

4.2. Sistema Propuesto

Para solventar los inconvenientes del sistema inicial descritos, se ha creado un sistema de síntesis a nivel de trazos para cualquier escenario de firma y escritura manuscrita. Con ello, se ha conseguido captar casi toda la información de los datos de entrada, hasta los detalles más minúsculos. A su vez, ha sido posible generalizar nuestro sistema para cualquier usuario y cualquier escenario ya que no se encuentra entrenado a nivel de usuario, si no a nivel de trazos. En la figura 4.1 se muestra el sistema propuesto compuesto por dos módulos:

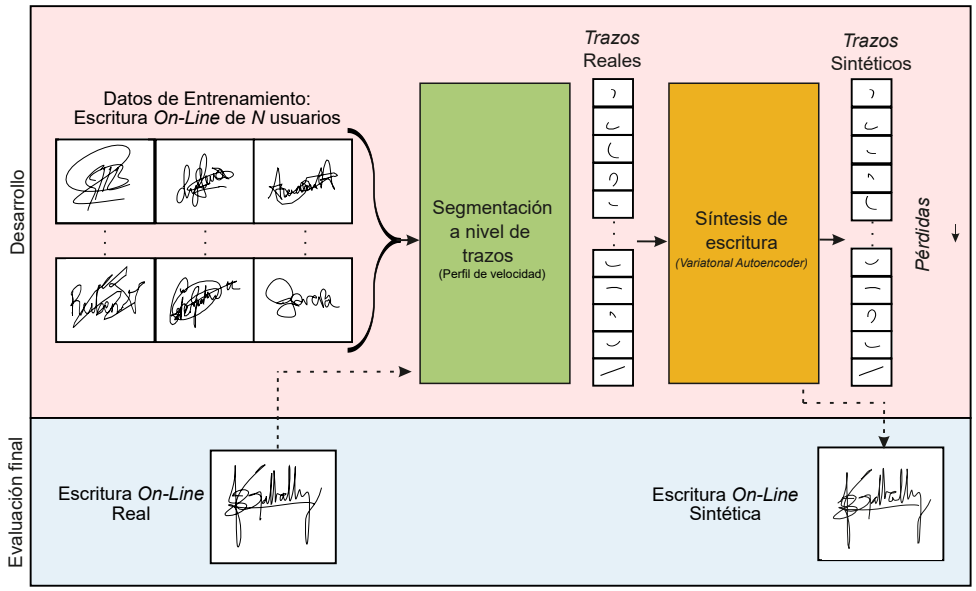

Figura 4.1: Esquema de la arquitectura propuesta.

- **Módulo de Segmentación a nivel de trazos** donde se adecúan las secuencias temporales para el posterior módulo. En él, se dividen las secuencias temporales en conjuntos de muestras más pequeños para evitar introducir secuencias temporales de más de 300 muestras.
- **Módulo de Síntesis de escritura *on-line*** en el que se describe el *VAE* a utilizar, su entrenamiento y los parámetros ajustables.

4.2.1. Módulo de Segmentación en Trazos

En muchos de los escenarios de la vida real, como es el caso de las firmas, el tamaño de las secuencias temporales suele ser mucho mayor de 300 muestras. Se ha decidido dividirlas

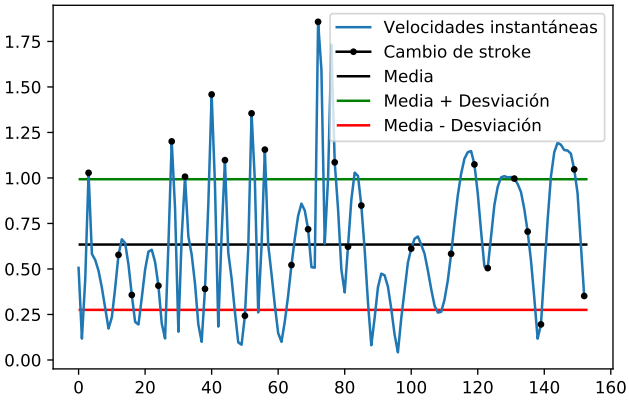

Figura 4.2: Velocidad de una firma perteneciente a *DeepSignDB* con las particiones realizadas por la media y la desviación estándar.

en diferentes trazos para evitar uno de los inconvenientes del *VAE*. A la hora de realizar esta partición de las muestras, se ha adoptado una postura basada en la extracción de la velocidad de la firma de los tramos en los que el usuario se encuentra dibujando, basándonos en la extracción que realizan en [102]. El primer paso de nuestro pre-procesamiento para cada muestra antes de introducir los trazos en el *VAE* es dividir la firma según el *Pen-State*. Se realiza una separación de las partes de la firma en las que el lápiz se encuentra dibujando (partes que se utilizarán para entrenar el sistema) y aquellas en las que el lápiz se halla levantado (partes que se utilizarán para la posterior reconstrucción de la firma). El siguiente paso es tratar todas las partes en las que el lápiz se encuentre esbozando para extraer la velocidad de la firma. Para cada muestra de este conjunto, se obtienen las posiciones X e Y como las coordenadas que tienen las secuencias de tiempo en la pantalla a la hora de realizar el esbozo. Cada una de estas señales se compone de:

$$X = x_1, x_2, ..., x_n, ..., x_N \tag{4.1}$$

$$Y = y_1, y_2, ..., y_n, ..., y_N \tag{4.2}$$

donde N indica el número total de muestras por cada secuencia temporal. El vector de velocidad V se calcula en cada punto n a partir de sus posiciones x_n e y_n como se puede observar en la siguiente ecuación:

$$v_n = \sqrt{\dot{x_n}^2 + \dot{y_n}^2} \tag{4.3}$$

donde $\dot{x_n}^2$ y $\dot{y_n}^2$ representan la primera derivada de muestras consecutivas de x e y respectivamente.

Tras ello, se realiza la extracción de los trazos de la secuencia temporal a partir de la media y la desviación estándar de la velocidad basándonos en el estudio realizado en [102] como se muestra en la figura 4.2.

En la figura 4.2 se divide la velocidad en cuatro particiones, siendo sus tres límites la suma de la media y la desviación estándar, la media, y la resta de la media menos la desviación estándar. A partir de estos tres límites se forman los trazos. En cada instante en el que el vector de velocidad sobrepasa uno de estos límites, dicho vector se particiona formándose un nuevo trazo. A su vez, se ha tenido en cuenta el tamaño de estos nuevos trazos, indicando que si cuentan con dos o menos muestras, estos se añadan al siguiente trazo. Con ello, los trazos de menor longitud contarán con tres muestras. En el ejemplo mostrado en la figura 4.3 se cuenta con 25 trazos. Por último, se ha comprobado que la longitud media de todos los trazos de las diferentes bases de

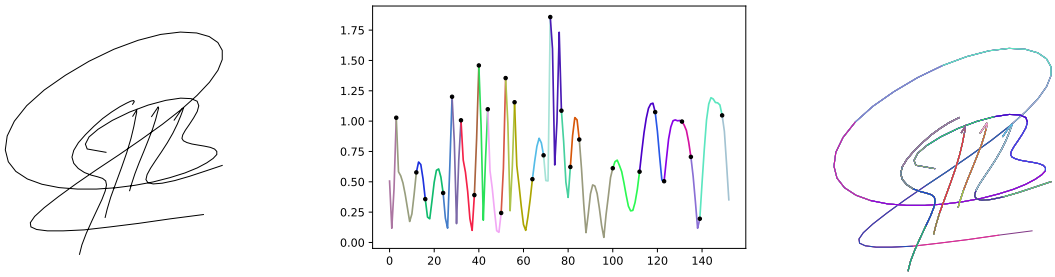

Figura 4.3: Firma perteneciente a *DeepSignDB*, su velocidad y división en trazos según la media y la desviación típica y la misma firma dividida en trazos respectivamente.

datos a utilizar, no supera el número máximo de muestras permitido para obtener una buena reconstrucción, contando el vector más largo con 172 muestras. Con ello se demuestra que este método consigue dividir una firma en vectores de menos de 300 muestras.

En la figura 4.3 se muestra como pasamos de tener una firma completa a su partición en trazos los cuales serán las secuencias temporales introducidas en el *VAE*. En ella se puede observar la firma original, la velocidad separada en trazos según el color de la misma y la firma completa dividida en sus diferentes trazos marcados por los cambios de color.

4.2.2. Módulo de Síntesis

4.2.2.1. Descripción de la Arquitectura

El sistema inicial utilizado en este TFM es *SketchRNN* [1], un *sequence-to-sequence VAE*, desarrollado por *Google* con el propósito de ayudar a la comunicación de los seres humanos por medio de dibujos vectoriales de bocetos.

El *sequence-to-sequence VAE* consta de dos partes: *Encoder* y *Decoder*, como se observa en la figura 4.4.

Dentro del *Encoder* podemos encontrar una *Bidirectional RNN*. Tiene como entrada un vector que representa a la secuencia temporal de dimensión *Stroke-3* convirtiéndolo posteriormente en *Stroke-5*. Dentro de esta red, se introducen la secuencia S y la secuencia invertida $S_{reverse}$, obteniendo dos estados ocultos $h\rightarrow$ y $h\leftarrow$. Se produce una concatenación de ambos estados h y se proyectan en dos vectores μ y σ, ambos de igual tamaño. Tras ello, se multiplican estos dos vectores por una distribución Gaussiana del mismo tamaño, siendo una normal de media 0 y desviación estándar 1. Así se obtiene el conocido *Latent Vector (z)*, teniendo cierto grado de aleatoriedad debido a la Gaussiana. En otras palabras, el *Latent Vector (z)* es un vector que contiene una secuencia temporal codificada en 128 caracteres, no siendo determinista, es decir, para los mismos datos de entrada se puede conseguir una secuencia codificada diferente.

El *Decoder* consta de una *Autorregresive RNN*. Tiene como entrada el *Latent Vector (z)* proveniente del *encoder* y a la salida muestra una secuencia temporal determinada para cada *Latent Vector*. Así se obtienen las muestras reconstruidas con cierta aleatoriedad. El estado inicial h_0 y los estados de celda opcionales c_0, son la salida de una pequeña red neuronal de una sola capa: $[h_0\ ;\ c_0] = tanh\ (W_z z + b_z)$. Cada entrada del *decoder* x_i se crea concatenando S_{i-1} y el *Latent Vector (z)*, donde S_0 corresponde con *(0, 0, 1, 0, 0)*. La salida en cada momento es S_i, parámetros procedentes del resultado de una distribución de probabilidad. Dentro de S_i se encuentran $(\Delta x, \Delta y)$ modelados por un *GMM* y $(q1, q2, q3)$, como una distribución categórica para forjar los datos de verdad básica $(p1, p2, p3)$, donde $(q1+q2+q3 = 1)$. La secuencia

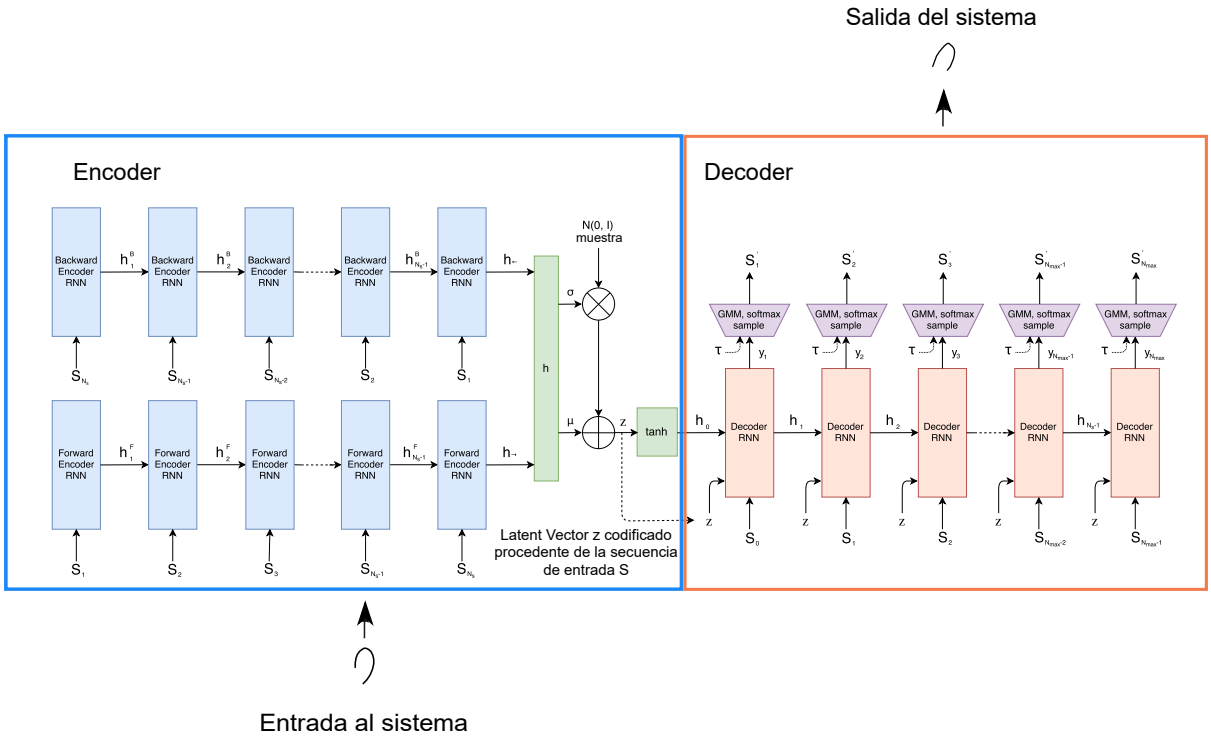

Figura 4.4: Esquema del módulo de síntesis.

temporal decodificada está determinada por el *Latent Vector (z)* resultante del *encoder*, el cual se entrena extremo a extremo junto con el decodificador. Para modelar *(Δx, Δy)* del primer estado del *RNN* con un *GMM* es necesario calcular:

$$p(\Delta x, \Delta y) = \sum_{j=1}^{M} \prod_{j} \mathcal{N}(\Delta x, \Delta y | \mu_{x,j}, \mu_{y,j}, \sigma_{x,j}, \sigma_{y,j}, \rho_{xy,j}), \;\; donde \;\; \sum_{j=1}^{M} \prod_{j} = 1 \tag{4.4}$$

La ecuación consta de dos partes principales:

- $\mathcal{N}(x, y | \mu_x, \mu_y, \sigma_x, \sigma_y, \rho_{xy})$ es la función de distribución de probabilidad para una distribución normal bivariada. Se tienen M distribuciones con cinco parámetros cada una: $(\mu_x, \mu_y, \sigma_x, \sigma_y, \rho_{xy})$ donde μ_x y μ_y son las medias, σ_x y σ_y las desviaciones estándar y ρ_{xy} corresponde con el parámetro de correlación de cada una de las distribuciones normales bivariadas.
- El vector $\prod$ con una longitud de M es una distribución categórica que contiene los pesos de mezcla del *GMM*.

A partir de ello se forma el vector y como salida del *decoder* con una longitud de $5M + M + 3$, que incluye lo necesario para generar $(q1, q2, q3)$.

Para calcular los siguientes estados ocultos del *RNN* se sigue el mismo patrón. Se realiza a partir de su operación de avance, calculando y_i a partir de una capa *fully-conected*:

$$x_i = [S_{i-1}; z], [h_0; c_0] = forward(x_i, [h_{i-1}; c_{i-1}]), \;\; y_i = W_y h_i + b_y, \;\; y_i \in \mathbb{R}^{6M+3} \tag{4.5}$$

El vector y_i se divide en los parámetros de distribución que indica la siguiente ecuación:

$$y_i = [(\widehat{\prod_1} \mu_x \mu_y \widehat{\sigma_x} \widehat{\sigma_y} \widehat{\rho_{xy}})_1 ... (\widehat{\prod_1} \mu_x \mu_y \widehat{\sigma_x} \widehat{\sigma_y} \widehat{\rho_{xy}})_M (\widehat{q_1}, \widehat{q_2}, \widehat{q_3})] \tag{4.6}$$

Para asegurar que los valores de desviación estándar sean positivos y que la correlacción se encuentre entre -1 y 1 se aplican las operaciones de *exp* y *tanh* a estos mismos valores:

$$\sigma_x = exp(\widehat{\sigma_x}), \;\; \sigma_y = exp(\widehat{\sigma_y}), \;\; \rho_{xy} = tanh(\widehat{\rho_{xy}}) \tag{4.7}$$

Las probabilidades para las distribuciones categóricas se calculan utilizando las salidas de la siguiente manera:

$$q_k = \frac{exp(\widehat{q_k})}{\sum_{j=1}^{3} exp(\widehat{q_j})}, \;\; k \in 1, 2, 3, \;\; \prod_k = \frac{exp(\widehat{\prod_k})}{\sum_{j=1}^{M} exp(\widehat{\prod_j})}, \;\; k \in 1, ..., M \tag{4.8}$$

Un gran desafío para este modelo es entrenarlo para conocer el momento en el que debe dejar de dibujar. Esta tarea se convierte en algo difícil debido al desbalanceamiento entre los tres eventos del esbozo (dibujar, levantar el lápiz del papel o acabar el boceto). La probabilidad de un evento *p1* (dibujar) es mucho mayor que la de un evento *p2* (levantar el lápiz del papel) e infinitamente mayor que la de un evento *p3* (acabar el boceto) que solo ocurrirá una vez por dibujo. Debido a este desafío, se desarrolló un enfoque en el que todas las secuencias son generadas con una longitud N_{max}, donde esta medida es la relativa a la longitud del boceto más largo dentro del *dataset* de entrenamiento. Debido a que normalmente la longitud de S es inferior a N_{max}, se configura el *dataset* de tal manera que todos los puntos S_i para $i > Ns$ sean *(0, 0, 0, 0, 1)*.

Una vez acabado el entrenamiento, se puede probar el modelo a partir de nuevas muestras. Durante el muestro, se generan los parámetros para la *GMM* y las distribuciones categóricas en cada instante de tiempo, obteniendo el resultado $S_i^{'}$ para ese instante. A diferencia del entrenamiento, se utiliza $S_i^{'}$ como entrada para el siguiente instante de tiempo. El proceso continua realizándose de igual manera hasta que ocurre uno de estos dos sucesos: *p3 = 1* o $i = N_{max}$. Al igual que en el *encoder* la salida no es determinista, si no que posee cierta aleatoriedad definida por el *Latent Vector (z)*, en el caso del *decoder* se puede introducir cierto nivel de incertidumbre a la secuencia temporal determinada para cada *Latent Vector* con un parámetro, la temperatura (τ):

$$\widehat{q_k} \rightarrow \frac{\widehat{q_k}}{\tau}, \;\; \prod_k \rightarrow \frac{\widehat{\prod_k}}{\tau}, \;\; \sigma_x^2 \rightarrow \sigma_x^2 \tau, \;\; \sigma_y^2 \rightarrow \sigma_y^2 \tau \tag{4.9}$$

En consecuencia, se demuestra que se puede escalar con la temperatura (τ) tanto los parámetros de la distribución categórica como los parámetros σ de la distribución normal bivariada. Con ello es posible controlar la aleatoriedad de las nuevas muestras. Este parámetro suele variar entre 0 y 1. Cuando τ es 0 se trata de un modelo determinista, siendo la salida el punto más probable de la función de densidad de probabilidad [1].

4.2.2.2. Entrenamiento del Sistema

En ell entrenamiento del *VAE* la función de pérdidas es la suma de dos términos: la pérdida de reconstrucción (L_R) y la pérdida de la divergencia *Kullback-Leibler* (L_{KL}). El algoritmo se entrena para optimizar ambos parámetros.

La pérdida de reconstrucción (L_R) maximiza la semejanza logarítmica de la distribución de probabilidad generada para explicar los datos de entrenamiento S. L_R se puede calcular a partir de las siguientes ecuaciones y de los datos de entrenamiento S. Las pérdidas de reconstrucción se constituyen con la suma de la pérdida de registro del desplazamiento de (x, y) , L_s, y la pérdida logarítmica de los términos del estado de la pluma (*p1*, *p2*, *p3*), L_p:

$$L_s = -\frac{1}{N_{max}} \sum_{i=1}^{N_s} log(\sum_{j=1}^{M} \prod_{j,i} \mathcal{N}(\Delta x_i, \Delta y_i | \mu_{x,j}, \mu_{y,j,i}, \sigma_{x,j,i}, \sigma_{y,j,i}, \rho_{xy,j,i})) \tag{4.10}$$

$$L_p = -\frac{1}{N_{max}} \sum_{i=1}^{N_s} \sum_{k=1}^{3} p_{k,i} log(q_{k,i}), \quad L_R = L_s + L_p \tag{4.11}$$

Ambos términos están normalizados por la longitud total de la secuencia N_{max}. Esta metodología permite que el algoritmo aprenda fácilmente cuando el usuario termina de realizar la secuencia.

La pérdida de la divergencia *Kullback-Leibler* (L_{KL}) mide la diferencia entre la distribución del *Latent Vector (z)* que se tiene y un vector Gaussiano que de primeras cuenta con media 0 y desviación estándar 1. Al optimizar este término se consigue reducir esta diferencia. Al calcular L_{KL} también se normaliza por N_{max}:

$$L_{KL} = -\frac{1}{2N_z}(1 + \widehat{\sigma} - \mu^2 - exp(\widehat{\sigma})) \tag{4.12}$$

Tras la obtención de ambos términos, obtenemos el resultado de las pérdidas como una suma ponderada de los mismos:

$$Loss = L_R + w_{KL} L_{KL} \tag{4.13}$$

Existe una compensación entre los términos. cuando $w_{KL} \to 0$ el modelo se asemeja a un *AE* puro. Con ello se consiguen mejores métricas de pérdida de reconstrucción pero apenas se tiene aleatoriedad sobre el *Latent Vector (z)*.

Tras el estudio se averiguó otra función de pérdidas que obtenía mejores resultados que la anterior. Esta segunda se utilizó solamente para el entrenamiento, manteniendo la anterior tanto en validación como en test:

$$\eta_{step} = 1 - (1 - \eta_{min} R^{step}) \tag{4.14}$$

$$Loss_{train} = L_R + w_{KL}\, \eta_{step}\, max(L_{KL}, KL_{min}) \tag{4.15}$$

Se ha comprobado que con esta nueva función de pérdidas se obtienen mejores resultados. Se debe a que con ello el entrenamiento de pérdidas se centra primero en el término L_R, el cual es más complicado de optimizar. Tras acabar con las pérdidas de reconstrucción, se optimizan L_{KL}, tarea más sencilla. En entre entrenamiento η_{step} comienza en η_{min} (típicamente 0 o 0.01) en el primer *step* del entrenamiento y llegando a converger en 1 al llegar a grandes *steps*. R es un término cercano pero nunca llega a 1. Otro término importante es $max(L_{KL}, KL_{min})$. Esta parte de la función se constituyó así debido a que en la practica se observó que variar de un valor L_{KL} grande ($L_{KL} > 1$) a un valor más pequeño ($L_{KL} \sim 0{,}3$) provoca un aumento en la calidad de las imágenes cuando se utiliza una distribución Gaussiana para el *Latent Vector (z)* de $\mathcal{N}(0, I)$. Sin embargo, cambiar de $L_{KL} \sim 0{,}3$ a valores más pequeños no producía ninguna

mejora notable. Por último, el término KL_{min} generalmente varía entre 0.10 y 0.50. Este término conseguirá que el optimizador se centre menos en el término L_{KL} cuando sea lo suficientemente pequeño, para conseguir mejores métrica en el término L_R [1].

4.2.2.3. Configuración del Sistema

El *encoder* está constituido por una *RNN* de 521 nodos mientras que el *decoder* cuenta con 2048 nodos. Dentro del modelo se cuenta con $M = 20$ para el *decoder*. El *Latent Vector (z)* cuenta con $N_z = 128$ dimensiones. Se aplica *Layer Normalization* y *Recurrent Dropout* durante el entrenamiento con una probabilidad de mantenimiento del 90 %. El modelo se entrena con bloques de 100 ejemplos, con *Adam*, una tasa de aprendizaje del 0.0001 y un recorte por gradiente de 1.0. A su vez, se entrena con $KL_{min} = 0,20$ y $R = 0{,}99999$. Durante el entrenamiento también se realiza un pequeño *Data Augmentation* multiplicando *(Δx, Δy)* por dos factores aleatorios elegidos uniformemente entre $\sigma - 0,1$ y $\sigma + 0{,}1$. Con ello, se permite modificar el valor de w_{KL} para el entrenamiento. Se recomienda que las secuencias temporales introducidas no superen las 300 muestras para garantizar una buena reconstrucción. El modelo se vuelve cada vez más difícil de entrenar más allá de esta longitud.

4.2.2.4. Parámetros Clave en el Proceso de Síntesis

Los siguientes parámetros permiten modificar la reconstrucción de las secuencias temporales. Dentro del proceso de entrenamiento se puede modificar el parámetro w_{KL}. Por otra parte, en el *decoder* se puede alterar el parámetro de la temperatura (τ).

- **Parámetro de pérdidas *Kullback-Leibler* (w_{KL})**: aporta la proporción de las pérdidas relativas a divergencia *Kullback-Leibler* con respecto a las pérdidas de reconstrucción. L_R se optimiza para la probabilidad logarítmica del conjunto de trazos que conforman una secuencia temporal. Aún así, esta métrica por si sola no aporta la información suficiente para afirmar que un modelo con L_R más bajo produzca mejores reconstrucciones que un L_R más alto. Esto es debido a que en algunas secuencias temporales los detalles más minuciosos consiguen la diferencia. Por ello es relevante ajustar el parámetro w_{KL}. Un valor de $w_{KL} \approx 1$ reconstruye mejor los generalidades de la secuencia en general sin centrarse en las particularidades, mientras que un valor de $w_{KL} \approx 0{,}25$ se centrará más en pequeños detalles sin tener en cuenta el conjunto de la secuencia. Con ello se puede afirmar que con un L_{KL} más bajo los vectores reconstruidos tienen mayor grado de información de las secuencias temporales que se han introducido que un L_{KL} mayor.
- **Parámetro de la temperatura (τ)**: esta variable también puede aportar cierta incertidumbre a la muestra decodificada. Con ello es posible controlar la aleatoriedad de las nuevas muestras. Es un factor que permite escalar el valor tanto de los parámetros de la distribución categórica como de los parámetros σ de la distribución normal bivariada. Sus valores suelen ser entre 0 y 1. Cuando τ es 0 se trata de un modelo determinista, siendo la salida el punto más probable de la función de densidad de probabilidad.

5

Experimentos y Resultados

En este capítulo se describen y analizan los experimentos llevados a cabo. En la sección 5.1 se detalla el protocolo experimental. A su vez, se van a explicar los distintos experimentos realizados en la sección 5.2. Finalmente, el método propuesto en este TFM ha sido validado en 3 escenarios distintos relacionados con la escritura: bocetos, contraseñas y firma.

5.1. Protocolo Experimental

El protocolo experimental se compone de tres experimentos diferentes. En el primero se reproduce el trabajo inicial de Google con bocetos procedentes de la base de datos *Quick Draw*, entrenando el algoritmo con diferentes clases de esta base de datos. En segundo lugar, se aplica a contraseñas manuscritas utilizando el *dataset eBioDigitDB*. Por último, se va a realizar un análisis del sistema biométrico en firma *on-line* con la base de datos *DeepSignDB*. Además de los detalles relacionados con el protocolo experimental, se describe a continuación para cada base de datos el pre-procesado realizado para obtener los datos en formato *Stroke-3* requerido por el *VAE*.

5.1.1. Quick Draw

Dentro de ***Quick Draw*** [1] se proporcionan los *datasets* ya particionados y procesados para introducirlos en el *VAE*. El único parámetro que hay que variar es N_{max}, la longitud más larga de todas las secuencias temporales. Según se modifica cada clase de bocetos de entrada es necesario ajustar este parámetro. Para ello, se ha propuesto una adaptación automática de este parámetro dentro del *VAE*. Esto se ha realizado en la función que obtiene el formato *Stroke-5* a partir del formato *Stroke-3*. La longitud de las secuencias en *Stroke-5* estaba predeterminada a 250 muestras por ejemplo, siendo ahora un parámetro que adopta la longitud de la secuencia temporal más larga de entrenamiento.

Se han realizado pruebas con distintos *datasets* como *cat.npz* o *pig.npz*. Cada una de las bases de datos cuenta con 75.000 ejemplos para el entrenamiento, 2.500 para la validación y 2.500 para el evaluación final.

5.1.2. eBioDigitDB

Han sido utilizados todos los dígitos de ***eBioDigitDB*** [3], desde la cifra 0 a la 9. El *dataset* aporta las posiciones x e y, el *timestamp* y la presión ejercida. En primer lugar, se han invertido las secuencias temporales respecto al eje de coordenadas debido a que por el sistema de captación éstas contaban con una rotación de 180 grados. El *VAE* toma como entrada (Δx , Δy , *Pen State*). Por ello, se calcula $(\Delta x, \Delta y)$ en cada instante i a partir de (x_i, y_i) y (x_{i-1}, y_{i-1}). Con respecto a la presión, al ser cifras dibujadas con el dedo, el *dataset* solo aporta dos valores: 255 cuando está dibujando el usuario y 0 cuando no. Para el formato *Stroke-3*, hace falta modificar los datos de la presión de 255 a 0 cuando está dibujado el usuario y de 0 a 1 cuando tiene el dedo levantado. Se ha comprobado que la longitud media de las secuencias temporales es de 33.90 muestras siendo la secuencia de tiempo más larga de 166 muestras. Con ello, se verifica como todas las cifras tienen una longitud menor de 300 muestras, como requiere el sistema propuesto. Por último, se ha realizado una división de 5.000 ejemplos de entrenamiento en total, 1.200 ejemplos de validación y 1.200 de test. Dentro de la partición, de cada dígito se tienen 500 ejemplos para entrenamiento, 120 para validación y 120 para evaluación final.

5.1.3. DeepSignDB

DeepSignDB [4] cuenta con dos conjuntos de datos, uno para el desarrollo y otro para la evaluación final del sistema donde los usuarios pertenecientes a cada grupo de datos son diferentes. Se ha utilizado el conjunto de datos de desarrollo para formar el grupo de ejemplos de entrenamiento y validación, utilizando las firmas genuinas pertenecientes a la captación *"Stylus"*. Por otro lado, dentro del conjunto de datos de evaluación se conforma el grupo de firmas que se utilizará para examinar el rendimiento final del sistema con aquellas genuinas pertenecientes a *"Stylus"*. Con ello, se realiza una división de 9.284 firmas de entrenamiento, 2.321 firmas de validación y 10.450 de evaluación final.

Las firmas provienen de los distintos *datasets* que conforman *DeepSignDB* pero todas ellas poseen la información común necesaria para nuestro análisis: para cada instante i la posición (x_i, y_i) y la presión ejercida. Se calcula $(\Delta x, \Delta y)$ en cada instante i a partir de (x_i, y_i) y (x_{i-1}, y_{i-1}). Con respecto a la presión, al ser cifras dibujadas con un lápiz, se aportan diferentes valores cuando está dibujando el usuario y 0 cuando no. Para el formato *Stroke-3*, hace falta modificar los datos de la presión a 0 cuando está dibujado el usuario y a 1 cuando tiene el lápiz levantado. En último lugar, debido a que las firmas enteras suelen tener un tamaño mayor a 300 muestras, se ha decidido dividirlas en diferentes trazos utilizando el método propuesto en la sección 4.2.1. Con ello se obtienen las firmas preparadas para el pre-procesamiento antes de ser introducidas en el *VAE*.

5.2. Resultados

5.2.1. Bocetos

Con respecto a la generación de bocetos se ha replicado el trabajo de Google, comprobando su correcto funcionamiento. Para ello, en primer lugar se ha reproducido el ejemplo mostrado con Google de "aaron-shep.npz" variando los valores de la temperatura. El valor de pérdidas Kullback-Leibler (w_{KL}) se mantiene siempre a 0.5, ya que es el valor que viene por defecto. Se puede observar como cuando el valor de temperatura es más próximo a 0 la reconstrucción es más parecida a la oveja de entrada al sistema, mientras que cuando este valor se va aproximando a 1, se aumenta la variabilidad como se muestra en la figura 5.1.

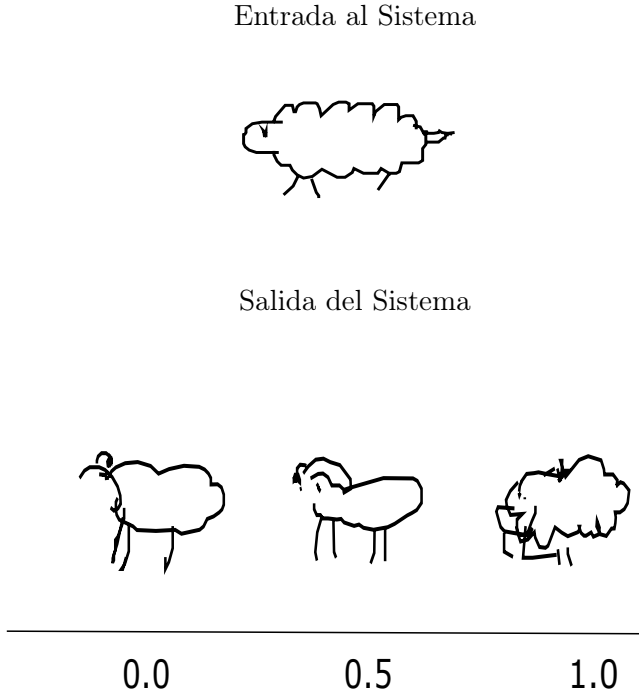

Figura 5.1: "Aaron Sheep" original y muestras reconstruidas tras pasar por el *VAE* con *Wkl* = 0.5 y diferentes valores de τ.

El parámetro τ determina parte de la aleatoriedad en la reconstrucción de las muestras. Con valores más próximos a 0, se trata de reconstruir lo mejor posible la secuencia temporal de la entrada, tratándose de un modelo determinista, siendo la salida el punto más probable de la función de densidad de probabilidad. Cada vez que el parámetro se va acercando al valor 1, el modelo deja de ser tan determinista.

A su vez, se ha probado el modelo con otros *dataset* incluidos en *Quick Draw* como *pig.npz*. Se ha mantenido el parámetro w_{KL} a 0.5 y se ha ido modificando el valor de la temperatura como se observa en la figura 5.2.

En todas las reconstrucciones se puede observar como el *VAE* muestra algunos inconvenientes. En primer lugar, como se ha indicado anteriormente, hay que ajustar el tamaño a N_{max}, la longitud más larga del *dataset* de entrada. Otro inconveniente es que esta técnica no es capaz de generalizar correctamente cuando el entrenamiento se realiza con más de una clase a la vez. En el caso de la figura 5.2 el entrenamiento se está realizando con una sola clase. En la figura 5.3 el algoritmo ha sido entrenado con los dos *datasets* al mismo tiempo, *cat.npz* y *pig.npz*. Se puede observar como al tener varias clases de entrada, las muestras reconstruidas presentan mucha variación, imposibilitando en muchos casos distinguir a que clase pertenece la secuencia

temporal sintética generada.

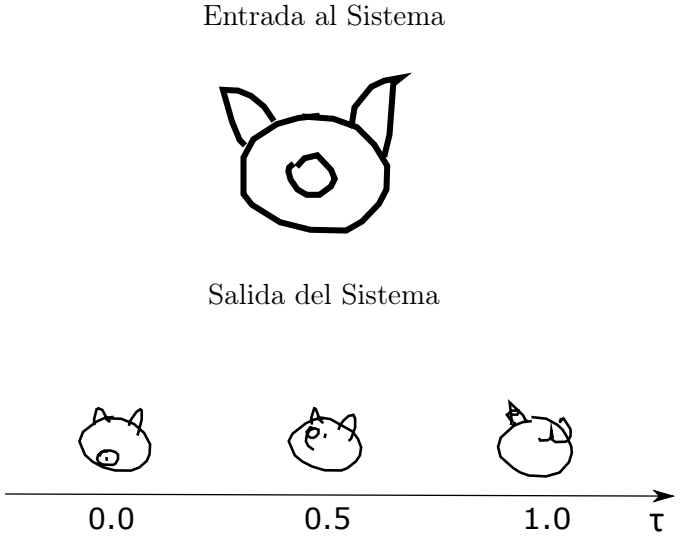

Figura 5.2: "Pig" original y muestras reconstruidas tras pasar por el *VAE* con $Wkl = 0.5$ y diferentes valores de τ.

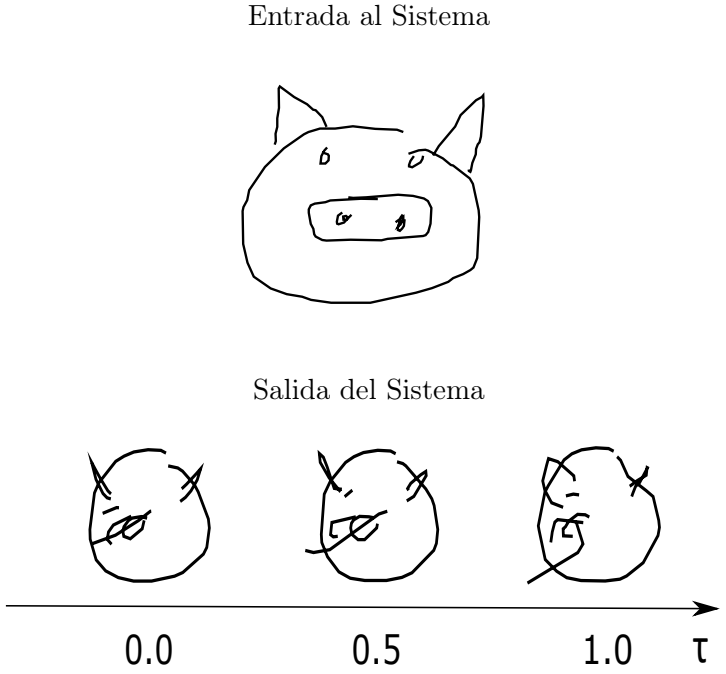

Figura 5.3: Muestra original y muestras reconstruidas tras pasar por el *VAE* con $Wkl = 0.5$ y diferentes valores de τ.

5.2.2. Contraseñas Manuscritas

Dentro de este capítulo han sido utilizados todos los dígitos de ***eBioDigitDB*** [3], desde la cifra 0 a la 9, para todos los entrenamientos del *VAE*. Se ha comprobado que todos los dígitos cumplen la restricción de tener menos de 300 muestras en cada serie temporal. Se han realizado numerosos estudios a partir de la variación de los parámetros w_{KL} y τ analizando visualmente la capacidad del sistema propuesto para generar nuevos dígitos.

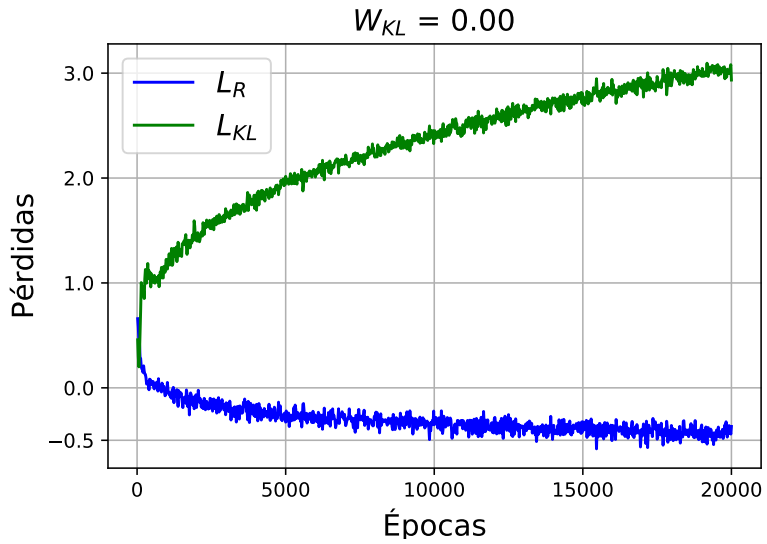

Figura 5.4: Evolución de las componentes de la función de coste del *VAE* para la base de datos *eBioDigitDB* [3] con $w_{KL} = 0$.

Se va a estudiar en primer lugar la función de pérdidas, seguida por la reconstrucción de la secuencia temporal de cada dígito. En el primer grupo las pérdidas afectadas principalmente son L_{KL} mientras que en el segundo grupo son L_R como se ha explicado en la sección anterior.

Se han realizado 3 entrenamientos diferentes dentro de esta sección. Todos ellos cuentan con un valor fijo para la distribución Gaussiana de $\mu = 0$ y $\sigma = 1$ como viene predeterminado. Se ha ido variando el parámetro w_{KL} dentro del *encoder* del *VAE* desde el valor 0.00 hasta el valor 1.00 con un paso de 0.5. En primer lugar, $w_{KL} = 0{,}00$. En este caso, la función de pérdidas $Loss = L_R + w_{KL}L_{KL}$ se resume en $Loss = L_R$ tratándose de un *autoenconder (AE)* puro.

En la figura 5.4 se muestran las pérdidas de un *VAE* con $w_{KL} = 0{,}00$. Estas pérdidas estan compuestas por dos términos, las pérdidas de reconstrucción (L_R) y las pérdidas *Kullback-Leibler* (L_{KL}). Por un lado, se puede observar como las pérdidas de reconstrucción van decreciendo con las épocas ya que muestran la semejanza logarítima de la distribución de probabilidad generada. Con ello se demuestra como según va aumentando el número de épocas, nuestros datos generados serán más semejantes a los datos de entrada del *VAE*. Por otro lado las pérdidas L_{KL} miden la diferencia entre la distribución Gaussiana de media 0 y desviación estándar 1 y la distribución Guassiana de nuestros *Latent Vectors*. A medida que disminuye L_R, el término L_{KL} tiende a aumentar debido a la compensación entre L_R y L_{KL} en la función de pérdidas. Al tratarse de un *VAE* con $w_{KL} = 0{,}00$, estas pérdidas no son relevantes para nuestro estudio ya que nos encontramos ante un *AE*.

Se puede observar en la figura 5.5 como varían las funciones de pérdidas para los distintos valores de w_{KL} seleccionados. A medida que aumenta el valor w_{KL}, se obtienen peores métricas de pérdidas de reconstrucción y aumenta la aleatoriedad en los *Latent Vectors*. Al estudiar la figura 5.5 se aprecia como todas las gráficas cuentan con una distribución similar. Por un lado, las pérdidas L_R disminuyen con las épocas teniendo una organización similar a un *VAE* con $w_{KL} = 0{,}00$. Con el aumento de las épocas de entrenamiento, los datos serán más semejantes a los datos de entrada. Esto es debido a que los parámetros que estamos modificando apenas afectan a estas pérdidas. Por otro lado, cuanto mayor es el término w_{KL} menor son las pérdidas L_{KL}, a causa de que con un w_{KL} próximo a 1 el algoritmo se centra más en los detalles pequeños. Por ello, aunque la reconstrucción no tenga un gran sentido global, los datos de salida se parecen más a los datos de entrada. Por el contrario, cuanto menor sea el parámetro de w_{KL} el entrenamiento

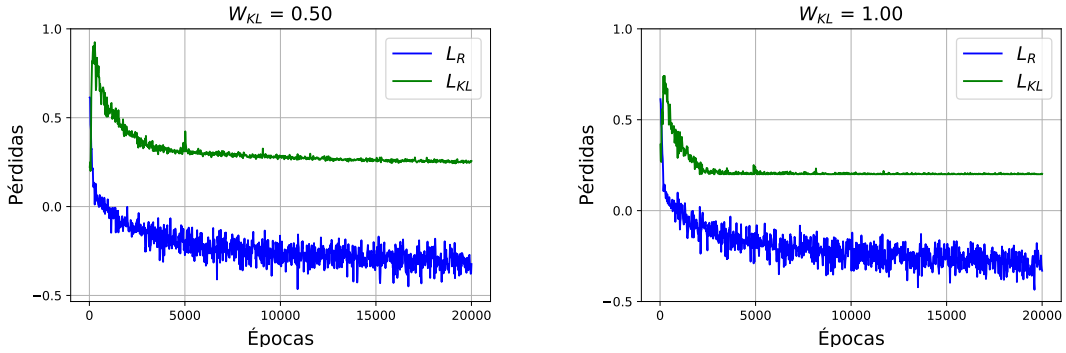

Figura 5.5: Evolución de las componentes de la función de coste del *VAE* para la base de datos *eBioDigitDB* [3] con diferentes valores de w_{KL}.

se centra más en la reconstrucción general. Aunque las pérdidas sean mayores, la reconstrucción en su totalidad tendrá un mayor sentido. La proporción de los términos en la función de pérdidas depende en cada caso del valor asignado al término w_{KL}.

Tras el entrenamiento del *VAE* con valores de w_{KL} desde 0.00 hasta 1.00 en un paso de 0.50, se va a analizar en concreto la reconstrucción de dos dígitos de toda la base de datos que ha sido empleada.

En la figura 5.6 se muestran los dos dígitos bajo estudio, un 8 y un 4. En ella se aprecia una comparación entre el dígito real y sus reconstrucciones variando los parámetros w_{KL} y τ entre los valores 0 y 1 con un paso de 0.5. Para cada una de las reconstrucciones se incluye tanto la imagen como las secuencias temporales X e Y correspondientes. Con las reconstrucciones como imágenes se pueden obtener varias conclusiones. Por un lado, se observa como con el término w_{KL} próximo a 1 varía mucho más la imagen del dígito con respecto a la original. Este parámetro influye aportando más variabilidad en aspectos geométricos como en las curvas o consiguiendo secuencias más largas en el tiempo. Sin embargo, un w_{KL} próximo a 0 reconstruye una imagen más semejante a la original. Con ello se puede afirmar que con un w_{KL} más pequeño la reconstrucción general de la imagen tiene más similitud con la imagen original. A partir de este estudio, se ha determinado que el valor del término w_{KL} no debe adoptar valores cercanos a 1 para una reconstrucción lograda. Por otro lado, se observa que cuanto mayor es la temperatura más aleatoriedad se introduce en las muestras.

Las reconstrucciones de las secuencias temporales en las coordenadas X e Y muestran como a medida que se va aumentando el valor de w_{KL} los trazos son menos abruptos. A su vez, se observa el cambio al introducir el parámetro de aleatoriedad τ. Cuanto mayor es este parámetro menos se parece la secuencia generada sintéticamente a la secuencia temporal original. De igual manera, la longitud de la secuencia suele verse afectada cuando este valor aumenta, obteniendo reconstrucciones menos extensas. Por ello, se ha determinado que valores cercanos a 1 la reconstrucción discerniría en gran medida de la secuencia temporal original.

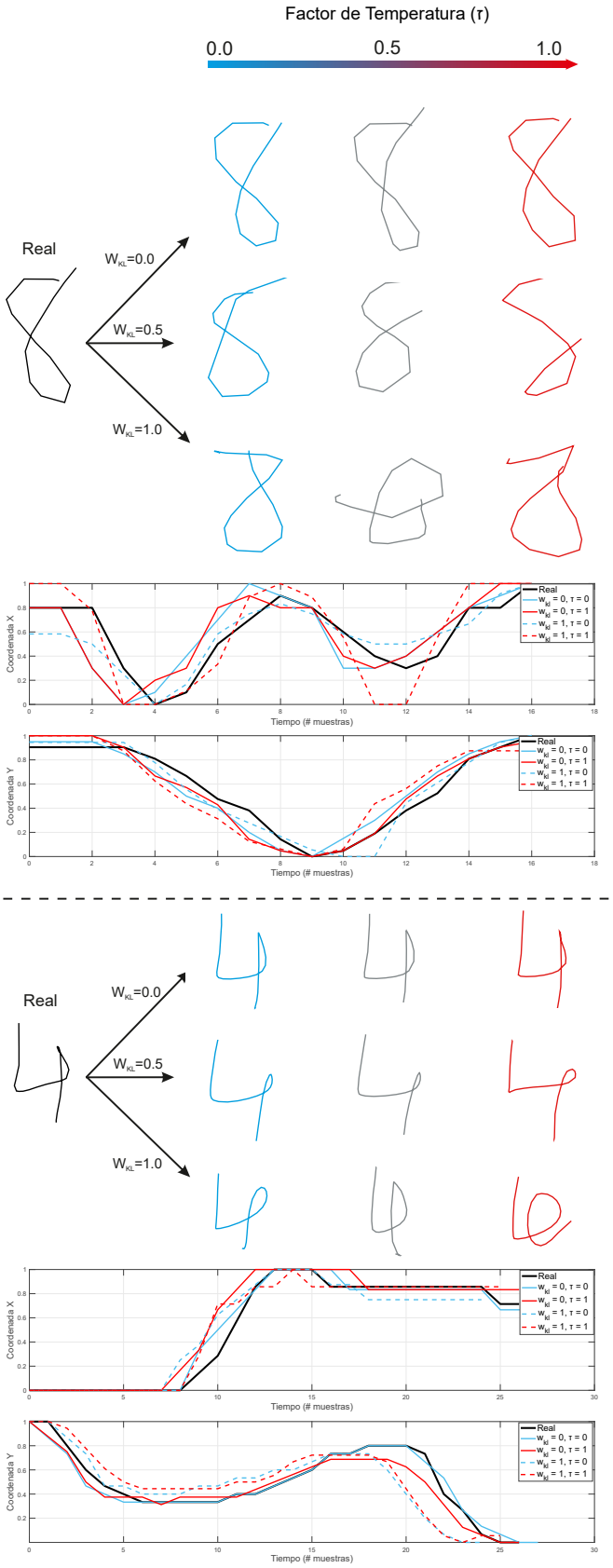

Figura 5.6: Dígitos 8 y 4 originales y reconstruidas tras el método propuesto pertenecientes la base de datos *eBioDigitDB* [3]. Se muestran las imágenes y correspondientes secuencias temporales.

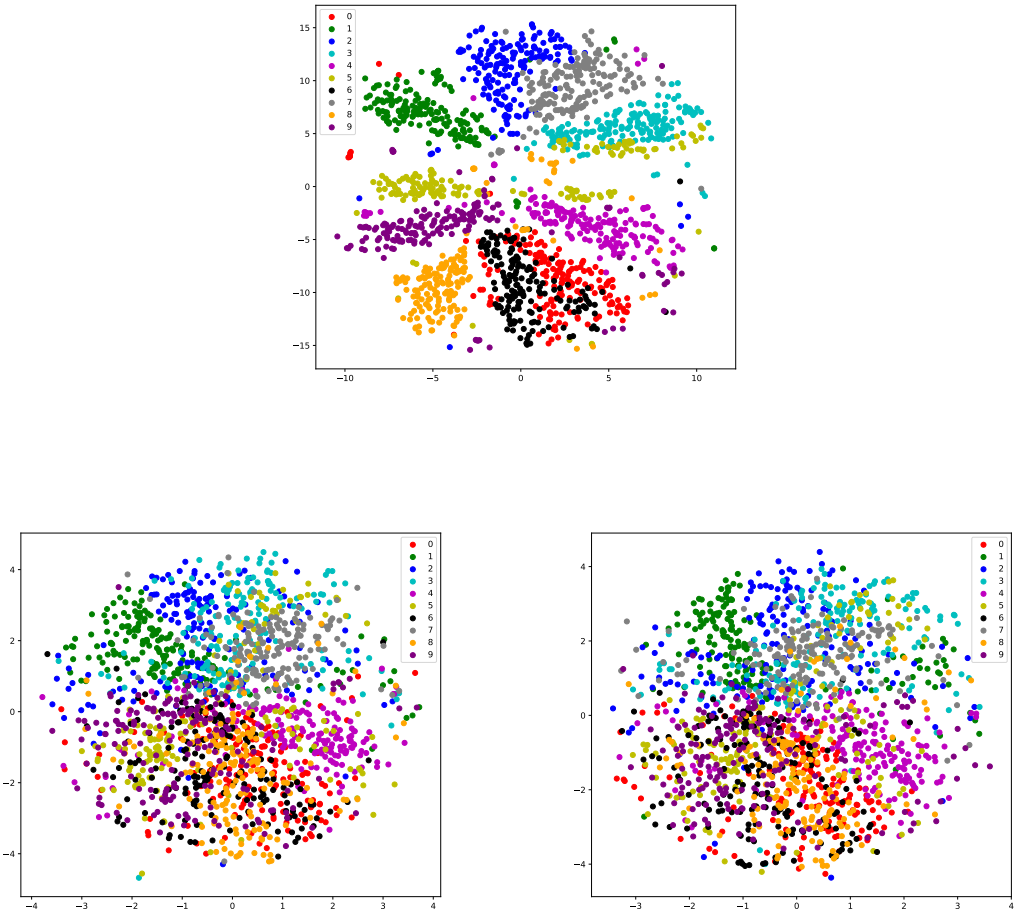

Figura 5.7: Arriba, representación de todos los dígitos de la base de datos *eBioDigitDB* [3] con el algoritmo *t-SNE*. Abajo a la izquierda, la misma representación con los dígitos reconstruidos tras pasar por el método propuesto con $w_{KL} = 0.25$ y $\tau = 0$. Abajo a la derecha, la misma representación con los dígitos reconstruidos tras pasar por el método propuesto con, $w_{KL} = 0.5$ y $\tau = 0$ $w_{KL} = 0.5$ y $\tau = 0$.

Por último, en la figura 5.7 se muestra una distribución con el algoritmo *t-Distributed Stochastic Neighbor Embedding (t-SNE)*, diseñado para la visualización de conjuntos de datos de alta dimensionalidad. En ella se aprecia en primer lugar la distribución de todos los dígitos originales del 0 al 9 como secuencia temporal. En segundo lugar, se muestra la misma distribución pero con los dígitos tras ser entrenado el sistema propuesto en este TFM con $w_{KL} = 0.25$ y $\tau = 0$. En tercer lugar, el sistema entrenado con $w_{KL} = 0.5$ y $\tau = 0$. En la primera imagen se observa como los dígitos están separados entre sí, permitiendo distinguir cada uno de los grupos pertenecientes a cada dígito. La segunda imagen muestra como las clases se encuentran más difusas entre sí pero se puede distinguir como los dígitos pertenecientes a una misma clase están medianamente agrupados en una misma zona. Por último, en la tercera imagen se puede observar como las representaciones de cada uno de los dígitos están más difusas entre sí. Con esto se puede demostrar que al generar sintéticamente secuencias temporales con cierta aleatoriedad no se presentan exactamente las mismas características que poseen los dígitos originales. Aún así, se preserva cierta similitud entre las secuencias temporales pertenecientes a cada clase, lo que ayuda a distinguirlas de otras clases diferentes. Por último, cuanta más aleatoriedad se introduce más difusas se encuentran las clases entre sí.

5.2.3. Firma Manuscrita

Dentro de firma manuscrita ha sido empleada la base de datos ***DeepSignDB*** [4]. Se han utilizado todos los usuarios del grupo *Stylus* como se especifica en el apartado 3.3. Debido a que la longitud de las muestras era superior a 300, se decidió dividir las firmas en trazos utilizando el método propuesto en la sección 4.2.1.

Se han realizado diferentes entrenamientos dentro de esta sección. Al igual que en la sección 5.2.2, todos los entrenamientos cuentan con un valor fijo para la distribución Gaussiana de $\mu = 0$ y $\sigma = 1$ como viene predeterminado. Para los diversos estudios se ha empleado diferentes combinaciones de los parámetros w_{KL} y τ analizando tanto cualitativa como cuantitativamente la capacidad del sistema propuesto para generar nuevas firmas. Las funciones de pérdidas presentan una forma similar a las de la sección 5.2.2.

Tras el entrenamiento del *VAE*, se va a estudiar el conjunto total de muestras de cada firma sintética. Para ello, tras aplicar la transformación de los trazos con el *VAE* entrenado, se produce una reconstrucción total de la firma. Esta reconstrucción se realiza uniendo los trazos transformados sintéticamente junto con los *pens-up* (conjuntos de muestras en los que el lápiz se halla levantado del papel) pertenecientes a la firma original.

En la figura 5.8 se muestra la representación tanto en imagen como en secuencia temporal, con sus coordenadas X e Y, de dos firmas estudiadas. La primera de las firmas presenta una forma similar a la escritura mientras que la segunda se trata de un grafo. El estudio de estas dos firmas diferentes se debe a comprobar la globalización de nuestro sistema. Ambas se muestran junto con sus reconstrucciones con los diferentes valores del parámetro w_{KL} y del factor de temperatura τ variando ambos entre 0 y 1 con un paso de 0.5.

5.2.3.1. Análisis Cualitativo

Gracias a la figura 5.8 se puede llegar a la conclusión de que tanto el parámetro w_{KL} como el factor τ aportan cierta aleatoriedad a las firmas reconstruidas, independientemente de su forma. A su vez, se observa como para una reconstrucción similar a la firma original, tanto el valor de w_{KL} como el de τ no deberían adoptar valores próximos a 1. Esto es debido a que cuando el valor de w_{KL} es mayor, se le da más importancia a las pérdidas de la divergencia *Kullback-Leibler* (L_{KL}) que a las pérdidas de reconstrucción (L_R). Por ello cuanto menor es este valor, la reconstrucción conserva más rasgos de la secuencia temporal original, teniendo la capacidad de generalizar el sistema para *trazos* nunca vistos. Al aumentar el valor de w_{KL} se puede observar como la variabilidad aumenta introduciendo trazos más largos o cortos que los originales y variando en mayor medida los aspectos geométricos. Con respecto al valor del factor de temperatura (τ), aporta cierta aleatoriedad a las muestras, por ello cuando este valor supera cierto umbral la reconstrucción difiere en mayor medida de la secuencia temporal original. Este parámetro afecta mayormente a la posición espacial de la secuencia temporal. A su vez, acorta o alarga el número de muestras de la secuencia temporal. Aplicar un alto valor de los parámetros w_{KL} y τ a la vez modifica en gran medida la firma original, aportando mayor variabilidad intrausuario a la misma. Sin embargo, aumentar demasiado estos valores puede distorsionar mucho la firma, consiguiendo como resultado una mala identificación de la misma.

Los parámetros ajustables han sido aplicados al conjunto de cada secuencia temporal, aportando la misma variabilidad a todos los trazos pertenecientes a una misma firma. Para crear muestras reconstruidas en las que se pretende enfatizar la variabilidad en algunos de sus trazos, se pueden aplicar diferentes valores a los parámetros w_{KL} y τ.

Como se puede observar en ambos casos, el sistema propuesto se puede aplicar en cualquier ámbito de firma y escritura manuscrita de secuencias temporales, ya sea desde bocetos, firmas

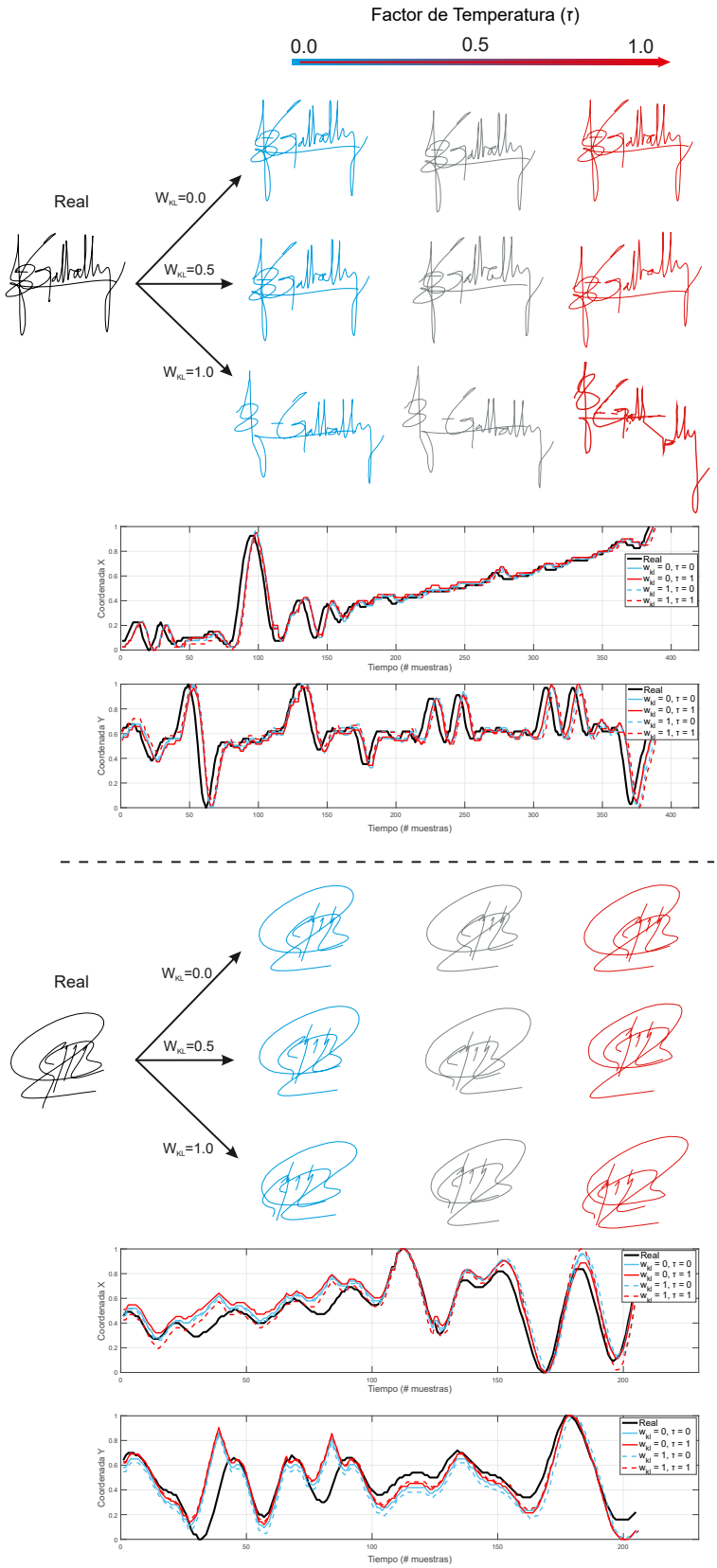

Figura 5.8: Representaciones de dos firmas originales y reconstruidas tras el método propuesto pertenecientes la base de datos *DeepSignDB* [4]. Se muestran las imágenes y correspondientes secuencias temporales.

similares a grafos hasta la propia escritura humana. El modelo creado con firma manuscrita se puede considerar universal, debido a su división en trazos. Con ello se demuestra como un solo *VAE* ha sido entrenado para cualquier clase de entrada aportando resultados notables.

5.2.3.2. Análisis Cuantitativo

Esta sección pretende realizar un análisis cuantitativo del potencial del sistema de síntesis de escritura desarrollado en el presente TFM. En concreto, se analiza un escenario biométrico de verificación de firma *on-line* con escasez de datos. Para ello han sido utilizados dos sistemas. En primer lugar, el sistema de síntesis propuesto en este TFM, dentro del cual se generan firmas sintéticas. En segundo lugar, un sistema de verificación de firma *on-line*[5] creado por *BiDAlab*, el cual permitirá estudiar los beneficios del sistema propuesto.

El sistema de verificación de firma *on-line*[5] utiliza una arquitectura *RNN* bidireccional, con un esquema de una red siamesa. Esta red, tiene 2 entradas con las cuales aprende las similitudes y diferencias entre pares de firmas. La primera entrada siempre es una firma real del usuario, mientras que la segunda firma puede ser real o impostora, generando así comparaciones tanto genuinas como impostoras. Tras la comparación la red aprende a distinguir entre firmas reales e impostoras, siendo la salida del sistema un 0 cuando se encuentra con una comparación genuina y un 1 en caso contrario.

Para dicho estudio se ha empleado la base de datos *DeepSignDB* [4]. Al centrar el trabajo en un escenario de escasez de datos, los usuarios utilizados para el entrenamiento y la validación han sido obtenidos únicamente del grupo de usuarios de desarrollo pertenecientes a *BiosecurID*. En total, 214 usuarios serán utilizados para el entrenamiento, mientras que la validación de los modelos se realizará con 54 usuarios diferentes. Por otro lado, dentro de la evaluación final se han utilizado los 132 usuarios restantes que pertenecen a *BiosecurID*, no utilizados durante el proceso de entrenamiento del sistema. De igual manera, para enfrentarnos a un problema de *Few-Shot* Learning, se ha simplificado el estudio a una única comparación genuina y una comparación impostora por usuario. Con todo ello se puede especificar que en la parte de desarrollo, en el experimento inicial, el entrenamiento cuenta con 428 comparaciones de las cuales 214 son genuinas y 214 impostoras, mientras que la validación cuenta con 108 comparaciones, 54 genuinas y 54 impostoras. Por último, para la evaluación final han sido utilizadas 12.672 comparaciones en total, de las cuales la mitad son genuinas y la mitad impostoras. Las comparaciones de la evaluación final son aquellas que se especifican en la base de datos *DeepSignDB* [4] dentro de las comparaciones con una firma real y una firma impostora por usuario.

Se ha observando los diferentes valores de *Equal Error Rate (EER)* y las curvas *Detection Error Tradeoff (DET)* que nos aporta la *RNN* entrenada con 1) firmas originales y con 2) el conjunto de firmas originales y sintéticas. Finalmente, se han obtenido los resultados con el conjunto de datos de evaluación final, el cual sólo tiene en cuenta firmas reales.

En primer lugar, se han realizado 6 experimentos diferentes utilizando diversas configuraciones de los parámetros ajustables del método propuesto. En cada experimento se generan 3 firmas sintéticas por usuario con el método propuesto por este TFM. Seguidamente, se aplica el sistema de verificación de firma *on-line*. Dentro de este segundo sistema, sólo en el entrenamiento como en la validación se utilizan las firmas generadas sintéticamente. Con ello se aumenta el número de firmas genuinas por usuario. Por último en la evaluación se mantiene durante todos los experimentos el mismo conjunto de firmas reales que se indica en *DeepSignDB*. A continuación se muestran los experimentos desarrollados, indicando en cada caso los parámetros ajustables que afectan al método propuesto en este TFM:

- Una firma original por usuario.

- Una firma original por usuario y 3 generadas sintéticamente con $w_{KL} = 0$ y $\tau = 0$.
- Una firma original por usuario y 3 generadas sintéticamente con $w_{KL} = 0$ y $\tau = 0.5$.
- Una firma original por usuario y 3 generadas sintéticamente con $w_{KL} = 0$ y $\tau = 1$.
- Una firma original por usuario y 3 generadas sintéticamente con $w_{KL} = 0.5$ y $\tau = 0$.
- Una firma original por usuario y 3 generadas sintéticamente con $w_{KL} = 1$ y $\tau = 0$.

El estudio compara el entrenamiento de dicha red neuronal con una firma real por usuario con el conjunto de cuatro firmas por usuario, una real y tres generadas sintéticamente mediante el método propuesto en este TFM. Con estos experimentos se han estudiando las variaciones de los parámetros ajustables comprobando los valores máximos que pueden adoptar dichos parámetros para mejorar el rendimiento del sistema inicial, entrenado sólo con una firma original por usuario.

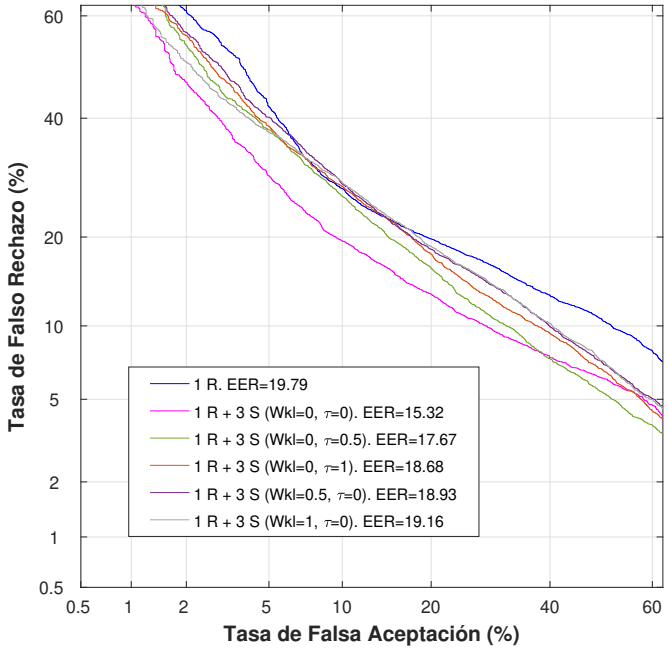

Figura 5.9: Curvas *DET* tras el entrenamiento de un sistema de *RNN* para firmas *on-line* [5] con firmas originales y diferentes configuraciones de w_{KL} y τ. R = Real, S = Sintética. Los resultados mostrados corresponden al conjunto de datos perteneciente a la evaluación final.

Los resultados mostrados en la figura 5.9 han sido evaluados con el conjunto de datos perteneciente a la evaluación final. Como se puede observar en dicha figura, los valores de w_{KL} iguales o superiores a 0.5 consiguen una mejora menor que los valores cercanos a 0. Esto es debido a que las secuencias temporales generadas difieren de las originales, siendo las firmas genuinas sintéticas muy parecidas a las falsificaciones consideradas en esta base de datos. Ocurre lo mismo con valores de τ superiores a 0.5, donde las firmas generadas difieren en mayor medida de las originales que cuando los valores τ son pequeños. Por ello, el sistema presenta una mayor mejora con valores de τ cercanos a 0. El mejor caso es con $w_{KL} = 0$ y $\tau = 0$ donde se ha conseguido una mejora absoluta de un *EER* del 4.47 % al pasar de entrenar con sólo una firma original por usuario a entrenar con una firma original por usuario y 3 generadas sintéticamente con $w_{KL} = 0$ y $\tau = 0$.

Con todo ello, es necesario realizar un estudio con menor variación de los parámetros ajustables. A su vez se ha optado por aumentar el número de firmas sintéticas por usuario, comprobando su aportación al estudio. Visto que el caso que mejor funciona es cuando $w_{KL} = 0$ y $\tau = 0$, se ha decidido acotar el estudio a $w_{KL} = 0$ y 0.25 y $\tau = 0$ realizando los siguientes experimentos:

- Una firma original por usuario.
- Una firma original por usuario y 3 generadas sintéticamente con $w_{KL} = 0$ y $\tau = 0$.
- Una firma original por usuario y 6 generadas sintéticamente con $w_{KL} = 0$ y $\tau = 0$.
- Una firma original por usuario y 10 generadas sintéticamente con $w_{KL} = 0$ y $\tau = 0$.
- Una firma original por usuario y 3 generadas sintéticamente con $w_{KL} = 0.25$ y $\tau = 0$.
- Una firma original por usuario y 6 generadas sintéticamente con $w_{KL} = 0.25$ y $\tau = 0$.
- Una firma original por usuario y 10 generadas sintéticamente con $w_{KL} = 0.25$ y $\tau = 0$.

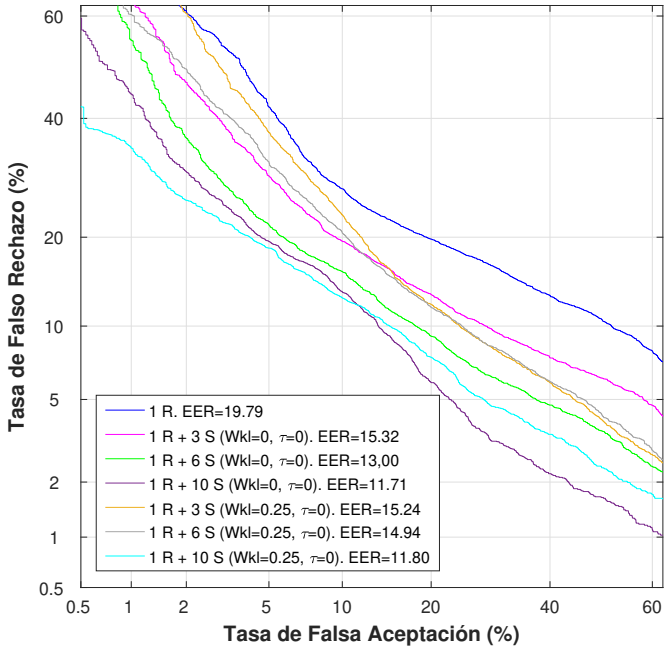

Figura 5.10: Curvas *DET* tras el entrenamiento de un sistema de *RNN* para firmas *on-line* [5] con firmas originales y diferente número de firmas y de configuraciones de w_{KL}. R = Real, S = Sintética. Los resultados mostrados corresponden al conjunto de datos perteneciente a la evaluación final.

Como se puede observar a partir de los datos de la figura 5.10, resultados evaluados con el conjunto de datos perteneciente a la evaluación final, al aumentar el número de firmas generadas sintéticamente se mejora el rendimiento del sistema. Por ello, ante las tres configuraciones aportadas en este estudio tanto para $w_{KL} = 0$ como para $w_{KL} = 0.25$, el número de firmas más idóneo es 10. Por otro lado, se aprecia como la mejor configuración de los parámetros w_{KL} y τ es ambos valores a 0, ya que es el experimento que mejores valores nos ha aportado. Con la mejor configuración, 10 firmas con $w_{KL} = 0$ y $\tau = 0$ se ha llegado a obtener un *EER* del 11.71 %, mejorando en 8.08 % el sistema original.

6

Conclusiones y Trabajo Futuro

6.1. Conclusiones

En este trabajo se ha estudiando e implementado una nueva técnica para la generación sintética de firma y escritura manuscrita dinámica a través de redes neuronales profundas.

Un rasgo muy importante a la hora del estudio es la variabilidad intraclase. Los métodos implantados hoy en día requieren de un número considerable de ejemplos del mismo conjunto para estudiar las características que comparten los datos intraclase e interclase. Una gran incógnita a la hora de investigar la firma y escritura manuscrita es la falta de ejemplos. Coleccionar una gran base de datos es algo costoso, sobre todo cuando se requieren numerosos ejemplos de un mismo usuario. Por todo ello, se presenta un método innovador como técnica de *Data Augmentation* para la generación sintética de firma y escritura manuscrita.

El método propuesto consta de dos módulos principales: el módulo de segmentación en trazos y el módulo de síntesis. En primer lugar, el módulo de segmentación de trazos, dentro del cual se adecúan las secuencias temporales para el posterior módulo. En él se dividen las secuencias en tramos de menor número de muestras, consiguiendo crear un modelo que permita generar sintéticamente cualquier muestra temporal manuscrita. En segundo lugar, el módulo de síntesis, en el cual se describe el *Variational Autoencoder* a utilizar, su entrenamiento y los parámetros ajustables. Gracias a dicho método se consigue captar gran parte de la variabilidad de las muestras generadas sintéticamente. A su vez, se trata de un sistema entrenado a nivel de trazos, es decir, es adaptable a cualquier usuario y cualquier situación. Desde nuestro conocimiento, se trata del primer enfoque de síntesis propuesto para contraseñas manuscritas y firmas *on-line* a través de aprendizaje profundo.

Este ha sido el primer enfoque de síntesis en línea propuesto para firma manuscrita y contraseña a través de aprendizaje profundo.

Se han realizado varias pruebas con el método propuesto demostrando buenos resultados visuales en la generación sintética de firma y contraseñas manuscritos. A su vez, se han modificando diferentes parámetros dentro del *Variational Autoencoder* aportando distintos grados de variabilidad a las muestras generadas.

Por último, el método propuesto ha sido utilizado para entrenar una red neuronal dentro del ámbito de la verificación de firma *on-line*[5]. Para ello se ha empleado la base de datos ***DeepSignDB*** [4], consiguiendo mejorar el *Equal Error Rate (EER)*. El estudio compara el entrenamiento de dicha red neuronal con una firma real por usuario con el conjunto de varias firmas por usuario, una real y el resto generadas sintéticamente con diferentes configuraciones del método propuesto.

Con todo ello, podemos concluir con la aportación de un sistema con el que se ha llegado a una mejora en un sistema de verificación de firma del *EER* del 8.08 % con las configuraciones estudiadas en este TFM.

6.2. Trabajo Futuro

Como trabajo futuro se propone investigar más a fondo sobre el método propuesto. Con ello se pretende conseguir una mejor compresión de como afectan la variación de los parámetros del sistema a la firma y escritura manuscrita.

A su vez, se plantea profundizar en la verificación de firma con diferentes estudios que propongan diversos niveles de variación de los parámetros ajustables.

Otra línea que se puede adoptar en el futuro es la sintetización de falsificaciones realistas con diferentes niveles de calidad.

Otro gran aspecto es, al igual que se ha realizado un pequeño estudio con firma, aplicar el método propuesto como técnica de *Data Augmentation* para escritura manuscrita.

Por último, se propone utilizar el método en otros ámbitos de secuencias temporales biométricas como puede ser el reconocimiento de la actividad humana, el estado de ánimo o la interacción humano-ordenador.

Los resultados obtenidos en este Trabajo Final de Máster han generado un artículo de investigación que será enviado al congreso internacional *35th AAAI Conference on Artificial Intelligence, 2021.*

Bibliografía

[1] D. Ha and D. Eck. A Neural Representation of Sketch Drawings. *Proc. International Conference on Learning Representations*, 2018.

[2] Magenta/sketchRNN. `https://github.com/tensorflow/magenta/`.

[3] R. Tolosana, R. Vera-Rodriguez, and J. Fierrez. BioTouchPass: Handwritten Passwords for Touchscreen Biometrics. *IEEE Transactions on Mobile Computing*, 2019.

[4] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, and J. Ortega-Garcia. DeepSign: Deep On-Line Signature Verification. *arXiv preprint*, 2020.

[5] Ruben Tolosana, Ruben Vera-Rodriguez, Julian Fierrez, and Javier Ortega-Garcia. Exploring Recurrent Neural Networks for On-Line Handwritten Signature Biometrics. *IEEE Access*, pages vol. 6, pp. 5128–5138, 2018.

[6] Qianru Sun, Yaoyao Liu, Tat-Seng Chua, and Bernt Schiele. Meta-Transfer Learning for Few-Shot Learning. *Proc. Conference on Computer Vision and Pattern Recognition*, 2019.

[7] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. *Proc. International Conference on Machine Learning*, 2017.

[8] Yanbin Liu, Juho Lee, Minseop Park, Saehoon Kim, Eunho Yang, Sung Ju Hwang, and Yi Yang. Learning to Propagate Labels: Transductive Propagation Network for Few-Shot Learning. *Proc. International Conference on Learning Representations*, 2019.

[9] C. Shorten and T. M. Khoshgoftaar. A Survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*, pages vol. 6, no. 1, 2019.

[10] T.T. Um, F.M.J. Pister, D. Pichler, S. Endo, M. Lang, S. Hirche, U. Fietzek, and D. Kulic. Data Augmentation of Wearable Sensor Data for Parkinson's Disease Monitoring using Convolutional Neural Networks. *Proc. ACM International Conference on Multimodal Interaction*, 2017.

[11] G. E. A. P. A. Batista, X. Wang, and E. J. Keogh. A Complexity-Invariant Distance Measure for Time Series. *Proc. SIAM International Conference on Data Mining*, 2011.

[12] H. Din, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh. Querying and Mining of Time Series Data: Experimental Comparison of Representations and Distance Measures. *Proc. of the VLDB Endowment*, 2008.

[13] Y. Zhen, Q. Li, E. Che, Y. Ge, and J. L. Zhao. Time Series Classification using Multi-Channels Deep Convolutional Neural Networks. *Lecture Notes in Computer Science*, pages vol. 1, no. 2, pp. 1542–1552, 2014.

[14] B. Zha, H. L, S. Che, J. Liu, and D. Wu. Multi-Scale Convolutional Neural Networks for Time Series Classification. *Journal of Systems Engineering and Electronics*, pages vol. 28, no. 1, pp. 162–169, 2017.

[15] R. Tolosana, J. Gismero-Trujillo, R. Vera-Rodriguez, J. Fierrez, and J. Ortega-Garcia. MobileTouchDB: Mobile Touch Character Database in the Wild and Biometric Benchmark. *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2019.

[16] S. Russell and P. Norvig. Artificial Intelligence: A Modern Appproach. *3rd ed., Pearson, Harlow*, page 2013.

[17] E. Alpaydın. Introduction to Machine Learning. *3rd ed., The MIT Press*, 2014.

[18] T.M. Mitchell. The Discipline of Machine Learning. *Mach. Learn. 17*, pages 1–7, 2006.

[19] Barret Zoph, Ekin D. Cubuk, Golnaz Ghiasi, Tsung-Yi Lin, Jonathon Shlens, and Quoc V. Le. Learning Data Augmentation Strategies for Object Detection. *arXiv.org*, 2019.

[20] G. Forman and I. Cohen. Learning from Little: Comparison of Classifiers Given Little Training. *Proc. European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2004.

[21] R. Lanouette, J. Thibault, and J.L. Valade. Process Modeling with Neural Networks using Small Experimental Datasets. *Comput. Chem. Eng. 23*, 1999.

[22] Guosheng Hu, Xiaojiang Peng, Yongxin Yang, Timothy M. Hospedales, and Jakob Verbeek. Frankenstein: Learning Deep Face Representations using Small Data. *IEEE Transactions on Image Processing*, page 293–303, 2017.

[23] Torgyn Shaikhina, Dave Lowe, Sunil Daga, David Briggs, Robert Higgins, and Natasha Khovanova. Decision Tree and Random Forest Models for Outcome Prediction in Antibody Incompatible Kidney Transplantation. *Biomedical Signal Processing and Control*, page 456–462, 2019.

[24] Ying Zhang and Chen Ling. A Strategy to Apply Machine Learning to Small Datasets in Materials Science. *npj Computational Materials*, pages 28–33, 2018.

[25] Miguel A. Ferrer, Moises Diaz, Cristina Carmona-Duarte, and Aythami Morales. A Behavioral Handwriting Model for Static and Dynamic Signature Synthesis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 39(6), 1041–1053, 2017.

[26] D. Lofaro, S. Maestripieri, R. Greco, T. Papalia, D. Mancuso, D. Conforti, and R.Bonofiglio. Prediction of Chronic Allograft Nephropathy using Classification Trees, Transplant. *Transplantation Proceedings*, 2010.

[27] H. I. Fawazand G. Forestie, J. Weberand L. Idoumgha, and P.-A. Muller. Data Augmentation using Synthetic Data for Time Series Classification with Deep Residual Networks. *Arxiv*, 2018.

[28] Junsuk Choe, Song Park, Kyungmin Kim, Joo Hyun Park, Dongseob Kim, and Hyunjung Shim. Face Generation for Low-Shot Learning Using Generative Adversarial Networks. *Proc. IEEE International Conference on Computer Vision Workshops*, 2017.

[29] Karl Weiss, Taghi M. Khoshgoftaar, and DingDing Wang. A Survey of Transfer Learning. *Journal of Big Data*, 2010.

[30] Yu-Xiong Wang, Ross Girshick, Martial Hebert, and Bharath Hariharan. Low-Shot Learning from Imaginary Data. *arXiv*, 2018.

[31] G. Forestier, F. Petitjean, H. Dau, G. I. Webb, and E. Keogh. Generating Synthetic Time Series to Augment Sparse Datasets. *Proc. IEEE International Conference on Data Mining*, 2017.

[32] Spyros Gidaris, Ponts Paristech, and Nikos Komodakis. Dynamic Few-Shot Visual Learning without Forgetting: Supplementary Material. *Proc. Conference on Computer Vision and Pattern Recognition*, 2018.

[33] Rui Gao, Xingsong Hou, Jie Qin, Jiaxin Chen, Li Liu, Fan Zhu, Zhao Zhang, and Ling Shao. Zero-VAE-GAN: Generating Unseen Features for Generalized and Transductive Zero-Shot L. *Proc. IEEE Transactions on Image Processing*, 2020.

[34] J. Snell, K. Swersky, and R. S. Zemel. Prototypical Networks for Few-Shot Learning. *Proc. NIPS*, 2017.

[35] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H.S. Torr, and Timothy M. Hospedales. Learning to Compare: Relation Network for Few-Shot Learning. *Proc. Conference on Computer Vision and Pattern Recognition*, 2018.

[36] B. N. Oreshkin, P. Rodrıguez, and A. Lacoste. TADAM: Task Dependent Adaptive Metric for Improved Few-Shot Learning. *Proc. Annual Conference on Neural Information Processing Systems*, 2018.

[37] N. Mishra, M. Rohaninejad, X. Chen, and P. Abbeel. Snail: A Simple Neural Attentive Meta-Learner. *Proc. International Conference on Learning Representations*, 2018.

[38] Sachin Ravi and Hugo Larochelle. Optimization as a Model for Few-Shot Learning. *Conference on Learning Representations*, 2017.

[39] Alex Nichol, Joshua Achiam, and John Schulman. On First-Order Meta-Learning Algorithms. *arXiv*, 2018.

[40] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching Networks for One Shot Learning. *Advances in Neural Information Processing Systems*, page 3630–3638, 2016.

[41] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. S. Torr, and T. M. Hospedales. Learning to Compare: Relation Network for Few-Shot Learning. *Proc. Conference on Computer Vision and Pattern Recognition*, 2018.

[42] T. Munkhdalai, X. Yuan, S. Mehri, and A. Trischler. Rapid Adaptation with Conditionally Shifted Neurons. *Proc. International Conference on Machine Learning*, 2018.

[43] Priyanka Gupta, Pankaj Malhotra, Lovekesh Vig, and Gautam Shroff. Transfer Learning for Clinical Time Series Analysis using Recurrent Neural Networks. *ArXiv*, 2018.

[44] Arthur Le Guennec, Simon Malinowski, and Romain Tavenard. Data Augmentation for Time Series Classification using Convolutional Neural Networks. *halshs.archives-ouvertes.fr*, 2016.

[45] M. Alzantot, S. Chakraborty, and M. Srivastava. SenseGen: A Deep Learning Architecture for synthetic Sensor Data Generation. *Proc. IEEE International Conference on Pervasive Computing and Communications Workshops*, 2017.

[46] J. Lemley, S. Bazrafkan, and P. Corcoran. Smart Augmentation Learning an Optimal Data Augmentation Strategy. *IEEE Access, 5:5858–5869*, 2017.

[47] T. Tran, T. Pham, G. Carneiro, L. Palmer, and I. Reid. A Bayesian Data Augmentation Approach for Learning Deep Models. *Advances in Neural Information Processing Systems*, page 2794–2803, 2017.

[48] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative Adversarial Nets. *Proc. Advances in Neural Information Processing System*, 2014.

[49] H. Zhang, T. Xu, H. Li, S. Zhang, X. Huang, X. Wang, and D. Metaxas. Stackgan: Text to Photo-Realistic Image Synthesis with Stacked Generative Adversarial Networks. *arXiv*, 2016.

[50] Yuchen Jiang, Bin Zhu, and Qi Ma. Data Augmentation Based on Wasserstein Generative Adversarial Nets under Few Samples. *Proc. in IOP Conference Series: Materials Science and Engineering*, 2020.

[51] Tero Karras, Samuli Laine, and Timo Aila. A Style-Based Generator Architecture for Generative Adversarial Networks. *Proc. Conference on Computer Vision and Pattern Recognition*, 2019.

[52] Ekin D. Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V. Le. Dataset Augmentation in Feature Space. *Proc. International Conference on Learning Representations*, 2017.

[53] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. *Proc. International Conference on Learning Representations*, 2014.

[54] Sajad Norouzi, David J. Fleet, and Mohammad Norouzi. Exemplar VAEs for Exemplar based Generation and Data Augmentation. *arXiv*, 2020.

[55] Guillaume Alain, Yoshua Bengio, Li Yao, Jason Yosinski, Eric Thibodeau-Laufer, Saizheng Zhang, and Pascal Vincent. GSNs : Generative Stochastic Networks. *Information and Inference*, 2016.

[56] Ekin D. Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V. Le. Pixel Recurrent Neural Networks. *Proc. International Conference on Machine Learning*, 2016.

[57] Ekin D. Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V. Le. Autoaugment: Learning Augmentation Strategies from Data. *Proc. Conference on Computer Vision and Pattern Recognition*, 2019.

[58] N.V. Chawla, K.W. Bowyer, L.O. Hall, and W.P. Kegelmeyer. SMOTE: Synthetic Minority Over-Sampling Technique. *J. Artif. Intell. Res*, pages 16, 321–357, 2017.

[59] G. Douzas and F. Bacao. Geometric SMOTE a Geometrically Enhanced Drop-In Replacement for SMOTE. *Inf. Sci.*, pages 501, 118–135., 2019.

[60] B. Zhao, H. Lu, S. Chen, J. Liu, and D. Wu. Multi-Scale Convolutional Neural Networks for Time Series Classification. *Journal of Systems Engineering and Electronics*, pages 28(1), 162–169, 2017.

[61] Krzysztof Kamycki, Tomasz Kapuscinski, and Mariusz Oszust. Data Augmentation with Suboptimal Warping for Time-Series Classification. *Sensors (Switzerland)*, 2020.

[62] T. Handhika, Murni. D.P. Lestari, and I. Sari. Multivariate Time Series Classification Analysis: State-of-the-art and Future Challenges. *J. Phys. Conf. Ser. Mater. Sci. Eng.*, 2019.

[63] T. Kapuscinski, M. Oszust, M. Wysocki, and D. Warchol. Recognition of Hand Gestures Observed by Depth Cameras. *Int. J. Adv. Rob. Syst*, 2015.

[64] N. Kanda, R. Takeda, and Y. Obuchi. Elastic Spectral Distortion for Low Resource Speech Recognition with Deep Neural Networks. *Proc. IEEE Automatic Speech Recognition and Understanding Workshop*, 2013.

[65] A. Ragni, K. M. Knill, S. P. Rath, and M. J. F. Gales. Data Augmentation for Low Resource Languages. *Proc. Annual Conference of the International Speech Communication Association*, 2014.

[66] N. Jaitly and G. Hinton. Vocal Tract Length Perturbation (VTLP) Improves Speech Recognition. *Proc. International Conference on Machine Learning*, 2013.

[67] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, and A. Ng. Deep Speech: Scaling Up End-to-end Speech Recognition. *arXiv*, 2014.

[68] C. Kim, A. Misra, K. Chin, T. Hughes, A. Narayanan, T. Sainath, and M. Bacchiani. Generation of Large-Scale Simulated Utterances in Virtual Rooms to Train Deep-Neural Networks for Far-Field speech Recognition in Google Home. *Proc. Annual Conference of the International Speech Communication Association*, 2017.

[69] R. Prabhavalkar, R. Alvarez, C. Parada, P. Nakkiran, and T. N. Sainath. Automatic Gain Control and Multi-Style Training for Robust Small-Footprint Keyword Spotting with Deep Neural Networks. *Proc. International Conference on Acoustics, Speech, and Signal Processing*, 2015.

[70] R. Prabhavalkar, R. Alvarez, C. Parada, P. Nakkiran, and T. N. Sainath. Automatic Gain Control and Multi-Style Training for Robust Small-Footprint Keyword Spotting with Deep Neural Networks. *Proc. International Conference on Acoustics, Speech, and Signal Processing*, 2015.

[71] Daniel S. Park, William Chan, Yu Zhang, Chung Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le. SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition. *Proc. of the Annual Conference of the International Speech Communication Association*, 2019.

[72] S. Haradal, H. Hayashi, and S. Uchida. Biosignal Data Augmentation Based on Generative Adversarial Networks. *Proc. Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 2018.

[73] M.M. Krell, A. Seeland, and S.K Kim. Data Augmentation for Brain-Computer Interfaces: Analysis on Event-Related Potentials Data. *ArXiv*, 2018.

[74] Varshaneya V, S Balasubramania, and Vineeth N Balasubramanian. Teaching GANs to Sketch in Vector Format. *arXiv*, 2019.

[75] Chandra Sekhar Vorugunti, Prerana Mukherjee, and Viswanath Pulabaigari. Online Signature Profiling Using Generative Adversarial Networks. *Proc. International Conference on COMmunication Systems and NETworkS*, 2020.

[76] I. M.vBaytas, C. Xiao, X. Zhang, F. Wang, A. K. Jain, and J. Zhou. Patient Subtyping Via Time-Aware LSTM Networks. *Proc. International Conference on Knowledge Discovery and Data Mining*, 2017.

[77] Jifei Song, Kaiyue Pang, Yi-Zhe Song, Tao Xiang, and Timothy Hospedales. Learning to Sketch with Shortcut Cycle Consistency. *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[78] Xingyuan Wu, Yonggang Qi, Jun Liu, and Jie Yang. Sketchsegnet: A Rnn Model for Labeling Sketch Strokes. *Proc. IEEE International Workshop on Machine Learning for Signal Processing*, 2018.

[79] Y. Qi and Z.-H. Tan. SketchSegNet+: An End-to-End Learning of RNN for Multi-Class Sketch Semantic Segmentation. *IEEE Access*, pages 7, 102717–102726, 2019.

[80] Meijuan Ye, Shizhe Zhou, and Hongbo Fu. DeepShapeSketch : Generating Hand Drawing Sketches from 3D Objects. *Proc. International Joint Conference on Neural Networks*, 2019.

[81] N. Cao, X. Yan, YShi, and C. Chen. AI-Sketcher: A Deep Generative Model for Producing High-Quality Sketches. *Proc. AAAI Conference on Artificial Intelligence*, 2019.

[82] Taichi Sumi, Brian Kenji Iwana, Hideaki Hayashi, and Seiichi Uchida. Modality Conversion of Handwritten Patterns by Cross Variational Autoencoders. *Proc. International Conference on Document Analysis and Recognition*, 2019.

[83] H. Kvamme, N. Sellereite, K. Aas, and S. Sjursen. Predicting Mortgage Default using Convolutional Neural Networks. *Expert Systems with Applications 102*, pages 207–217, 2018.

[84] J. Dahmen, D., and Cook. SynSys: A Synthetic Data Generation System for Healthcare Applications. *Proc. Sensors*, 2019.

[85] Cristóbal Esteban, Stephanie L. Hyland, and Gunnar Rätsch. Real-valued (Medical) Time Series Generation with Recurrent Conditional GANs. *Retrieved from http://arxiv.org/abs/1706.02633*, 2017.

[86] Mikołaj Bińkowski, Jeff Donahue, Sander Dieleman, Aidan Clark, Erich Elsen, Norman Casagrande, Luis C. Cobo, and Karen Simonyan. High Fidelity Speech Synthesis with Adversarial Networks. *arxiv.org*, page 1–15, 2019.

[87] A. Graves. Generating Sequences With Recurrent Neural Networks. *arXiv preprint arXiv:1308.0850*, 2013.

[88] Anil K. Jain, Karthik Nandakumar, and Arun Ross. 50 Years of Biometric Research: Accomplishments, Challenges, and Opportunities. *Pattern Recognition Lett.*, pages vol. 79, pp. 80–105, 2016.

[89] D. Impedovo and G. Pirlo. Automatic Signature Verification: The State of the Art. *EEE Transactions on Systems, Man, and Cybernetics*, page pp. 609–635, 2008.

[90] M. Diaz, M. A. Ferrer, D. Impedovo, M. I. Malik, G. Pirlo, and R. Plamondon. A Perspective Analysis of Handwritten Signature Technology. *Proc. ACM Computing Surveys*, 2019.

[91] Rejean Plamondon and Sargur N. Srihar. On-Line and Off-Line Handwriting Recognition: A Comprehensive Survey. *1EEE Transactions on Pattern Analysis and Machine Learning*, pages vol. 22, no. 1, 2000.

[92] M. A. Ferrer, M. Diaz-Cabrera, and A. Morales. Static Signature Synthesis: A Neuromotor Inspired Approach for Biometrics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 37(3), 667–680, 2015.

[93] Enrique Frias-Martinez, Angel Sanchez, and Jose Velez. Support Vector Machines versus Multi-LayerPerceptrons for Efficient Off-Line Signature Recognition. *Engineering Applications of Artificial Intelligence*, pages vol. 19, no. 6, pp. 693–704, 2006.

[94] A. O. Thomas, A. Rusu, and V. Govindaraj. Synthetic Gandwritten CAPTCHAs. *Proc. Pattern Recog*, 2009.

[95] U. Midic. Automatic Generation of Synthetic Handwriting Samples with Application to CAPTCHA . *Proc. Comput. Inf. Sci. Dep*, 2014.

[96] Ruben Tolosana, Ruben Vera-Rodriguez, and Julian Fierrez. BioTouchPass: Handwritten Passwords for Touchscreen Biometrics. *IEEE Transactions on Mobile Computing*, 2019.

[97] Ruben Tolosana, Ruben Vera-Rodriguez, Julian Fierrez, and Javier Ortega-Garcia. BioTouchPass2: Touchscreen Password Biometrics Using Time-Aligned Recurrent Neural Networks. *IEEE Transactions on Information Forensics and Security*, 2020.

[98] Moises Diaz, Andreas Fischer, Miguel A. Ferrer, and Réjean Plamondon. Dynamic Signature Verification System Based on One Real Signature. *IEEE Transactions on Cybernetics*, pages vol. 48, no. 1, pp. 228–239, 2016.

[99] Songxuan Lai, Lianwen Jin, Luojun Lin, Yecheng Zhu, and Huiyun Mao. SynSig2Vec: Learning Representations from Synthetic Dynamic Signatures for Real-world Verification. *Proc. AAAI Conference on Artificial Intelligence*, 2020.

[100] Ruben Tolosana, Ruben Vera-Rodriguez, Julian Fierrez, and Javier Ortega-Garcia. Reducing the Template Aging Effect in On-Line Signature Biometrics. *IET Biometrics*, 2019.

[101] Ruben Tolosana, Ruben Vera-Rodriguez, Julian Fierrez, and Javier Ortega-Garcia. DeepSign: Deep On-Line Signature Verification. *arXiv preprint arXiv:2002.10119*, 2020.

[102] Mohammad A. U. Khan, Muhammad Khalid Khan Niaz, and Muhammad Aurangzeb Khan. Velocity-Image Model for Online Signature Verification. *IEEE Transactions on Imagen Processing*, pages vol. 15, no. 11, pp. 3540–3549, 2006.