UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

TFG del Grado en Ingeniería Informática

Sign2Text - Transcripción de lenguaje de signos (a nivel de palabra) mediante DL

Presentado por Iván Ruiz Gázquez
en Universidad de Burgos — 30 de junio de 2022
Tutores: Dr. Daniel Urda Muñoz y Dr. Bruno Baruque Zanon

UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

D. nombre tutor, profesor del departamento de nombre departamento, área de nombre área.

Expone:

Que el alumno D. Iván Ruiz Gázquez, con DNI dni, ha realizado el Trabajo final de Grado en Ingeniería Informática titulado título de TFG.

Y que dicho trabajo ha sido realizado por el alumno bajo la dirección del que suscribe, en virtud de lo cual se autoriza su presentación y defensa.

En Burgos, 30 de junio de 2022

Vº. Bº. del Tutor:	Vº. Bº. del co-tutor:
D. nombre tutor	D. nombre co-tutor

Resumen

En este primer apartado se hace una **breve** presentación del tema que se aborda en el proyecto.

Descriptores

Palabras separadas por comas que identifiquen el contenido del proyecto Ej: servidor web, buscador de vuelos, android ...

Abstract

A **brief** presentation of the topic addressed in the project.

Keywords

keywords separated by commas.

Índice general

Índice de figuras

Índice de tablas

Introducción

Descripción del contenido del trabajo y del estrucutra de la memoria y del resto de materiales entregados.

Objetivos del proyecto

La principal motivación del proyecto es:

Mejorar la interacción entre personas que necesitan comunicación por signos y personas que no conocen el lenguaje de signos, intentando aumentar la calidad de vida de las primeras.

Objetivos teóricos

1. Creación de un modelo de DL[1] capaz de clasificar ASL[2] a nivel palabra.

2. Poner en producción un método para que los usuarios puedan probar el modelo de forma libre y transparante.

3. Generar una API de código abierto para que otros desarrolladores puedan crear plataformas de cliente, con el objeto de aumentar la audencia y alcance del proyecto.

4. Liberar y contenedorizar el modelo para su libre distribución.

5. Creación de herramientas que permitan el tratamiento de datos y la transformación entre distintos formatos[3].

6. Estudio del comportamiento, estructura e hiperparametrización de redes neuronales convolucionales.

[1] *Deep Learning*
[2] *American Sign Language*
[3] i.e.: extracción de fotogramas de un video, concatenación de imágenes

7. Aprender a fondo el uso de `PyTorch`[4], un *framework* para acelerar la creación y prototipado de redes neuronales.

8. Exportación del modelo entrenado a un formato estándar que facilite la compatibilidad con el mayor número de librerías en distintos lenguajes de programación.

9. Mantenimiento de una *codebase* que favorezca la integración continua (*linting*[4], *formatting*[5] y *type-checking*[6]) de código, siguiendo así los estándares en contribuciones *Open Source*.

Estos objetivos representan en general la filosofía y los objetivos teóricos del proyecto. Si entramos más en detalle sobre los aspectos técnicos y las *features* que se esperan obtener al finalizar el proyecto, obtenemos los siguientes puntos:

Features **esperadas**

- Debemos ser capaces de inferir resultados del modelo entrenado a tiempo real.

- Se desarrollará un modelo fácilmente escalable y adaptable a distintos dataset de cualquier tamaño.[7]

- Se realizará una fase de *data augmentation* sobre los datos iniciales.

- Se implementarán distintas redes neuronales para los ditintos formatos de datos (imagen y video). Deben ser fácilmente refactorizables y mantenibles.

- Cada vez que se estudie el uso de un nuevo formato de dataset, se creará una nueva red, manteniendo la usabilidad de la anterior intacta.

- A lo largo de las pruebas y entrenamientos de la red, vamos a probar distintos *schedulers*, *optimizers* y *criterions*[8] buscando el que mejor se adecúe al formato de los datos y la estructura de la red.

[4] Voz ingl. Estudio de limpieza, orden, calidad y redundancia
[5] Formatear: Correcta estructura y legibilidad
[6] Comprobación de tipados de variables y funciones
[7] El formato de los datasets debe ser el mismo. Un modelo entrenado para clasificación de imágenes no podrá clasificar en formato video o audio.
[8] Funciones de cálculo de pérdida

- Se mantendrán unas estadísticas a tiempo real de la fase de *trainning* y *test* mediante el uso de `Tensorboard`, una herramienta analítica que permite mantener un *log* de imágenes generadas en la ejecución; así como gráficas de costes y *accuracies*.

Conceptos teóricos

Este proyecto es un proyecto de *machine learning*. Dentro del *machine learning*, es de aprendizaje supervisado. Más concretamente intenta resolver un problema de clasificación mediante deep learning. Las redes neuronales aplicadas para la clasificación serán *ResNets* y CNN (*Convolutional Neural Networks*). Veamos en detalle todos estos terminos en los siguientes apartados.

3.1. Machine Learning

Según Tom Mitchell [5], un programa de ordenador aprende de una experiencia E con respecto a alguna tarea T si su medida de desempeño P (del inglés *performance*) mejora con E.

El uso del machine learning se origina en la búsqueda de soluciones a problemas que no podemos resolver programáticamente. Determinar las soluciones se centra en la recogida y análisis de datos[9] con la finalidad de buscar relaciones entre ellos.

Para aclarar este concepto, veamos un ejemplo [1]: Imaginemos que debemos estimar el precio de un coche usado pero no tenemos la fórmula exacta para valorar su estado. Lo que si sabemos es que el precio del coche aumenta y disminuye dependiendo de sus propiedades, como la marca, el kilometraje, o el desgaste. No sabemos tampoco cuales de las propiedades

[9] A lo largo del documento nos referiremos a los datos indistintamente como «datos» o «instancias»

tiene más importancia. Sabemos seguro que cuantos más kilómetros tenga el coche, menor será su precio, pero no sabemos a que escala ocurre esto.

Para determinar la solución necesitamos estudiar el precio de coches en el mercado y anotar sus características. Estos datos son los que daremos a nuestro programa para que busque las relaciones entre propiedades e indique el precio óptimo de los coches.

Saliendo del ejemplo, el machine learning se divide en distintos tipos según la cantidad de datos (o supervisión) que otorgamos a nuestro programa.

Unsupervised Learning

El *unsupervised learning* o aprendizaje no supervisado, tiene su principal característica en que es capaz de detectar las relaciones entre los datos sin ayuda de etiquetas o respuestas externas. Si volvemos al ejemplo de los coches, seríamos capaces de determinar el precio de venta sin necesidad de recibir datos de coches reales.

La técnica principal en el aprendizaje no supervisado es el agrupamiento de los datos (o *clustering*) según la similaridad de sus características. Pero hay más técnicas.

Clustering

El clustering consiste en el agrupamiento de datos no etiquetados basándonos en sus similaridades o diferencias. Los algoritmos de *clustering* son usados para procesar datos crudos sin alteraciones en grupos representados por patrones o estructuras de información. Estos algoritmos pueden clasificar en:

- **Mutualmente excluyentes**: En estos algoritmos se estipula que una instancia de datos solo puede existir en un y solo un *cluster* (o grupo). Este tipo de *clustering* también se denomina *«hard» clustering*. Un ejemplo de algoritmo mutualmente excluyente es el algoritmo de *K-means*. En *K-means*, la «K» indica el número de *clusters* basándonos en su centroide[10]. Los puntos, representando a un dato, caeran en

[10] O centro geométrico, es la posición media aritmética entre todos los puntos de una figura

el grupo con el centroide más cercano. Un valor de *K-means* mayor indicará mayor número de agrupamientos.

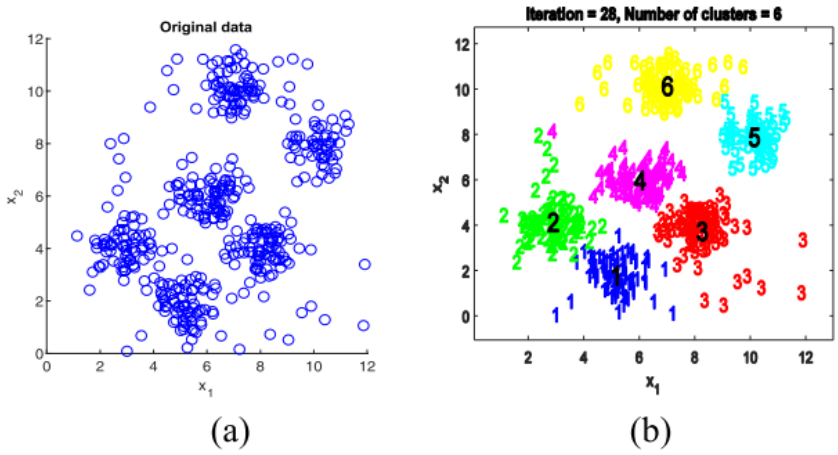

Figura 3.1: Ejemplo de datos antes y después del *clustering* para un algoritmo mutualmente exclutente *K-means* [7].

- **Superpuestos**: En el caso de los algoritmos superpuestos, la diferencia es que un dato puede pertenecer a más de un *cluster* de forma simultánea con distinto grado de pertenencia. Un ejemplo de algoritmo superpuesto es el *«soft» K-means*.
- **Jerárquicos**: También conocido como HCA (*hierarchical cluster analysis*), es un algoritmo no supervisado que se puede categorizar de dos modos, en aglomerativo o divisivo. El primero funciona con acercamiento *«bottom-up»*. Los datos comienzan de forma separada y se van uniendo de forma iterativa según similitud hasta conseguir un único cluster. La similaridad se puede medir de las siguientes formas:

  1. Método de *Ward*: La distancia entre dos clusters se define por el aumento de la distancia cuadrática después de que los clusters sean fusionados.
  2. Promedio de distancias: Este método usa la distancia media entre dos puntos en cada *cluster*.
  3. Distancia máxima: La distancia usada es la distancia mayor entre dos puntos en distintos clusters.
  4. Distancia mínima: Se define por la distancia mínima entre dos puntos de dos clusters diferentes.

La medida más comun a la hora de calcular estas distancias es la distancia euclídea, aunque otra muy común en la literatura es la distancia *Manhattan*. Por otro lado, el acercamiento divisivo, o *«top-down»* funciona dividinendo un único *cluster* basándose en las diferencias entre instancias (puntos del cluster). Visualmente se parece a un dendrograma[11].

- **Probabilísticos**: En un algoritmo probabilístico, los puntos se agrupan basándose en la probabilidad de que pertenezcan a una distribución paramétrica [12]. El GMM (*Gaussian Mixture Model*) es el método más usado en *clustereing* probabilístico.

  - *Gaussian Mixture Model*: Este algoritmo se clasifica como mixto debido a que se compone de un número no especificado de funciones de densidad probabilísitca. Normalmente se encargará de comprobar si un instancia es parte de una distribución Gaussiana o normal. El algoritmo más usado para determinar la distribución es el E-M (Expectation-Maximizarion) [2].

Reglas de asociación

Las reglas de asociación es un método que busca buscar las relaciones entre variables de un *dataset*. Suelen ser usados en *marketing*, sistemas de recomendación, relación entre productos, estudio de hábitos de consumición [6], entre otros. El algoritmo más usado en reglas de asociación es el algoritmo Apriori.

- **Algoritmo Apriori**: Este algoritmo ppermite encontrar de forma frecuente «conjuntos de elementos frecuentes». Estos conjuntos sirven para generar reglas de asociación que permiten extender los sets a sets de mayor tamaño.

[11] Representación gráfica de datos en forma de árbol para organizar datos en categorias y subcategorias hasta alcanzar el nivel de detalle deseado

[12] La estadística paramétrica es una rama de la estadística con la que se deciden valores basándose en distribuciones desconocidas. Estas se determinan según un número finito de parámetros.

Algoritmo 1: Algoritmo Apriori

Input: Para un dataset T, un umbral de soporte ε y un conjunto de datos C_k para el nivel k

```
1  L_1 ← findFrequentOneItemsets(T)
2  k = 2
3  while L_{k-1} not empty do
       /* Generación de Apriori                                  */
4      C_k ← empty list
5      forall p ⊆ L, q ⊆ L do
6          c ← p ∪ q_{k-1}
7          forall u ∈ L do
8              if u ⊆ c then
9                  C_k ← C_k ∪ {c}
10             end if
11         end forall
12     end forall
       /* Cálculo de candidatos                                  */
13     for t ∈ T do
14         D_t ← {c ∈ C_k : c ⊆ t}
15         for c ∈ D_t do
16             count[c] ← count[c] + 1
17         end for
18     end for
19     k ← k + 1
20     L_k ← {c ∈ C_k : count[c] ≥ ε}
21 end while
22 return Union(L_k)
```

Reducción de la dimensionalidad

Normalmente cuando disponemos de una mayor cantidad de datos, tendemos a obtener mejores soluciones, aunque esto tiene impacto en el rendimiento de nuestros algritmos y puede dificultarnos comprender de forma completa nuestro *dataset*. La reducción de la dimensionalidad se usa cuando el número de características o dimensiones de un *dataset* es muy grande. Reduce el número de instancias a un tamaño manejable manteniendo la integridad de los datos lo máximo posible. Algunos métodos de reducción

de la simensionalidad son [9][3]: PCA (del inglés *Principal Component Analysis*), SVD (*Singular Value Decomposition*) y *Autoencoders*.

Semi-supervised Learning

El aprendizaje semi-supervisado se caracteriza por tener la capacidad de aprender de conjuntos de datos cuyas instancias no están completamente etiquetadas. En nuestro ejemplo 3.1«cálculo de precio de un coche según sus propiedades», tendríamos precios reales solo para algunos de los coches, mientras que otros no tendrían una etiqueta establecida.

La característica principal que diferencia el aprendizaje semi-supervisado del aprendizaje no supervisado y del aprendizaje supervisado (que veremos a continuación) es que la capacidad de aprender del algoritmo se nutre del hecho de que algunos datos no estén etiquetados. A la hora de intentar averiguar las etiquetas faltantes en base a las presentes, podemos observar comportamientos de aprendizaje que no se verían en los otros dos tipos de *machine learning*.

De este modo, no es útil solo como algoritmo de aprendizaje si no como algoritmo de generación de nuevos datos en aprendizaje supervisado [11]. Si la obtención de datos y etiquetas nos supone un proceso costoso y/o el acceso a los datos es limitado [13], podemos crear un algoritmo de aprendizaje semi-supervisado para completar la falta de datos.

Algunos de los métodos usados en aprendizaje semi-supervisado son los siguientes:

- Modelos generativos: Con este modelo se busca estimar $p(x \mid y)$, la distribución de los puntos pertenecientes a cada clase. La probabilidad $p(x \mid y)$ de que un punto x pertenezca a una etiqueta y es proporcional a $p(x \mid y) \cdot p(y)$ por la Regla de Bayes.

  Extrayéndolo de [11], los modelos generativos asumen que las distribuciones tienen la forma $p(x \mid y, \theta)$ para metrizada por el vector θ. Si esta suposición es incorrecta, los datos no etiquetados pueden incluso disminuir la capacidad de acierto de la solución. Pero, en el caso contrario, si la suposición es correcta, los datos etiquetados mejoraran el rendimiento del modelo.

[13] i.e.: La recopilación de datos médicos es algo compleja, ya que se debe tener permiso del paciente, así como hay enfermedades cuyos datos son muy escasos y difíciles de conseguir.

Los datos no etiquetados se suelen distribuir mediante una distribución Gaussiana mixta. La distribución de probabilidad conjunta se puede escribir como $p(x, y \mid \theta) = p(y \mid \theta) \cdot p(x \mid \theta)$ mediante el teorema de probabilidad compuesta. Cada vector θ se asocia con una función de decisión:

$$f_\theta(x) = \underset{x}{\text{argmax}} p(y \mid x, \theta)$$

El parámetro se elige entonces basándonos en la adaptación a los datos etiquetados y no etiquetados, usamos un peso de balanceo λ:

$$\underset{\theta}{\text{argmax}} \left(\log p \left(\{x_i, y_i\}_{i=1}^{l} \mid \theta \right) + \lambda \log p \left(\{x_i\}_{i=l+1}^{l+u} \mid \theta \right) \right)$$

- *Support Vector Machines* (SVM): Según Van Engelen *et al.*[10] un SVM es un clasificador que intenta buscar la frontera de decisión que maximice el margen, que a su vez se define como la distancia entre la fontera de decisión y los puntos cercanos a ella[14]. En el escenario del aprendizaje semi-supervisado, un objeto adicional gana importancia: también queremos minimizar el número de datos «no etiquetados» que viola el margen. Como no podemos saber la etiqueta de los datos «no etiquetados», aquellos puntos que violan el margen son penalizados basándonos en su distancia a la frontera de decisión.

- Modelos basados en gráfos: Este modelo asume un grafo $G = V, E$ tal que los vértices V son las instancias de datos (etiquetados y no etiquetados) y E las aristas no dirigidas que conectan las instancias i, j con un peso $w_i j$. La construcción de un grafo se cumple en los siguientes pasos [8][10] 3.2:

  1. **Construcción del grafo**: Se suele asumir una instanciación inicial del grafo aleatoria
  2. ***Weighting***: Cálculo de pesos de los datos etiquetados para la fase de inferencia.
  3. **Inferencia de etiquetas**: La tarea a cumplir aquí es propagar información de las instancias etiquetadas a las «no etiquetadas» incorporando la estructura de grafo creada en el paso anterior.

[14] A veces el margen también se refiere a la zona delimitada por la fontera de decisión en la que caen los puntos

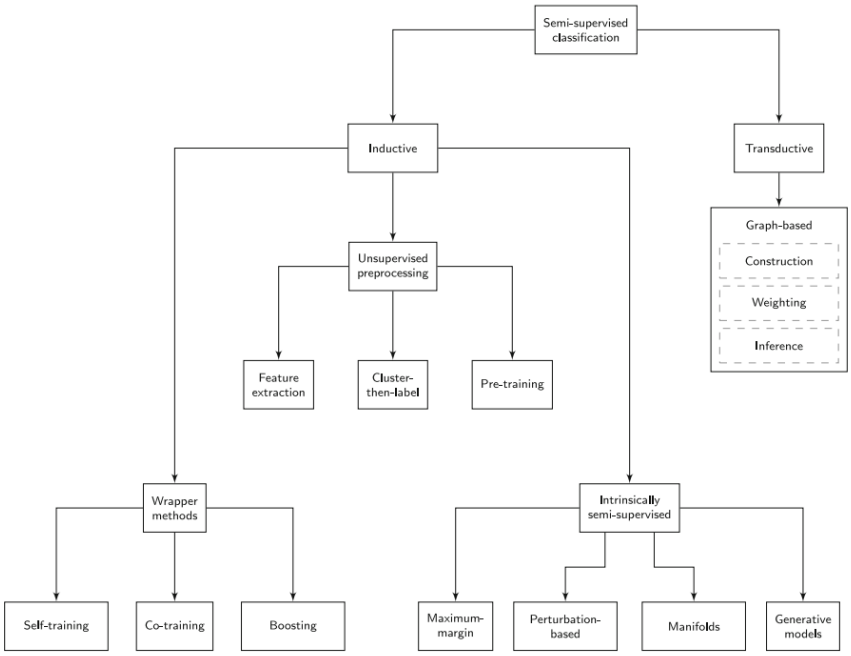

Figura 3.2: Visualización de la taxonomia de los métodos de aprendizaje semi-supervisado. En la rama correspondiente a los métodos basados en grafo podemos observar las distintas fases de clasificación

Supervised Learning

Regression

Classification

1. Redes Neuronales

    - *Perceptron*
    - *Feedforward neural network*
    - *Backpropagation*
    - *Recurrent neural network*
    - *Convolutional neural network*
    - *Autoencoder*

3.2. Carga de datos

Extracción de frames de videos

3.3. Tratamiento de los datos

Normalización

Data Augmentation

Transformación de datos a tensores

Batching

Subsampling

Eliminación de datos innecesarios

3.4. Red Neuronal ResNet

3.5. Estructura de la red convolucional

Capas convolucionales

Downsampling

MaxPooling

BatchNormalization

Flatten layer

Dense layers

Dropout

Lineal layers

Leaky ReLU

ReLU

Capas de salida

Sigmoide

Softmax

3.6. Entrenamiento e hiperparametrización

Pipeline del proceo

Optimizer

Scheduler

3.9. Exportación del modelo

Formatos estándares

ONNX

3.10. Copresión del modelo

Consecuencias

Velocidad

Accuracy

Métodos de compresión

Quantization

Prunning

3.11. Estudio de escalabilidad del modelo

Vamos a comprobar hasta que punto la red neuronal es capaz de mantener la precisión de clasificación con el aumento de las etiquetas. Para todo lo anterior el número de etiquetas a clasificar han sido 8. Vamos a aumentar esto hasta 100 etiquetas.

En la siguiente tabla blabla

Técnicas y herramientas

Esta parte de la memoria tiene como objetivo presentar las técnicas metodológicas y las herramientas de desarrollo que se han utilizado para llevar a cabo el proyecto. Si se han estudiado diferentes alternativas de metodologías, herramientas, bibliotecas se puede hacer un resumen de los aspectos más destacados de cada alternativa, incluyendo comparativas entre las distintas opciones y una justificación de las elecciones realizadas. No se pretende que este apartado se convierta en un capítulo de un libro dedicado a cada una de las alternativas, sino comentar los aspectos más destacados de cada opción, con un repaso somero a los fundamentos esenciales y referencias bibliográficas para que el lector pueda ampliar su conocimiento sobre el tema.

Aspectos relevantes del desarrollo del proyecto

Este apartado pretende recoger los aspectos más interesantes del desarrollo del proyecto, comentados por los autores del mismo. Debe incluir desde la exposición del ciclo de vida utilizado, hasta los detalles de mayor relevancia de las fases de análisis, diseño e implementación. Se busca que no sea una mera operación de copiar y pegar diagramas y extractos del código fuente, sino que realmente se justifiquen los caminos de solución que se han tomado, especialmente aquellos que no sean triviales. Puede ser el lugar más adecuado para documentar los aspectos más interesantes del diseño y de la implementación, con un mayor hincapié en aspectos tales como el tipo de arquitectura elegido, los índices de las tablas de la base de datos, normalización y desnormalización, distribución en ficheros3, reglas de negocio dentro de las bases de datos (EDVHV GH GDWRV DFWLYDV), aspectos de desarrollo relacionados con el WWW... Este apartado, debe convertirse en el resumen de la experiencia práctica del proyecto, y por sí mismo justifica que la memoria se convierta en un documento útil, fuente de referencia para los autores, los tutores y futuros alumnos.

Trabajos relacionados

Este apartado sería parecido a un estado del arte de una tesis o tesina. En un trabajo final grado no parece obligada su presencia, aunque se puede dejar a juicio del tutor el incluir un pequeño resumen comentado de los trabajos y proyectos ya realizados en el campo del proyecto en curso.

Conclusiones y Líneas de trabajo futuras

Todo proyecto debe incluir las conclusiones que se derivan de su desarrollo. Éstas pueden ser de diferente índole, dependiendo de la tipología del proyecto, pero normalmente van a estar presentes un conjunto de conclusiones relacionadas con los resultados del proyecto y un conjunto de conclusiones técnicas. Además, resulta muy útil realizar un informe crítico indicando cómo se puede mejorar el proyecto, o cómo se puede continuar trabajando en la línea del proyecto realizado.

Bibliografía

[1] Ethem Alpaydin. *Machine learning*. MIT Press, 2021.

[2] Chuong B Do and Serafim Batzoglou. What is the expectation maximization algorithm? *Nature biotechnology*, 26(8):897–899, 2008.

[3] Ali Ghodsi. Dimensionality reduction a short tutorial. *Department of Statistics and Actuarial Science, Univ. of Waterloo, Ontario, Canada*, 37(38):2006, 2006.

[4] Pytorch main Mantainer. Pytorch developers guideline and design philosophy, 2022.

[5] Tom M Mitchell and Tom M Mitchell. *Machine learning*. Number 9. McGraw-hill New York, 1997.

[6] Suprianto Panjaitan, Muhammad Amin, Sri Lindawati, Ronal Watrianthos, Hengki Tamando Sihotang, Bosker Sinaga, et al. Implementation of apriori algorithm for analysis of consumer purchase patterns. In *Journal of Physics: Conference Series*, volume 1255, page 012057. IOP Publishing, 2019.

[7] Kristina P Sinaga and Miin-Shen Yang. Unsupervised k-means clustering algorithm. *IEEE access*, 8:80716–80727, 2020.

[8] Zixing Song, Xiangli Yang, Zenglin Xu, and Irwin King. Graph-based semi-supervised learning: A comprehensive review. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[9] Carlos Oscar Sánchez Sorzano, Javier Vargas, and A Pascual Montano. A survey of dimensionality reduction techniques. *arXiv preprint arXiv:1403.2877*, 2014.

[10] Jesper E Van Engelen and Holger H Hoos. A survey on semi-supervised learning. *Machine Learning*, 109(2):373–440, 2020.

[11] Xiaojin Zhu and Andrew B Goldberg. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, 3(1):1–130, 2009.