# Generación de pictogramas con inteligencia artificial para la comunicación aumentativa y alternativa
# Pictogram generation with artificial intelligence for augmentative and alternative communication


**Trabajo de Fin de Grado**
**Curso 2024–2025**

**Autor**
**Alejandro Antuña Rodríguez**

**Directores**
**Gonzalo Méndez Pozo**
**Pablo Gervás Gómez-Navarro**

**Grado en Ingeniería Informática**
**Facultad de Informática**
**Universidad Complutense de Madrid**

# Generación de pictogramas con inteligencia artificial para la comunicación aumentativa y alternativa
# Pictogram generation with artificial intelligence for augmentative and alternative communication

**Trabajo de Fin de Grado en Ingeniería Informática**


## Autor
**Alejandro Antuña Rodríguez**

## Directores
**Gonzalo Méndez Pozo**
**Pablo Gervás Gómez-Navarro**



**Convocatoria:** *Mayo 2025*
**Calificación:** *8.2*



**Grado en Ingeniería Informática**
**Facultad de Informática**
**Universidad Complutense de Madrid**



**26 de Mayo de 2025**

# Dedicatoria

*Hay muchas personas que me han ayudado en mi camino hasta aquí, pero pocas a las que podría dedicar este logro. Sin embargo, mentiría si no me lo dedicara también a mí mismo. Al final, el chico que pasaba noches sin dormir, que pasaba horas en la facultad y que soportaba el estrés de toda esta carrera, fuiste tú, Alejandro.*

*Por eso, muchas gracias, y a por lo siguiente.*

# Agradecimientos

Muchísimas gracias a mi familia por haber estado ahí para mí en mis días buenos, pero sobre todo en mis días malos, esos que no compartes con nadie. Mamá, David, Pablo, Caye: os quiero.

A toda mi gente de la universidad que ha estado ahí: joya, siempre. Pelli, Rúben, Curro, César, Epi, Irem, Jorge, Patín, Omar, Charly, Carlos, gracias por las risas, por las lágrimas, por todo lo que hemos vivido juntos. Gracias por cambiarme tanto, por hacerme mejor en todos los sentidos. *El Prota* en esta historia también sois vosotros.

# Resumen


## Generación de pictogramas con inteligencia artificial para la comunicación aumentativa y alternativa

La generación de pictogramas mediante inteligencia artificial (IA) tiene como objetivo principal apoyar a educadores, médicos, psicólogos y otros profesionales que trabajan con personas con dificultades comunicativas. Al permitir la creación instantánea de pictogramas en diversos estilos a partir de descripciones simples, se elimina la dependencia de bases de datos preexistentes, mejorando significativamente el flujo y las posibilidades de comunicación que estos recursos ofrecen. Este Trabajo de Fin de Grado (TFG) se centra en el desarrollo de LoRAs (*Low-Rank Adaptations*) utilizando tecnologías de software libre, con el propósito de expandir las aplicaciones de la inteligencia artificial generativa de imagen en el ámbito de la Comunicación Aumentativa y Alternativa (CAA).

Para este proyecto, se utilizó la herramienta *Kohya_ss*, que permitió un entrenamiento altamente personalizable de los modelos. Los modelos base seleccionados se fundamentan en *Stable Diffusion*, específicamente la versión 1.5, por su equilibrio entre calidad, velocidad de generación y bajos requisitos de hardware.

Tras numerosos ensayos y ajustes, se lograron entrenar LoRAs que generan pictogramas en dos estilos distintos de manera rápida, coherente y sin necesidad de equipos informáticos de alta gama. Además, se realizó un acercamiento al desarrollo de estos modelos LoRA en tecnologías más potentes como *Stable Diffusion XL*.

Estos avances representan un paso importante hacia una comunicación más inclusiva y eficiente, mejorando la calidad de vida de personas con dificultades comunicativas y facilitando el trabajo de sus cuidadores y profesionales.



## Palabras clave

[ Inteligencia Artificial Generativa de Imagen, Stabble Diffusion, LoRA (Low-Rank Adaptation), Pictogramas, Comunicación Aumentativa y Alternativa (CAA) ]

# Abstract


## Pictogram generation with artificial intelligence for augmentative and alternative communication

The main objective of generating pictograms through AI is to support educators, doctors, psychologists, and other professionals working with individuals with communication difficulties. By allowing the instant creation of pictograms in various styles from simple descriptions, it eliminates the dependency on pre-existing databases, significantly improving the flow and communication possibilities these resources offer. This Bachelor's Thesis (TFG) focuses on the development of LoRAs (*Low-Rank Adaptations*) using free software technologies, with the goal of expanding the applications of generative image artificial intelligence (AI) in the field of Augmentative and Alternative Communication (AAC).

For this project, the *Kohya_ss* tool was used, which allowed for highly customizable model training. The selected base models are based on *Stable Diffusion*, specifically version 1.5, due to its balance between quality, generation speed, and low hardware requirements.

After numerous tests and adjustments, LoRAs were successfully trained to generate pictograms in two distinct styles quickly, consistently, and without the need for high-end computing equipment. Additionally, an approach was made to develop these LoRA models using more powerful technologies such as *Stable Diffusion XL*.

These advances represent an important step toward more inclusive and efficient communication, improving the quality of life for individuals with communication difficulties and facilitating the work of their caregivers and professionals.



## Keywords

[ Generative Image artificial intelligence, Stable Diffusion, LoRA (Low-Rank Adaptation), Pictograms, Augmentative and Alternative Communication (AAC) ]

# Índice

# Índice de figuras

# Índice de tablas

Capítulo 1

# Introducción

## 1.1. Motivación e importancia de los pictogramas en la comunicación

Desde la antigüedad, las civilizaciones han recurrido al uso de símbolos e imágenes para expresar ideas. Ejemplos como los jeroglíficos egipcios o el arte rupestre paleolítico muestran cómo los seres humanos han utilizado la representación visual para comunicar conceptos complejos sin necesidad de un lenguaje hablado. Estos sistemas pictográficos sentaron las bases de la comunicación visual, que en muchas culturas precedió al desarrollo de lenguajes escritos formales.

En la actualidad, los pictogramas juegan un papel crucial en la comunicación aumentativa y alternativa (CAA), especialmente para personas con dificultades cognitivas o interpretativas. Diversos estudios han demostrado su utilidad en mejorar la comunicación de personas con condiciones como el síndrome de Down, el autismo o el deterioro cognitivo asociado a la vejez (Mirenda (2003); Lorah et al. (2015); Beukelman y Mirenda (2013)). Estas herramientas permiten a las personas expresar sus necesidades y comprender su entorno de manera más efectiva.

Sin embargo, el principal inconveniente de los pictogramas es que, para cada idea, se requiere uno específico, y disponer de una base de datos lo suficientemente amplia como para abarcar cualquier concepto imaginable resulta prácticamente imposible.

Es aquí donde surge la idea de este TFG: aprovechar la inteligencia artificial generativa para crear pictogramas personalizados en tiempo real, adaptados a cada situación concreta. Este enfoque no solo facilitará la comunicación en escenarios cotidianos, sino que también promete empoderar a educadores, terapeutas y cuidadores, proporcionándoles una herramienta versátil y accesible que puede transformar el modo en que interactúan con personas con necesidades comunicativas especiales.

## 1.2. Objetivos

El objetivo principal de este proyecto es desarrollar un sistema basado en inteligencia artificial capaz de generar pictogramas de manera rápida, eficiente y consistente, siguiendo el estilo visual de los pictogramas del centro aragonés ARASAAC[1], (ver figura 1.1). Este sistema debe ser accesible y flexible, permitiendo la creación de pictogramas a partir de descripciones simples y en tiempo real, sin depender de bases de datos preexistentes. Para ello, se utilizarán tecnologías de software libre, garantizando tanto la gratuidad del sistema como su libertad de uso y adaptación.

(a) (b) (c)

Figura 1.1: Pictogramas del centro ARASAAC, estilo objetivo de generación de este proyecto.

### Objetivos secundarios

- Garantizar que el sistema sea compatible con dispositivos de recursos computacionales limitados, manteniendo una alta calidad en la generación de pictogramas.
- Asegurar la capacidad del sistema para adaptarse a diversas necesidades de la Comunicación Aumentativa y Alternativa, como el uso de paletas de colores simples, la especificación de colores en objetos o la inclusión de elementos adicionales al pictograma.
- Facilitar la implementación y el uso del sistema, mediante una guía clara y accesible que permita generar pictogramas a partir de descripciones sencillas, sin requerir conocimientos técnicos avanzados ni configuración de herramientas complementarias.
- Mantener una coherencia visual con el estilo de los pictogramas de ARASAAC, favoreciendo así su integración con materiales existentes y su comprensión por parte de los usuarios finales.

[1] ARASAAC: `https://arasaac.org/`

## 1.3. Plan de trabajo

Para alcanzar los objetivos, se realizaron las tareas descritas a continuación:

1. **Análisis de limitaciones de modelos de IA generativa:** Analizar las limitaciones de los modelos de IA generativa más populares al intentar crear pictogramas simples que faciliten la comunicación con personas con dificultades cognitivas o interpretativas.
2. **Generación de pictogramas:** Producir pictogramas utilizando distintos modelos disponibles en plataformas como GitHub, Hugging Face y CivitAI.
3. **Evaluación de resultados:** Analizar los resultados de los diferentes modelos para evaluar si alguno puede generar pictogramas adecuados. En caso de que ninguno lo consiga, seleccionar un modelo específico sobre el cual trabajar con el objetivo de desarrollar un nuevo modelo que permita generar pictogramas con un estilo cercano al de ARASAAC. Además, este modelo deberá cumplir con las características necesarias para generar pictogramas con pocos recursos y tiempos de generación reducidos.
4. **Preparación del conjunto de datos:** Para entrenar un modelo que genere pictogramas, será necesario preparar un conjunto de datos (*dataset*). Este *dataset* deberá ajustarse al estilo de ARASAAC.
5. **Entrenamiento del modelo:** Con el *dataset* y el modelo seleccionado, profundizar en el proceso de entrenamiento para lograr resultados que cumplan con los objetivos propuestos.
6. **Análisis de los resultados:** Realizar un análisis de los resultados obtenidos.

## 1.4. Contribuciones esperadas

Las contribuciones esperadas de este proyecto se derivan directamente del cumplimiento de los objetivos principales y secundarios definidos anteriormente. En conjunto, buscan optimizar el uso de pictogramas en el contexto de la Comunicación Aumentativa y Alternativa (CAA), mejorando la accesibilidad para personas con dificultades comunicativas mediante la generación rápida y eficiente de pictogramas con un estilo visual sencillo y coherente con el de ARASAAC.

Entre las contribuciones clave del proyecto se encuentran:

- **Generación eficiente de pictogramas:** El proyecto permite la creación de pictogramas de alta calidad a partir de descripciones simples, lo cual facilitará su aplicación en la CAA en tiempo real y con recursos limitados. Esto permitirá que profesionales de la educación, la medicina y otros campos puedan generar pictogramas rápidamente, sin depender de bases de datos preexistentes o software especializado.

Esta contribución se considerará alcanzada si el sistema es capaz de generar pictogramas de calidad utilizando modelos que ofrezcan una buena relación entre calidad y coste computacional.

- **Mejora en la comunicación con personas con dificultades cognitivas o interpretativas:** Al proporcionar pictogramas adaptados a las necesidades específicas de los usuarios, se espera que el proyecto facilite la interacción en contextos donde las personas enfrentan barreras de comunicación, promoviendo una inclusión efectiva y mejorando la calidad de vida de los usuarios.

  Esta contribución se valorará positivamente si el sistema es capaz de representar conceptos que normalmente no cuentan con pictogramas disponibles, como situaciones específicas o personalizadas. Algunos ejemplos incluyen la capacidad de generar pictogramas con determinadas paletas de colores, asignar colores específicos a objetos o representar personas con herramientas o prendas concretas.

- **Fomento de la investigación en IA generativa aplicada a la CAA:** Este trabajo también busca fomentar el interés y la exploración de la inteligencia artificial generativa en el campo de la CAA, abriendo nuevas vías para la investigación y el desarrollo de tecnologías accesibles. Al emplear tecnologías de código abierto, se espera que otros investigadores y desarrolladores se inspiren para continuar mejorando estas herramientas, ampliando sus aplicaciones y aumentando su precisión y utilidad.

  Esta contribución se considerará alcanzada si el proyecto demuestra resultados de calidad que evidencien el potencial de la IA generativa aplicada a la CAA, sirviendo como base o inspiración para futuros trabajos en esta línea.

Estas contribuciones se alinean con el objetivo de mejorar la comunicación en contextos donde las barreras cognitivas o interpretativas dificultan la interacción, y buscan establecer una base para futuras investigaciones centradas en tecnologías accesibles y personalizables.

Capítulo 2

# Estado de la Cuestión

En este capítulo se presentan los principales fundamentos y trabajos previos que contextualizan el desarrollo de este proyecto. Se parte de una revisión del campo de la Comunicación Aumentativa y Alternativa (CAA), con especial atención al papel de los pictogramas como herramientas clave para facilitar la comunicación en personas con dificultades comunicativas.

A continuación, se analiza el estado actual de las técnicas de generación de imágenes mediante inteligencia artificial, prestando especial atención a los modelos de Stable Diffusion y al uso de adaptaciones ligeras como los LoRA (Low-Rank Adaptation), empleados para personalizar su comportamiento. Se revisan también herramientas y entornos populares en la comunidad para entrenar y ajustar este tipo de modelos, así como sus implicaciones técnicas y limitaciones prácticas.

Seguidamente, se examinan los proyectos existentes que aplican IA generativa para la creación de pictogramas, evaluando sus capacidades actuales. Además, se exploran las soluciones disponibles para utilizar pictogramas en contextos comunicativos, analizando sus limitaciones y señalando las mejoras que podría aportar el presente trabajo.

El objetivo de este capítulo es proporcionar una base teórica y técnica que justifique las decisiones tomadas en el desarrollo del trabajo, facilitando la comprensión de los retos abordados y de las soluciones implementadas en los capítulos posteriores.

## 2.1. Pictogramas en la Comunicación Aumentativa y Alternativa (CAA)

La Comunicación Aumentativa y Alternativa (CAA) engloba todos aquellos sistemas, métodos y recursos que se utilizan para compensar o potenciar la comunicación de personas con dificultades en el habla o el lenguaje. Dentro de estos sistemas, los pictogramas ocupan un lugar central como herramienta visual que facilita la comprensión, la expresión y la interacción social (Bertola López (2017)).

Los pictogramas son representaciones gráficas de conceptos, acciones, objetos o emociones. Su uso es especialmente común en personas con Trastorno del Espectro Autista (TEA), parálisis cerebral, discapacidades intelectuales o enfermedades neurodegenerativas, ya que proporcionan un medio accesible para estructurar el lenguaje, anticipar rutinas y reducir la frustración en la comunicación cotidiana (Paolieri y Marful (2018)).

Un aspecto importante del diseño de pictogramas es su estilo visual. Diversos estudios han señalado que el nivel de abstracción, color, detalle y consistencia gráfica pueden influir en la eficacia comunicativa de estos símbolos (Rankin et al. (1994); Beukelman y Light (2020)). Por ejemplo, pictogramas con muchos detalles pueden ser más expresivos, pero también pueden sobrecargar visualmente a algunos usuarios. En cambio, estilos más simplificados, como los conocidos *stickman*, eliminan elementos decorativos para centrar la atención en la acción o el mensaje clave, siendo a menudo más eficaces en contextos de atención reducida o dificultades cognitivas.

La base de datos de pictogramas ARASAAC[1] (Portal Aragonés de la Comunicación Aumentativa y Alternativa) ofrece una amplia colección de recursos de uso libre que sigue esta lógica. En concreto, incluye múltiples estilos gráficos diseñados para responder a diferentes perfiles de usuario. Entre ellos se encuentran estilos con figuras humanas detalladas (con color de piel, ropa o expresiones) y estilos simplificados como el *stickman*, que representan a las personas mediante líneas y formas básicas. Esta diversidad de estilos refleja la importancia de adaptar los materiales visuales al nivel cognitivo y sensorial de cada persona usuaria, facilitando una comunicación más efectiva y personalizada.

## 2.2. Introducción a la inteligencia artificial generativa

La inteligencia artificial generativa ha avanzado significativamente en los últimos años, permitiendo la creación de modelos que generan imágenes de alta calidad a partir de descripciones textuales. Estos avances están impactando sectores como el arte, el diseño, la educación y la publicidad, cambiando la forma en que interactuamos con la tecnología y la creatividad. Lo que hace que estos modelos sean populares es no solo la calidad de las imágenes generadas, sino también su accesibilidad, ya que muchas de estas herramientas permiten a los usuarios sin experiencia técnica participar en el proceso creativo.

Los modelos de IA generativa han ganado notoriedad a través de su uso extendido en redes sociales y comunidades en línea, siendo herramientas clave en la democratización de la creatividad. Estas tecnologías están proporcionando acceso a herramientas de generación de imágenes avanzadas que antes solo estaban disponibles para profesionales.

[1] ARASAAC: `https://arasaac.org/`

## 2.3. Descripción de las herramientas y modelos de IA generativa

Actualmente, los modelos más conocidos por el público general son:

1. Imagen 3[2] (Imagen-Team-Google et al. (2024)), de Google.
2. MidJourney[3] (Midjourney (2022)).
3. DALL-E 3[4] y sus versiones (Ramesh et al. (2022)), de OpenAI.
4. Stable Diffusion[5], el único modelo de software libre.

La siguiente tabla compara varios modelos de inteligencia artificial generativa ampliamente conocidos según tres criterios: **Acceso** (si son gratuitos o de pago), **Desarrollo** (si el modelo es de código abierto o cerrado), y **Flexibilidad para desarrolladores**, entendida como la facilidad para adaptar, modificar o ampliar el sistema por parte de la comunidad de usuarios.

| **Modelo** | **Acceso** | **Desarrollo** | **Flexibilidad para desarrolladores** |
|---|---|---|---|
| **Imagen 3 (Google)** | Pago | Cerrado | Limitada |
| **MidJourney** | Pago | Cerrado | Limitada |
| **DALL-E 3 (OpenAI)** | Pago | Cerrado | Limitada |
| **Stable Diffusion** | Gratuito | Abierto (Código libre) | Alta (comunidad activa) |

Tabla 2.1: Comparación de modelos de IA generativa más conocidos

Entre estas opciones, todos los modelos, excepto Stable Diffusion, presentan restricciones en cuanto a generación gratuita y desarrollo de software. Este último destaca por ser de código abierto y contar con una amplia comunidad que proporciona abundantes recursos, soporte y herramientas de desarrollo, lo que lo convierte en la opción más adecuada para este trabajo.

### 2.3.1. Stable Diffusion: Una alternativa versátil y accesible

Stable Diffusion ofrece diversos modelos base, desarrollados por equipos como *Runway*[6], el grupo *CompVis*[7] de la Universidad Ludwig Maximilian de Múnich y

[2]Imagen 3: `https://deepmind.google/technologies/imagen-3/`
[3]MidJourney: `https://www.midjourney.com/`
[4]DALL-E 3: `https://openai.com/dall-e`
[5]Stable Diffusion: `https://stability.ai/stable-diffusion`
[6]Runway: `https://runwayml.com/`
[7]CompVis (Computer Vision Group): `https://compvis.github.io/`

*Stability AI*[8], con contribuciones de *datasets* por parte de *EleutherAI*[9] y *LAION*[10]. Aunque la versión más reciente es Stable Diffusion 3.0 (Esser et al. (2024)), gran parte de la comunidad se centra en los modelos Stable Diffusion 1.5 y Stable Diffusion XL, que analizaremos a continuación.

### Stable Diffusion 1.5 (SD 1.5)

Stable Diffusion 1.5 ((Rombach et al., 2022)), desarrollado por *Stability AI*, es uno de los modelos de generación de imágenes más populares debido a su personalización y versatilidad. Su comunidad ha desarrollado numerosas herramientas para su entrenamiento y generación. La resolución nativa de este modelo es de 512x512 píxeles, lo cual es adecuado para la creación de pictogramas simples como los de ARASAAC, que tienen una resolución estándar de 500x500 píxeles. Además, SD 1.5 es asequible en términos de requisitos de hardware, necesitando únicamente 3-4 GB de VRAM, un procesador de 4 núcleos y 3-6 GB de RAM.

Sin embargo, SD 1.5 presenta limitaciones al generar detalles complejos, como luces avanzadas o representaciones precisas de manos humanas. Aunque estas carencias no afectan gravemente la creación de pictogramas, sí pueden ser problemáticas en escenarios que requieren mayor detalle. Otro inconveniente es la dificultad para interpretar *prompts* largos o complejos, lo que puede complicar la especificación de elementos específicos como colores o posturas. No obstante, estas limitaciones pueden paliarse con modelos derivados entrenados sobre SD 1.5.

Cabe destacar que SD 1.5 ha sido retirado de muchos repositorios públicos, incluidos los oficiales de *Stability AI*, posiblemente debido a problemas legales relacionados con su base de datos de entrenamiento o para promover el uso de versiones más recientes como SDXL.

### Stable Diffusion XL (SDXL)

Stable Diffusion XL (SDXL) ((Podell et al., 2023)) representa una evolución significativa respecto a SD 1.5, gracias a su mayor capacidad de procesamiento e interpretación de *prompts* complejos. Este modelo genera imágenes de resolución nativa 1024x1024 píxeles, lo que permite obtener mayor detalle sin necesidad de reescalado.

Sin embargo, su mayor complejidad implica requisitos de hardware más altos: entre 6-8 GB de VRAM, un procesador moderno como un I5 o Ryzen 5 y 8-16 GB de RAM. Esto podría dificultar su uso en proyectos que buscan generar pictogramas de manera rápida y con recursos limitados.

Existen numerosas variantes de SD 1.5 y SDXL que mejoran la calidad en estilos específicos como el fotorrealismo, el anime o el *cartoon*. Algunos de los modelos

[8]Stability AI: `https://stability.ai/`
[9]EleutherAI: `https://www.eleuther.ai/`
[10]LAION (Large-scale Artificial Intelligence Open Network): `https://laion.ai/`

derivados más populares son DreamShaper[11] (SD 1.5), Realistic Vision[12] (SD 1.5), DreamShaper XL[13] (SDXL) y Juggernaut XL[14] (SDXL), disponibles en plataformas como *HuggingFace*[15].

### 2.3.2. Limitaciones de los modelos más populares generando pictogramas

Para evaluar el estado actual de la generación de pictogramas mediante IA, se realizaron pruebas con los modelos mencionados. Los resultados evidenciaron limitaciones significativas en su capacidad para generar pictogramas adecuados al estilo requerido en la comunicación aumentativa y alternativa (CAA), como el estilo ARASAAC.

**Imagen 3** Imagen 3, pese a su sofisticación, no fue capaz de generar pictogramas adecuados, como se observa en la figura 2.1. Su enfoque se centra en representaciones realistas, alejándose del propósito deseado.

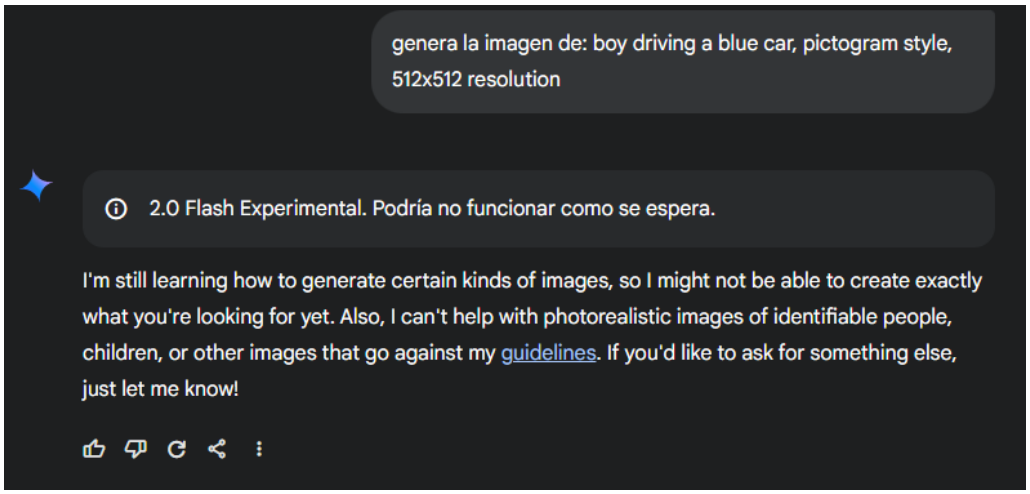



Figura 2.1: Intento de generación en Imagen 3.

**MidJourney** El acceso a MidJourney presentó una limitación significativa, ya que se requiere una suscripción activa para realizar pruebas, como se evidencia en la figura 2.2.

**DALL-E 3** Aunque DALL-E 3 generó pictogramas (figura 2.3), estos carecían de color y tenían un estilo más cercano al de íconos de aplicaciones que al de ARA-

[11] DreamShaper: `https://huggingface.co/Lykon/dreamshaper-8`
[12] Realistic Vision: `https://huggingface.co/SG161222/Realistic_Vision_V6.0_B1_noVAE`
[13] DreamShaper XL: `https://huggingface.co/Lykon/dreamshaper-xl-turbo`
[14] Juggernaut XL: `https://huggingface.co/RunDiffusion/Juggernaut-X-v10`
[15] *HuggingFace*: `https://huggingface.co`

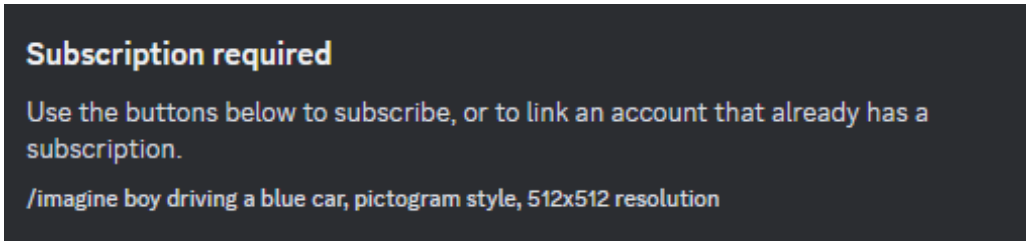
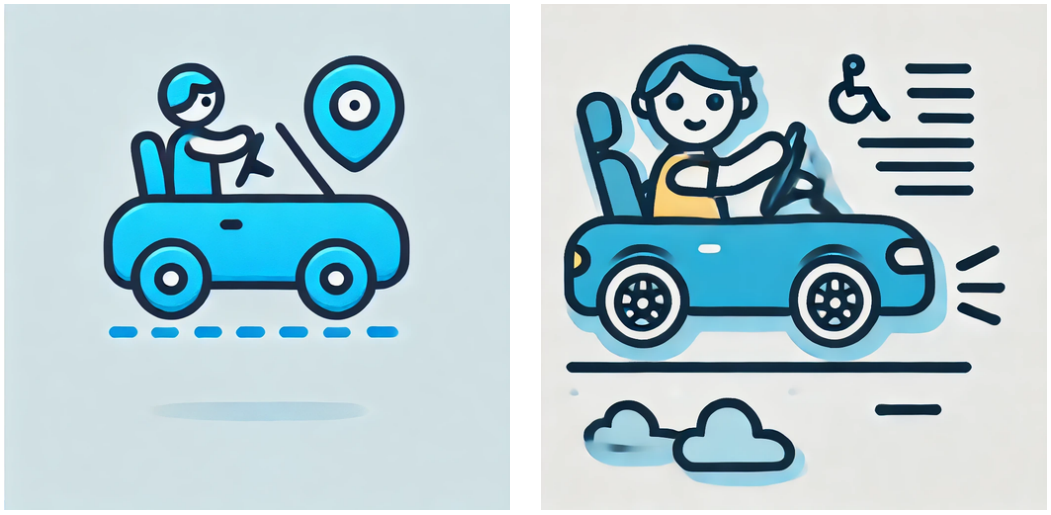



Figura 2.2: Intento de generación en MidJourney.

SAAC (figura 1.1). Además, mostró inconsistencias estilísticas entre generaciones consecutivas.



(a) Primera generación (b) Segunda generación

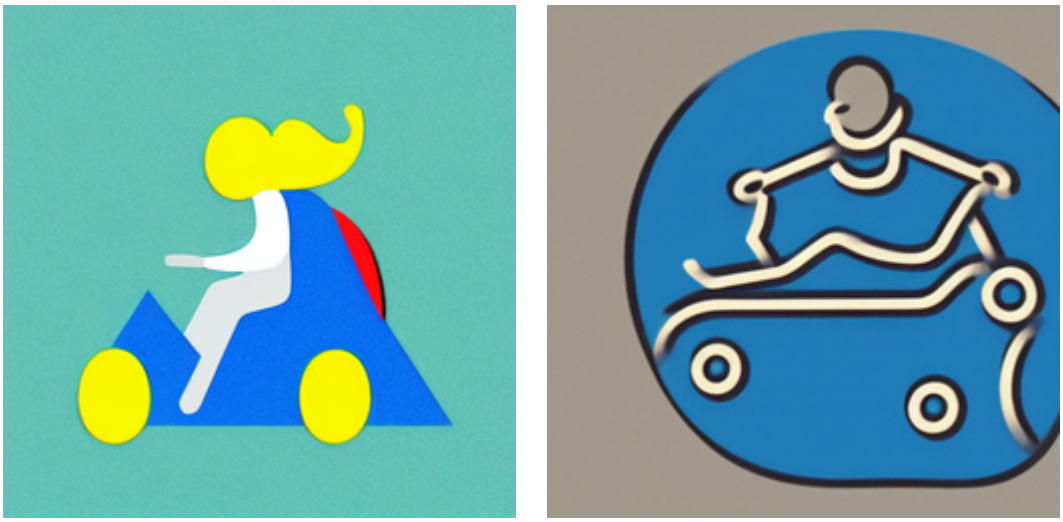
Figura 2.3: Generación con DALL-E 3.
Prompt: "boy driving a blue car, pictogram style, 512x512 resolution."

**Stable Diffusion** Tanto Stable Diffusion 1.5 como SDXL generaron imágenes de baja calidad y sin consistencia estilística (figuras 2.4 y 2.5), lo que evidencia la necesidad de ajustes específicos para generar pictogramas.

**Conclusión** Aunque la IA generativa ha avanzado significativamente, aún enfrenta grandes desafíos en su aplicación a la CAA. El desarrollo de una herramienta gratuita y de software libre que permita generar pictogramas accesibles y efectivos tendría un impacto positivo en este campo, mejorando la inclusión y la comunicación para personas con necesidades especiales.

### 2.3.3. Tecnología LoRA

Los LoRAs (*Low-Rank Adaptations*) (Hu et al. (2021); Face (n.d.)), son adaptaciones que se aplican a los modelos de IA generativa para mejorar o ajustar sus



(a) (b)

Figura 2.4: Generación con Stable Diffusion 1.5.
Prompt: "boy driving a blue car, pictogram style."

(a) (b)

Figura 2.5: Generación con Stable Diffusion XL.
Prompt: "boy driving a blue car, pictogram style."

resultados sin necesidad de reentrenar el modelo base de forma completa. Para entender mejor cómo funcionan los LoRAs, es útil recordar que un modelo generativo de imágenes, como Stable Diffusion, está basado en una red neuronal profunda entrenada con grandes cantidades de datos. Esta red neuronal tiene múltiples capas que procesan la información de manera jerárquica, desde la entrada (el texto o la descripción) hasta la salida (la imagen generada).

Los LoRAs funcionan modificando ciertos parámetros en las capas superiores de la red neuronal. Estas capas son responsables de generar características más

abstractas y complejas de la imagen, como formas, texturas y detalles finos. Al ajustar solo estas capas, los LoRAs permiten personalizar el comportamiento del modelo sin necesidad de cambiar todo el sistema. Esto es ventajoso, ya que entrenar un modelo completo de nuevo puede ser muy costoso en términos de tiempo y recursos computacionales.

Por ejemplo, los LoRAs pueden ser utilizados para modificar el estilo visual de las imágenes generadas. Esto incluye la incorporación de estilos artísticos específicos, como los de pintores famosos o de estilos gráficos como el arte de videojuegos (por ejemplo, el estilo de *Super Mario*), o mejorar la representación de detalles complejos como las manos o las partes del cuerpo humano. De igual manera, los LoRAs permiten adaptaciones que mejoran la generación de personajes humanos o incluso permiten crear imágenes de personas reales.

Esta capacidad de ajustar el modelo a necesidades específicas hace que los LoRAs sean una herramienta poderosa y eficiente para personalizar los modelos de IA generativa sin requerir un reentrenamiento completo. Además, debido a su naturaleza más ligera, los LoRAs pueden ser aplicados con un aumento mínimo en los tiempos de generación y los recursos computacionales necesarios.

### 2.3.4. Técnicas para Mejorar la Generación de Imágenes

Además del entrenamiento personalizado de modelos, como ocurre con los LoRAs, existen diversas estrategias utilizadas para mejorar la calidad de las imágenes generadas. Entre las más comunes se encuentran el diseño avanzado de *prompts*, el uso de *word embeddings* y los *refiners*.

**Diseño de *prompts* complejos** (White et al. (2023)) El proceso de creación de descripciones detalladas para guiar al modelo se conoce como *prompt engineering*. Un *prompt* bien diseñado no solo incluye el objeto o sujeto deseado, sino también atributos visuales como colores, acciones, estilo gráfico o disposición espacial. Por ejemplo, un *prompt* como "a purple apple on a white table, minimalistic style, centered" tiene más probabilidad de generar una imagen coherente y precisa que uno genérico como "an apple".

Asimismo, el uso de *negative prompts* permite indicar al modelo qué elementos se deben evitar, como "bad anatomy", "extra limbs" o "poorly drawn face".

Algunos modelos permiten además ajustar el peso de ciertas palabras clave dentro del *prompt*, otorgándoles mayor influencia en la imagen generada —por ejemplo, escribiendo "(purple apple:1.5)" para reforzar el color morado. Otros modelos incorporan palabras clave especiales que activan estilos o efectos concretos durante la generación, como "masterpiece" o "detailed illustration", aunque su eficacia depende del entrenamiento previo del modelo base.

***Word embeddings*** (Gal et al. (2022)) Los *embeddings* son representaciones numéricas del significado de las palabras en un espacio vectorial. Algunos modelos

permiten ajustar estos vectores para que ciertas palabras o expresiones se interpreten de forma más precisa o específica. En el contexto de generación de imágenes, ajustar un *embedding* puede ayudar al modelo a diferenciar entre conceptos similares (como "boxer dog" y "boxer girl") o mejorar su comprensión de conceptos poco frecuentes o especializados.

***Refiners*** (Parmar et al. (2023)) Los *refiners* son modelos adicionales que se aplican después de la generación inicial de la imagen. Su objetivo es mejorar ciertos aspectos visuales, como la anatomía de figuras humanas, el color de los objetos o la coherencia estilística. En muchos casos, los refiners forman parte de una arquitectura de generación por etapas, donde un modelo base produce una imagen preliminar y otro modelo la refina hasta alcanzar mayor calidad. Esta técnica resulta especialmente útil en escenas complejas o que requieren un alto grado de fidelidad visual.

No obstante, aunque estas estrategias pueden mejorar notablemente los resultados, su uso también introduce una mayor complejidad técnica y, en algunos casos, mayor coste computacional. En el contexto de este proyecto, cuyo objetivo es facilitar la generación de pictogramas con estilo ARASAAC de forma sencilla y accesible, se prioriza una solución que no dependa de técnicas avanzadas como tener que utilizar obligatoriamente *word embeddings* personalizados, el uso sistemático de *refiners* o el diseño manual de *prompts* complejos. En su lugar, se apuesta por un modelo base eficiente, complementado con LoRAs bien entrenados, capaces de ofrecer resultados de alta calidad con un mínimo esfuerzo por parte del usuario.

Estas otras estrategias podrían tomarse en cuenta si el objetivo fuera únicamente obtener la mejor calidad posible en cada imagen, sin considerar el tiempo, la sencillez o la facilidad de uso por parte de personas no expertas. Sin embargo, para aplicaciones prácticas como la comunicación aumentativa, donde la accesibilidad es clave, es más valioso contar con un método rápido, intuitivo y reproducible, aunque eso implique un pequeño sacrificio en la perfección visual.

### 2.3.5. Aplicaciones de generación y entrenamiento

Los distintos modelos de Stable Diffusion y LoRAs, como los mencionados en 2.3.3, pueden ser probados en distintas páginas web, la mayoría de las cuales son de pago. Sin embargo, en algunos casos, ofrecen unas pocas generaciones gratuitas. Una de las ventajas de estas plataformas es que no dependen del hardware del usuario, ya que se ejecutan en la nube. No obstante, la mejor opción para este trabajo de investigación, por su gratuidad y comodidad, es utilizar las interfaces de software libre que desarrolla la comunidad. A través de estas interfaces, aunque el usuario depende de su propio hardware, se obtiene la gran ventaja de poder generar imágenes de manera gratuita y sin restricciones de cantidad, además de la posibilidad de experimentar con extensiones, parámetros y tecnologías que mejoran la calidad y precisión de las imágenes generadas.

Existen numerosas interfaces de software libre disponibles en plataformas como

GitHub, siendo *WebUI Forge*[16] una de las más destacadas por su amplia gama de opciones, extensiones y personalización. Esta interfaz ha sido utilizada en este trabajo debido a su flexibilidad y las mejoras continuas que la comunidad implementa en ella.

Otras herramientas igualmente importantes para lograr los objetivos de este trabajo son las utilizadas en el entrenamiento de modelos y LoRAs, así como aquellas que automatizan la generación de *captions* (descripciones de imágenes). Estas herramientas son esenciales en el proceso de entrenamiento, ya que las *captions* guían a los modelos en la relación entre las imágenes y las palabras asociadas. Durante el entrenamiento, el modelo ajusta los valores de sus capas internas, lo que le permite asociar características visuales con los elementos descritos en las *captions*. Este ajuste es crucial, ya que habilita a los modelos para generar respuestas más precisas y coherentes cuando se les presentan *prompts* que involucran estilos artísticos u otros elementos presentes en los conjuntos de datos de entrenamiento.

Otras herramientas igualmente importantes para lograr los objetivos de este trabajo son las utilizadas en el entrenamiento de modelos y LoRAs, así como aquellas que automatizan la generación de *captions* (descripciones de imágenes). Estas incluyen, entre otras:

- **Kohya_SS GUI**[17]: herramienta de interfaz gráfica basada en Python para entrenar modelos LoRA, con soporte para múltiples arquitecturas de difusión. Permite configurar hiperparámetros, gestionar datasets y lanzar entrenamientos sin necesidad de codificación manual.
- **BLIP Captioning**[18]: modelo de visión-lenguaje diseñado para generar descripciones automáticas de imágenes. Es ampliamente empleado en herramientas de entrenamiento como *Kohya SS GUI*, que lo integran para automatizar el etiquetado de datasets a partir de imágenes sin anotaciones previas.

Estas herramientas son esenciales en el proceso de entrenamiento, ya que las *captions* guían a los modelos en la relación entre las imágenes y las palabras asociadas. Durante el entrenamiento, el modelo ajusta los valores de sus capas internas para aprender estas asociaciones, lo que le permite generar resultados más coherentes y relevantes al recibir *prompts* que incluyan estilos artísticos u otros elementos presentes en el dataset.

## 2.4. Uso de IA generativa en otros campos relacionados

La inteligencia artificial, y en particular el aprendizaje automático, ha abierto nuevas posibilidades en el análisis y la reinterpretación de lenguajes de comunicación

[16] WebUI Forge: `https://github.com/lllyasviel/stable-diffusion-webui-forge?tab=readme-ov-file`

[17] Kohya_SS GUI: `https://github.com/bmaltais/kohya_ss`

[18] BLIP (*Bootstrapping Language-Image Pretraining*): `https://github.com/salesforce/BLIP`

pictográfica antiguos, como los jeroglíficos. Un ejemplo destacado de este enfoque es el trabajo descrito por Nawar (2018), que explora cómo las técnicas de aprendizaje automático pueden identificar patrones en estos lenguajes con el fin de desarrollar un sistema pictográfico moderno. Otro estudio relevante en este campo es el presentado en Llorente-Hernando y Romano-Ramos (2024), que se centra en la creación de recuerdos representados por imágenes generadas automáticamente en tiempo real mediante inteligencia artificial, con el objetivo de ayudar a personas con Alzheimer.

La IA generativa ha demostrado ser una herramienta poderosa en una amplia variedad de campos artísticos.En particular, la tecnología LoRA ha sido clave para adaptar modelos de generación de imágenes a estilos visuales específicos con un coste computacional reducido. La creación de imágenes que emulan técnicas como óleo[19], acuarela[20], lápices de colores y carboncillo. Además, han sido empleados para recrear estilos visuales específicos, como los de pintores históricos[21], mosaicos[22], animes y series de dibujos animados. Estos avances destacan la capacidad de los LoRA para capturar y reproducir características visuales detalladas.

En este contexto, el potencial de la IA generativa para desarrollar un sistema pictográfico simplificado y coherente es evidente. Adaptado al estilo de ARASAAC, este sistema podría facilitar la creación de herramientas más efectivas y accesibles para mejorar la comunicación visual de personas con dificultades cognitivas, ampliando las posibilidades de expresión y comprensión.

## 2.5. Soluciones existentes para la creación de pictogramas y sus limitaciones

Actualmente existen distintos enfoques para la creación automática de pictogramas en el ámbito de la Comunicación Aumentativa y Alternativa (CAA), que pueden clasificarse en dos grandes grupos: aquellos basados en la selección desde bases de datos cerradas, y los que emplean inteligencia artificial generativa.

### Sistemas basados en bases de datos predefinidas

Un ejemplo representativo de estos sistemas es el proyecto *PICTAR* (Martín-Guerrero (2018)), desarrollado como Trabajo de Fin de Máster en la Facultad de Informática (FDI). Este sistema permite expresar el significado de frases mediante análisis semántico y selección de pictogramas desde una base de datos preexistente. Sin embargo, presenta varias limitaciones notables. En primer lugar, depende completamente del contenido disponible en la base de datos. Esto implica que, si no existe un pictograma específico para una idea concreta, el sistema se ve obligado a combinar varios pictogramas. Por ejemplo, para representar una "hoja de árbol

[19]LoRA óleo: `https://civitai.com/models/84542/oil-paintingoil-brush-stroke`
[20]LoRA acuarela: `https://civitai.com/models/604440/aquarelle-watercolor-painting`
[21]LoRA Velázquez: `https://civitai.com/models/668818/velazquex-flux-lora`
[22]LoRA mosaico: `https://civitai.com/models/553752?modelVersionId=666202`

rosa", si no existe un pictograma exacto, se seleccionaría uno para "hoja" y otro para "color rosa". Esta combinación puede resultar poco clara, especialmente para personas con dificultades cognitivas, quienes podrían tener problemas para asociar correctamente los conceptos.

En general, todos los sistemas basados en bases de datos cerradas presentan un inconveniente evidente: es prácticamente imposible abarcar todas las ideas posibles sin que el tamaño de la base crezca en exceso, dificultando su mantenimiento, escalabilidad y usabilidad. Por ello, la inteligencia artificial generativa representa un avance prometedor en el contexto de la comunicación mediante pictogramas.

## Sistemas basados en inteligencia artificial generativa

En el ámbito de la inteligencia artificial generativa aplicada a la comunicación aumentativa, aún existen pocos trabajos que aborden específicamente la generación de pictogramas simples como los utilizados en ARASAAC. Si bien hay proyectos enfocados en la creación de iconos para aplicaciones o señalizaciones (como señales de tráfico), su enfoque no está orientado a las necesidades de la Comunicación Aumentativa y Alternativa (CAA). En este contexto, destaca como excepción el proyecto *PictoAI* (Drews et al. (2025)), una herramienta integrada en ChatGPT que permite generar pictogramas a partir de descripciones textuales.

*PictoAI* ofrece funcionalidades interesantes como la segmentación semántica de conceptos, el diseño de pictogramas guiado por *prompts*, y la posibilidad de hacer referencia a pictogramas existentes en ARASAAC. No obstante, presenta limitaciones importantes. Al estar basada en tecnologías propietarias de *OpenAI* (GPT y DALL·E), su uso gratuito es limitado y depende de la disponibilidad y carga de la plataforma. Además, no garantiza una fidelidad total al estilo visual de ARASAAC, aunque puede generar ilustraciones de calidad que comparten algunas de sus características gracias a la potencia de los modelos de *OpenAI*.

En el terreno del software libre, existen numerosos modelos y LoRAs entrenados para generar imágenes en estilos como la animación japonesa, el *cartoon* occidental o ilustración digital. Sin embargo, estos modelos suelen centrarse en objetivos artísticos o estéticos más que comunicativos. Además, algunos de estos LoRAs pueden incorporar contenido inapropiado, lo cual los hace inadecuados para contextos educativos o de uso con personas vulnerables.

Esta falta de enfoques centrados en la simplicidad visual y la claridad semántica refuerza la necesidad de desarrollar modelos específicos adaptados a los requisitos de la CAA. En este tipo de comunicación, la comprensión inmediata y sin ambigüedad del mensaje visual es esencial, por lo que resulta fundamental contar con soluciones que prioricen estos criterios.

## Necesidad de soluciones especializadas

La revisión de estas soluciones pone de manifiesto una carencia significativa de herramientas capaces de generar pictogramas personalizados que cumplan con los

estándares de simplicidad y coherencia estilística del estilo ARASAAC. Por tanto, es necesario desarrollar soluciones que combinen la flexibilidad de la IA generativa con modelos entrenados específicamente para los fines de la comunicación aumentativa. Este TFG pretende cubrir ese vacío mediante el uso de tecnologías abiertas, como Stable Diffusion y *Kohya_ss*, entrenando modelos adaptados a los requerimientos reales de la CAA.

## 2.6. Conclusión

El análisis de este capítulo ha permitido contextualizar la relevancia de este proyecto dentro del ámbito de la Comunicación Aumentativa y Alternativa (CAA), mostrando cómo puede contribuir a mejorar este campo al permitir la generación de pictogramas personalizados para cada situación, sin depender de su existencia previa en bases de datos cerradas. Se han identificado las principales herramientas, técnicas y limitaciones actuales en la generación de imágenes mediante inteligencia artificial, centrándose especialmente en su aplicación a pictogramas con estilo ARASAAC.

Aunque existen algunas soluciones previas, ninguna ofrece un control suficiente sobre el estilo, la simplicidad y la coherencia necesarias para su uso en entornos de CAA. La herramienta propuesta en este trabajo, basada en software libre y modelos abiertos, permite una generación más versátil, accesible y personalizable.

Asimismo, se ha analizado el estado actual de la inteligencia artificial generativa aplicada a imágenes, detallando qué aplicaciones y herramientas están disponibles, qué funcionalidades ofrecen y cómo la comunidad ha logrado avances significativos mediante el uso de modelos base como Stable Diffusion y técnicas como los LoRA. Estas soluciones permiten generar imágenes personalizadas a cualquier persona con conocimientos técnicos básicos, de forma gratuita y con mayor adaptabilidad que los sistemas cerrados de las grandes compañías.

Por último, aunque han comenzado a surgir propuestas como *PictoAI*, integradas en asistentes conversacionales como ChatGPT, estas aún presentan limitaciones importantes en cuanto a consistencia estilística. Su dependencia de tecnologías propietarias y el acceso restringido refuerzan la necesidad de desarrollar soluciones más específicas, controlables y adaptadas, como la que se plantea en este trabajo.

En definitiva, los avances recientes en IA generativa suponen una oportunidad real para mejorar la accesibilidad y la personalización en el uso de pictogramas. El trabajo desarrollado en este TFG se presenta como una respuesta a estas carencias, ofreciendo una alternativa gratuita, flexible y mejor alineada con las necesidades reales de la comunicación aumentativa.

Capítulo 3

# Descripción del Trabajo

En este capítulo se describen las etapas seguidas para lograr el objetivo principal del proyecto: generar pictogramas adecuados para su uso en Comunicación Aumentativa y Alternativa (CAA), optimizando el proceso para que requiera un bajo costo computacional y un tiempo de entrenamiento reducido.

A continuación, se presenta un esquema general del flujo de trabajo seguido. Cada una de las fases mostradas se desarrolla en detalle en los apartados posteriores.

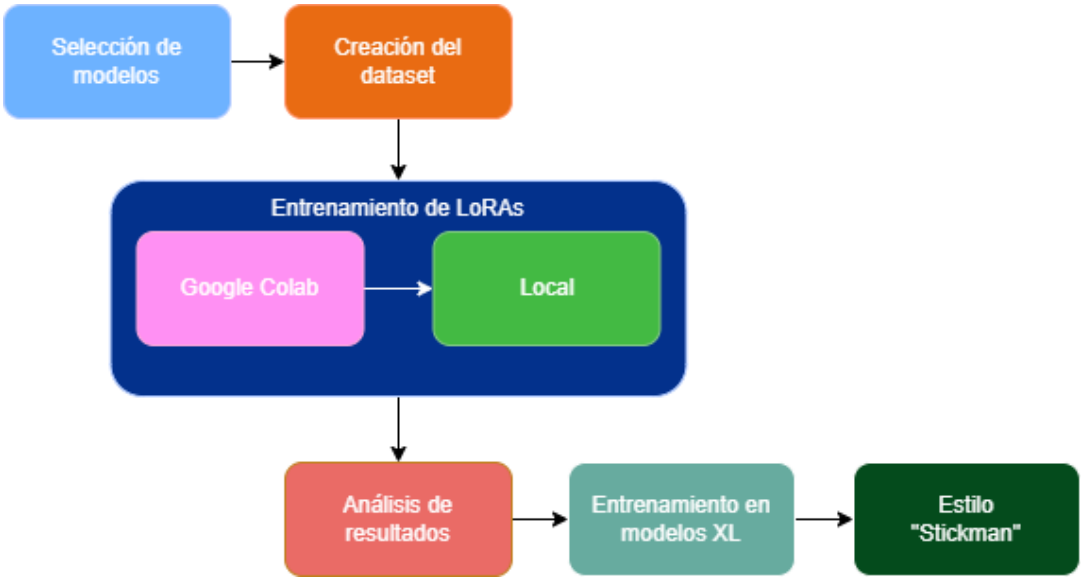



Figura 3.1: Diagrama de flujo del trabajo realizado.

De forma resumida, estas etapas son:

- **Selección de modelos:** elección de un modelo base compatible con entrenamiento LoRA y generación de imágenes coherentes.
- **Creación del dataset:** recopilación, depuración y etiquetado de imágenes en estilos visuales relevantes.

- **Entrenamiento de LoRAs:** experimentación inicial en Google Colab y mejoras posteriores en entorno local.
- **Análisis de resultados:** evaluación visual y comparativa de la calidad generada con distintos parámetros.
- **Entrenamiento en modelos XL:** pruebas con versiones más potentes del modelo base para explorar mejoras cualitativas.
- **Estilo *Stickman*:** entrenamiento con estilo alternativo "stickman" para evaluar adaptabilidad del enfoque.

## 3.1. Selección de Modelos

Para desarrollar un sistema de generación de pictogramas eficiente, accesible y de calidad, es fundamental seleccionar correctamente los modelos base sobre los que se entrenarán los LoRA y se generarán las imágenes. La elección del modelo afecta directamente al rendimiento, la calidad visual, la capacidad de personalización y la compatibilidad con distintos dispositivos.

En esta sección se analizan diferentes modelos, evaluando sus características clave en relación con los requisitos del proyecto: bajos requerimientos computacionales, calidad de generación, tiempos de generación y entrenamiento, y facilidad de entrenamiento. A partir de esta evaluación, se justifica la elección del modelo base, el modelo de generación y el modelo de entrenamiento.

### 3.1.1. Elección del Modelo Base

Para este proyecto, se requiere trabajar con unos modelos robustos, ampliamente probados y respaldados por una gran comunidad. Por ello, se consideraron dos modelos principales: **Stable Diffusion 1.5 (SD 1.5)** y **Stable Diffusion XL (SDXL)**. La elección del modelo base influye directamente en las características de los modelos de generación y entrenamiento que se usarán.

**Criterios de selección del modelo:** Al elegir el modelo base, se tuvieron en cuenta varios factores clave:

1. **Requisitos de hardware:** Se priorizan modelos accesibles en dispositivos con recursos limitados. Por ejemplo, un modelo como FLUX, que requiere 10 GB de VRAM, 32-64 GB de RAM y una CPU de alto rendimiento, sería inviable en la mayoría de entornos.
2. **Calidad de imagen:** Se busca una calidad de imagen que permita generar pictogramas claros y con líneas definidas. Aunque SDXL tiene mejor calidad de imagen que SD 1.5, esta última ofrece un balance adecuado entre calidad y requisitos técnicos.

3. **Capacidad de *prompt*:** Es importante que el modelo interprete con precisión indicaciones textuales para generar imágenes coherentes con el concepto deseado. SDXL tiene una capacidad más avanzada para interpretar *prompts* complejos, mientras que SD 1.5 es suficiente para la creación de pictogramas más simples.

4. **Velocidad de generación:** En el contexto de la comunicación aumentativa y alternativa (CAA), la velocidad es esencial para garantizar una respuesta ágil. Modelos como SD 1.5 permiten generar imágenes en segundos, mientras que SDXL requiere más tiempo debido a su mayor complejidad.

5. **Facilidad de entrenamiento:** Modelos con requisitos de hardware más bajos, como SD 1.5, son más sencillos y económicos de entrenar en comparación con SDXL, que demanda más recursos y tiempo. Esto hace que SD 1.5 sea una mejor opción para proyectos con recursos limitados.

6. **Disponibilidad:** La comunidad ha desarrollado múltiples variantes y mejoras basadas en SD 1.5, lo que facilita su adaptación a diversos proyectos. Por otro lado, SDXL también cuenta con una amplia comunidad, y su versión inicial sigue estando oficialmente disponible.

| **Característica** | **SD 1.5** | **SDXL** |
|---|---|---|
| **Requisitos de hardware** | Bajos | Altos |
| **Calidad de imagen** | Buena | Excelente |
| **Capacidad de *prompt*** | Limitada | Avanzada |
| **Velocidad de entrenamiento** | Rápida | Más lenta |
| **Facilidad de entrenamiento** | Menos costoso | Más costoso |
| **Disponibilidad** | No oficial (derivados) | Oficial |

Tabla 3.1: Comparativa entre modelos base SD 1.5 y SDXL

**Decisión:** Se seleccionó SD 1.5 como modelo base para el entrenamiento debido a sus menores requisitos de hardware, bajo coste en el proceso de entrenamiento (crucial dado los recursos limitados del proyecto) y velocidad de generación. Aunque su capacidad de *prompt* es más limitada que la de un modelo XL, en la generación de pictogramas no se prevé la necesidad de *prompts* altamente complejos.

### 3.1.2. Elección del Modelo de Generación

Dado que el modelo base seleccionado es Stable Diffusion 1.5, los modelos de generación deberán basarse en este. Se priorizan modelos capaces de producir ilustraciones estilizadas y claras, adecuadas para pictogramas.

#### Modelos derivados de SD 1.5

Los modelos derivados de SD 1.5 ofrecen mejoras específicas sobre el modelo base. Entre los más relevantes para este proyecto se encuentran:

1. **Realistic Vision**[1]**:** Modelo ampliamente popular en la comunidad, con millones de descargas en todas sus versiones. Su principal enfoque es el realismo, logrando resultados sobresalientes en la generación de imágenes fotorrealistas. Aunque también puede generar ilustraciones, su calidad artística es inferior a la de otros modelos.

2. **DreamShaper**[2]**:** Este modelo nació como una alternativa gratuita y de software libre a *MidJourney*, conocido por sus políticas más restrictivas y su modelo de negocio basado en suscripciones. A diferencia de *Realistic Vision*, *DreamShaper* está diseñado principalmente para la creación de arte e ilustraciones, un enfoque más acorde con los estilos necesarios para la generación de pictogramas. En sus versiones más recientes, también ha mejorado significativamente en la generación de imágenes realistas.

**Comparativa entre DreamShaper y Realistic Vision:**

| **Característica** | **DreamShaper** | **Realistic Vision** |
|---|---|---|
| **Estilo principal** | Ilustraciones y arte | Realismo fotográfico |
| **Calidad artística** | Alta | Moderada |
| **Detalles finos** | Buena | Excelente |
| **Velocidad** | Rápida | Rápida |

Tabla 3.2: Comparativa entre *DreamShaper* y Realistic Vision

**Decisión:** Se seleccionó *DreamShaper* como el modelo principal de generación debido a su enfoque en ilustraciones estilizadas, que se alinean mejor con los objetivos del proyecto. A pesar de sus diferencias, ambos modelos pueden usarse dependiendo del *prompt*, ya que comparten el mismo modelo base (SD 1.5).

### 3.1.3. Elección del Modelo de Entrenamiento

**Relación entre generación y entrenamiento:** Es fundamental que el modelo de generación y el modelo de entrenamiento compartan le mismo modelo base. Esta compatibilidad asegura que los resultados de la generación mantengan la calidad y coherencia esperadas. Por ejemplo, si se genera con *DreamShaper*, el LoRA debe ser entrenado en *DreamShaper*, SD 1.5 u otro modelo que derive de la misma base. Sin embargo, entrenar en un modelo derivado puede ocasionar problemas como el *text encoder corruption*[3].

**Decisión:** Por esta razón, inicialmente se decidió utilizar DreamShaper como modelo de entrenamiento, asegurando una total compatibilidad con el modelo de

[1]Realistic Vision: `https://civitai.com/models/4201?modelVersionId=130072`

[2]DreamShaper: `https://civitai.com/models/4384?modelVersionId=128713`

[3]El *text encoder corruption* se refiere a la pérdida de precisión o coherencia en la interpretación de los textos durante el proceso de generación, debido a incompatibilidades entre modelos derivados y su modelo base.

generación seleccionado. No obstante, pruebas posteriores con SD 1.5 *vanilla* (versión base, sin modificaciones ni ajustes personalizados) como modelo de entrenamiento ofrecieron mejores resultados en términos de generalización y adaptabilidad, permitiendo que los LoRA fueran completamente compatibles con otros modelos derivados de SD 1.5, como *DreamShaper* y *Realistic Vision*. Además, esta elección redujo significativamente el *text encoder corruption*, logrando los mejores resultados de entre todos los entrenamientos realizados.

## 3.2. Creación del *dataset* de entrenamiento

El *dataset* de imágenes de entrenamiento es una de las partes más importantes para entrenar un LoRA efectivo. Este debe estar compuesto por imágenes de alta calidad, ricas en información relevante sobre el estilo que se desea lograr. Imágenes de baja calidad no solo resultan inútiles, sino que perjudican el entrenamiento, empeorando los resultados finales. En este contexto, más cantidad no siempre es mejor; la calidad es crucial.

### 3.2.1. Resolución de las imágenes

La resolución de las imágenes es un factor fundamental. Imágenes de mayor resolución contienen más información útil, lo que puede mejorar el entrenamiento. Sin embargo, este incremento en resolución también aumenta el tiempo necesario para entrenar el modelo, un recurso limitado en este proyecto. Es importante destacar que la resolución de las imágenes no tiene que coincidir exactamente con la resolución base del modelo. Por ejemplo, se pueden usar imágenes de 1024x1024 para entrenar un modelo con resolución base de 512x512.

En nuestro caso, utilizamos los pictogramas de ARASAAC, cuya resolución es de 500x500 píxeles. Aunque esta resolución es menor que la base de 512x512 del modelo de entrenamiento, no presenta ningún problema práctico y reduce significativamente el tiempo de entrenamiento. Además, los pictogramas no requieren el mismo nivel de detalle que, por ejemplo, replicar el arte de un gran maestro.

### 3.2.2. Etiquetado del *dataset*: *Captions*

Además de las imágenes, cada una de ellas necesita estar acompañada por un documento de texto, generalmente en formato .txt, que describa los elementos importantes presentes en la imagen. Estos documentos, conocidos como *captions*, son esenciales para que el modelo relacione las características visuales de la imagen con el texto correspondiente durante el entrenamiento.

**Funcionamiento de las *captions*** Durante el entrenamiento de modelos de generación de imágenes, las *captions* (descripciones textuales) sirven para vincular los elementos visuales de una imagen con palabras concretas. Esto permite que el modelo aprenda a asociar ciertos patrones visuales con conceptos lingüísticos específicos. Cuanta más información contenga la *caption* sobre la imagen, más precisas serán estas asociaciones durante el entrenamiento.

Sin embargo, en algunos casos hay características visuales relevantes que no tienen una etiqueta textual clara. Para esos casos, se puede utilizar una palabra inventada que actúe como marcador, ayudando al modelo a aprender a asociar esa palabra con el conjunto visual correspondiente.

En el caso de los pictogramas de ARASAAC, generé las *captions* automáticamente usando *Kohya_ss*[4] junto con el modelo *BLIP*[5]. Después, añadí manualmente la palabra inventada "arasaac", que no forma parte del vocabulario original del modelo. Esta palabra permite etiquetar y aislar el estilo visual característico de los pictogramas de ARASAAC, diferenciándolo de otros estilos durante el entrenamiento.

#### 3.2.2.1. Métodos alternativos y elección final

Una alternativa habría sido reutilizar una etiqueta existente, como *pictograma*, para representar el estilo visual de ARASAAC. Por ejemplo, una imagen de un pictograma que muestre a un gato negro jugando con un ovillo podría haberse descrito con una *caption* como: ‘‘`Un gato negro jugando con un ovillo, pictograma`’’.

Este enfoque habría reforzado la asociación entre la palabra *pictograma* y el estilo gráfico deseado. Sin embargo, descarté esta opción porque los modelos base utilizados previamente generaban pictogramas de baja calidad y con un estilo visual muy diferente al de ARASAAC. Usar la etiqueta *pictograma* podría haber inducido al modelo a asociar el nuevo estilo deseado con ejemplos previos de menor calidad y estilo visual distinto, dificultando así un aprendizaje preciso del estilo de ARASAAC. Por esta razón, opté por definir una nueva etiqueta inventada, *arasaac*, asociada únicamente al estilo visual específico de los pictogramas utilizados en este proyecto.

### 3.2.3. Formato de las imágenes: RGB

Al trabajar con los pictogramas de ARASAAC, fue necesario asegurarse de que todas las imágenes estuvieran codificadas en formato RGB. Esto es crucial para evitar la pérdida de información visual, especialmente en imágenes con capas transparentes, como sucede en muchos pictogramas del centro aragonés.

Para resolver este problema, desarrollé un *script* en Python que convierte las imágenes al formato RGB de manera automática. Este *script* puede encontrarse en el repositorio de *Git* vinculado al proyecto[6].

[4] *Kohya_ss*: `https://github.com/bmaltais/kohya_ss`

[5] *BLIP* (*Bootstrapped Language-Image Pre-training*) es un modelo diseñado para generar descripciones de imágenes (*captions*) a partir de sus características visuales.

[6] Repositorio Git: `https://github.com/NILGroup/TFG-2324-Pictogramas/tree/main`

### 3.2.4. Tamaños de *dataset*

Se crearon varios *datasets* con diferentes tamaños para comparar los resultados al entrenar un LoRA utilizando distintas cantidades de imágenes de entrenamiento. Inicialmente, se pensó en emplear conjuntos de 20, 100, 200 y 400 imágenes.

Sin embargo, finalmente solo se trabajó con dos tamaños: uno pequeño de 20 imágenes y otro más grande de 100 imágenes. No se utilizaron conjuntos mayores debido a que el *dataset* de 100 imágenes ya resultaba muy costoso en términos de tiempo y recursos computacionales. Además, no se consideró necesario, ya que los resultados obtenidos con 20 imágenes fueron satisfactorios, y al emplear 100 imágenes se logró una mejora apreciable, aunque no proporcional. Esto sugiere que incrementar aún más la cantidad de imágenes podría ser inútil o incluso contraproducente, al aumentar el riesgo de sobreajuste, lo que limitaría la capacidad de generalización del modelo. A partir de ese punto, resultaba más eficiente optimizar la parametrización del entrenamiento que ampliar el tamaño del *dataset*.

### 3.2.5. *Dataset* de Regularización

Un *dataset* compuesto por imágenes y *captions* de regularización es un complemento importante en algunos entrenamientos. Este tipo de *dataset* se diferencia del de entrenamiento en que incluye conceptos y categorías que el modelo base ya domina. Su propósito principal es garantizar un equilibrio en el aprendizaje cuando se entrena un nuevo concepto.

Durante el entrenamiento de un LoRA, existe el riesgo de que el modelo se sobreajuste al *dataset* de entrenamiento, lo que podría influir negativamente en la generación de imágenes. Esto ocurre porque el sobreajuste puede hacer que el modelo base pierda versatilidad, generando únicamente imágenes muy similares a las presentes en el *dataset* del LoRA. Este problema es especialmente evidente al entrenar LoRAs de personas o personajes, ya que el modelo puede perder su capacidad de generar humanos de manera diversa y flexible.

En el caso de entrenar un estilo específico, como los pictogramas, este efecto es menos pronunciado. Esto se debe a que un estilo artístico es un concepto más acotado en comparación con la representación de seres humanos, aunque el uso exclusivo del *dataset* de entrenamiento aún puede limitar las capacidades del modelo base.

En este proyecto, se realizó una prueba utilizando un *dataset* de regularización. Sin embargo, el tiempo de entrenamiento se incrementó significativamente, ya que cada imagen, tanto del *dataset* de regularización como del de entrenamiento, se procesa de forma individual. Además, dado que el estilo de los pictogramas era un concepto concreto que el modelo base no dominaba completamente y considerando que los resultados sin este complemento eran ya muy buenos, se decidió prescindir de este tipo de *dataset*.

A pesar de ello, es importante señalar la existencia y utilidad de los *datasets* de regularización. En futuros entrenamientos donde exista un mayor riesgo de sobreajuste que pueda comprometer el conocimiento general del modelo base, este

complemento puede ser una herramienta valiosa para preservar su versatilidad y equilibrio.

### 3.2.6. Problemas con los estilos de pictogramas en ARASAAC

El *dataset* proporcionado por ARASAAC facilitó significativamente este trabajo, pero presenta ciertos problemas, como el del formato RGB, discutido en la subsección 3.2.3. Uno de los desafíos más relevantes es la selección cuidadosa de las imágenes. Para lograr un estilo consistente y uniforme entre las generaciones, es fundamental que las imágenes compartan una estética similar.

(a) Estilo base y objetivo (b) Estilo stickman (c) Estilo extraño

Figura 3.2: Distintos estilos presentes en el *dataset* de la web ARASAAC

En la figura 3.2 se observan tres estilos comunes dentro del *dataset* de ARASAAC. Aunque existen otros, estos tres predominaban en los primeros *datasets* que utilicé para el entrenamiento. Tras emplear estos *datasets* en los primeros LoRAs entrenados, descubrí que la coexistencia de estos estilos artísticos generaba resultados inconsistentes y alejados del objetivo deseado.

Si el propósito era alcanzar una sencillez como la del estilo base mostrado en 3.2a, la presencia de estilos como el stickman (3.2b), con una falta de detalle más pronunciada, o un estilo más caricaturesco y exagerado, como el de 3.2c, contaminaba el entrenamiento y perjudicaba la coherencia del modelo.

La eliminación de estos estilos heterogéneos y la homogeneización del *dataset* en torno a un único estilo resultaron en una mejora notable de los resultados, logrando una mayor consistencia en los LoRAs generados.

### 3.2.7. Características finales del *dataset*

El *dataset* final utilizado para entrenar el modelo LoRA fue el resultado de un proceso de depuración y refinamiento orientado a maximizar la calidad del entrenamiento. A continuación, se resumen sus principales características:

- **Formato:** Todas las imágenes se convirtieron al formato RGB para evitar pérdidas de información.
- **Estilo visual:** Se realizó una selección entre los distintos estilos de pictogramas disponibles en la base de datos de ARASAAC, creando dos tipos de conjuntos: estilo base (figura 3.2a) y estilo *stickman* (figura 3.2b). Esta selección garantizó la coherencia visual necesaria para lograr un entrenamiento consistente.
- **Tamaño:** Se definieron dos variantes: una de 20 imágenes, con menor coste de entrenamiento, y otra de 100 imágenes, que ofrecía una mayor variedad de información a cambio de un mayor coste computacional.
- **Etiquetado:** Cada imagen fue acompañada por un archivo `.txt` con la *caption* correspondiente. Este archivo se generó con el modelo de etiquetado BLIP y se añadió la palabra "arasaac" como marcador de estilo exclusivo.

Estas características permitieron construir conjuntos de datos variados, adecuados para experimentar con diferentes niveles de costo computacional y obtener resultados de calidad con distintos volúmenes de recursos.

## 3.3. Entrenamiento de LoRAs: desde Google Colab hasta entrenamiento Local

En esta sección se describen los entrenamientos realizados con el objetivo de obtener un LoRA de calidad, optimizado para funcionar de manera eficiente y con un uso reducido de recursos. Se detalla la evolución del proceso, desde los primeros intentos en plataformas *online* como Google Colab hasta la migración a un entorno local más flexible y controlado.

### 3.3.1. Entrenamiento inicial en Google Colab

Los primeros intentos de entrenamiento del LoRA se realizaron utilizando notebooks en Google Colab. Esto se debió a los exigentes requisitos de memoria RAM y VRAM necesarios para entrenar modelos de este tipo. Sin embargo, este enfoque presentó numerosos problemas. Al tratarse de un campo en constante evolución, los notebooks quedaban rápidamente desactualizados, ya sea por incompatibilidades entre versiones de PyTorch y Python o porque algunas herramientas dejaban de funcionar. A pesar de explorar múltiples notebooks, finalmente encontré el repositorio de Abhishek Thakur (Thakur-Git-Notebooks (2024)), que proporcionaba una solución funcional utilizando Hugging Face junto con ngrok[7].

Aun así, el entrenamiento en Google Colab estaba limitado por restricciones en el tiempo de uso de la GPU gratuita. Las sesiones se interrumpían tras 30 minutos

[7]Notebook de entrenamiento: `https://colab.research.google.com/github/huggingface/autotrain-advanced/blob/main/colabs/AutoTrain_ngrok.ipynb`

de inactividad o debido a alta demanda, lo que resultaba en la cancelación del entrenamiento y la pérdida de todo el progreso.

Ajustando el tamaño del *dataset*, el número de épocas y los pasos (steps), logré obtener los primeros LoRAs, pero los resultados no fueron satisfactorios. Además de las limitaciones de tiempo, los parámetros disponibles en estas herramientas eran limitados en comparación con los métodos de entrenamiento locales.

### 3.3.2. Resultados de los LoRAs entrenados en Google Colab

Varios LoRAs fueron entrenados en esta etapa, pero muchos no lograron alterar significativamente la red neuronal. Los que sí lo hicieron generaron resultados inconsistentes, aunque demostraron que alcanzar el objetivo era posible con una mejor parametrización.

En las figuras 3.3 y 3.4 se comparan imágenes generadas con el modelo *DreamShaper*, utilizando un mismo *prompt* y semilla. La primera fila muestra la imagen base (sin LoRA aplicado) y las siguientes filas ilustran la evolución de los resultados a medida que se entrenaban nuevos LoRAs.

Los primeros experimentos se realizaron con *datasets* pequeños (20 imágenes) y 500 pasos de entrenamiento concentrados en una única época, es decir, una sola pasada completa por el conjunto de datos. Esta configuración, aunque simple, puede limitar el aprendizaje al no permitir al modelo reforzar patrones útiles en iteraciones sucesivas. Posteriormente, en el entrenamiento local, se descubrió que era más eficaz utilizar múltiples épocas con menos pasos en cada una. Esta estrategia permite al modelo ajustar sus pesos de forma más gradual, reforzar información importante para el estilo objetivo al ver repetidamente las mismas imágenes, y evitar tanto el sobreajuste (solo poder generar imágenes muy parecidas a las del entrenamiento) como la convergencia errónea (cuando el modelo aprende relaciones inadecuadas o inestables, que degradan la calidad de las imágenes generadas).

Con el aumento del número de pasos a 2000 (manteniendo una sola época) y la ampliación del tamaño del *dataset*, los resultados se aproximaron al estilo buscado, similar al de ARASAAC (figura 1.1), caracterizado por ilustraciones simples con líneas gruesas.

La figura 3.5 muestra el mejor LoRA obtenido en Colab, correspondiente a la sexta imagen de las figuras 3.3 y 3.4. Aunque este LoRA aprendió correctamente ciertas características, como el grosor de las líneas, los estilos de los humanos eran inconsistentes, alternando entre artes detallados y simplificados. Además, los ojos tendían a un estilo anime o a un aspecto espeluznante, y las paletas de colores se tornaban apagadas, como se observa en las imágenes de la montaña o el hada, donde predominan tonos grises y oscuros.

Estos resultados no pudieron mejorarse en Colab, lo que llevó a la decisión de migrar a un entorno de entrenamiento local.

Figura 3.3: Resultados en Colab: prompt “chef”

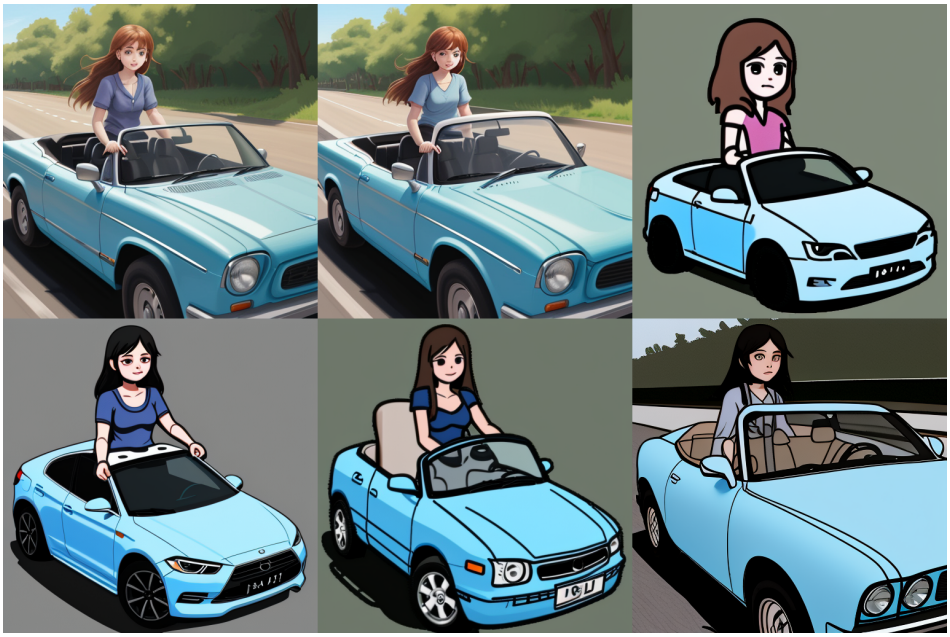


Figura 3.4: Resultados en Colab: prompt “girl driving a blue car”

### 3.3.3. Entrenamientos en Kohya_ss

A continuación, se describe el proceso de entrenamiento realizado en local mediante herramientas de software libre, en contraste con las aplicaciones online utilizadas anteriormente desde Google Colab. Esta transición permitió una mayor personalización y control sobre los parámetros del entrenamiento, mejorando la eficiencia y calidad del modelo generado.

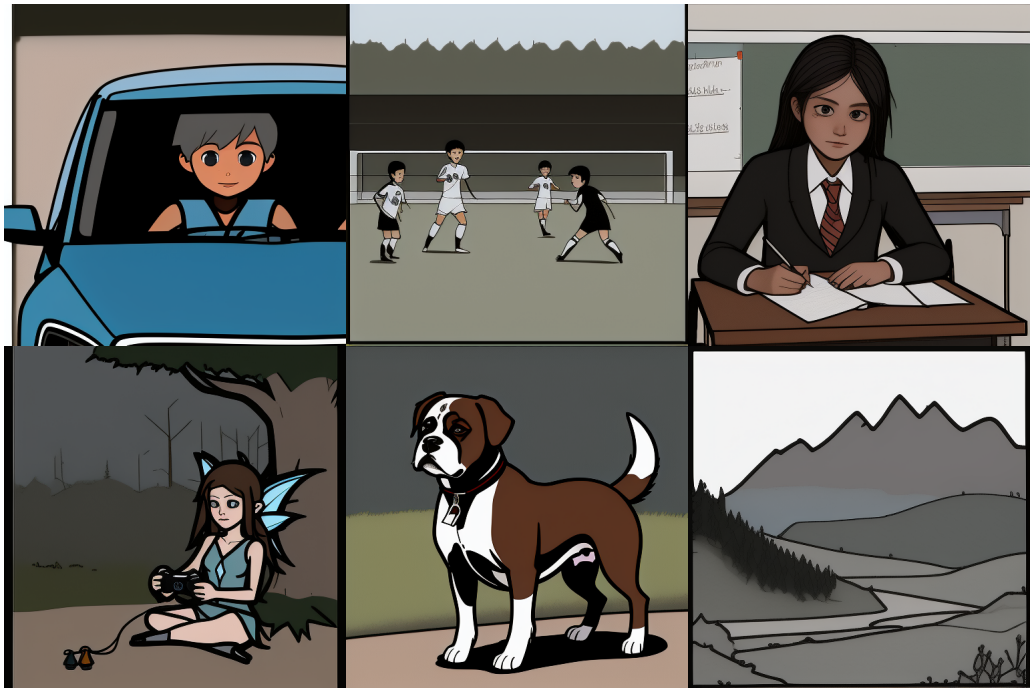


Figura 3.5: Mejor LoRA entrenado en Colab. Prompt: "girl driving a blue car"

#### 3.3.3.1. Migración y Configuración Inicial

Debido a los problemas recurrentes con el entrenamiento en Google Colab y las limitaciones de sus LoRAs, opté por buscar una alternativa para entrenar el LoRA localmente. Mi PC cuenta con una tarjeta gráfica adecuada (una 3070 RTX con 8 GB de RAM virtual), lo que me permitiría utilizar herramientas avanzadas para parametrizar el entrenamiento sin preocuparme por las desconexiones de Google Colab. Aunque seguiría enfrentándome a la limitación de tiempo para evitar un uso excesivo de la GPU, dispondría de más flexibilidad.

Tras investigar y consultar qué herramientas utilizaban otros usuarios, decidí emplear Kohya_ss (bmaltais (2024)), una herramienta que permite entrenar LoRAs localmente. Además, ofrece métodos avanzados de *captions* por IA y permite configurar múltiples parámetros que no estaban disponibles en notebooks o aplicaciones web. No obstante, al principio enfrenté problemas de compatibilidad con PyTorch, Python y XFormers.

Finalmente, tras probar distintas soluciones sin éxito, encontré una guía en un post de Reddit en r/StableDiffusion[8], que explicaba cómo instalar *Kohya_ss* utilizando un entorno virtual de Python. El principal problema era que tenía instalada una versión de PyTorch (una biblioteca muy utilizada para el desarrollo de modelos de aprendizaje profundo) que solo funcionaba con la CPU, mientras que *Kohya_ss* requería una versión compatible con CUDA para poder usar la tarjeta gráfica. Además, Xformers (una biblioteca complementaria que mejora la eficiencia del entrenamiento de modelos de difusión) también presentaba conflictos de versión. La guía me

[8]Guía en un post de Reddit: `https://www.reddit.com/r/StableDiffusion/comments/1dxvjya/how_to_install_kohya_ss_gui_on_cuda_121_easy/`

permitió crear un entorno virtual con las versiones correctas, lo que resolvió todos los problemas de compatibilidad y me permitió aprovechar la GPU para entrenar de forma óptima.

Una vez instalada correctamente la herramienta, comencé a trabajar con Kohya_ss. El *dataset*, descrito en la sección 3.2, fue completado en esta etapa. Hasta ese momento había trabajado sin *captions,* ya que las herramientas online que utilicé anteriormente. Por ejemplo, el notebook de Hugging Face con *ngrok*, solo permitía subir imágenes a través de una interfaz web predefinida. Dado que este notebook simplemente enlazaba Google Colab con una aplicación externa y no ofrecía opciones para modificar el flujo de entrenamiento, no era posible añadir *captions* personalizados, lo cual limitaba considerablemente la calidad del entrenamiento y del LoRA resultante.

#### 3.3.3.2. Evolución de los Entrenamientos

**Primeras versiones** Los primeros entrenamientos en Kohya_ss mostraron mejoras significativas en comparación con los realizados en Google Colab. Inicialmente, distribuía todos los pasos de entrenamiento en una sola época. Sin embargo, descubrí que era más eficiente realizar muchas épocas con pocos pasos, alrededor de 100-200 por época.

En cuanto a los parámetros, en un principio me guié por información dispersa, suponiendo erróneamente que valores más altos siempre generarían mejores resultados. Esto llevó a entrenamientos excesivamente largos, con más de 10,000 pasos distribuidos en 60 épocas, algunos de los cuales fueron interrumpidos al considerar inviable arriesgar la GPU durante 24 horas de uso continuo. Aunque completé entrenamientos de hasta 7 horas, sus resultados, si bien superiores a los obtenidos en Colab, no justificaban el esfuerzo, pues el sobreentrenamiento afectaba negativamente la calidad del LoRA.

**Iteraciones y ajustes** A medida que identificaba problemas, realicé diversos ajustes a la parametrización:

- **Ajuste en el número de pasos y épocas:** Opté por trabajar en un rango de 2000 a 5000 pasos, con 100 pasos por época y entre 20 y 40 épocas. Esto mejoró significativamente la precisión del LoRA.
- **Remodelación del *dataset*:** Al principio utilicé el mismo *dataset* que en Colab, pero, como se discutió en 3.2.6, contenía estilos de pictogramas inconsistentes. Al solucionar este problema, logré un *dataset* más homogéneo, lo que mejoró la consistencia de los resultados.
- **LR Scheduler:** El *LR Scheduler* (planificador de tasa de aprendizaje) es un componente que ajusta automáticamente la learning rate (velocidad de aprendizaje) a lo largo del entrenamiento. Inicialmente utilicé el scheduler "constant", que mantiene la tasa de aprendizaje fija durante todo el entrenamiento. Sin

embargo, investigando descubrí que optimizadores como Adafactor o Prodigy funcionan mejor con un scheduler de tipo "cosine", el cual reduce progresivamente la tasa de aprendizaje con forma de onda coseno. Esto permite un entrenamiento más estable y evita sobreajustes en fases finales.

- **LR warmup (% of total steps):** El *LR warmup* es una técnica que consiste en comenzar el entrenamiento con una tasa de aprendizaje muy baja y aumentarla gradualmente hasta alcanzar la tasa configurada. En este caso, establecí un LR warmup del 10 % de los pasos totales, lo cual está recomendado para modelos de tamaño pequeño o medio. Esta configuración ayuda a evitar inestabilidades iniciales en el entrenamiento.
- **Network Rank (Dimension):** Define el número de dimensiones o "direcciones" que el LoRA puede usar para modificar el modelo base. Técnicamente, LoRA introduce una pequeña red dentro del modelo grande, que aprende solo una parte del comportamiento. Cuanto mayor sea el rank, mayor será la capacidad de esta red para representar variaciones respecto al modelo base. Si el rank es demasiado bajo, el LoRA puede quedarse corto en expresividad; si es demasiado alto, ocupa más memoria y puede sobreentrenar. En mi caso, un rank de 32 fue suficiente para aprender el estilo de los pictogramas sin sobrecargar la GPU.
- **Network Alpha:** Es un parámetro de escalado que regula cuánto influye la red LoRA sobre el modelo base durante el entrenamiento. Si se deja muy alto, el modelo puede desviarse demasiado del original, si es muy bajo, no aprenderá cambios significativos. Un valor de 1 es un punto de equilibrio recomendado y ayuda a mantener la estabilidad del entrenamiento.
- **Cambio en el modelo base:** Originalmente entrenaba los LoRAs sobre *DreamShaper*, pero finalmente opté por Stable Diffusion 1.5 vanilla, cuyos pesos y text encoder son más limpios. Este cambio mejoró la consistencia y versatilidad del LoRA, mejorando su compatibilidad con otros modelos basados en SD 1.5.

Estos cambios impactaron positivamente en la calidad de las imágenes generadas, reduciendo problemas de líneas y mejorando la representación del estilo ARASAAC.

**Impacto del tiempo de entrenamiento** Inicialmente asumí que tiempos largos de entrenamiento, con más pasos y un mayor tamaño de *dataset*, producirían mejores resultados. Sin embargo, con la optimización de parámetros, observé que entrenamientos de 3-4 horas generaban LoRAs de alta calidad. Posteriormente, con un *Network Rank* reducido a 32 y el uso del optimizador *Prodigy*, conseguí entrenamientos de calidad equivalente en tan solo 1 hora.

**Progresión hacia el LoRA más exitoso** Tras múltiples iteraciones, llegué a una combinación de parámetros que ofreció buenos resultados con bajo coste computacional:

- *Dataset* homogéneo en estilo
- 2000–4000 pasos totales
- 20–40 épocas (épocas cortas, con 100 pasos por época)
- *clip skip*: 2 (para omitir las dos primeras capas del *text encoder* y mejorar la expresividad de los prompts)
- Optimizadores: *Adafactor* o *Prodigy*
- *LR Scheduler*: *cosine*
- *LR warmup*: 10 %
- *Network Rank*: 32–64
- *Network Alpha*: 1

Esta configuración equilibró calidad y eficiencia, permitiendo entrenamientos efectivos en tiempos de entre 1 y 2 horas. Otros parámetros apenas variaron y se documentan en el archivo de configuración del proyecto en Hugging Face[9].

| **LoRA** | **Número de pasos** | **Número de épocas** | **Tamaño del *dataset*** | **Resultados** |
|---|---|---|---|---|
| LoRA-1 | 1600 | 4 | 20 imágenes | Estilo inconsistente |
| LoRA-2 | 2000 | 10 | 20 imágenes | Mejora moderada |
| LoRA-3 | 10000 | 10 | 20 imágenes | Estilo aceptable |
| LoRA-4 | 12000 | 60 | 100 imágenes | Mejora considerable |
| LoRA-5 | 8000 | 32 | 100 imágenes | Muy buenos resultados |
| LoRA-6 | 4000 | 40 | 100 imágenes | Excelente calidad |

Tabla 3.3: Comparación de parámetros entre los LoRAs.

#### 3.3.3.3. Influencia de los Parámetros en los Resultados

Los parámetros más importantes para el aprendizaje y la optimización del LoRA son:

- **Número de pasos y épocas:** Lo ideal es trabajar entre 2000 y 5000 pasos, distribuidos en épocas de 100-200 pasos. Un número excesivo de pasos puede llevar a sobreentrenamiento.
- **Tamaño del *dataset*:** Aunque es preferible trabajar con al menos 100 imágenes, es posible obtener resultados decentes con menos, siempre que las imágenes sean representativas del estilo deseado.

[9]Hugging Face del proyecto: `https://huggingface.co/antuna01/Pictogram-LoRAs/tree/main`

- **Network Rank (Dimension) y Network Alpha:** Un *Network Rank* de 32 es suficiente para modelos básicos, mientras que el *Network Alpha* debe ajustarse a 1 con optimizadores adaptativos y siempre mantenerse inferior al *Network Rank*.
- **Learning rate:** Valores inadecuados pueden causar sobreajuste o un aprendizaje deficiente. Los optimizadores adaptativos ajustan este parámetro automáticamente.

#### 3.3.3.4. Fallos y Lecciones Aprendidas

Durante el proceso de entrenamiento se cometieron diversos errores que permitieron identificar buenas prácticas para mejorar los resultados. Los errores más relevantes fueron:

- **Usar un *LR Scheduler* constante con optimizadores adaptativos**: al mantener una tasa de aprendizaje fija, se desaprovechaba la capacidad de ajuste dinámico del optimizador, lo que dificultó la convergencia y estabilidad del entrenamiento.

  **Solución:** Planificadores dinámicos como *cosine*.

- **Utilizar un número inadecuado de pasos y épocas**: entrenamientos excesivamente largos provocaban sobreajuste, reduciendo la calidad y adaptabilidad del modelo. En cambio, entrenamientos demasiado breves no lograban capturar bien el estilo del dataset.

  **Solución:** Distribución de pocos pasos (100 - 200) y muchas épocas (20 - 40) fue la mejor solución.

- **Emplear valores elevados de *Network Rank***: esto incrementó innecesariamente el consumo de VRAM, sin mejoras visibles en los resultados. Además, redujo la eficiencia al obtener una red LoRA mas grande

  **Solución:** Dimensiones mas limitadas entre 16 y 32.

- **Crear un *dataset* con estilos artísticamente diversos**: la falta de homogeneidad impidió que el modelo aprendiera un estilo coherente, lo que resultó en pictogramas con rasgos contradictorios o inconsistentes.

  **Solución:** Eliminar del *dataset* imágenes que pudieran distanciarse del estilo objetivo.

- **Colocar el *dataset* en la misma carpeta que los resultados**: Debido a cómo funciona Kohya_ss, al preparar los datos para el entrenamiento, el programa genera una carpeta con las imágenes procesadas y sus correspondientes *captions*. Si esta carpeta se encuentra dentro del mismo directorio que el *dataset* original, las nuevas imágenes generadas pueden volver a incluirse en el conjunto de entrenamiento, creando un bucle que introduce duplicados y ruido, lo que deriva en un entrenamiento ineficiente y con menor calidad.

**Solución:** Colocar el *dataset* en una carpeta distinta a la que se usará para guardar la información y los resultados del entrenamiento.

La lección principal es que disponer de más recursos o tiempo de entrenamiento no garantiza mejores resultados. Con parámetros optimizados y un *dataset* cuidadosamente depurado, se pueden obtener LoRAs de alta calidad en menos tiempo y con un uso eficiente de los recursos disponibles.

## 3.4. Análisis de Resultados

En esta sección se comparan los distintos LoRAs entrenados con el objetivo de identificar el más adecuado para la generación de pictogramas. Tras seleccionar el modelo con mejor desempeño cualitativo, se realizan pruebas más exhaustivas para evaluar su capacidad de generar imágenes con la calidad, el estilo y las personalizaciones necesarias para su uso en contextos de comunicación aumentativa y alternativa.

### 3.4.1. Prompts de Prueba

Para evaluar los LoRAs desarrollados, se diseñaron una serie de *prompts de prueba* con el objetivo de comparar los resultados obtenidos en distintos entrenamientos. Estos *prompts* representan conceptos diversos, desde simples hasta complejos, permitiendo identificar fortalezas y debilidades en las imágenes generadas. Los *prompts* utilizados fueron los siguientes:

1. chef
2. kids playing soccer
3. marriage proposal
4. school class
5. boxer girl
6. boxer dog
7. socks
8. car race
9. mountains landscape
10. a boy driving a blue car
11. forest fairy playing video games, video game controller

El diseño de estos *prompts* buscaba evaluar diferentes aspectos. Por ejemplo, la generación de figuras humanas, tanto en situaciones simples como el caso de un chef (*prompt* 1) como en escenarios más complejos, como un grupo de niños jugando al fútbol (*prompt* 2). También se evaluó la posible corrupción semántica de los textos con *prompts* similares, como "boxer girl" y "boxer dog" (*prompts* 5 y 6). Si el LoRA funciona adecuadamente, debería generar imágenes coherentes en ambos casos.

Además, se analizaron la generación de objetos y paisajes con *prompts* como "socks" (*prompt* 7), "car race" (*prompt* 8) y "mountains landscape" (*prompt* 9). Finalmente, se probaron *prompts* más complejos, como "a boy driving a blue car" (*prompt* 10) y "forest fairy playing video games" (*prompt* 11), para evaluar la capacidad del modelo de manejar combinaciones conceptuales.

Se verificaron, además, características útiles para aplicaciones reales, como:

- Generar pictogramas en blanco y negro, lo cual resulta útil en casos donde los colores puedan distraer al receptor.
- Generar pictogramas con fondo blanco para mayor claridad.

### 3.4.2. Comparación entre los Mejores LoRAs

Todos los LoRAs evaluados cumplieron con el requisito de ser generados en equipos con especificaciones modestas, logrando tiempos de generación de aproximadamente cinco segundos por imagen (una vez cargado el modelo). En concreto, es posible utilizarlos con un entorno con los requisitos mínimos para ejecutar Stable Diffusion 1.5: una GPU NVIDIA con 4 GB de VRAM, 8 GB de RAM y un procesador convencional. Esto permitió generar pictogramas incluso en el despacho del tutor, evidenciando la viabilidad del sistema en equipos no especializados.

La comparación entre ellos se realizó agrupando los *prompts de prueba* en cuatro categorías principales:

- Generación de personas
- Corrupción de texto
- Objetos y lugares
- Generaciones complejas

Para cada categoría se evaluaron cualitativamente las imágenes generadas por los diferentes modelos siguiendo tres criterios principales: la fidelidad al *prompt* (es decir, que la imagen represente correctamente lo solicitado), la similitud visual con el estilo de los pictogramas de ARASAAC, y la coherencia interna de la imagen (por ejemplo, que las figuras humanas estén bien formadas o que no haya errores evidentes). Aunque esta evaluación fue principalmente visual y, por tanto, subjetiva, se intentó mantener un criterio constante y equitativo a lo largo de todas las comparaciones.

**Generación de personas** Se evaluó la precisión en la representación de figuras humanas, tanto en escenarios simples como en contextos más complejos. Las figuras 3.6 y 3.8 muestran que, al generar imágenes de una o dos personas, casi todos los modelos producen resultados lógicos y comprensibles. Sin embargo, cuando se intenta representar escenas con múltiples figuras humanas o acciones más complejas, como jugar al fútbol, los resultados presentan mayor variabilidad. En las figuras 3.9 y 3.7, los primeros LoRAs evidencian inconsistencias y errores en las generaciones, errores como perder mucha coherencia en el entorno que situa la acción, en este caso las clases de alumnos y los campòs de futbol. El otro error claro seria el detalle de la anotomia humana al generar muchas personas, este problema se ve bien en la 3.9 con los 4 primeros LoRAs.

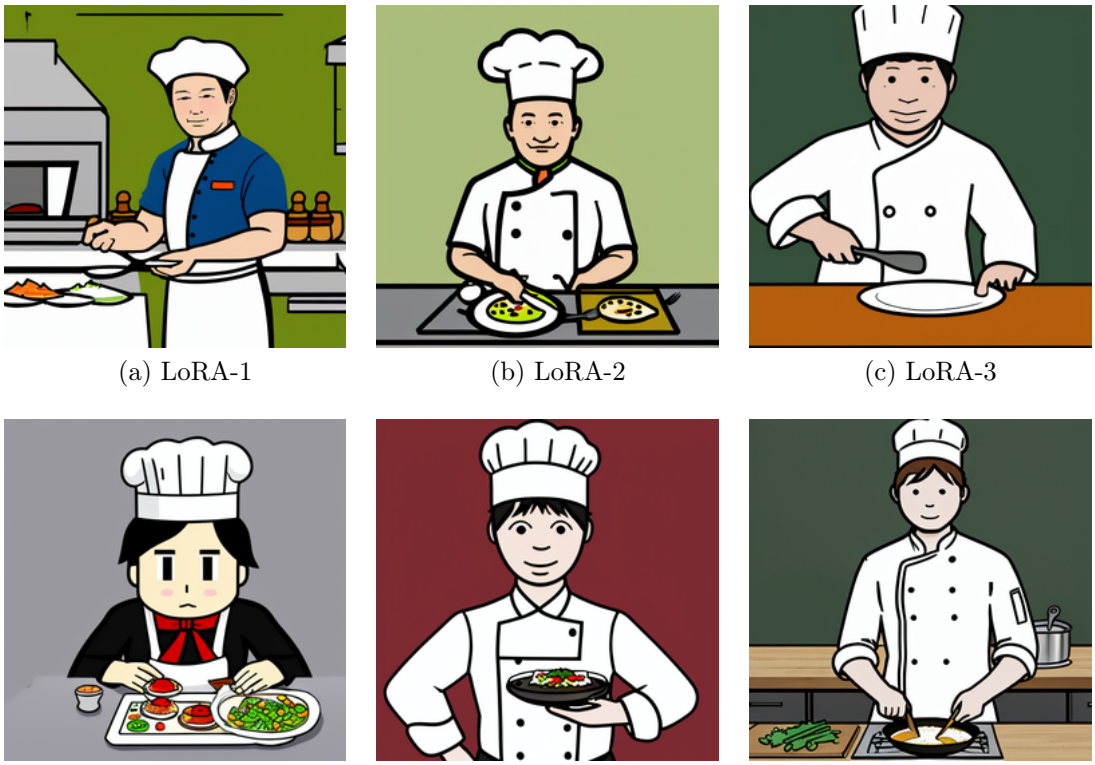
A medida que se mejoraron los parámetros de entrenamiento, incluyendo el número de pasos, las épocas y el tamaño del *dataset*, los modelos ganaron significativamente en consistencia y calidad. Los resultados más destacados corresponden al LoRA-5 y al LoRA-6, cuyos outputs no solo muestran mayor calidad, sino también una coherencia estilística en las diferentes pruebas realizadas.

**Generación de personas** Se evaluó la precisión en la representación de figuras humanas, tanto en escenarios simples como en contextos más complejos. Las figuras 3.6 y 3.8 muestran que, al generar imágenes de una o dos personas, casi todos los modelos producen resultados lógicos y comprensibles. Sin embargo, cuando se intenta representar escenas con múltiples figuras humanas o acciones más complejas, como jugar al fútbol, los resultados presentan mayor variabilidad. En las figuras 3.9 y 3.7, los primeros LoRAs evidencian inconsistencias claras, como la pérdida de coherencia en el entorno en el que se sitúa la acción (por ejemplo, el aula de clase o el campo de fútbol), y errores anatómicos al generar varias figuras humanas. Este último problema es especialmente evidente en la figura 3.9, donde los cuatro primeros LoRAs muestran cuerpos mal formados.

A medida que se mejoraron los parámetros de entrenamiento, incluyendo el número de pasos, las épocas y el tamaño del *dataset*, los modelos ganaron significativamente en consistencia y calidad. Los resultados más destacados corresponden al LoRA-5 y al LoRA-6, cuyos outputs no solo muestran mayor calidad, sino también una coherencia estilística en las diferentes pruebas realizadas.

**Corrupción de texto** En esta categoría se analizó la capacidad del modelo para distinguir entre conceptos semánticamente similares pero distintos, como los utilizados en los *prompts* “boxer girl” y “boxer dog”. Los resultados se presentan en las figuras 3.10 y 3.11.

Todos los LoRAs lograron generar un perro con una consistencia y calidad notable. Sin embargo, el problema surge al intentar generar la chica boxeadora. En la mayoría de los casos, los LoRAs confundieron ambos conceptos, mezclando características del perro bóxer con las de la figura humana. Solo el LoRA-4 y el LoRA-6 consiguieron aislar correctamente los conceptos de humano y animal, siendo el LoRA-6 el más destacado. Este no solo representó el concepto humano de forma precisa, sino



(a) LoRA-1 (b) LoRA-2 (c) LoRA-3

(d) LoRA-4 (e) LoRA-5 (f) LoRA-6

Figura 3.6: Comparación de generación de personas.
Prompt: “chef, arasaac” Modelo: DreamShaper

que también añadió la característica de que el pictograma fuera una boxeadora.

(a) LoRA-1 (b) LoRA-2 (c) LoRA-3

(d) LoRA-4 (e) LoRA-5 (f) LoRA-6

Figura 3.7: Comparación de generación de personas.
Prompt: "kids playing soccer, arasaac" Modelo: DreamShaper

**Objetos y lugares** Esta categoría se enfocó en la generación de objetos y paisajes, evaluando la consistencia en su representación visual. Todos los LoRAs superaron las pruebas presentadas en las figuras 3.13 y 3.14, aunque con variaciones en la calidad del dibujo.

Es importante destacar un fallo observado en los LoRAs 1 y 2 al generar calcetines (3.12), donde confundieron el concepto deseado y lo combinaron con características de animales. Este error subraya la importancia de mejorar la parametrización para evitar confusiones conceptuales en escenarios similares.

**Generaciones complejas** En esta categoría se evaluó la capacidad de los LoRAs para manejar *prompts* que combinan múltiples conceptos o incluyen detalles específicos. Los resultados en este ámbito fueron sorprendentes, ya que ninguno de los modelos falló al representar conceptos extraños o muy alejados de los buscados.

Sin embargo, se observó una clara mejora en la calidad de las generaciones cuando se utilizó un *dataset* de 100 imágenes en comparación con uno de 20. Dentro del grupo entrenado con 100 imágenes, los resultados mejoraron significativamente a medida que la parametrización del entrenamiento se optimizó. El LoRA-6, al igual que en los apartados anteriores, sobresalió por su calidad excepcional, destacándose por encima de los demás modelos.

(a) LoRA-1 (b) LoRA-2 (c) LoRA-3

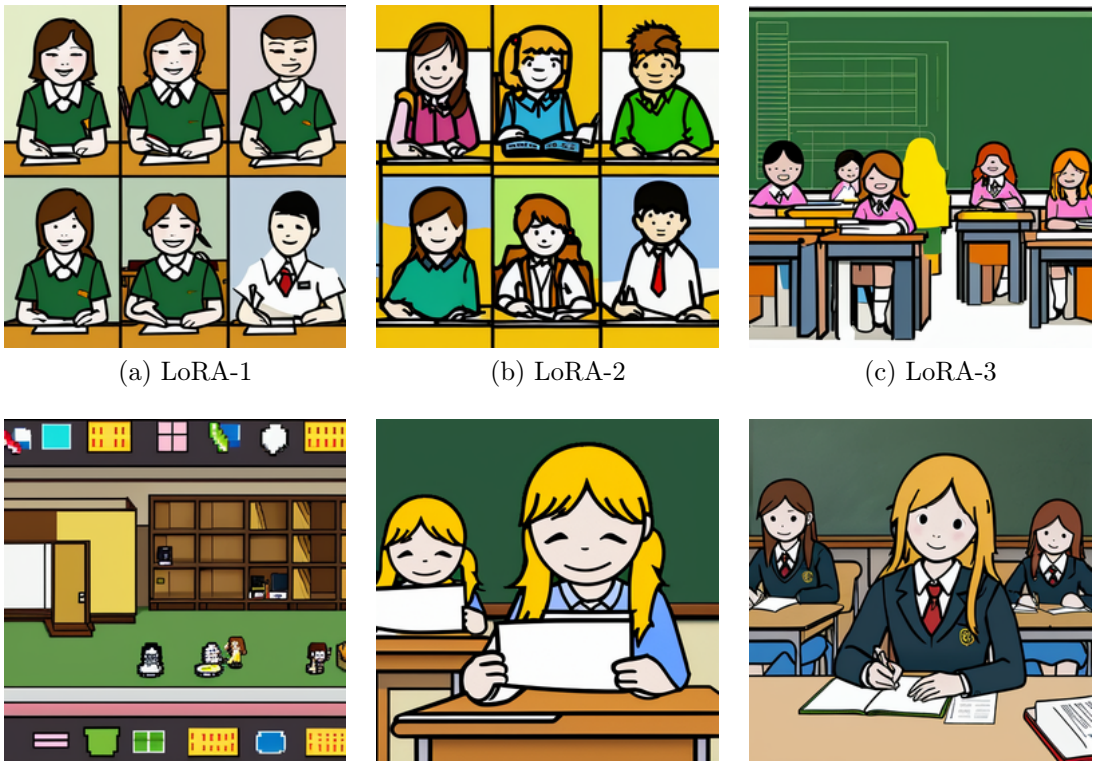
(d) LoRA-4 (e) LoRA-5 (f) LoRA-6

Figura 3.8: Comparación de generación de personas.
Prompt: "marriage proposal, arasaac" Modelo: DreamShaper

De todos los LoRAs evaluados, el LoRA-6 se destacó como el mejor en todas las pruebas realizadas, mostrando un estilo consistente, claro y casi idéntico al de ARASAAC. Los primeros LoRAs (1, 2 y 3) lograron algunas imágenes aceptables, pero sus limitaciones y falta de fiabilidad fueron evidentes. Estas deficiencias se atribuyen al uso de un *dataset* reducido y a una parametrización menos refinada, con un número insuficiente o excesivo de pasos y épocas de entrenamiento, como se detalla en la tabla 3.3.

El LoRA-4 mostró un desempeño notable, aunque mejorable en algunos aspectos, como la representación de niños jugando al fútbol o en la clase de colegio. Además, su estilo se acerca más al *pixel art* que al estilo de ARASAAC, lo que lo posiciona por debajo del LoRA-6 en términos de fidelidad estilística.

Por otro lado, el LoRA-5 ocupó el segundo lugar en calidad general. Aunque logró buenos resultados en la mayoría de las pruebas, se quedó atrás respecto al LoRA-6 en la categoría de corrupción semántica, fallando gravemente al generar la chica boxeadora (3.10e). Además, aunque consistente, su representación de figuras humanas fue menos uniforme que la del LoRA-6. Esto se observa claramente en la generación de humanos en la comparativa de la clase de colegio (3.9), donde el LoRA-6 destaca por mantener rasgos visuales coherentes entre todas las figuras.



(a) LoRA-1 (b) LoRA-2 (c) LoRA-3

(d) LoRA-4 (e) LoRA-5 (f) LoRA-6

Figura 3.9: Comparación de generación de personas.
Prompt: "school class, arasaac" Modelo: DreamShaper

### 3.4.3. LoRA Seleccionado: LoRA-6

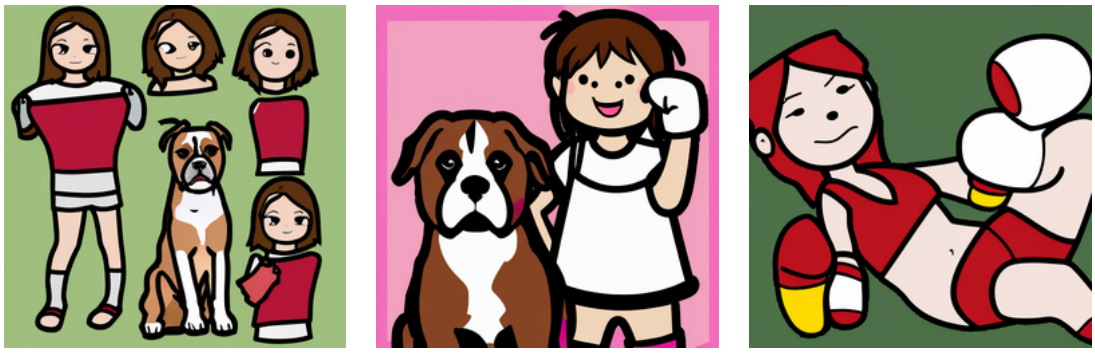
El LoRA seleccionado fue el que se entrenó utilizando el optimizador *Prodigy* y el modelo base *Stable Diffusion 1.5 vanilla*. Este último incrementó significativamente la versatilidad, permitiendo que el LoRA generara resultados consistentes incluso al usarse con modelos distintos al de entrenamiento, como *Realistic Vision* o *DreamShaper*.

#### Parámetros de entrenamiento

Los parámetros del entrenamiento utilizados en este LoRA están disponibles en el archivo *configuration.json* del repositorio en *Hugging Face*[10], aunque se destacan los siguientes:

- Épocas: 40
- *Batch Size*[11]: 4

[10] `https://huggingface.co/antuna01/Pictogram-LoRAs/tree/main`
[11] El *Batch Size* es el número de muestras procesadas simultáneamente durante una iteración del entrenamiento de un modelo.



(a) LoRA-1 (b) LoRA-2 (c) LoRA-3

(d) LoRA-4 (e) LoRA-5 (f) LoRA-6

Figura 3.10: Comparación de corrupción de texto.
Prompt: "boxer girl, arasaac" Modelo: DreamShaper

- *Dataset*: 100 imágenes con 4 repeticiones por imagen
- Pasos: 4000 pasos (100 imágenes x 4 repeticiones x 40 épocas / 4 *Batch Size*)
- Optimizador: *Prodigy*
- Planificador de LR (*Scheduler*): *Cosine*
- *LR warmup* ( % del total de pasos): 10 %
- Resolución máxima: 512x512
- *Network Rank (Dimension)*: 32
- *Network Alpha*: 1

A pesar de contar con un *dataset* de solo 100 imágenes y un total de 4000 pasos, el tiempo de entrenamiento se redujo a aproximadamente una hora gracias a las optimizaciones realizadas. Entre estas, el ajuste del *Network Rank* y el uso de un planificador y optimizador de *learning rate* adecuados fueron determinantes para mejorar tanto el rendimiento como los resultados obtenidos.

(a) LoRA-1 (b) LoRA-2 (c) LoRA-3

(d) LoRA-4 (e) LoRA-5 (f) LoRA-6

Figura 3.11: Comparación de corrupción de texto.
Prompt: "boxer dog, arasaac" Modelo: DreamShaper

### Configuración de generación

Para obtener los mejores resultados al usar este LoRA, se recomienda la siguiente configuración en la interfaz de *WebUI Forge*[12]:

- *Sampling steps*: ±20
- Método de muestreo: *DPM++ 2M*
- Tipo de planificador (*Schedule type*): *Karras* o *Automatic* (aunque casi cualquier opción ofrece buenos resultados).
- Size: 512x512
- *Distilled CFG Scale*: 3.5
- *CFG Scale*: 7
- *Clip skip*: 2

[12]WebUI Forge: `https://github.com/lllyasviel/stable-diffusion-webui-forge?tab=readme-ov-file`

(a) LoRA-1 (b) LoRA-2 (c) LoRA-3

(d) LoRA-4 (e) LoRA-5 (f) LoRA-6

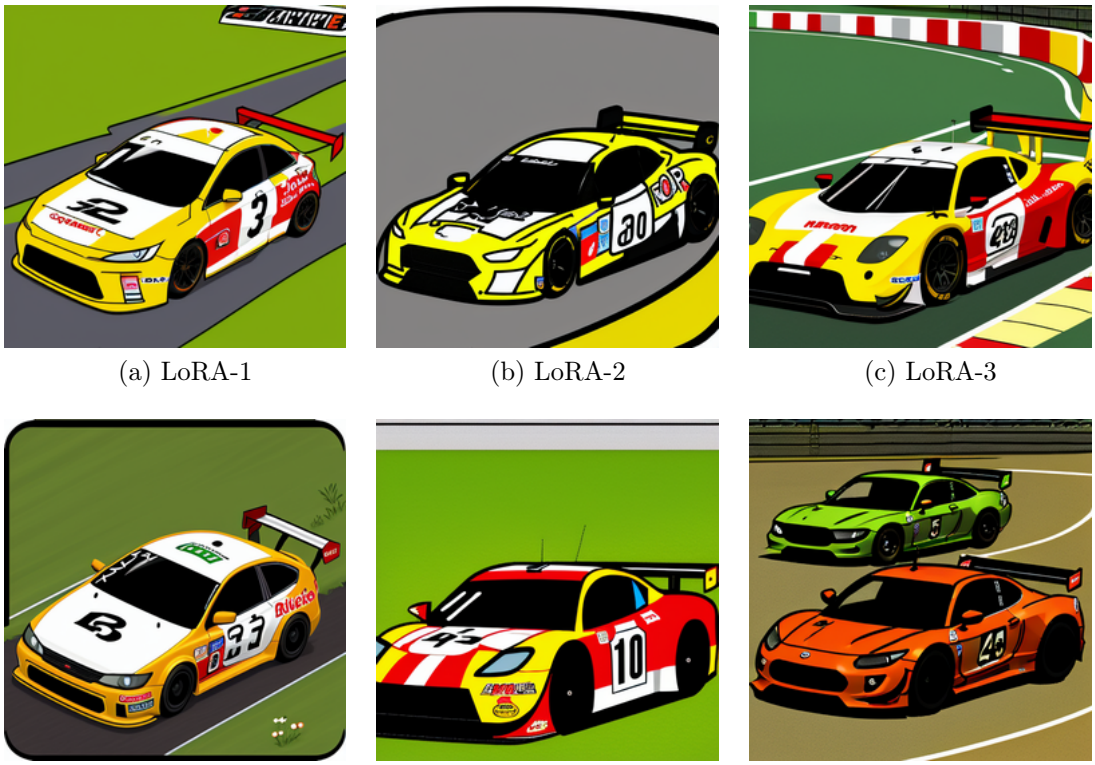
Figura 3.12: Comparación de objetos y lugares.
Prompt: "socks, arasaac" Modelo: DreamShaper

- Peso del LoRA: 0.7 (funciona bien en un rango de [0.6, 1.1], dependiendo del resultado deseado).
- Modelos base compatibles: Basados en *Stable Diffusion 1.5*, como *Realistic Vision* y *DreamShaper*.

#### 3.4.3.1. Pruebas Adicionales

Con este LoRA se profundizaron más las pruebas y se intentaron generar en distintos modelos basados en SD 1.5, como se pueden ver las figuras de esta sección donde hay imágenes tanto en *DreamShaper* como en *Realistic Vision*. En estas pruebas se intentaron añadir características a los *prompts* para ver cómo respondía el LoRA seleccionado:

**Nacionalidades y edades en las personas** Se quería probar cómo de bueno era a la hora de especificar que una persona fuese asiática o africana, (figura 3.17). A pesar de los pocos detalles que deja en los humanos el estilo de pictogramas, se puede diferenciar que unas personas sean asiáticas o africanas. La que más sorprende es la asiática, pues logra su diferenciación con conceptos más complejos que cambiar el color de piel. Además, se probó a indicar edades o, más bien, si una persona debía



(a) LoRA-1 (b) LoRA-2 (c) LoRA-3

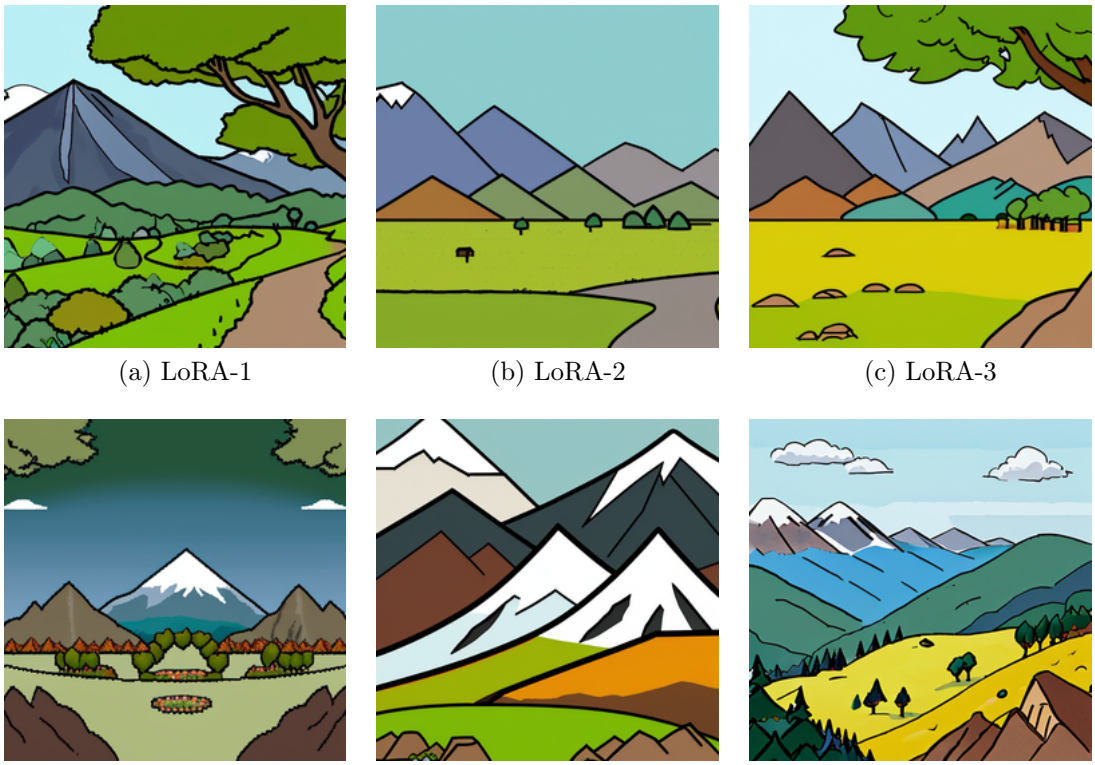
(d) LoRA-4 (e) LoRA-5 (f) LoRA-6

Figura 3.13: Comparación de objetos y lugares.
Prompt: “car race, arasaac” Modelo: DreamShaper

ser un niño, un adulto o un anciano, (figura 3.18). Esto también fue un éxito, y su diferenciación es clara.

**Corrupción del entrenamiento** Un entrenamiento deficiente o mal ejecutado puede generar problemas en el modelo al momento de diferenciar conceptos cercanos, como el ya mencionado caso de “boxer dog” y “boxer girl.” Sin embargo, como se observa en las figuras 3.19a y 3.19b, el LoRA entrenado sobre SD 1.5 demuestra un buen desempeño, sin evidencias de contaminación entre conceptos. Este modelo es capaz de diferenciarlos perfectamente, lo que refleja la solidez alcanzada gracias a un correcto entrenamiento.

**Añadir vestimentas u objetos** Una funcionalidad útil a la hora de comunicarse con imágenes es poder especificar un objeto a una persona o una vestimenta específica. Explicar que para limpiar el suelo hace falta una fregona (figura 3.20), para las hojas de los árboles un rastrillo (figura 3.21), que un obrero utiliza una pala (figura 3.22) o que para poder ir a correr con mucho calor hace falta una gorra (figura 3.23). Son situaciones en las que un educador puede emplear los pictogramas para enseñar a niños o a personas que necesiten de CAA. Todas las pruebas fueron satisfactorias.



(a) LoRA-1 (b) LoRA-2 (c) LoRA-3

(d) LoRA-4 (e) LoRA-5 (f) LoRA-6

Figura 3.14: Comparación de objetos y lugares.
Prompt: "mountains landscape, arasaac" Modelo: DreamShaper

**Condiciones climáticas** Se probó en añadir a pictogramas las condiciones climáticas de que lloviese, nevase o hiciese calor, (figura 3.24). Fue un gran éxito, se pueden diferenciar las características climatológicas claramente con la aparición de elementos como gotas de lluvia, nieve, un sol, ropa de invierno o ropa de verano. Lo mejor es que solo se necesita añadir una o dos palabras al *prompt* original.

**Fondo de la imagen** Especificar un fondo concreto al generar un pictograma puede mejorar significativamente su claridad y utilidad en ciertos contextos. Por ejemplo, un fondo blanco resalta al máximo el objeto o acción representada, facilitando su interpretación. Durante las pruebas, se evaluó la capacidad del LoRA para generar pictogramas con diferentes fondos, como un bosque (figura 3.25), una playa (figura 3.26) o un fondo completamente blanco (figura 3.27). Este tipo de pruebas permite determinar si el modelo mantiene el estilo uniforme del pictograma independientemente del entorno especificado, y si logra resaltar los elementos principales sin perder claridad visual. El aplicar el color al fondo es un éxito, pero cabe destacar que puede llegar a condicionar con qué color genere el objeto, como en el caso de la mochila (figura 3.28), que también adopta el color blanco.

(a) LoRA-1 (b) LoRA-2 (c) LoRA-3

(d) LoRA-4 (e) LoRA-5 (f) LoRA-6

Figura 3.15: Comparación de generaciones complejas.
Prompt: "a boy driving a blue car, arasaac" Modelo: DreamShaper

**Especificar colores** Especificar colores puede ser una funcionalidad muy útil a la hora de generar pictogramas para la comunicación. En una situación en la que a una persona se le quiera especificar que necesita un alimento de un color específico (figura 3.29) o una hoja de árbol morada para su experimento de ciencias (figura 3.31). En situaciones como esta, es realmente útil este trabajo, ya que, al momento, sin necesidad de que esté en una base de datos, el pictograma que necesitas puede ser generado.

Esta funcionalidad no está lograda a la perfección; a veces, le cuesta aplicar el color al elemento o exclusivamente al elemento deseado (figura 3.30). Pero, si se prueban varias semillas o se mejora el *prompt*, se puede lograr el resultado objetivo.

**Paleta de colores** El objetivo aquí es lograr que se genere con una paleta de colores específica, en situaciones de comunicación con personas con dificultades comunicativas, o cuando es fácil que distraigan su atención de la idea principal. La eliminación de colores y la capacidad de generar exclusivamente en blanco y negro (figura 3.32) o con una paleta de grises (figura 3.33) ayudan a reducir el número de distracciones del pictograma final.

(a) LoRA-1 (b) LoRA-2 (c) LoRA-3

(d) LoRA-4 (e) LoRA-5 (f) LoRA-6

Figura 3.16: Comparación de generaciones complejas.
Prompt: "forest fairy playing video games, video game controller, arasaac"
Modelo: DreamShaper

(a) personas sin especificar. Nacionalidad = "african"

(b) personas africanas. Nacionalidad = "african"

(c) personas asiáticas. Nacionalidad = "asian"

Figura 3.17: Generación con personas de distintas etnias/nacionalidades.
Prompt: "[Nacionalidad] people running a marathon, arasaac"
Modelo: Realistic Vision

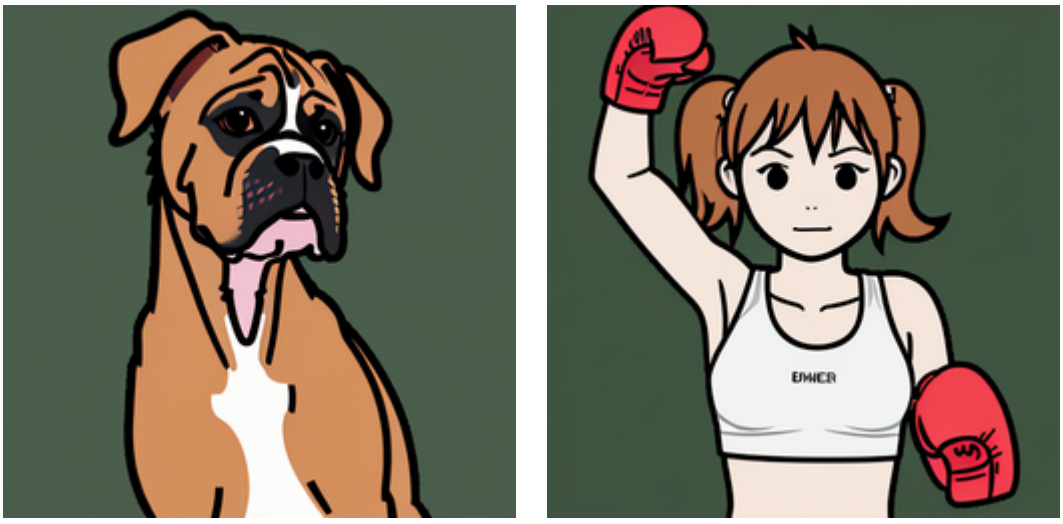
La palabra que más éxito ha tenido a la hora de que el modelo comprenda que se quiere la paleta de esos colores es "scale". Probé con otras como "palette" o "gray colors", pero sus resultados eran más inconsistentes y añadían colores fuera de la paleta especificada.


(a) Persona niño.
Edad = “child”

(b) Persona adulta.
Edad = “adult”

(c) Persona anciana.
Edad = “old man”

Figura 3.18: Generación con personas de distintas edades.
Prompt: “[Edad] using a computer, arasaac”
Modelo: Realistic Vision



(a) Perro de raza boxer.

(b) Chica boxeador.

Figura 3.19: Prueba de corrupción de texto.
Prompt: “boxer dog/girl, arasaac” Modelo: DreamShaper

### Limitaciones Detectadas

Durante las pruebas adicionales se identificaron algunas limitaciones relevantes en el comportamiento del modelo entrenado, especialmente en lo referente a la personalización por color. En particular, al indicar un color para un objeto o lugar específico, el modelo tiende a aplicarlo también a otros elementos no deseados. Esta problemática se observa, por ejemplo, en la figura 3.35, donde el color azul se extiende a los ojos u otros objetos secundarios, a diferencia de la versión base. Asimismo, en la figura 3.34, el modelo tiene dificultades para aplicar correctamente el color morado a la manzana, no lográndolo en la mayoría de los casos o extendiéndolo a otras zonas de la imagen. Es importante señalar que incluso en la versión base

Figura 3.20: Generar especificando un objeto.
Prompt: "kid using a mop, arasaac"
Modelo: DreamShaper

Figura 3.21: Generar especificando un objeto.
Prompt: "kid using a rake, arasaac"
Modelo: DreamShaper

Figura 3.22: Generar especificando un objeto.
Prompt: "worker using a shovel, arasaac"
Modelo: DreamShaper

Figura 3.23: Generar especificando una vestimenta.
Prompt: "child wearing a hat running a marathon, arasaac"
Modelo: DreamShaper

(figura 3.34a), el modelo ya mostraba limitaciones similares, lo que sugiere que esta dificultad no es exclusiva del LoRA entrenado.

Otra de las limitaciones es la generación de caras y de conceptos complejos, como que alguien está cayendo de un árbol. Donde la imagen base (figuras 3.36a y 3.36b), ya no es capaz de generar con éxito algo con sentido, y esto se transfiere a

(a) Lloviendo.
Clima = "raining"

(b) Nevando.
Clima = "snowing"

(c) Clima caluroso.
Clima = "hot weather"

Figura 3.24: Generación con diferentes climatologías.
Prompt: "people wearing hats running a marathon,[Clima],arasaac"
Modelo: Realistic Vision

Figura 3.25: Generar especificando un fondo de un bosque.
Prompt: "astronaut, arasaac, forest background"
Modelo: DreamShaper

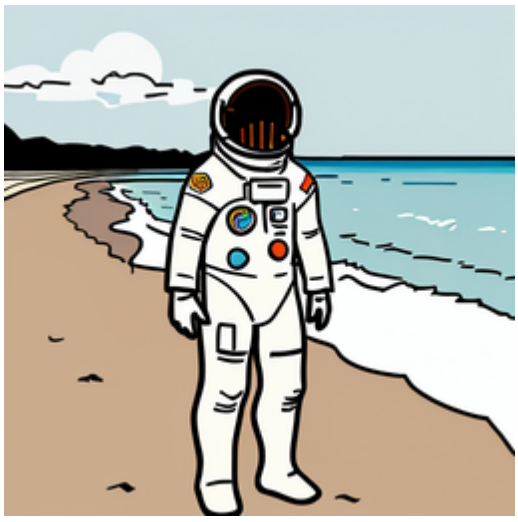


Figura 3.26: Generar especificando un fondo de una playa.
Prompt: "astronaut, arasaac, beach background"
Modelo: DreamShaper

los pictogramas generados (figura 3.36c).

Estas limitaciones se entienden como propias del modelo base utilizado, que ya presenta dificultades para aplicar correctamente colores, generar posturas humanas creíbles o representar caras, especialmente en escenas con múltiples personas. No obstante, con un mayor conocimiento sobre generación de imágenes, estas carencias pueden mitigarse. Algunas de las estrategias más comunes incluyen el uso de *prompts* más complejos y específicos (por ejemplo, añadiendo detalles como "defined faces"), la utilización de *negative prompts* ("bad anatomy", "poorly drawn"), así como el em-

Figura 3.27: Generar especificando un fondo blanco.
Prompt: "people playing volleyball, arasaac, white background"
Modelo: DreamShaper

Figura 3.28: Generar especificando un fondo blanco.
Prompt: "backpack, arasaac, white background"
Modelo: DreamShaper

(a) Pimiento azul.
Color = "blue"

(b) Pimiento rojo.
Color = "red"

(c) Pimiento amarillo.
Color = "yellow"

Figura 3.29: Generación de un pimiento con distintos colores.
Prompt: "[Color] pepper,arasaac,white background"
Modelo: DreamShaper

pleo de técnicas adicionales como *word embeddings* o *refiners*.[13] Estas herramientas, combinadas con este LoRA entrenado específicamente para el estilo ARASAAC, obtendrían resultados de mayor calidad, mejorando la anatomía, la fidelidad cromática o la coherencia de los pictogramas generados.

[13]Se detallan estas técnicas en la sección 2.3.4.

(a) Pimiento azul.
Color = “blue”

(b) Pimiento rosa.
Color = “pink”

(c) Pimiento amarillo.
Color = “yellow”

Figura 3.30: Fallos al generar pimientos de colores.
Prompt: “[Color] pepper,arasaac,white background”
Modelo: DreamShaper

Figura 3.31: Generación especificando el color de un objeto.
Prompt: “purple leaf,arasaac,white background”
Modelo: DreamShaper

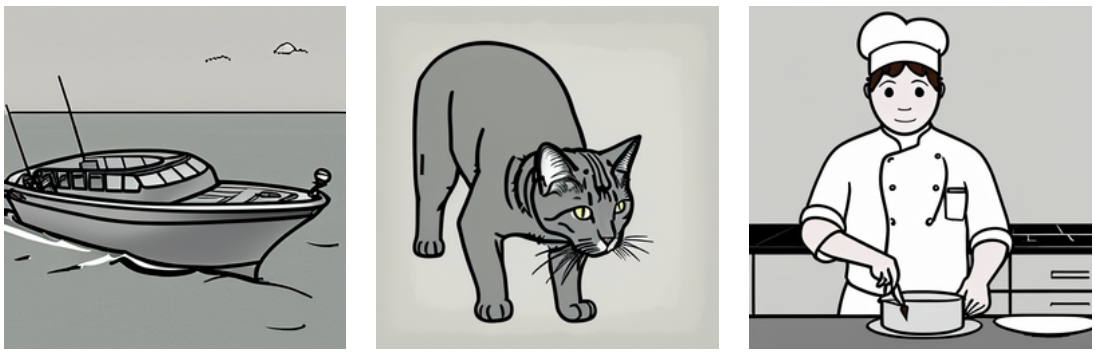
## 3.5. Entrenamientos en Modelos XL

El proyecto se desarrolló principalmente sobre Stable Diffusion 1.5 debido a su popularidad en la comunidad y a sus requisitos de hardware más accesibles, mientras que Stable Diffusion XL fue descartado inicialmente por sus elevados requerimientos. Sin embargo, una vez finalizado el proyecto principal y con el conocimiento adquirido durante los entrenamientos en Stable Diffusion 1.5, decidí explorar la posibilidad de realizar entrenamientos en modelos XL.

El principal desafío al trabajar con modelos XL radica en sus elevados requisitos de hardware, tanto para la generación como para el entrenamiento. En mi caso,

(a) Reunión de doctores en blanco y negro. (b) Anciano usando un PC en blanco y negro. (c) Niña corriendo con un gorro en blanco y negro.

Figura 3.32: Generación de imágenes en blanco y negro.
Prompt: "[prompt específico],arasaac,black and white scale"
Modelo: Realistic Vision



(a) Barco en escala de grises. (b) Gato en escala de grises. (c) Chef en escala de grises.

Figura 3.33: Generación de imágenes en escala de grises.
Prompt: "[prompt específico],arasaac,gray scale"
Modelo: Realistic Vision

(a) Imagen normal. (b) Pictograma exitoso. (c) Pictograma fallido.

Figura 3.34: Fallos aplicando colores 1. Prompt: "grandmother eating an apple"
Modelo: Realistic Vision

(a) Imagen normal. (b) Pictograma exitoso. (c) Pictograma fallido.

Figura 3.35: Fallos aplicando colores 2. Prompt: "a boy driving a blue car" Modelo: Realistic Vision

(a) Imagen normal. (b) Pictograma exitoso. (c) Pictograma fallido.

Figura 3.36: Fallos en anatomía y posturas. Prompt: "boy falling from a tree" Modelo: Realistic Vision

contaba con una NVIDIA RTX 3070 con 8 GB de VRAM en mi PC personal. Tras investigar, descubrí que, limitando ciertas características del entrenamiento, sería posible llevarlo a cabo, aunque con resultados inciertos debido a estas restricciones.

Después de experimentar con distintos valores de *Network Rank*, número total de pasos y otras configuraciones, me encontré con tiempos de entrenamiento excesivamente largos (más de 24 horas). Esto no solo representaba una carga considerable para mi equipo, sino que también planteaba dudas sobre posibles daños a la GPU por uso prolongado al 100 % de capacidad. Finalmente, logré un entrenamiento en aproximadamente 4 horas, obteniendo un LoRA de calidad aceptable pese a las limitaciones. Las restricciones aplicadas fueron las siguientes:

1. **Número total de pasos**
   En modelos XL, el tiempo de entrenamiento aumenta considerablemente con el número de pasos, lo que hizo necesario limitar esta configuración.

   - En el primer intento, utilicé un *dataset* de 100 imágenes, con 4 repeticiones por imagen, 30 épocas y un tamaño de *batch size* de 4. Esto resultaba

en un total de 3000 pasos [(100 IMG × 4 Rep × 30 Ep)/4 BS] y un tiempo estimado de entrenamiento de aproximadamente 24 horas. Este intento fue cancelado al inicio, ya que no era seguro mantener mi GPU trabajando al 100 % durante tanto tiempo.

- Opté por reducir el tamaño del *dataset* a 20 imágenes, una configuración que había demostrado ser eficaz en los entrenamientos con SD 1.5. Manteniendo los mismos valores para repeticiones, épocas y tamaño de *batch*, el número total de pasos se redujo a 600 [(20 IMG × 4 Rep × 30 Ep)/4 BS], con un tiempo de entrenamiento de 4 horas. Aunque este número de pasos es inferior al recomendado (2000-5000), fue suficiente para obtener resultados aceptables dadas las restricciones de hardware.

2. **Tamaño de *Network Rank* (Dimensión) y *Network Alpha***
   La dimensión de estas redes tiene un impacto directo en el uso de VRAM y el tiempo de entrenamiento. Reducir estos valores no solo disminuye los requisitos de hardware, sino que también acelera el proceso. Según la guía del usuario *Valstrix* (Training-Guide-Valstrix (2024)), las configuraciones óptimas para entrenamientos XL incluyen:

   - Mantener *Network Rank* en valores de hasta 64, ya que valores mayores pueden afectar negativamente los resultados.
   - Usar valores cercanos a 16 para *Network Rank* y *Network Alpha*. En mi caso, empleé un valor de 16 para *Network Rank* y reduje *Network Alpha* a 1, siguiendo la recomendación para optimizadores como Adafactor.

Además, se aplicaron otros ajustes menores:

- Se configuró el *LR warmup* ( % de los pasos totales) al 20 %.
- La resolución máxima se fijó en 1024 × 1024 px.
- Se ajustaron parámetros relacionados con el ruido del entrenamiento.

Todos los parámetros se encuentran documentados en el archivo de configuración *configuration.json*, disponible junto al modelo en el repositorio del proyecto[14].

El modelo empleado para el entrenamiento fue *Stable Diffusion XL base 1.0*[15], que ofrece versatilidad para ser utilizado con cualquier modelo basado en SDXL. Además, al tratarse de un modelo base, tiene menor probabilidad de presentar *text encoder corruption*.

[14]Repositorio Hungging Face: `https://huggingface.co/antuna01/Pictogram-LoRAs/tree/main`.

[15]Referencia del modelo, Stability AI, Stable Diffusion XL base 1.0: `https://huggingface.co/stabilityai/stable-diffusion-xl-base-1.0`.

### 3.5.1. Resultados del LoRA XL

**Configuración de generación** Para obtener los mejores resultados al usar este LoRA, se recomienda la siguiente configuración en la interfaz de *WebUI Forge*:

- *Sampling steps*: ±20
- Método de muestreo: *DPM++ 2M*
- Tipo de planificador (*Schedule type*): *Karras* o *Automatic* (aunque casi cualquier opción ofrece buenos resultados).
- Size: 1024x1024
- *Distilled CFG Scale*: 3.5
- *CFG Scale*: 7
- *Clip skip*: 2
- Peso del LoRA: 1 (dependiendo del modelo de generación hace falta subir o bajar el peso, rango [0.9,1.3]).
- Modelos base compatibles: Los LoRAs desarrollados son compatibles con modelos basados en *Stable Diffusion XL*, como *Juggernaut XL*[16] y *DreamShaper XL*[17]. Entre ellos, *Juggernaut XL* destaca por ofrecer resultados superiores en términos generales, aunque requiere más recursos computacionales en comparación con *DreamShaper XL*, que es más ligero y accesible.


(a) Chico conduciendo un coche azul. (b) Niño usando una fregona. (c) Clase de colegio.

Figura 3.37: Pruebas variadas del LoRA XL. Modelo: Juggernaut XL

Como se observa en las figuras presentadas en este apartado, los resultados obtenidos son notables, destacando un estilo consistente y evidenciando una mejora significativa en la generación de rostros al utilizar un modelo mas avanzado, como la

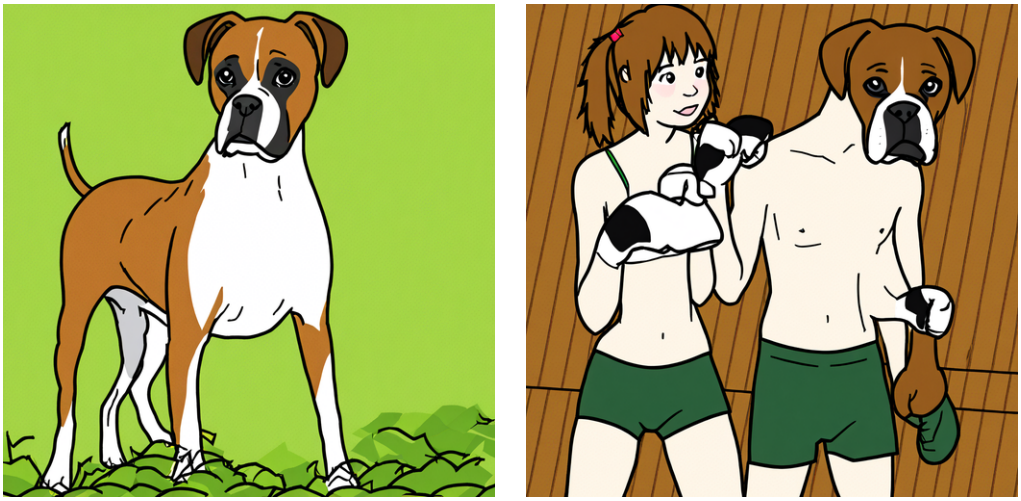
[16]Juggernaut XL: `https://civitai.com/models/133005/juggernaut-xl`
[17]DreamShaper XL: `https://civitai.com/models/112902?modelVersionId=351306`

(a) Hada del bosque jugando videojuegos. (b) Niño cayendo de un árbol. (c) Chef.

Figura 3.38: Pruebas variadas del LoRA XL. Modelo: Juggernaut XL

(a) Montañas. (b) Calcetines. (c) Carrera de coches.

Figura 3.39: Pruebas variadas del LoRA XL. Modelo: Juggernaut XL



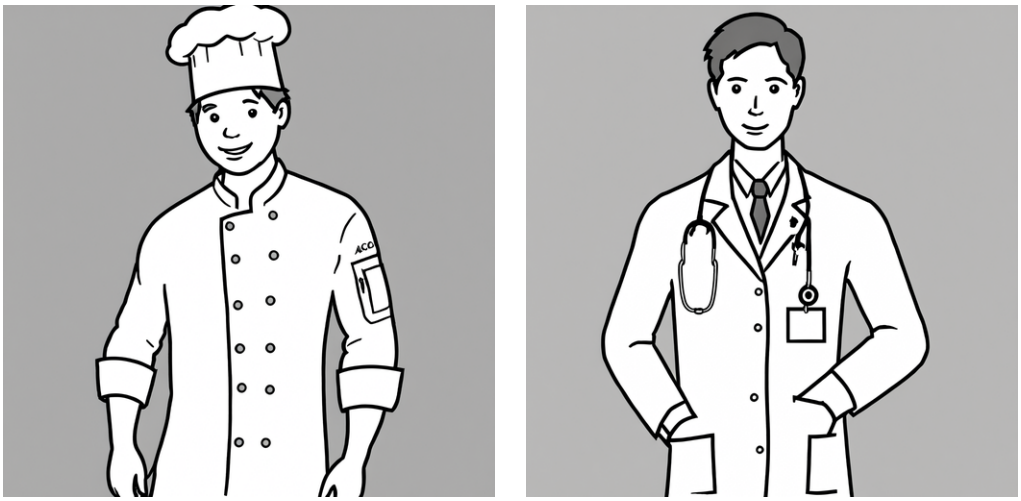
(a) Acierto con perro de raza boxer. (b) Fallos con chica boxeador.

Figura 3.40: Prueba de corrupción de texto. Modelo: Juggernaut XL

de SDXL. Además, la capacidad de trabajar con *prompts* complejos es evidente, permitiendo generar escenas con múltiples personas, como la maratón (3.41), o aplicar

(a) Generación base de una maratón. (b) Generación con LoRA de una maratón.

Figura 3.41: Prueba de generación de muchas personas en un modelo distinto. Modelo: DreamShaper XL



(a) Chef paleta de grises. (b) Doctor paleta de grises.

Figura 3.42: Prueba de generación con paleta de colores grises. Modelo: Juggernaut XL

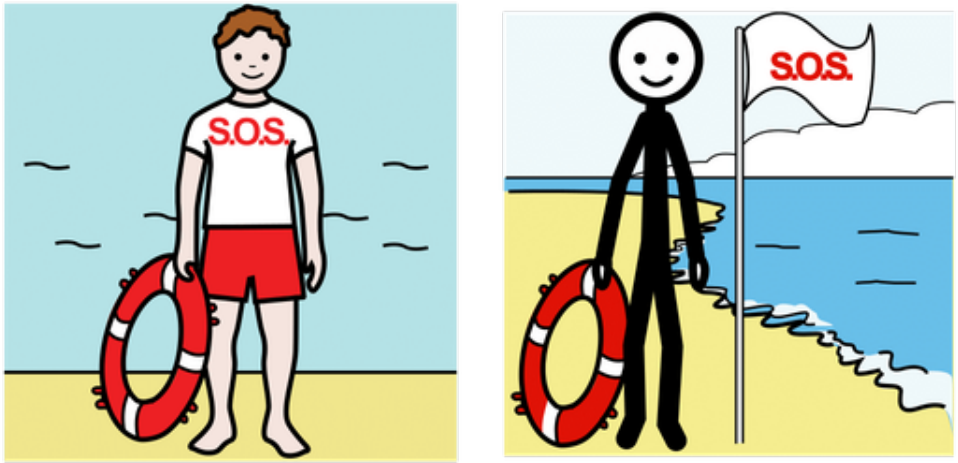
colores específicos tanto a objetos como a la paleta general de la imagen (3.42).

En comparación con la versión basada en SD 1.5, los resultados son similares a pesar de las limitaciones impuestas durante el entrenamiento. Sin embargo, estas restricciones son visibles en ejemplos como la figura 3.40, donde el modelo mezcla conceptos debido a un entrenamiento más acotado, o en la figura 3.37c, en la que el modelo intenta escribir la palabra de activación “arasaac”, un problema recurrente en algunas generaciones.

El entrenamiento en SDXL puede considerarse un éxito rotundo incluso bajo condiciones restringidas, lo que sugiere que trabajar con modelos más potentes no solo incrementa los costos de generación y entrenamiento, sino que también permite alcanzar resultados sobresalientes. Por ello, explorar esta posibilidad representa una dirección prometedora que merece atención en futuros desarrollos.

## 3.6. Otro estilo de pictogramas: Estilo Stickman

La base de pictogramas de ARASAAC cuenta con diferentes estilos diseñados para adaptarse mejor a las necesidades de comunicación de cada persona. Su base de datos ofrece dos estilos claramente diferenciados, tal como se muestra en la figura 3.43. El primer estilo representa a las personas con detalles como color de piel, ojos y cabello, mientras que el segundo simplifica las figuras humanas a dibujos de palos con una cabeza, conocidos como *stickman* u "hombre palo".




(a) Estilo con detalles humanos

(b) Estilo stickman, objetivo del nuevo LoRA

Figura 3.43: Estilos principales en el *dataset* de la web ARASAAC en los que se centran los LoRAs desarrollados.

Tal y como se analiza en la sección 2.1, la elección del estilo visual en los pictogramas puede influir en su eficacia comunicativa. En algunos contextos, especialmente con personas con dificultades cognitivas, el estilo *stickman* puede resultar más adecuado al eliminar detalles que pueden distraer del mensaje principal.

Con este propósito, desarrollé un nuevo *dataset* siguiendo el mismo procedimiento descrito en secciones anteriores, adaptado para entrenar un LoRA orientado a generar pictogramas en estilo *stickman*. Este LoRA permitiría, mediante los mismos *prompts* empleados en el estilo detallado, obtener resultados en este estilo minimalista simplemente activando el LoRA correspondiente.

### 3.6.1. Resultados del LoRA estilo Stickman en SD 1.5

El entrenamiento del LoRA estilo *stickman* se llevó a cabo en Stable Diffusion 1.5, utilizando los parámetros y aprendizajes previos obtenidos en el desarrollo del LoRA principal (sección 3.3.3). El objetivo principal de este LoRA era generar representaciones humanas simplificadas en el estilo característico de los pictogramas *stickman*, reduciendo los detalles al mínimo esencial.

Para evaluar la eficacia del entrenamiento, se centró la generación en *prompts* relacionados con figuras humanas, ya que este es el elemento distintivo del estilo *stickman*.

**Configuración de generación** Para obtener los mejores resultados al usar este LoRA, se recomienda la siguiente configuración en la interfaz de *WebUI Forge*:

- *Sampling steps*: ±20
- Método de muestreo: *DPM++ 2M*
- Tipo de planificador (*Schedule type*): *Karras* o *Automatic* (aunque casi cualquier opción ofrece buenos resultados).
- Size: 512x512
- *Distilled CFG Scale*: 3.5
- *CFG Scale*: 7
- *Clip skip*: 2
- Peso del LoRA: ±1,1
- Modelos base compatibles: Basados en *Stable Diffusion 1.5*, como *Realistic Vision* y *DreamShaper*.

A continuación, se presentan algunas de las generaciones obtenidas:

En muchos casos, las generaciones lograron plasmar el estilo *stickman* en los humanos (3.45), lo que representa un avance prometedor. Sin embargo, la capacidad del modelo para mantener este estilo disminuyó al intentar incorporar otros elementos o detalles en las imágenes. Esto se refleja en generaciones donde los coches aparecen con formas inusuales (3.47), el fondo de una clase resulta confuso (3.46), o los elementos distintivos que deberían diferenciar a un humano como doctor no siempre son representados con claridad (3.44).

Estos resultados apuntan a una dificultad del modelo para equilibrar la simplicidad del estilo *stickman* con la inclusión de detalles adicionales, algo que podría estar influido tanto por las limitaciones del *dataset* como por las características del modelo base, Stable Diffusion 1.5. Además, las numerosas capas transparentes presentes en

(a) (b)

Figura 3.44: Resultados modelo LoRA Stickman en Stable Diffusion 1.5.

(a) (b)

Figura 3.45: Resultados modelo LoRA Stickman en Stable Diffusion 1.5.

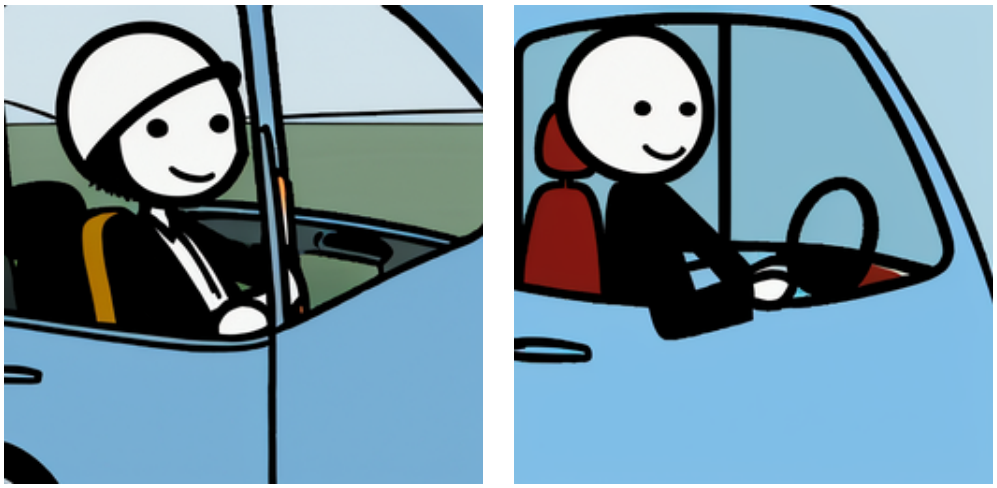
el *dataset* parecen haber dificultado el aprendizaje y contribuido a la inconsistencia en algunas generaciones con los fondos de las imágenes.

En conclusión, aunque el LoRA en su estado actual no alcanza la consistencia necesaria para un uso fiable, los resultados obtenidos indican que el estilo *stickman* es alcanzable con optimizaciones adicionales. Mejoras en la calidad del *dataset*, particularmente eliminando las capas transparentes, junto con una mayor cantidad de datos y un cambio en los pasos de entrenamiento, podrían conducir a un LoRA más robusto y efectivo.

(a) Estilo con detalles humanos

(b) Estilo stickman, objetivo del nuevo LoRA

Figura 3.46: Resultados modelo LoRA Stickman en Stable Diffusion 1.5.



(a)

(b)

Figura 3.47: Resultados modelo LoRA Stickman en Stable Diffusion 1.5.

### 3.6.2. Resultados del LoRA estilo Stickman en SDXL

Tras decidir ampliar los entrenamientos a modelos XL, también se consideró oportuno desarrollar un LoRA de estilo *stickman* en modelos XL. Este experimento tenía como objetivo evaluar la capacidad de SDXL para generar imágenes en un estilo simplificado y minimalista, utilizando los mismos recursos limitados empleados en versiones anteriores.

Para este entrenamiento, se adoptaron los mismos parámetros que en el modelo de estilo detallado en SDXL (sección 3.5.1) y se utilizó el mismo *dataset* reducido de 20 imágenes (sección 3.6.1). Estas decisiones respondieron a las restricciones de hardware descritas en la sección 3.5, donde se destacaron los elevados requisitos de

VRAM y el tiempo de entrenamiento asociados con modelos XL.

En esta sección, las generaciones se centran en humanos, ya que la principal característica distintiva de este LoRA frente al estilo detallado es su capacidad para representar figuras humanas con el diseño simplificado propio del estilo *stickman*.

**Configuración de generación** Para obtener los mejores resultados al usar este LoRA, se recomienda la siguiente configuración en la interfaz de *WebUI Forge*:

- *Sampling steps*: ±20
- Método de muestreo: *DPM++ 2M*
- Tipo de planificador (*Schedule type*): *Karras* o *Automatic* (aunque casi cualquier opción ofrece buenos resultados).
- Size: 1024x1024
- *Distilled CFG Scale*: 3.5
- *CFG Scale*: 7
- *Clip skip*: 2
- Peso del LoRA: ±1,2
- Modelos base compatibles: Los LoRAs desarrollados son compatibles con modelos basados en *Stable Diffusion XL*, como *Juggernaut XL* y *DreamShaper XL*.

(a) Chico conduciendo coche azul Stickman. (b) Chef Stickman. (c) Chef Stickman.

Figura 3.48: Pruebas variadas del LoRA de estilo Stickman en modelos XL. Modelo: Juggernaut XL

El entrenamiento produjo resultados visibles y prometedores, aunque en ocasiones no logra mantener completamente la simplicidad característica del estilo *stickman*. Esto se observa en imágenes como el coche azul (3.48a) o el hada (3.50c),

(a) Niños jugando al futbol Stickman.

(b) Niño usando una fregona Stickman.

(c) Boxer girl contaminacion de text encoder Stickman.

Figura 3.49: Pruebas variadas del LoRA de estilo Stickman en modelos XL. Modelo: Juggernaut XL

(a) Doctor Stickman.

(b) Niño cayendo Stickman.

(c) Hada del bosque jugando videojuegos Stickman.

Figura 3.50: Pruebas variadas del LoRA de estilo Stickman en modelos XL. Modelo: DreamShaper XL

donde se añaden colores o detalles adicionales como el pelo, alejándose del diseño minimalista. Además, el modelo muestra cierta confusión en conceptos, como en la figura 3.49c, donde mezcla la idea de un perro bóxer con una chica boxeadora, lo que evidencia cierta limitación en la precisión semántica.

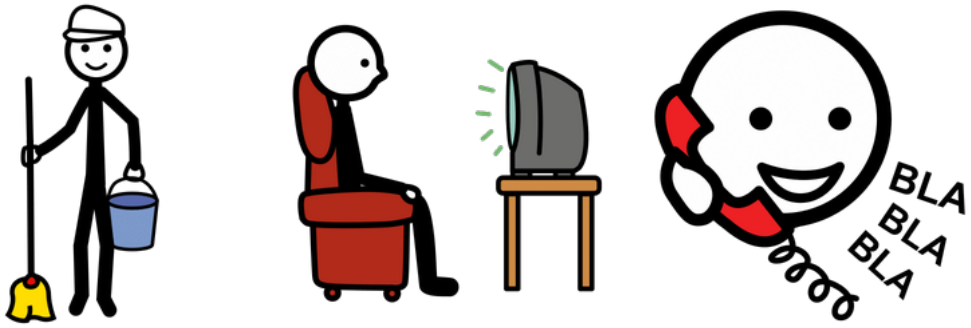
A pesar de estas inconsistencias, es sorprendente que los resultados obtenidos con SDXL sean más consistentes que los logrados en la versión de SD 1.5, incluso cuando ambos compartieron el mismo *dataset* reducido y un número de pasos total más bajo en SDXL. Esto puede atribuirse a la mayor capacidad de los modelos XL, que parece ser más eficiente en la comprensión y representación del concepto *stickman*, compensando las limitaciones del *dataset* y del entrenamiento.

Con un *dataset* mejorado, que elimine los problemas relacionados con las capas transparentes, y mayores recursos para realizar entrenamientos más prolongados, es probable que se pueda alcanzar una calidad y consistencia comparables a las obtenidas con el LoRA de estilo detallado. Los resultados actuales refuerzan la idea de que SDXL ofrece un potencial significativo incluso bajo condiciones limitadas, consolidando su relevancia para proyectos futuros.

### 3.6.3. Conclusiones del estilo Stickman

A partir de los entrenamientos realizados en ambos modelos base, Stable Diffusion 1.5 y Stable Diffusion XL, se pueden extraer dos conclusiones principales sobre el estilo *stickman*:

- **Problemas con el *dataset*:** La calidad del *dataset* utilizado fue un factor determinante en la inconsistencia observada en los resultados. En particular, muchas imágenes del estilo *stickman* contenían capas transparentes, lo que dificultó el proceso de entrenamiento. Como se puede observar en la figura 3.51, este tipo de capas interfiere con la capacidad del modelo para generar figuras consistentes. Mejorar la calidad del *dataset*, eliminando estas capas transparentes y asegurando una representación más clara y precisa del estilo, podría generar resultados más consistentes y de mayor calidad.
- **Eficacia de la parametrización:** A pesar de los desafíos del *dataset*, los parámetros utilizados en el LoRA principal demostraron ser efectivos para replicar entrenamientos en otros estilos de pictogramas. Esto facilitó la creación de modelos adicionales basados en el LoRA de estilo detallado, como el estilo *stickman*. La capacidad de aplicar estos parámetros a diferentes estilos sugiere que con ajustes adecuados, es posible crear modelos de otros estilos visuales sin necesidad de reconfigurar completamente los parámetros de entrenamiento.




(a) (b) (c)

Figura 3.51: Problemas con capas transparentes en el *dataset* de estilo *stickman*.

## 3.7. Conclusión

Este capítulo ha detallado el proceso completo seguido para desarrollar modelos LoRA de generación de pictogramas, desde la selección inicial de modelos (SD 1.5 y SDXL) hasta la exploración de estilos alternativos (*stickman*). Se han tomado decisiones fundamentadas para elegir modelos base, de generación y de entrenamiento, teniendo en cuenta tanto criterios técnicos como de compatibilidad con los objetivos del proyecto.

La construcción del dataset ha supuesto una etapa fundamental, especialmente por las particularidades estilísticas de los pictogramas de referencia (como los de ARASAAC) y por la necesidad de un etiquetado coherente y representativo. Además, se identificaron y resolvieron problemas técnicos importantes durante esta fase, como la representación de datos en la imagen (formato RGB) o la diversidad estilística no deseada, que rompía la homogeneidad del *dataset* y contaminaba el aprendizaje del LoRA.

En cuanto al entrenamiento de LoRAs, se abordó tanto en entornos gratuitos como Google Colab, como en local, con las limitaciones del hardware disponible. Optimizando recursos y parámetros tras un análisis exhaustivo de resultados, se lograron unos entrenamientos eficientes y de calidad. Esta fase permitió validar el funcionamiento de los modelos entrenados y seleccionar las mejores versiones según sus capacidades de generación vistas en el análisis de resultados.

También se exploraron ampliaciones al trabajo inicial, como el entrenamiento sobre modelos XL y la prueba de un estilo visual alternativo (*stickman*), lo que permitió evaluar la versatilidad del sistema propuesto y sus posibilidades de adaptación a distintas necesidades comunicativas.

En conjunto, este capítulo ha mostrado el proceso completo necesario para crear modelos capaces de capturar estilos visuales concretos mediante el uso de tecnologías libres como Stable Diffusion, *BLIP* y *Kohya_ss*. Este recorrido práctico no solo sienta las bases experimentales del proyecto, sino que también pone de manifiesto el potencial de estas herramientas accesibles para desarrollar soluciones personalizadas en el ámbito de la comunicación aumentativa.

Capítulo 4

# Conclusiones y Trabajo Futuro

## 4.1. Introducción

En este capítulo se presentan las conclusiones principales del proyecto y se proponen posibles líneas de trabajo futuro para mejorar los resultados obtenidos. Se exponen los resultados obtenidos de los entrenamientos en distintos modelos base. Además, se analizan las limitaciones encontradas, el impacto de las decisiones y estrategias adoptadas a lo largo del proyecto, así como las oportunidades de mejora a través del uso de nuevos recursos y técnicas.

## 4.2. Conclusiones Principales

En esta sección se sintetizan los aprendizajes clave extraídos del desarrollo del proyecto, así como las decisiones y resultados más relevantes. Para ello, se analiza el grado de cumplimiento de los objetivos planteados, la calidad alcanzada en la generación de pictogramas, las limitaciones derivadas del entorno técnico y las elecciones estratégicas adoptadas durante el proceso. Esta reflexión final permite evaluar el impacto real del trabajo realizado.

### 4.2.1. Cumplimiento de los objetivos y contribuciones esperadas

El desarrollo del sistema ha permitido cumplir de forma satisfactoria tanto el objetivo principal como los objetivos secundarios planteados al inicio del proyecto.

En cuanto al **objetivo principal**, se ha conseguido implementar un sistema basado en inteligencia artificial capaz de generar pictogramas de manera rápida y consistente, siguiendo el estilo visual de los pictogramas de ARASAAC. El sistema permite generar pictogramas a partir de descripciones simples sin depender de bases de datos preexistentes, cumpliendo también con los principios de accesibilidad y

flexibilidad definidos. Al utilizar una tecnología libre y escalable como Stable Diffusion, se ha garantizado la gratuidad y posibilidad de adaptación del sistema de generación, tal y como se planteó.

Respecto a los **objetivos secundarios**:

- Se ha demostrado que el sistema puede ejecutarse en entornos con recursos computacionales limitados. Al no requerir recursos adicionales durante la inferencia, los modelos LoRA permiten generar pictogramas en ordenadores personales con GPUs modestas, siempre que cumplan con los requisitos mínimos para ejecutar SD 1.5 o SDXL.

- El sistema ha demostrado adaptarse a distintas necesidades de la Comunicación Aumentativa y Alternativa (CAA). Ha sido capaz de generar pictogramas con paletas de colores simples, asignar colores específicos a objetos, e incluir elementos adicionales como herramientas o prendas en figuras humanas (3.4.3.1), lo que valida su utilidad en contextos donde se requieren pictogramas personalizados.

- Se creo una guía[1] en GitHub clara y accesible que permite utilizar el sistema sin conocimientos técnicos avanzados. No se requiere manipulación directa de herramientas complejas como *embeddings* personalizados, *refiners* o configuraciones internas del modelo, lo que facilita su uso por parte de usuarios no expertos.

- Se ha mantenido una coherencia y consistencia visual notables con el estilo de ARASAAC, lo cual se evidenció en los resultados obtenidos.

Estas evidencias permiten afirmar que el sistema no solo cumple con sus objetivos funcionales, sino que también logra materializar las **contribuciones esperadas**:

- **Generación eficiente de pictogramas:** El sistema permite la creación de pictogramas a partir de texto en tiempo real, sin depender de bases de datos y utilizando modelos optimizados que equilibran calidad y consumo de recursos. Esta contribución se considera alcanzada, ya que se han obtenido pictogramas de calidad utilizando modelos LoRA, los cuales no requieren recursos adicionales y pueden ejecutarse con los requisitos mínimos de Stable Diffusion 1.5.

- **Adaptabilidad a necesidades específicas:** El sistema ha demostrado flexibilidad en la generación de pictogramas personalizados, siendo capaz de representar conceptos poco comunes mediante *prompts* específicos. Además, la ampliación del estilo *stickman* representa una contribución adicional frente al reto de cubrir necesidades particulares. Esto valida su utilidad en contextos reales de CAA, donde las bases de datos cerradas no ofrecen suficiente cobertura.

[1] Guía de uso en el README:`https://github.com/NILGroup/TFG-2324-Pictogramas`

- **Fomento de la investigación en IA y CAA:** Los resultados obtenidos, así como la documentación generada, evidencian el potencial de la IA generativa en el ámbito de la CAA. Esta contribución se considera cumplida al proporcionar una base sólida, con resultados de calidad, que puede ser reutilizada y ampliada por futuros proyectos o investigaciones.

### 4.2.2. Calidad de los resultados obtenidos

A continuación, se analiza en mayor profundidad la calidad de los resultados obtenidos, tal como se mencionó previamente. Los LoRAs desarrollados, especialmente el principal basado en SD 1.5, destacaron por su calidad y adaptabilidad, incluso en dispositivos con recursos computacionales limitados. Esto refleja la efectividad de las decisiones de entrenamiento adoptadas para optimizar la parametrización y aprovechar al máximo los recursos disponibles.

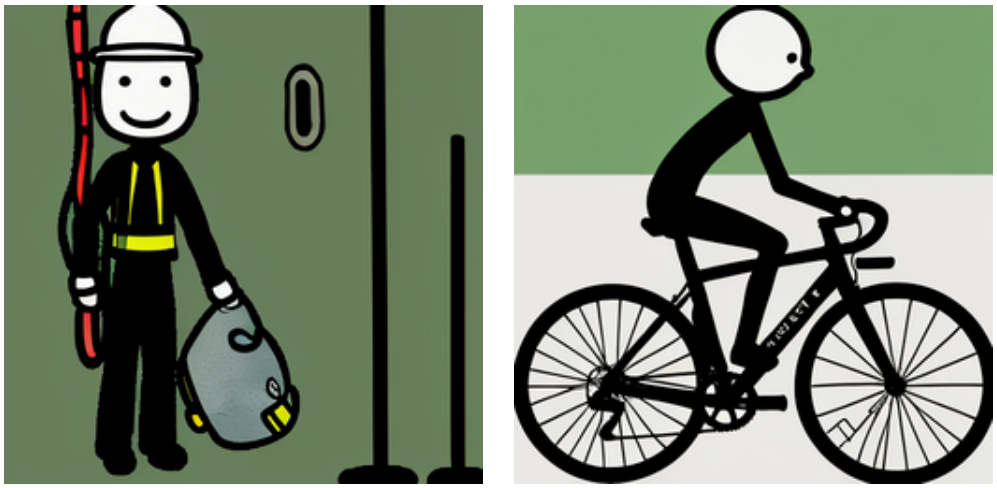
En la figura 4.1, se presentan ejemplos generados en el estilo principal de ARASAAC utilizando el LoRA basado en SD 1.5, mostrando su capacidad para replicar fielmente las características distintivas del estilo.

(a) Bombero (b) Ciclistas

Figura 4.1: Pictogramas generados en el estilo de principal de ARASAAC (3.2a) utilizando el LoRA basado en SD 1.5.

En el caso del estilo *stickman*, aunque se observaron limitaciones en entornos complejos o características específicas, las figuras humanas mostraron un arte característico de hombre palo, validando el potencial del sistema para adaptarse a distintos estilos de pictogramas simples y funcionales para comunicación aumentativa. La figura 4.2 ilustra algunos ejemplos generados en este estilo, destacando su capacidad para capturar los elementos fundamentales de los pictogramas de hombre palo.

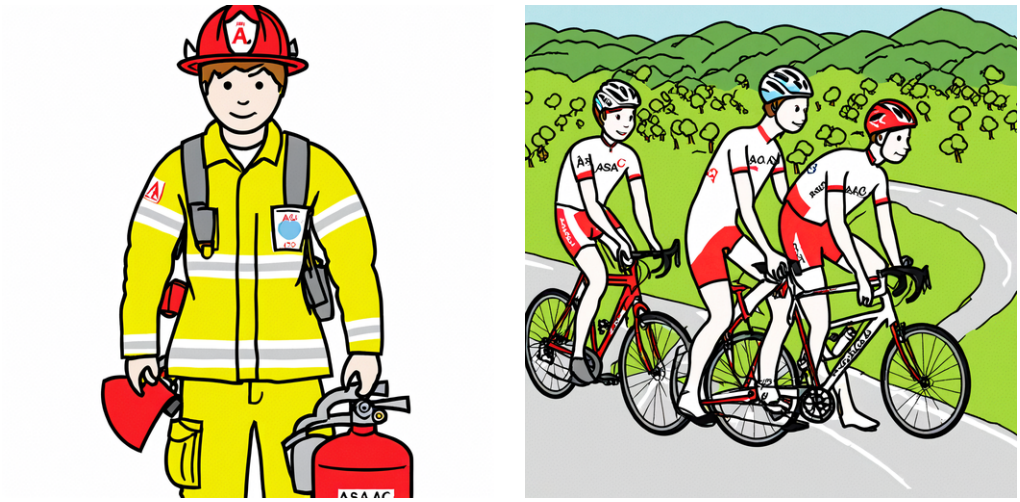
Por otro lado, los experimentos con los modelos XL revelaron su capacidad superior para generar imágenes más complejas y detalladas. La figura 4.3 muestra



(a) Bombero (b) Ciclistas

Figura 4.2: Pictogramas generados en el estilo *stickman* (3.2b) utilizando el LoRA basado en SD 1.5.

ejemplos generados en el estilo principal de ARASAAC con SDXL, destacando la riqueza de detalles y la consistencia obtenida incluso bajo restricciones de hardware.



(a) Bombero (b) Ciclistas

Figura 4.3: Pictogramas generados en el estilo principal de ARASAAC (3.2a) utilizando el LoRA basado en SDXL.

Basándome en el análisis de los resultados obtenidos y en el conocimiento adquirido durante el desarrollo del proyecto, se identificaron áreas clave de mejora que podrían contribuir a un rendimiento más consistente y optimizado:

- Incrementar la homogeneidad y la calidad del dataset, eliminando capas transparentes y optimizando el formato de las imágenes desde el inicio, mejoraría el aprendizaje del modelo.
- La exploración de modelos mas avanzados, como SDXL, mostró gran potencial para manejar estilos visuales complejos, abriendo posibilidades para lograr resultados aún más consistentes.
- Diseñar datasets más precisos, incluso con volúmenes de datos reducidos, puede maximizar el potencial de los modelos y facilitar entrenamientos más eficientes.

En conclusión, el sistema desarrollado cumple con los objetivos planteados al demostrar versatilidad, eficiencia y capacidad de adaptación a estilos visuales específicos. Con un refinamiento en los recursos y técnicas utilizadas, es posible alcanzar niveles aún más altos de calidad en la generación de pictogramas.

### 4.2.3. Impacto de las limitaciones de hardware

El hardware disponible para este proyecto (una RTX 3070 con 8 GB de VRAM) presentó restricciones que afectaron especialmente a los modelos más avanzados, como SDXL. Mientras que los modelos basados en SD 1.5 pudieron aprovechar los recursos limitados para ofrecer resultados consistentes y de alta calidad, los LoRAs en modelos XL se vieron afectados por la imposibilidad de realizar entrenamientos prolongados y por la necesidad de reducir significativamente el tamaño del dataset.

Estas limitaciones se hicieron evidentes en la diferenciación de conceptos complejos. Por ejemplo, en los experimentos para distinguir entre “boxer girl” y “boxer dog,” el LoRA entrenado sobre SDXL mostró dificultades para capturar con precisión los conceptos, como se observa en las figuras 3.40a y 3.40b. Este comportamiento puede atribuirse al reducido número de pasos de entrenamiento y al tamaño limitado del dataset, lo que afectó negativamente la capacidad del modelo para generalizar.

En contraste, el LoRA entrenado sobre SD 1.5 logró resultados significativamente mejores en estos escenarios, como se muestra en las figuras 3.19a y 3.19b. Gracias a la capacidad de realizar entrenamientos más extensos y con datasets más amplios, este modelo pudo diferenciar de manera más consistente entre los conceptos mencionados.

A pesar de estas diferencias, los resultados obtenidos con SDXL en condiciones subóptimas fueron prometedores, como lo demuestra su capacidad para captar algunos aspectos estilísticos complejos incluso con recursos limitados. Esto subraya la necesidad de disponer de hardware más avanzado para maximizar las capacidades de los modelos XL, reduciendo compromisos en la calidad del entrenamiento y explorando completamente el potencial de los modelos más potentes.

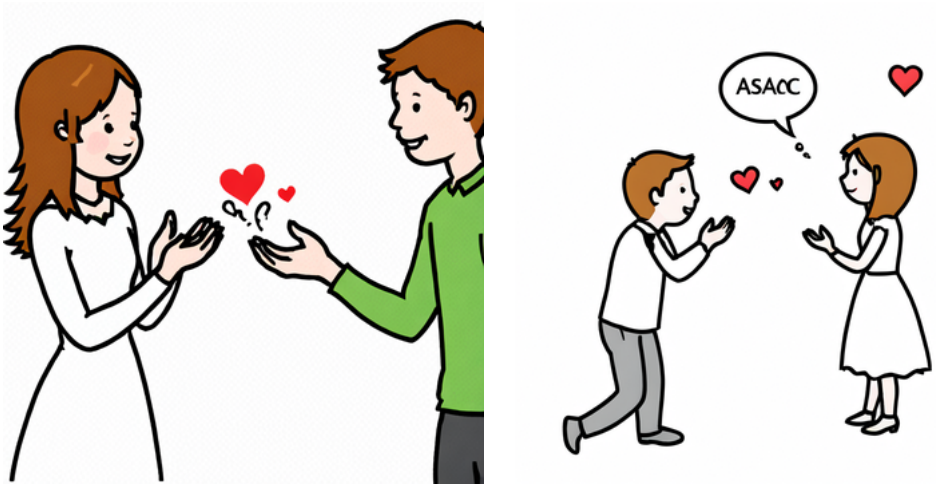
### 4.2.4. Decisiones técnicas y estratégicas del desarrollo

A lo largo del proyecto se tomaron una serie de decisiones clave que influyeron en la eficacia y versatilidad del sistema desarrollado. La elección de herramientas como *Kohya_ss* permitió un control detallado sobre los parámetros de entrenamiento, superando las limitaciones de plataformas más restrictivas ejecutadas en Google Colab.

Asimismo, el uso del *dataset* de ARASAAC se reveló como una base sólida gracias a su validación por profesionales del ámbito de la comunicación aumentativa. No obstante, se detectaron oportunidades de mejora en el formato y resolución de las imágenes, especialmente al trabajar con modelos más exigentes como SDXL.

Durante el proceso de entrenamiento, se comprobó que era preferible utilizar los modelos base como punto de partida, incluso aunque fueran más limitados en generación que los modelos derivados. Esto se debía a que presentaban menos corrupción en el codificador de texto (*text encoder corruption*) y facilitaban el aprendizaje. Además, los LoRA entrenados sobre modelos base como SDXL podían reutilizarse en versiones como DreamShaper XL o Juggernaut XL sin pérdida de calidad (ver figura 4.4).

Estas decisiones y estrategias permitieron alcanzar un estilo consistente, muy cercano al deseado, y añadieron el valor de experimentar con distintos modelos (SD 1.5 y SDXL). Se demostró que, aunque los modelos más limitados pueden cumplir con el objetivo, el uso de modelos más potentes como SDXL ofrece mejores resultados, a pesar de requerir mayores recursos computacionales para su entrenamiento.




(a) Generación sobre DreamShaper XL (b) Generación sobre Juggernaut XL

Figura 4.4: Prompt: “white background,arasaac, marriage proposal”; imágenes generadas en distintos modelos con SDXL como base, utilizando un LoRA entrenado para modelos XL.

## 4.3. Trabajo Futuro

A lo largo del desarrollo de este trabajo se han obtenido resultados prometedores, pero también se han identificado diversas áreas de mejora que podrían abordarse en el futuro. Se plantean dos líneas principales: la creación de un dataset más consistente y alineado con los requisitos técnicos de los modelos, y la mejora del proceso de entrenamiento mediante el uso de hardware más potente y modelos base más avanzados.

### 4.3.1. Optimización de un dataset

**Propuesta:** Crear un nuevo dataset que mantenga el estilo ARASAAC pero solucione los errores con capas transparentes y resoluciones alineadas con los modelos de entrenamiento.

**Ventajas esperadas:**

- Consistencia en el estilo de las imágenes, asegurando que todas las generaciones estén alineadas con los objetivos del entrenamiento.
- Formato homogéneo, con resoluciones de 512x512 px o 1024x1024 px, dependiendo de si el modelo es SD 1.5 o SDXL, y un espacio de color RGB que elimina problemas como capas transparentes o formatos inconsistentes.
- Mejora en la calidad del entrenamiento, evitando sesgos no deseados, como la generación automática de fondos de color verde oscuro cuando no se especifica un fondo explícitamente.

### 4.3.2. Optimización del entrenamiento con mejores recursos

**Propuesta:** Utilizar hardware más potente para realizar entrenamientos más largos y con modelos base avanzados como SDXL o SD 3.0.

**Impacto esperado:**

- Incrementar el número de pasos (de 600 a un rango mínimo de 3000-5000 pasos) para mejorar la estabilidad y calidad del LoRA.
- Ampliar el dataset a 50-100 imágenes en SDXL para obtener resultados más consistentes y robustos, incluso explorar datasets más grandes para evaluar si hay mejoras significativas en la calidad.
- En entrenamientos de sobre SD 3.0 las mejoras irían por el mismo camino que con SDXL. La limitación es el hardware que se usa para entrenar.

## 4.4. Conclusión Final

A lo largo de este trabajo se ha logrado un avance significativo en la creación y optimización de modelos LoRA para la generación de pictogramas enfocados a la Comunicación Aumentativa y Alternativa (CAA). A pesar de las limitaciones iniciales en cuanto a hardware y disponibilidad de datos, los resultados obtenidos han sido satisfactorios, permitiendo replicar de forma consistente las características visuales del estilo de los pictogramas de ARASAAC.

El uso de herramientas como `Kohya_ss` y el entrenamiento sobre modelos libres de Stable Diffusion como SD 1.5 ha permitido alcanzar un buen equilibrio entre calidad y eficiencia. Las pruebas realizadas demuestran que, mediante técnicas de IA generativa, es posible alcanzar la calidad necesaria para generar pictogramas aplicables en contextos reales de CAA.

No obstante, la optimización del *dataset* y el acceso a recursos computacionales más potentes permitirían alcanzar resultados aún más robustos. Por tanto, el trabajo futuro debería centrarse en mejorar la calidad del conjunto de datos de entrenamiento y en aprovechar modelos base más potentes, como SDXL. Asimismo, disponer de acceso a modelos como Stable Diffusion 3.0, que incorpora los codificadores de texto más avanzados, abriría la puerta a generar pictogramas de calidad superior. Con estos avances, sería posible superar incluso a sistemas de generación cerrados desarrollados por grandes compañías como OpenAI o Google, en el ámbito específico de la creación de pictogramas.

En resumen, este trabajo sienta unas bases sólidas para la mejora continua en la generación de pictogramas personalizados, accesibles y de calidad. Los resultados alcanzados muestran el enorme potencial de la inteligencia artificial generativa aplicada a la CAA, y abren el camino hacia herramientas más inclusivas, adaptables y eficaces para personas con dificultades comunicativas.

Chapter 5

# Introduction

## 5.1. Motivation and Importance of Pictograms in Communication

Since ancient times, civilizations have used symbols and images to convey ideas. Examples such as Egyptian hieroglyphs or Paleolithic cave art show how humans have relied on visual representation to communicate complex concepts without the need for spoken language. These pictographic systems laid the foundation for visual communication, which in many cultures preceded the development of formal written languages.

Today, pictograms play a crucial role in augmentative and alternative communication (AAC), especially for individuals with cognitive or interpretive difficulties. Various studies have demonstrated their usefulness in improving communication for people with conditions such as Down syndrome, autism, or age-related cognitive decline (Mirenda (2003); Lorah et al. (2015); Beukelman y Mirenda (2013)). These tools enable individuals to express their needs and better understand their environment.

However, the main drawback of pictograms is that each idea requires a specific one, and building a database large enough to cover every imaginable concept is virtually impossible.

This is where the idea for this bachelor's thesis arises: to leverage generative artificial intelligence to create custom pictograms in real time, adapted to each specific situation. This approach not only facilitates communication in everyday scenarios, but also empowers educators, therapists, and caregivers by providing them with a versatile and accessible tool that can transform the way they interact with people with special communication needs.

## 5.2. Objectives

The main objective of this project is to develop an artificial intelligence-based system capable of generating pictograms quickly, efficiently, and consistently, following the visual style of pictograms from the Aragonese center ARASAAC[1], (see Figure 5.1). This system must be accessible and flexible, allowing the creation of pictograms from simple descriptions in real time, without relying on pre-existing databases. To achieve this, free and open-source technologies will be used, ensuring the system remains free of charge and open to use and adaptation.

(a) (b) (c)

Figure 5.1: Pictograms from the ARASAAC center, the target style for generation in this project.

### 5.2.1. Secondary Objectives

- Ensure that the system is compatible with devices with limited computational resources, while maintaining high-quality pictogram generation.
- Guarantee that the system can adapt to diverse AAC needs, such as the use of simple color palettes, specifying object colors, or including additional elements in the pictograms.
- Facilitate system deployment and use through a clear and accessible guide that enables pictogram generation from simple descriptions, without requiring advanced technical knowledge or the configuration of additional tools.
- Maintain visual consistency with the ARASAAC pictogram style to support integration with existing materials and enhance user comprehension.

## 5.3. Work Plan

To achieve the objectives, the following tasks were carried out:

[1]ARASAAC: `https://arasaac.org/`

1. **Analysis of limitations in generative AI models:** Evaluate the limitations of the most popular generative AI models when attempting to generate simple pictograms to support communication for people with cognitive or interpretive difficulties.

2. **Pictogram generation:** Generate pictograms using various models available on platforms such as GitHub, Hugging Face, and CivitAI.

3. **Evaluation of results:** Analyze the output from different models to assess whether any are suitable for generating appropriate pictograms. If none succeed, select a specific model to further develop a new model capable of generating pictograms with a style similar to ARASAAC. Additionally, the model should be able to generate pictograms efficiently using few resources and minimal generation time.

4. **Dataset preparation:** In order to train a model for pictogram generation, a dataset must be prepared. This dataset should adhere to the ARASAAC visual style.

5. **Model training:** With the dataset and selected model, delve into the training process to achieve results that meet the defined objectives.

6. **Analysis of results:** Perform an analysis of the final results obtained.

## 5.4. Expected Contributions

The expected contributions of this project directly stem from the fulfillment of the primary and secondary objectives previously outlined. Together, they aim to optimize the use of pictograms in the context of Augmentative and Alternative Communication (AAC), improving accessibility for individuals with communication difficulties by enabling fast and efficient generation of pictograms with a simple, coherent visual style aligned with ARASAAC.

Key contributions of the project include:

- **Efficient pictogram generation:** The project enables the creation of high-quality pictograms from simple descriptions, facilitating real-time AAC applications with limited resources. This will allow professionals in education, healthcare, and other fields to generate pictograms quickly, without relying on pre-existing databases or specialized software.

  This contribution will be considered achieved if the system can generate quality pictograms using models that offer a good balance between output quality and computational cost.

- **Improved communication with individuals with cognitive or interpretive difficulties:** By providing pictograms tailored to users' specific needs,

the project is expected to enhance interaction in contexts where communication barriers exist, promoting effective inclusion and improving users' quality of life.

This contribution will be positively evaluated if the system is capable of representing concepts that do not typically have existing pictograms, such as specific or personalized situations. Examples include generating pictograms with specific color palettes, assigning specific colors to objects, or representing people with specific tools or garments.

- **Promotion of generative AI research applied to AAC:** This work also seeks to encourage further exploration of generative AI in the AAC field, opening new avenues for research and the development of accessible technologies. By using open-source technologies, the project aims to inspire other researchers and developers to continue improving these tools, expanding their applications, and increasing their accuracy and utility.

  This contribution will be considered achieved if the project demonstrates high-quality results that highlight the potential of generative AI applied to AAC, serving as a foundation or inspiration for future work in this area.

These contributions align with the goal of improving communication in contexts where cognitive or interpretive barriers hinder interaction, and they aim to establish a foundation for future research focused on accessible and customizable technologies.

Chapter 6

# Conclusions and Future Work

## 6.1. Introduction

This chapter presents the main conclusions of the project and proposes potential future work lines to improve the results obtained. The outcomes from training on different base models are discussed. In addition, limitations encountered, the impact of decisions and strategies adopted throughout the project, and opportunities for improvement through the use of new resources and techniques are analyzed.

## 6.2. Main Conclusions

This section summarizes the key insights gained throughout the development of the project, along with the most relevant technical decisions and results. It includes an analysis of how well the initial objectives were met, the quality achieved in the pictogram generation process, the impact of hardware limitations, and the strategic decisions taken during development. This final reflection provides a comprehensive evaluation of the work carried out.

### 6.2.1. Achievement of Objectives and Expected Contributions

The development of the system has successfully fulfilled both the main and secondary objectives defined at the beginning of the project.

Regarding the **main objective**, a system based on artificial intelligence has been implemented, capable of generating pictograms quickly and consistently, following the visual style of ARASAAC pictograms. The system enables pictogram generation from simple descriptions without relying on pre-existing databases, while also complying with the defined principles of accessibility and flexibility. By using a free and scalable technology such as Stable Diffusion, the system remains cost-free and

adaptable, as originally intended.

Regarding the **secondary objectives**:

- It has been demonstrated that the system can run on environments with limited computational resources. Since no additional resources are required during inference, LoRA models allow pictogram generation on personal computers with modest GPUs, as long as they meet the minimum requirements for running SD 1.5 or SDXL.
- The system has proven to adapt to various needs of Augmentative and Alternative Communication (AAC). It has been able to generate pictograms with simple color palettes, assign specific colors to objects, and include additional elements such as tools or clothing on human figures (3.4.3.1), validating its usefulness in contexts that require personalized pictograms.
- A clear and accessible guide[1] has been created on GitHub, allowing the system to be used without advanced technical knowledge. There is no need for direct manipulation of complex tools such as custom embeddings, refiners, or internal model configurations, making it accessible to non-expert users.
- Notable visual consistency with the ARASAAC style has been maintained, as evidenced by the generated results.

These findings support the conclusion that the system not only meets its functional objectives but also achieves the **expected contributions**:

- **Efficient pictogram generation:** The system enables real-time pictogram creation from text, without relying on databases, using optimized models that balance quality and resource consumption. This contribution is considered achieved, as high-quality pictograms were obtained using LoRA models, which do not require additional resources and can run with the minimum requirements of Stable Diffusion 1.5.
- **Adaptability to specific needs:** The system has demonstrated flexibility in generating customized pictograms, being capable of representing uncommon concepts using specific prompts. Moreover, the extension of the stickman style represents an additional contribution in addressing the challenge of meeting particular needs. This validates its usefulness in real AAC contexts, where closed databases lack sufficient coverage.
- **Promotion of research in AI and AAC:** The results obtained, as well as the documentation generated, show the potential of generative AI in the AAC domain. This contribution is considered fulfilled, as the project provides a solid foundation with high-quality results that can be reused and expanded by future projects or research.

[1] Usage guide in the README: `https://github.com/NILGroup/TFG-2324-Pictogramas`

### 6.2.2. Quality of the Results Obtained

A deeper analysis of the quality of the results obtained is presented below. The LoRAs developed, especially the main one based on SD 1.5, stood out for their quality and adaptability, even on devices with limited computational resources. This reflects the effectiveness of the training decisions made to optimize parameterization and make the most of the resources available.

Figure 6.1 shows examples generated in the main ARASAAC style using the LoRA based on SD 1.5, demonstrating its ability to faithfully replicate the distinctive features of the style.

(a) Firefighter (b) Cyclists

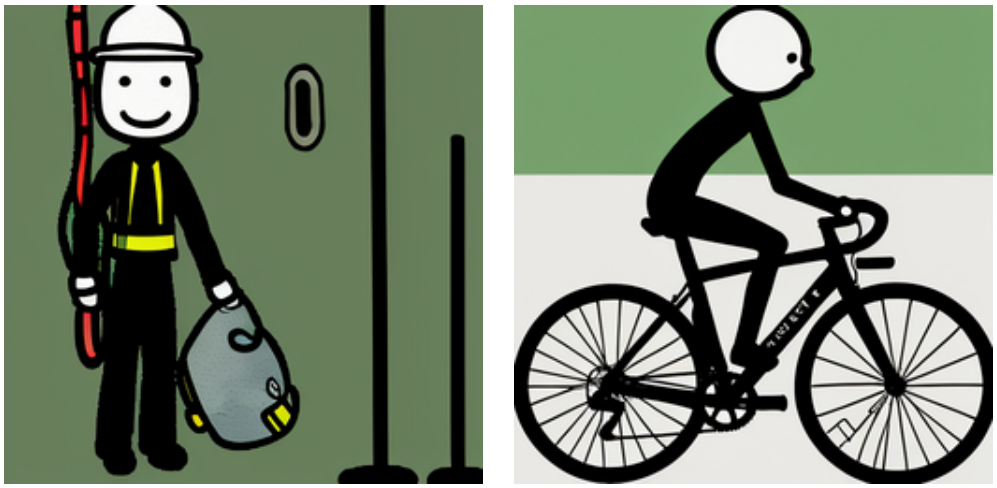
Figure 6.1: Pictograms generated in the main ARASAAC style (3.2a) using the LoRA based on SD 1.5.

In the case of the stickman style, although limitations were observed in complex environments or specific features, the human figures displayed characteristic stickman artwork, validating the system's potential to adapt to simple pictogram styles functional for augmentative communication. Figure 6.2 illustrates some examples in this style, highlighting its ability to capture the fundamental elements of stickman pictograms.

Experiments with XL models revealed their superior ability to generate more complex and detailed images. Figure 6.3 presents examples generated in the main ARASAAC style with SDXL, emphasizing the richness of detail and consistency achieved, even under hardware constraints.

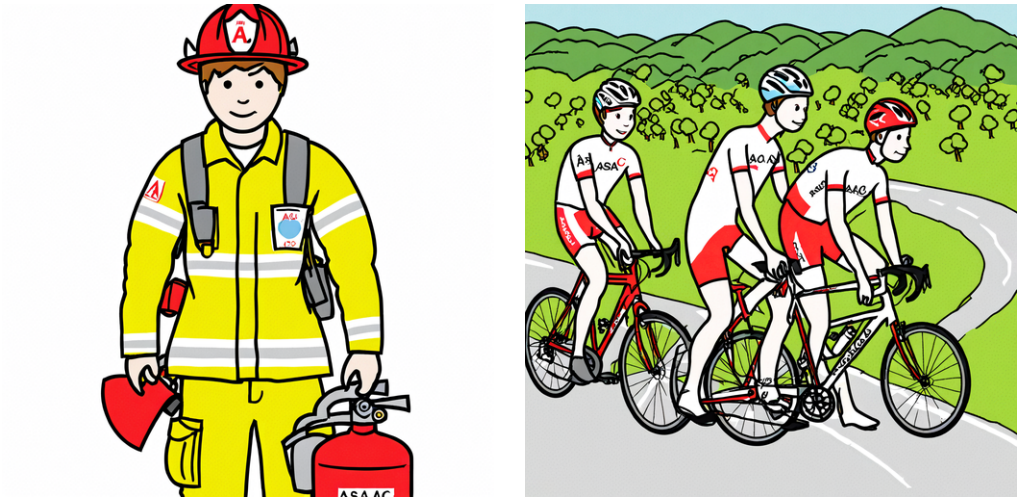
Based on the analysis of the results and the knowledge gained during the project, several key improvement areas were identified to further enhance performance:

- Increasing the homogeneity and quality of the dataset, removing transparent layers and optimizing image format from the beginning, would improve model learning.



(a) Firefighter (b) Cyclists

Figure 6.2: Pictograms generated in the *stickman* style (3.2b) using the LoRA based on SD 1.5.



(a) Firefighter (b) Cyclists

Figure 6.3: Pictograms generated in the main ARASAAC style (3.2a) using the LoRA based on SDXL.

- Exploring more advanced models, such as SDXL, showed great potential for handling complex visual styles, opening possibilities for achieving even more consistent results.
- Designing more precise datasets, even with smaller volumes of data, can maximize model potential and facilitate more efficient training.

In conclusion, the developed system fulfills its objectives by demonstrating versatility, efficiency, and adaptability to specific visual styles. With refinement of resources and techniques used, even higher quality in pictogram generation is achievable.

### 6.2.3. Impact of Hardware Limitations

The hardware available for this project (an RTX 3070 with 8 GB of VRAM) imposed constraints that especially affected more advanced models like SDXL. While models based on SD 1.5 could leverage limited resources to produce consistent and high-quality results, LoRAs on XL models were affected by the inability to perform long training sessions and the need to significantly reduce dataset size.

These limitations became apparent in the differentiation of complex concepts. For example, in experiments to distinguish between "boxer girl" and "boxer dog," the LoRA trained on SDXL struggled to accurately capture the concepts, as seen in Figures 3.40a and 3.40b. This behavior can be attributed to the limited number of training steps and the small dataset, which negatively impacted the model's generalization capabilities.

In contrast, the LoRA trained on SD 1.5 achieved significantly better results in these scenarios, as shown in Figures 3.19a and 3.19b. Thanks to the ability to conduct longer training sessions and use larger datasets, this model could consistently differentiate between the mentioned concepts.

Despite these differences, the results obtained with SDXL under suboptimal conditions were promising, demonstrating its ability to capture some complex stylistic elements even with limited resources. This highlights the need for more advanced hardware to fully unleash the potential of XL models, minimizing training quality compromises.

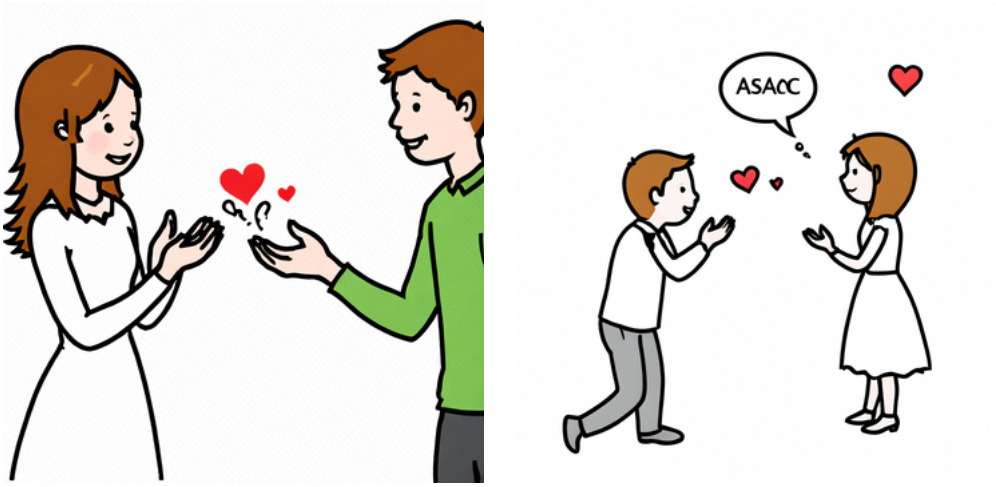
### 6.2.4. Technical and Strategic Development Decisions

Several key decisions were made throughout the project that influenced the system's effectiveness and versatility. The choice of tools like `Kohya_ss` enabled fine control over training parameters, overcoming the limitations of more restrictive platforms like Google Colab.

Likewise, the use of the ARASAAC dataset proved to be a solid foundation due to its validation by AAC professionals. However, opportunities for improvement were detected in image format and resolution, particularly when working with more demanding models like SDXL.

During training, it was found preferable to use base models as starting points, even if they were more limited in generation compared to derived models. This was because they exhibited less text encoder corruption and facilitated learning. Additionally, LoRAs trained on base models like SDXL could be reused on versions such as DreamShaper XL or Juggernaut XL without quality loss (see Figure 6.4).

These decisions and strategies allowed for achieving a consistent style, closely matching the desired result, and added value by experimenting with different models (SD 1.5 and SDXL). It was shown that although more limited models can achieve the objective, the use of more powerful models like SDXL offers better results, albeit requiring greater computational resources for training.




(a) Generation on DreamShaper XL (b) Generation on Juggernaut XL

Figure 6.4: Prompt: "white background, arasaac, marriage proposal"; images generated in different models with SDXL as the base, using a LoRA trained on the SDXL architecture.

## 6.3. Future Work

Promising results have been obtained throughout the development of this project, but several areas of improvement have also been identified that could be addressed in the future. Two main lines are proposed: the creation of a more consistent dataset aligned with the technical requirements of the models, and the improvement of the training process through the use of more powerful hardware and more advanced base models.

### 6.3.1. Dataset Optimization

**Proposal:** Create a new dataset that retains the ARASAAC style but resolves issues related to transparent layers and image resolutions aligned with training models.

**Expected Benefits:**

- Consistent image style, ensuring all generations align with the training objectives.
- Homogeneous format, with resolutions of 512x512 px or 1024x1024 px depending on whether the model is SD 1.5 or SDXL, and an RGB color space to eliminate problems such as transparent layers or inconsistent formats.
- Improved training quality, avoiding unwanted biases like the automatic generation of dark green backgrounds when no background is explicitly specified.

### 6.3.2. Training Optimization with Better Resources

**Proposal:** Use more powerful hardware to conduct longer training sessions with advanced base models like SDXL or SD 3.0.

**Expected Impact:**

- Increase the number of steps (from 600 to a minimum range of 3000–5000 steps) to improve LoRA stability and quality.
- Expand the dataset to 50–100 images in SDXL to achieve more consistent and robust results, and explore larger datasets to assess potential improvements in quality.
- For training on SD 3.0, improvements would follow a similar path to SDXL. The main limitation remains the hardware used for training.

## 6.4. Final Conclusion

Throughout this work, significant progress has been made in the creation and optimization of LoRA models for pictogram generation aimed at Augmentative and Alternative Communication (AAC). Despite initial limitations in terms of hardware and data availability, the results obtained were satisfactory, enabling consistent replication of the visual characteristics of ARASAAC pictograms.

The use of tools like `Kohya_ss` and training on open Stable Diffusion models like SD 1.5 allowed for a good balance between quality and efficiency. The tests carried out show that, through generative AI techniques, it is possible to reach the quality required to generate pictograms applicable in real AAC contexts.

Nevertheless, optimizing the dataset and accessing more powerful computing resources would allow for even more robust results. Therefore, future work should focus on improving the quality of the training dataset and leveraging more powerful base models, such as SDXL. Additionally, access to models like Stable Diffusion 3.0, which includes the most advanced text encoders, would open the door to generating even higher-quality pictograms. With these advances, it would be possible to surpass even closed-generation systems developed by large companies such as OpenAI or Google, in the specific domain of pictogram creation.

In summary, this work lays a solid foundation for continued improvement in the generation of high-quality, accessible, and personalized pictograms.

# Bibliografía


Bertola López, E. *Análisis empírico de las características formales de los símbolos pictográficos ARASAAC*. Tesis Doctoral, Universidad de Murcia, 2017. Tesis doctoral.

Beukelman, D. y Light, J. Augmentative and alternative communication: Supporting children and adults with complex communication needs. 2020.

Beukelman, D. R. y Mirenda, P. *Augmentative and Alternative Communication: Supporting Children and Adults with Complex Communication Needs*. Paul H. Brookes Publishing, 2013.

bmaltais. kohya_ss: A stable diffusion fine-tuning toolkit. 2024. Available at: `https://github.com/bmaltais/kohya_ss` (Accessed: 2024-11-08).

Drews, J., Weissmann, M., Keil, J., Dickmann, F. y Edler, D. A new ai tool for the design of cartographic pictograms (pictoai) and its potentials for increasing their meaningfulness. *KN-Journal of Cartography and Geographic Information*, páginas 1–13, 2025.

Esser, P., Kulal, S., Blattmann, A., Entezari, R., Müller, J., Saini, H., Levi, Y., Lorenz, D., Sauer, A., Boesel, F., Podell, D., Dockhorn, T., English, Z., Lacey, K., Goodwin, A., Marek, Y. y Rombach, R. Scaling rectified flow transformers for high-resolution image synthesis. *arXiv preprint arXiv:2403.03206*, 2024.

Face, H. Training with lora. `https://huggingface.co/docs/diffusers/en/training/lora`, n.d. Documentación práctica sobre LoRA en generación de imágenes.

Gal, R., Alaluf, Y., Atzmon, Y., Patashnik, O., Bermano, A. H., Chechik, G. y Cohen-Or, D. An image is worth one word: Personalizing text-to-image generation using textual inversion. 2022.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L. y Chen, W. Lora: Low-rank adaptation of large language models. 2021.

Imagen-Team-Google, :, Baldridge, J., Bauer, J., Bhutani, M., Brichtova, N., Bunner, A., Castrejon, L., Chan, K., Chen, Y., Dieleman, S., Du, Y., Eaton-Rosen, Z., Fei, H., de Freitas, N., Gao, Y., Gladchenko, E., Colmenarejo, S. G., Guo, M., Haig, A., Hawkins, W., Hu, H., Huang, H., Igwe, T. P., Kaplanis, C., Khodadadeh, S., Kim, Y., Konyushkova, K., Langner, K., Lau, E., Lawton, R., Luo, S., Mokrá, S., Nandwani, H., Onoe, Y., van den Oord, A., Parekh, Z., Pont-Tuset, J., Qi, H., Qian, R., Ramachandran, D., Rane, P., Rashwan, A., Razavi, A., Riachi, R., Srinivasan, H., Srinivasan, S., Strudel, R., Uria, B., Wang, O., Wang, S., Waters, A., Wolff, C., Wright, A., Xiao, Z., Xiong, H., Xu, K., van Zee, M., Zhang, J., Zhang, K., Zhou, W., Zolna, K., Aboubakar, O., Akbulut, C., Akerlund, O., Albuquerque, I., Anderson, N., Andreetto, M., Aroyo, L., Bariach, B., Barker, D., Ben, S., Berman, D., Biles, C., Blok, I., Botadra, P., Brennan, J., Brown, K., Buckley, J., Bunel, R., Bursztein, E., Butterfield, C., Caine, B., Carpenter, V., Casagrande, N., Chang, M.-W., Chang, S., Chaudhuri, S., Chen, T., Choi, J., Churbanau, D., Clement, N., Cohen, M., Cole, F., Dektiarev, M., Du, V., Dutta, P., Eccles, T., Elue, N., Feden, A., Fruchter, S., Garcia, F., Garg, R., Ge, W., Ghazy, A., Gipson, B., Goodman, A., Górny, D., Gowal, S., Gupta, K., Halpern, Y., Han, Y., Hao, S., Hayes, J., Heek, J., Hertz, A., Hirst, E., Hoogeboom, E., Hou, T., Howard, H., Ibrahim, M., Ike-Njoku, D., Iljazi, J., Ionescu, V., Isaac, W., Jana, R., Jennings, G., Jenson, D., Jia, X., Jones, K., Ju, X., Kajic, I., Kaplanis, C., Ayan, B. K., Kelly, J., Kothawade, S., Kouridi, C., Ktena, I., Kumakaw, J., Kurniawan, D., Lagun, D., Lavitas, L., Lee, J., Li, T., Liang, M., Li-Calis, M., Liu, Y., Alberca, J. L., Lorrain, M. K., Lu, P., Lum, K., Ma, Y., Malik, C., Mellor, J., Mensink, T., Mosseri, I., Murray, T., Nematzadeh, A., Nicholas, P., Nørly, S., Oliveira, J. G., Ortiz-Jimenez, G., Paganini, M., Paine, T. L., Paiss, R., Parrish, A., Peckham, A., Peswani, V., Petrovski, I., Pfaff, T., Pirozhenko, A., Poplin, R., Prabhu, U., Qi, Y., Rahtz, M., Rashtchian, C., Rastogi, C., Raul, A., Razavi, A., Rebuffi, S.-A., Ricco, S., Riedel, F., Robinson, D., Rohatgi, P., Rosgen, B., Rumbley, S., Ryu, M., Salgado, A., Salimans, T., Singla, S., Schroff, F., Schumann, C., Shah, T., Shaw, E., Shaw, G., Shillingford, B., Shivakumar, K., Shtatnov, D., Singer, Z., Sluzhaev, E., Sokolov, V., Sottiaux, T., Stimberg, F., Stone, B., Stutz, D., Su, Y.-C., Tabellion, E., Tang, S., Tao, D., Thomas, K., Thornton, G., Toor, A., Udrescu, C., Upadhyay, A., Vasconcelos, C., Vasiloff, A., Voynov, A., Walker, A., Wang, L., Wang, M., Wang, S., Wang, S., Wang, Q., Wang, Y., Ágoston Weisz, Wiles, O., Wu, C., Xu, X. F., Xue, A., Yang, J., Yu, L., Yurtoglu, M., Zand, A., Zhang, H., Zhang, J., Zhao, C., Zhaxybay, A., Zhou, M., Zhu, S., Zhu, Z., Bloxwich, D., Bordbar, M., Cobo, L. C., Collins, E., Dai, S., Doshi, T., Dragan, A., Eck, D., Hassabis, D., Hsiao, S., Hume, T., Kavukcuoglu, K., King, H., Krawczyk, J., Li, Y., Meier-Hellstern, K., Orban, A., Pinsky, Y., Subramanya, A., Vinyals, O., Yu, T. y Zwols, Y. Imagen 3. 2024.

Llorente-Hernando, S. y Romano-Ramos, I. Herramientas para la construcción de narraciones con apoyo gráfico. Trabajo de Fin de Grado, 2024. Facultad de Informática, UCM.

Lorah, E. R., Tincani, M., Dodge, J., Gilroy, S. y Holloway, J. Picture exchange and related alternative communication intervention for individuals with autism spectrum disorder: A meta-analysis. *Developmental Neurorehabilitation*, vol. 18(1), páginas 1–12, 2015.

Martín-Guerrero, A. PICTAR: una herramienta de elaboración de contenido para personas con TEA basada en la traducción de texto a pictogramas. Trabajo de Fin de Máster, 2018. Facultad de Informática, UCM.

Midjourney. Midjourney documentation. 2022.

Mirenda, P. Toward functional augmentative and alternative communication for students with autism: Manual signs, graphic symbols, and voice output communication aids. *Language, Speech, and Hearing Services in Schools*, vol. 34(3), páginas 203–216, 2003.

Nawar, H. Designing a generative pictographic language. En *Design, User Experience, and Usability: Designing Interactions* (editado por A. Marcus y W. Wang), páginas 285–296. Springer International Publishing, Cham, 2018. ISBN 978-3-319-91803-7.

Paolieri, D. y Marful, A. Norms for a pictographic system: The aragonese portal of augmentative/alternative communication (arasaac) system. *Frontiers in Psychology*, vol. Volume 9 - 2018, 2018. ISSN 1664-1078.

Parmar, G., Singh, K. K., Zhang, R., Li, Y., Lu, J. y Zhu, J.-Y. Zero-shot image-to-image translation. 2023.

Podell, D., English, Z., Lacey, K., Blattmann, A., Dockhorn, T., Müller, J., Penna, J. y Rombach, R. Sdxl: Improving latent diffusion models for high-resolution image synthesis. 2023.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C. y Chen, M. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, vol. 1(2), página 3, 2022.

Rankin, J., Harwood, K. y and, P. M. Influence of graphic symbol use on reading comprehension. *Augmentative and Alternative Communication*, vol. 10(4), páginas 269–281, 1994.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P. y Ommer, B. High-resolution image synthesis with latent diffusion models. En *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, páginas 10684–10695. 2022.

Thakur-Git-Notebooks. AutoTrain: No-code training for state-of-the-art models. En *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, páginas 419–423. Association for Computational Linguistics, Miami, Florida, USA, 2024.

Training-Guide-Valstrix. Valstrix's crash course guide to lora and lycoris training. `https://civitai.com/articles/3522/valstrixs-crash-course-guide-to-lora-and-lycoris-training`, 2024. Accessed: 2024-11-09.

White, J., Fu, Q., Hays, S., Sandborn, M., Olea, C., Gilbert, H., Elnashar, A., Spencer-Smith, J. y Schmidt, D. C. A prompt pattern catalog to enhance prompt engineering with chatgpt. 2023.

*He aquí el último suspiro de mi camino en esta carrera.*
*Tal vez no fue la aventura más memorable,*
*pero dejará una cicatriz en mi vida que jamás sanará.*
*Alejandro Antuña Rodríguez*