

UNIVERSIDAD DE LOS ANDES
FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS



MULTIMODAL GARMENT DESIGN

JOSÉ TOMÁS ANABALÓN DÍAZ

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN CIENCIAS DE LA COMPUTACIÓN

PROFESOR GUÍA: JOSÉ MANUEL SAAVEDRA

ING-IN-001/11

SANTIAGO, DICIEMBRE DE 2024

Certifico que he leído esta memoria y que en mi opinión su alcance y calidad son completamente adecuados como para ser considerada una memoria conducente al título de Ingeniero.

José Manuel Saavedra

Certifico que he leído esta memoria y que en mi opinión su alcance y calidad son completamente adecuados como para ser considerada una memoria conducente al título de Ingeniero.

Nombre Representante del Decano

Certifico que he leído esta memoria y que en mi opinión su alcance y calidad son completamente adecuados como para ser considerada una memoria conducente al título de Ingeniero.

Nombre Profesor Invitado

© José Tomás Anabalón Díaz 2024

Todos los derechos reservados.

Resumen

En este trabajo, se aborda la investigación de distintos modelos y técnicas generativas de imágenes tanto históricas como en el estado del arte, enfocandolo en el diseño de ropa, específicamente los vestidos. También la creación de un dataset para luego ser usado en el entrenamiento, validación y testo de diversos modelos. Junto a modelos preentrenados y los entrenados por nuestro dataset, se comparan resultados y en conjunto a las técnicas investigadas se desarrollan propuestas para las problemáticas de los modelos generativos.

Agradecimientos

Muchas gracias a todas las personas que han ayudado, aportado o que han guiado a lo largo de este trabajo y carrera. A todos los que han apoyado y han tenido fe, sobre todo a mis padres que han tenido paciencia. A mis hermanas que me han alentado en las distintas circunstancias. A mi amigo de carrera y de la vida, Julio Galindo. A la psicóloga asignada por el CAP, Ana María Lepeley Jungjohann, y mi psicóloga personal Laura Müller, por ayudarme a organizarme la vida y ver las cosas con otra perspectiva. A todas las personas que conocí a lo largo de la carrera y sobre todos a los amigos que forme en esta. A mi profesor guía, José Manuel Saavedra quien me ha estado exigiendo, apoyando y ayudando en esta travesía. Por último, a la Facultad de Ingeniería, por su comprensión y ayuda.

Este trabajo se lo dedico a mis padres, abuela, necke, bacon, rosa y nero.

Índice general

Resumen	IV
Agradecimientos	V
1. Introducción	1
1.1. Objetivos	2
1.1.1. Objetivo principal	2
1.1.2. Objetivos específicos	2
2. Revisión Bibliográfica	4
3. Experimento y Metodología	7
3.1. Construcción del dataset	7
3.1.1. Recolección de datos	7
3.1.2. Segmentación	7
3.1.3. Generación de bocetos	8
3.2. Modelos generativos seleccionados	8
3.2.1. Arquitecturas de GAN	9
3.2.2. Arquitectura de Transformadores	13
3.2.3. Arquitectura de Difusión	17
3.2.4. Configuración de los entrenamientos	24
4. Resultados	26
4.1. Validación del set de datos	26
4.2. Modelos Generativos y métricas	27
4.2.1. Pix2Pix	28
4.2.2. CycleGan	28
4.2.3. Taming Transformers (VQGan)	29
4.2.4. BBDM (Brownian Bridge Diffusion Model)	29
4.2.5. DDIM (Palette)	30

4.2.6. ControlNet con Stable Diffusion 1.5	30
5. Conclusiones del Análisis	40
Bibliografía	42
Anexos	44
A. Tablas y ranking	44

Índice de figuras

3.1. Generación de bocetos	8
3.2. Gráfico de <i>loss</i> vs <i>steps</i> en el entrenamiento Pix2Pix	10
3.3. De Izq. a Drch: Boceto, vestido, gen. pix2pix, gen. resnet	12
4.1. En la primera fila se tiene los dibujos de testo, lo siguen de Pix2Pix, después ambas CycleGan(Pix y ResNet respectivamente), más abajo está ControlNaet y después ambos Pallet(64x64 y 96x96 respectivamente)	32
4.2. Generación de bocetos mediante IP2P	33
4.3. Generación de bocetos mediante PiDiNet	34
4.4. Generación de bocetos mediante combinación de ambas	35
4.5. Bocetos hechos a mano	36
4.6. Datos del entrenamiento y validación	36
4.7. Imágenes generadas por Pix2Pix del data set de validación	36
4.8. Imágenes generadas por CyclePix del data set de validación	37
4.9. Imágenes generadas por CycleResnet del data set de validación	37
4.10. Imágenes de testeo para VQGAN	37
4.11. Imágenes generadas por VQGAN Nopixa en testeo	37
4.12. Imágenes generadas por VQGAN Det en testeo	37
4.13. Imágenes generadas por VQGAN sin preentrenamiento despues de 30 epochs	38
4.14. Imágenes de input a modo testeo en BBDM	38
4.15. Imágenes generadas por BBDM en los datos detesteo	38
4.16. En la primera fila se encuentra los vestidos reales, abajo de estos los sketches y las dos filas de más abajo corresponden a las imagenes generadas por Pallet 64x64 y 96x96 respectivamente	39
4.17. Imágenes generadas por ControlNet en testeo	39
A.1. Collage de vestidos obtenidos online	44
A.2. Ranking FID en el dataset Cifar-10	45

Índice de cuadros

4.1. Tabla comparativa de FID, LPIPS y SSIM por modelo en diferentes datasets	27
---	----

Capítulo 1

Introducción

El diseño de prendas de vestir ha experimentado una transformación significativa en los últimos años gracias al avance de las tecnologías digitales. Entre estas, los modelos generativos destacan como herramientas clave en la creación de imágenes que simulan bocetos y prototipos de prendas, lo cual permite a diseñadores y empresas optimizar procesos creativos y de producción. La capacidad de estas tecnologías para generar contenidos visuales personalizados y de alta calidad abre nuevas posibilidades en varias industria, como la moda, especialmente en el contexto de plataformas de comercio electrónico, en las cuales la presentación visual de los productos es esencial para atraer a los consumidores.

A pesar de los avances, la generación de diseños de moda presenta desafíos significativos, como, por ejemplo, la calidad limitada de las imágenes, la falta de diversidad en los diseños y la dificultad para integrar de manera efectiva características específicas de las prendas en las imágenes generadas. Estos problemas limitan la aplicabilidad de los modelos generativos en escenarios reales y destacan la necesidad de desarrollar métodos más robustos y personalizados para este propósito. En este contexto, surge la oportunidad de explorar modelos generativos avanzados y estrategias multimodales que combinen datos visuales y textuales para mejorar la calidad y versatilidad en el diseño de prendas. En particular, se ha dado prioridad a modelos *image-to-image* por su enfoque directo en la transformación de imágenes para generar resultados visuales realistas y por lo que se va a trabajar en esta memoria.

1.1. Objetivos

1.1.1. Objetivo principal

El objetivo principal de esta investigación es analizar y evaluar modelos generativos avanzados y las técnicas más destacadas del estado del arte, por medio de diferentes métricas con la finalidad de abordar los principales desafíos que enfrentan los modelos actuales en la generación de imágenes de prendas de vestir.

1.1.2. Objetivos específicos

Estos desafíos incluyen la alta dependencia de grandes volúmenes de datos de entrenamiento, la lentitud en los procesos de aprendizaje, y la inconsistencia en la representación visual de las características deseadas en las imágenes generadas. A través de esta investigación, se busca identificar estrategias que permitan mejorar la diversidad, la calidad y la precisión de las imágenes generadas. Además, se pretende proponer soluciones que optimicen estos modelos, que facilite tanto la personalización como la aplicabilidad en contextos reales. Sobre la base en estos hallazgos, se desarrollará una visión clara sobre qué enfoques pueden ser más efectivos para resolver estos problemas y guiar futuros avances en el diseño multimodal de prendas de vestir.

- a) **Comprender los fundamentos de los modelos generativos modernos para síntesis de imágenes** Tanto la investigación y selección de los distintos modelos y técnicas están basadas en, su importancia histórica , contexto en el estado del arte, y su aplicabilidad para este trabajo, considerando criterios como la facilidad de implementación, disponibilidad de recursos preentrenados, eficiencia computacional, y su capacidad para adaptarse al dominio específico de imágenes de vestidos.
- b) **Generar un conjunto de datos para el contexto** En cuanto a la creación del dataset propio, se realizó un *web scraping* de dos grandes tiendas de ropa. Luego, se segmentaron las imágenes para obtener solo la prenda de vestir. Para la realización de los bocetos, se utilizó una combinación de los resultados generados por dos modelos generativos diferentes, integrando sus salidas mediante un enfoque combinado para mejorar la diversidad y realismo de los bocetos, seguido de un filtrado para introducir distorsiones que simulen las irregularidades presentes en *sketches* manuales. Con este proceso, cada imagen obtenida tiene tres bocetos asociados. Este dataset se separó en 1/3 para la evaluación

y el resto corresponde al entrenamiento. En cuanto al *testeo*, se realizarán 101 dibujos de vestidos a mano.

- c) **Desarrollo de un framework para entrenar modelos de generación de imágenes desde bocetos** Se entrenaron diversos modelos generativos utilizando el dataset previamente construido. Aunque el preprocesamiento de imágenes y la configuración del conjunto de datos no variaron significativamente entre ellos, fue necesario realizar ajustes específicos debido a la incompatibilidad de algunas librerías con el hardware utilizado. Además, se hicieron modificaciones menores en ciertos modelos para admitir bocetos como datos de entrada y vestidos reales como condición asociada para la generación de imágenes.
- d) **Evaluación de resultados usando conjunto de datos sintéticos y reales** Una vez entrenados los modelos y preparados los preentrenados, se generaron imágenes a partir del conjunto de datos de prueba, seleccionando cuidadosamente ejemplos representativos para la evaluación. Con estas imágenes, se analizaron distintas métricas cuantitativas y se compararon los resultados visualmente, así como los rendimientos de cada modelo en términos de tiempo de entrenamiento, función de pérdida (*loss*) y error de evaluación (*evaluation loss*).

Capítulo 2

Revisión Bibliográfica

En los últimos años, los modelos generativos han avanzado significativamente en la tarea de generación de imágenes y han logrado resultados sorprendentes en diversos conjuntos de datos, incluidos aquellos que se enfocan en áreas como la moda. Existen varias arquitecturas prominentes, entre las cuales destacan las Generative Adversarial Networks (GAN, por sus siglas en inglés) Goodfellow & others (2014) y los Diffusion Models Ho, Jain & Abbeel (2020)d, por ejemplo. Ambas tecnologías, aunque parten de fundamentos distintos, han demostrado ser herramientas poderosas para la creación de imágenes sintéticas, pero con diferencias clave que las hacen únicas. A pesar de los avances logrados por varios enfoques, persisten problemas como la inestabilidad en el entrenamiento, el colapso de modo en las GANs y, en los Diffusion Models, la necesidad de grandes cantidades de datos y poder computacional para generar imágenes realistas. Estos obstáculos, aunque comunes en muchos sectores, son aún más críticos en el ámbito del diseño de moda y el comercio online, donde la generación de imágenes de productos o bocetos de prendas debe ser rápida, eficiente y de alta calidad para satisfacer las demandas del mercado y del usuario. Este trabajo explora el potencial de los modelos generativos en el diseño de moda y e-commerce, evaluando su capacidad para generar imágenes realistas de productos a partir de bocetos.

Las GAN se basan en un sistema de competencia entre dos redes: el generador y el discriminador. El primero crea imágenes sintéticas a partir de un ruido aleatorio, mientras que el discriminador intenta distinguir entre imágenes reales y generadas. Este enfoque es clave porque fomenta una competencia dinámica en la que el generador mejora sus imágenes para engañar.^a al discriminador, mientras que este último mejora para ser más preciso en su tarea de clasificación . Con el paso del tiempo, este concepto básico ha evolucionado a modelos más sofisticados como StyleGAN y, más recientemente, StyleGAN-XL, que ocupan posiciones destacadas Takida, Imai-

zumi, Shibuya, Lai, Uesaka, Murata & Mitsufuji (2023) en la tabla de clasificación de Fréchet Inception Distance (FID, por sus siglas en inglés) en el conjunto de datos CIFAR-10, lo que demuestra su capacidad para producir imágenes realistas con un FID de 1.36 Karras, Laine & Aila (2019) .

El FID se ha convertido en una métrica clave para evaluar la calidad de las imágenes generadas, que compara la distribución de las características de las imágenes reales con las generadas . Un valor más bajo de FID indica una mayor similitud entre estas distribuciones, lo que sugiere una mejor calidad en la generación de imágenes. Los modelos GAN han sido muy eficaces en reducir este valor, gracias a arquitecturas como StyleGAN, que introdujo una mejor controlabilidad sobre los estilos de las imágenes, lo que permite la generación de variaciones realistas a partir de un ruido latente .

Sin embargo, las GAN no están exentas de problemas. Uno de los principales desafíos es no poder capturar toda la diversidad y tampoco tener la estabilidad en el entrenamiento, ya que el proceso competitivo puede generar oscilaciones que resulten en la incapacidad del generador para mejorar, conocido como el problema de colapso de modo. En este punto, los Diffusion Models han comenzado a destacarse y aportan una solución robusta a la generación de imágenes. A diferencia de las GAN, que generan imágenes con de una sola iteración, los Diffusion Models descomponen la tarea en un proceso de múltiples pasos, lo que difumina progresivamente el ruido hasta llegar a una imagen clara . Esta técnica no solo evita los problemas de inestabilidad típicos de las GAN, sino que también ofrece un control más preciso sobre la calidad del resultado final.

El modelo más reciente y destacado en la tabla FID de CIFAR-10 es el *Generative Modeling with Explicit Memory* (GMem:, por sus siglas en inglés), que lidera el ranking con un FID de 1.33. GMem lo que hace es optimizar el uso de memoria para bajar la demanda computacional con lo que puede entrenar mejor y más rápido, esto aunque suene simple o no muy innovador, lo llevó al primer lugar de la tabla comparativa Tang et al. (2025). Los Diffusion Models, como Gmem, GDD-I, y CTM, ocupan varias de las primeras posiciones en el ranking de FID, lo que demuestra la robustez de estos modelos en la creación de imágenes ultra realistas.

Los Diffusion Models son únicos en la forma en que manejan el ruido durante el proceso de generación. En lugar de partir de un punto fijo como las GAN, comienzan con ruido puro y aplican una serie de pasos graduales de denoising para llegar a la imagen final . Este proceso de refinamiento ha demostrado ser extremadamente efectivo para evitar el problema de colapso de modo que afecta a las GAN. Modelos

como GDD-I Zheng & Yang (2024), que combina tanto técnicas de difusión como GAN, han logrado un equilibrio que maximiza tanto la precisión como la velocidad y han logra un FID de 1.54 y puntuaciones de Inception superiores a 10.10, lo cual demuestra que esta combinación puede ser una poderosa herramienta .

Otro aspecto relevante de los Diffusion Models es su capacidad para realizar ajustes precisos en cada paso de denoising. Por ejemplo, modelos como Discriminator Guidance (con un FID de 1.64) aprovechan los discriminadores de las GAN Ho et al. (2020)para guiar el proceso de denoising, que mejora la precisión de la imagen final . Esta técnica híbrida, que combina los mejores aspectos de ambos enfoques, ha ganado popularidad por su capacidad para generar imágenes altamente detalladas con menos pasos computacionales.

Aplicando estos avances a la moda, las posibilidades son enormes. En un contexto como el de la generación de imágenes de ropa o bocetos para diseñadores, los Diffusion Models pueden ofrecer un control excepcional sobre el nivel de detalle y la coherencia de los patrones. Modelos como StyleSAN-XL, con su controlabilidad de estilos, podrían ser utilizados para crear imágenes de prendas con distintas variaciones de textura, color y forma, mientras que los Diffusion Models podrían asegurar que las imágenes generadas mantengan un alto nivel de realismo en cada iteración, lo que evita inconsistencias o patrones irreales en las prendas generadasDhariwal & Nichol (2021).

En términos de estado del arte, se observa una clara tendencia hacia los Diffusion Models, como se evidencia en la tabla de clasificación de CIFAR-10 Figura A.2. Aunque las GAN aún dominan ciertos espacios, como la generación controlada de estilos con StyleGAN, los modelos de difusión, como SIMS y CTM, continúan mejorando los estándares de calidad. Además, los desarrollos recientes, como PFGM++, están explorando formas de mejorar aún más los tiempos de convergencia y la estabilidad en el entrenamiento, lo que indica que estos modelos seguirán siendo una pieza fundamental en el futuro de la generación de imágenes Liuet al. (2023).

En resumen, tanto las GAN como los Diffusion Models y otras técnicas como Transformadores, presentan ventajas claras para la generación de imágenes, y ambos pueden ser aplicados eficazmente en el diseño de moda. Los avances en el control de estilos, la mejora de la calidad de las imágenes generadas y la estabilidad del entrenamiento han posicionado a estos modelos como los líderes indiscutibles en la creación de imágenes realistas . Al integrar estas tecnologías en el proyecto de diseño de prendas, se podría aprovechar estas arquitecturas para obtener imágenes coherentes y detalladas de alta calidad, adecuadas para ser utilizadas en la industria de la moda.

Capítulo 3

Experimento y Metodología

Se aborda el problema de generación de imágenes de prendas desde una perspectiva multimodal, utilizando modelos generativos para convertir imágenes de referencia en bocetos y viceversa. Este enfoque permite explorar nuevas formas de diseño asistido por inteligencia artificial, facilitando aplicaciones en e-commerce y diseño de moda.

El diseño experimental incluye la construcción de un dataset extenso y la implementación de diversos modelos generativos, seleccionados por sus capacidades de conversión imagen-a-imagen, su robustez para manejar diferentes entradas modales y dificultad o requerimientos para su entrenamiento/uso.

3.1. Construcción del dataset

3.1.1. Recolección de datos

Se recolectaron 18,586 imágenes de vestidos mediante web scraping desde plataformas de comercio electrónico, asegurando una diversidad en estilos, colores y diseños. Se aplicaron criterios de selección para eliminar imágenes duplicadas o de baja calidad.

3.1.2. Segmentación

Para aislar las prendas del fondo y otros elementos irrelevantes, se usaron modelos como YOLOv8 para detección de objetos y Segment Anything Model (SAM) para segmentación. Los resultados se validaron manualmente para garantizar precisión.

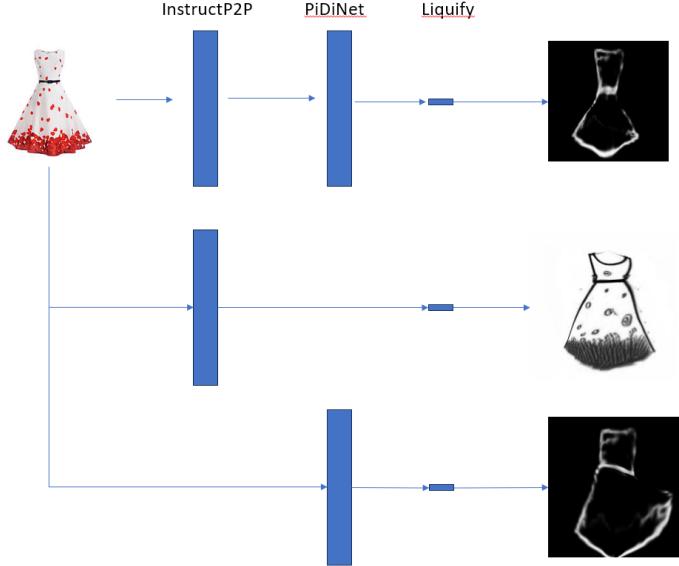


Figura 3.1: Generación de bocetos

3.1.3. Generación de bocetos

Se generaron tres tipos de *sketches* por imagen, utilizando distintas técnicas y filtro los cuales son:

InstructPix2Pix(IP2P): Para generar bocetos estilizados de las imágenes a partir de un *prompt* como : "dress drawing in black and white". PiDiNet: Enfocado en la detección precisa de bordes. Filtro personalizado Liquify: Para aplicar deformaciones estilizadas en función de imitar el error humano.

Elegidos especialmente gracias a sus cualidad de diversidad para IP2P o la recreación de un boceto parecidos a los humanos como PiDiNet. Con estos 3 elementos se hizo una combinación de IP2P con Liquify, después PiDiNet con Liquify y para terminar se utilizó primero IP2P y sobre sus resultados aplicamos PiDiNet y se finaliza con el filtro Liquify, como se puede ver en la figura 3.1. Los resultados se combinaron en un dataset estructurado con 55,758 imágenes (18,586 imágenes reales y 55,758 bocetos).

Se realizó una división del dataset : 67 porcientos para entrenamiento. 33 porcientos para validación. Y para el testeо se realizaron 101 dibujos a mano.

3.2. Modelos generativos seleccionados

Se seleccionaron seis modelos por su capacidad para convertir imágenes en diferentes dominios, garantizando versatilidad y calidad: Pix2Pix, CycleGAN, BBDM,

Palette, Taming Transformers y ControlNet.

3.2.1. Arquitecturas de GAN

Los modelos GAN son un algoritmo de inteligencia artificial que se utilizan en el aprendizaje no supervisado, implementadas por un sistema de dos redes neuronales que compiten mutuamente Wikipedia (2024), el generador y el discriminador. Como dice su nombre, el generador crea una imagen y el discriminador evalúa si la imagen producida es real o generada.

Pix2Pix

Este es un modelo de inteligencia artificial basado en redes generativas adversarias condicionales (cGAN) desarrollado para tareas de traducción de imagen a imagen Isola et al. (2018).

Arquitectura Generador El generador de Pix2Pix utiliza una arquitectura U-Net, que es ideal para tareas de traducción de imagen a imagen. Esta arquitectura se caracteriza por un *stack* de *downsampling* (codificador) que reduce progresivamente la dimensionalidad espacial mientras aumenta la profundidad de características, un *stack* de *upsampling* (decodificador) que recupera la dimensión espacial original, y conexiones de salto (*skip connections*) que conectan capas del codificador con sus correspondientes en el decodificador, lo que ayuda a preservar detalles espaciales y facilita el flujo del gradiente durante el entrenamiento. La implementación sigue un patrón de 8 capas para el *downsampling* (de 256×256 hasta 1×1) y 7 capas para el *upsampling*, usando convoluciones y convoluciones transpuestas respectivamente, con normalización por lotes y *dropout* en algunas capas para estabilizar el entrenamiento.

Discriminador El discriminador sigue una arquitectura PatchGAN, que evalúa la autenticidad de la imagen en parches, no globalmente. Esta estructura permite capturar mejor las texturas y estructuras locales de la imagen. El discriminador toma como entrada tanto la imagen de origen como la imagen objetivo (real o generada), concatena ambas imágenes en el canal, aplica una serie de capas convolucionales con normalización por lotes y LeakyReLU, y produce un mapa de características donde cada valor representa la probabilidad de que un parche específico de la imagen sea real.

Función de pérdida La pérdida del generador combina múltiples términos:

- **Pérdida adversarial (GAN loss)**: Mide qué tan bien el generador engaña al discriminador.
- **Pérdida L1**: Calcula la diferencia absoluta entre la imagen generada y la imagen objetivo, incentivando la similitud píxel a píxel.
- **Pérdida Total Variation**: Este es la suma entre GAN loss y el loss L1 multiplicado por (lambda), un hiperparámetro que definimos en la configuración. Esta pérdida penaliza cambios bruscos en la imagen, promoviendo suavidad.

La pérdida del discriminador usa dos componentes:

- **Pérdida real**: Evalúa la capacidad del discriminador para identificar imágenes reales como reales.
- **Pérdida generada**: Evalúa la capacidad del discriminador para identificar imágenes generadas como falsas.

La función de pérdida final del discriminador se calcula como la suma de ambas componentes.

Experiencia El modelo se pudo entrenar sin mayores problemas, los resultados eran esperados y se pudo apreciar una problemática de las GANs y es la inconsistencia del loss en su entrenamiento como se puede ver en la figura 3.2.

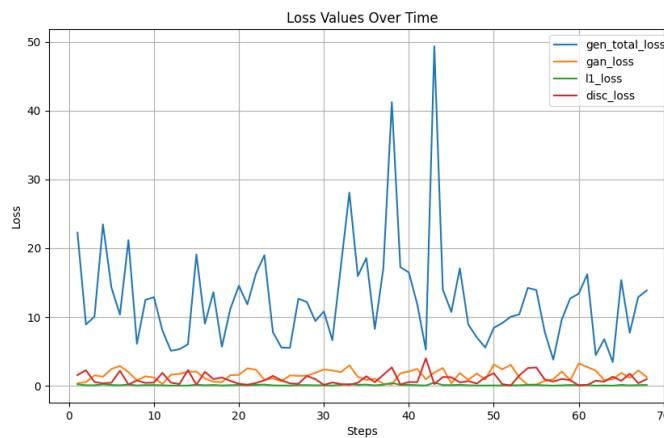


Figura 3.2: Gráfico de *loss* vs *steps* en el entrenamiento Pix2Pix

CycleGAN

Este es un modelo de inteligencia artificial basado en redes generativas adversarias que permite la traducción de imagen a imagen sin necesidad de pares de imágenes alineadas Zhu et al. (2020). El objetivo principal de CycleGAN es establecer un mapeo entre dos dominios distintos (X e Y) de manera que las imágenes traducidas sean indistinguibles de las imágenes del dominio objetivo.

CycleGAN genera un ciclo de aprendizaje donde se transforma una imagen de un dominio X (en nuestro caso, los bocetos) hacia otro dominio Y (vestidos). El mapeo se realiza con dos generadores: $G : X \rightarrow Y$ y $F : Y \rightarrow X$, completando un ciclo de consistencia donde $F(G(X)) \approx X$ y $G(F(Y)) \approx Y$.

Arquitecturas de Generador Para nuestros experimentos, implementamos y comparamos dos arquitecturas de generador diferentes:

- **Generador ResNet:** Basado en una arquitectura con bloques residuales que permite una mejor retención de características visuales. La implementación consta de:
 - Una capa inicial de padding reflexivo y convolución 7×7 seguida de normalización y ReLU.
 - Capas de downsampling que duplican la profundidad de características mientras reducen las dimensiones espaciales, con normalización y dropout opcional (con probabilidad 0.5).
 - 9 bloques residuales consecutivos que mantienen la resolución espacial pero permiten un aprendizaje más profundo.
 - Capas de upsampling mediante convolución transpuesta para recuperar la resolución original.
 - Una capa final de convolución 7×7 con activación $tanh$.
- **Generador Pix2Pix:** Descrito en la sección 3.2.1, basado en la arquitectura U-Net con conexiones de salto.

Arquitectura del Discriminador El discriminador implementado (**CycleConvDiscriminator**) sigue una arquitectura PatchGAN, similar a la utilizada en Pix2Pix .



Figura 3.3: De Izq. a Drch: Boceto, vestido, gen. pix2pix, gen. resnet

Funciones de Pérdida La implementación utiliza múltiples funciones de pérdida que son fundamentales para el entrenamiento estable y efectivo del modelo:

- **Pérdida Adversarial:** Implementamos varias opciones, incluyendo GAN estándar, hinge v1, hinge v2, LSGAN (Mean Squared Error) y WGAN. Para nuestros experimentos principales utilizamos LSGAN, donde:
 - La pérdida del discriminador minimiza el error cuadrático medio entre las predicciones de imágenes reales y el valor 1, y entre las predicciones de imágenes falsas y el valor 0.
 - La pérdida del generador minimiza el error cuadrático medio entre las predicciones de imágenes generadas y el valor 1.
- **Pérdida de Ciclo (Cycle Loss):** Implementada como error absoluto medio (MAE) entre la imagen original y la imagen reconstruida después de pasar por ambos generadores ($F(G(X)) \approx X$ y $G(F(Y)) \approx Y$). Esta pérdida es fundamental para mantener la consistencia estructural entre la imagen original y la reconstruida.
- **Pérdida de Identidad (Identity Loss):** También implementada como MAE, penaliza cuando un generador aplicado a una imagen de su propio dominio objetivo produce cambios significativos ($G(Y) \approx Y$ y $F(X) \approx X$). Esto ayuda a preservar características como el color cuando no es necesario transformarlas.
- **Penalización de Gradiente (Gradient Penalty):** Aplicada opcionalmente para estabilizar el entrenamiento.,

La función de pérdida total del generador se calcula como:

$$\begin{aligned}
 G_loss = & (A2B_g_loss + B2A_g_loss) + \\
 & (A2B2A_cycle_loss + B2A2B_cycle_loss) \times \lambda_{cycle} + \\
 & (A2A_id_loss + B2B_id_loss) \times \lambda_{identity} \quad (3.1)
 \end{aligned}$$

Donde λ_{cycle} y $\lambda_{identity}$ son hiperparámetros que controlan el peso relativo de cada componente de la pérdida.

La función de pérdida del discriminador incluye las pérdidas para imágenes reales y generadas, junto con la penalización de gradiente cuando está habilitada:

$$D_loss = (A_d_loss + B2A_d_loss) + (B_d_loss + A2B_d_loss) + (D_A_gp + D_B_gp) \times \lambda_{gp} \quad (3.2)$$

Donde λ_{gp} es el peso de la penalización de gradiente.

Experiencia En cuanto a su entrenamiento no hubo mayores problemas. Con el *cycle loss* como función de perdida hubo una mayor estabilización en el entrenamiento, esto se ve reflejado también en la calidad de las fotos y variación. Siguiendo, se probó con distintos generadores, vimos una clara mejora con el ResNet sobre el generador Pix2Pix. Dando una mejor reconstrucción tanto vestidos a sketches como sketches a vestidos. Se ve un mejor control de la imagen, sobre todo calidad y variación en comparación a Pix2Pix, como observamos en la figura 3.3, pero esta mejora se aprecia mejor en cuando reconstruyen imágenes desde el testeo, donde no se tiene vestido real, solo bocetos, estos se muestran más adelante.

3.2.2. Arquitectura de Transformadores

Los *Transformers* Vaswani et al. (2023) han demostrado ser una arquitectura fundamental en tareas de modelado secuencial, como el procesamiento de lenguaje natural (NLP, por sus siglas en inglés) en varios modelos dentro del estado del arte como los famosos GPT(*Generative Pre-trained Transformer*). Su capacidad para modelar dependencias a largo plazo dentro de secuencias ha permitido su aplicación en la síntesis de imágenes de alta resolución. A diferencia de las redes convolucionales (CNNs), que aplican filtros espaciales locales, los *Transformers* se basan en un mecanismo de *self-attention*, lo que les permite capturar relaciones de largo alcance en una imagen.

El principio central de los *Transformers* es la atención auto-regresiva, donde cada elemento de la secuencia se relaciona con todos los demás mediante pesos de atención. Matemáticamente, el mecanismo de *self-attention* se define como:

$$\text{Attention}(Q, K, V) = \text{softmax} \left(\frac{QK^T}{\sqrt{d_k}} \right) V, \quad (3.3)$$

donde:

- Q (query), K (key) y V (value) son transformaciones lineales de la entrada.
- d_k es la dimensionalidad de las claves.
- La normalización por $\sqrt{d_k}$ estabiliza la propagación de gradientes.

Los *Transformers* pueden ser utilizados de manera auto-regresiva en imágenes tratándolas como una secuencia de tokens visuales, lo que permite modelar dependencias globales en la composición visual.

Taming Transformers Condicional (VQGAN)

Para aplicar *Transformers* en la generación de imágenes de alta resolución, se requiere una representación comprimida y semánticamente rica de la imagen. Taming transformers (Esser, Rombach & Ommer, 2021) se introduce como un modelo de dos etapas que permite aprender representaciones discretas de imágenes en un espacio latente eficiente con la ayuda de VQGAN(*Vector Quantized GAN*) y luego modelar su composición con un *Transformer*.

Codificación y Compresión con VQGAN El modelo VQGAN se basa en la cuantización vectorial para convertir una imagen en una secuencia de códigos discretos. Se compone de:

- Un **encoder** $E(x)$ que transforma la imagen en una representación latente z .
- Una cuantización vectorial $q(\cdot)$ que asigna cada vector latente al código más cercano en un diccionario aprendido.
- Un **decoder** $G(z_q)$ que reconstruye la imagen a partir de la representación cuantizada.

Este proceso se define matemáticamente como:

$$z_q = q(E(x)) = \arg \min_{z_k \in Z} \|E(x) - z_k\|_2, \quad (3.4)$$

donde $Z = \{z_k\}_{k=1}^K$ es el código aprendido con K vectores en el diccionario. La reconstrucción de la imagen se obtiene como:

$$\hat{x} = G(z_q). \quad (3.5)$$

El entrenamiento de VQGAN se realiza con la siguiente función de pérdida:

$$L_{\text{VQ}}(E, G, Z) = \|x - \hat{x}\|_2^2 + \|\text{sg}[E(x)] - z_q\|_2^2 + \|\text{sg}[z_q] - E(x)\|_2^2. \quad (3.6)$$

Donde:

- El primer término es una **pérdida de reconstrucción** que minimiza la diferencia entre la imagen original y la generada.
- El segundo término es una **pérdida de compromiso**, que obliga al encoder a asignar códigos válidos del diccionario.
- $\text{sg}[\cdot]$ es una operación *stop-gradient* que evita la actualización de ciertos parámetros en la cuantización.

Mejorando la Representación con un Discriminador Adversarial Para mejorar la calidad visual de las imágenes generadas, se introduce una pérdida adversarial con un discriminador D , similar a una GAN:

$$L_{\text{GAN}}(E, G, Z, D) = \mathbb{E}_{x \sim p(x)}[\log D(x) + \log(1 - D(\hat{x})�)]. \quad (3.7)$$

El modelo se entrena conjuntamente para minimizar:

$$Q^* = \arg \min_{E, G, Z} \max_D \mathbb{E}_{x \sim p(x)}[L_{\text{VQ}} + \lambda L_{\text{GAN}}], \quad (3.8)$$

donde λ es un peso adaptativo que equilibra ambas pérdidas.

Modelado de la Composición de Imágenes con Transformers Una vez aprendida la representación discreta de las imágenes, el siguiente paso es modelar su distribución con un *Transformer*. Para ello, cada imagen es representada como una secuencia de índices en el diccionario de códigos:

$$s = \{s_{ij}\}, \quad s_{ij} = k \quad \text{si} \quad (z_q)_{ij} = z_k. \quad (3.9)$$

El *Transformer* aprende a predecir cada índice s_i de manera auto-regresiva:

$$p(s) = \prod_i p(s_i | s_{<i}), \quad (3.10)$$

maximizando la probabilidad logarítmica:

$$L_{\text{Transformer}} = \mathbb{E}_{x \sim p(x)}[-\log p(s)]. \quad (3.11)$$

Generación de Imágenes Condicional Para controlar la generación de imágenes, se introduce un código condicional c proveniente de otra imagen (como un boceto de vestido). En este caso, el modelo aprende:

$$p(s|c) = \prod_i p(s_i|s_{<i}, c). \quad (3.12)$$

En la práctica, se aprende una representación discreta r del condicionamiento con otro VQGAN, lo que permite concatenar la información de la condición con la de la imagen generada.

Síntesis de Imágenes de Alta Resolución Dado que la complejidad de los *Transformers* crece cuadráticamente con el tamaño de la secuencia, VQGAN aplica una estrategia de *downsampling* para reducir la dimensionalidad. Si la imagen original tiene tamaño $H \times W$, después de m bloques de reducción, la representación latente es de:

$$h = \frac{H}{2^m}, \quad w = \frac{W}{2^m}. \quad (3.13)$$

Para generar imágenes de resolución megapíxel, el *Transformer* opera de manera *patch-wise*, generando regiones de la imagen en ventanas deslizantes y asegurando la coherencia espacial mediante el contexto aprendido.

Experiencia El uso de *Transformers* junto con VQGAN ha permitido modelar la estructura global y los detalles locales de las imágenes de vestidos, logrando una síntesis de alta calidad en un marco condicional de generación de imágenes. Pero esto conlleva a un alto consumo de recursos computacionales. También hay que apuntar a las dificultades para entrenar este modelo sobre todo, ya que, aunque habían varios preprocesamientos para las distintas tareas como *depth-to-image* o *segmentation-to-image*, de ninguno pudo aprender el modelo condicionado directo. Para lograr que aprendiera el modelo, primero tuvimos que crear un preprocesamiento simple de imágenes rgb tanto para los vestidos como dibujos. Para luego, entrenar cada etapa por separado, cada una con sus funciones de perdidas con discriminadores GAN, osea que aprenda las características del objetivo y condicional para luego poder aprender a transformar los condicionales a imágenes objetivos. Una vez aprendido por separado, ahí sí se logró que el modelo pueda aprender a generar imágenes a partir de los dibujos, esta ahora, tiene una función loss de *crossentropy* para el entrenamiento

condicionado.

$$\mathcal{L} = \text{CrossEntropy}(\text{logits}, \text{target}) \quad (3.14)$$

En esta la generación de imágenes condicionadas, se dan dos técnicas dentro de este modelo:

- **x_sample_nopix**: Se genera la imagen con aleatoriedad, usando temperatura y top-k para controlar la diversidad. Cada vez que ejecutes el código, la imagen generada podría ser ligeramente diferente.
- **x_sample_det**: Se genera la imagen de forma totalmente determinista, siempre eligiendo el índice más probable en cada paso. Si ejecutas el código varias veces con los mismos datos de entrada, la imagen generada será idéntica en cada ejecución.

3.2.3. Arquitectura de Difusión

Los modelos de difusión han emergido como una de las arquitecturas más avanzadas. Su fundamento teórico se basa en la aproximación de una distribución de datos compleja mediante un proceso de difusión estocástico y su posterior reversión a través de un modelo entrenado.

Un modelo de difusión consiste en dos procesos clave:

- a) **Proceso de Difusión (Forward Process)**: Se aplica ruido gaussiano iterativamente a una imagen, transformándola en una distribución cercana a la normal estándar.
- b) **Proceso de Generación (Reverse Process)**: Se entrena un modelo para revertir este proceso, eliminando el ruido paso a paso y reconstruyendo la imagen original a partir de una muestra de ruido.

Matemáticamente, el proceso de difusión se define como:

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1 - \beta_t}x_{t-1}, \beta_t I), \quad (3.15)$$

donde β_t es un parámetro de ruido creciente en cada paso t . Para revertir esta transformación, se aprende un modelo de predicción de ruido ϵ_θ :

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)). \quad (3.16)$$

La optimización del modelo se realiza minimizando la pérdida de predicción de ruido:

$$L_{\text{diff}} = \mathbb{E}_{x,\epsilon,t} [||\epsilon - \epsilon_\theta(x_t, t)||^2]. \quad (3.17)$$

ControlNet1 con Stable Diffusion 1.5

El modelo **ControlNet** extiende la arquitectura de **Stable Diffusion 1.5**, permitiendo controlar la estructura y composición de la imagen generada, usando entradas como bordes, mapas de profundidad o poses humanas (Zhang, Rao & Agrawala, 2023)."

Arquitectura ControlNet utiliza un enfoque de red bloqueada con copia" donde el modelo original de difusión permanece congelado mientras se entrena una red adicional que procesa las condiciones. La arquitectura consta de:

- **Red base bloqueada:** Un modelo de difusión preentrenado cuyos pesos permanecen inalterados. En este caso es Stable Diffusion 1.5 que está basado en DDPM(Denoising Diffusion Probabilistic Model), que consta de los siguientes componentes:
 - **VAE (Variational Auto Encoder):** Comprime las imágenes a una representación latente de menor dimensionalidad al inicio del proceso y posteriormente descomprime esta representación para generar la imagen final.
 - **UNet condicional:** Es el núcleo del modelo que realiza la predicción de ruido paso a paso. Contiene bloques de atención cruzada que permiten incorporar información del prompt textual en el proceso de generación.
 - **Codificador de texto CLIP:** Convierte las descripciones textuales en embeddings que guían la generación de la imagen mediante los mecanismos de atención cruzada del UNet.
 - **Scheduler de difusión:** Controla la secuencia de pasos de ruido y la velocidad a la que se elimina el ruido durante la generación.
- **Red copiada entrenable:** Una copia de la red base que se puede entrenar y que recibe las señales de condición.
- **Módulos de adaptación de características:** Capas de convolución 1×1 que conectan la red copiada con la red base, permitiendo que la información condicionada se integre en el proceso generativo.

- **Red de codificación:** Procesa las condiciones de entrada (como bordes, poses o mapas de profundidad) antes de alimentarlas a la red copiada.

Función de pérdida La función de pérdida de ControlNet es similar a la de los modelos de difusión estándar, siguiendo principalmente:

- **Pérdida de predicción de ruido:** Mide la diferencia entre el ruido real agregado durante el proceso de difusión y el ruido predicho por el modelo durante la inversión.
- **Pérdida de reconstrucción condicional:** Evalúa qué tan bien el modelo puede reconstruir la imagen original dada la imagen ruidosa y la condición espacial.

La función de pérdida se puede expresar como:

$$\mathcal{L} = \mathbb{E}_{x_0, c, \epsilon, t} [||\epsilon - \epsilon_\theta(x_t, t, c)||^2] \quad (3.18)$$

donde x_0 es la imagen original, c es la condición, ϵ es el ruido aleatorio agregado, t es el paso de tiempo, x_t es la imagen ruidosa en el tiempo t , y ϵ_θ es la red que predice el ruido.

Experiencia Este modelo no fue entrenado, solo probamos con los pesos que ya venían en el Stabl Difussion, hay que destacar que hay varios pesos de los Stable-Diffusion 1.5 dependiendo de las tareas y lo que se quiere. Dicho esto ,se eligió el *scribble2image*(de boceto a imagen). Dando imágenes de todo tipo, algunas buenas, otra incoherentes, siempre con el mismo prompt para no añadir variabilidad.

BBDM (Brownian Bridge Diffusion Model)

El modelo **BBDM** introduce un enfoque novedoso para tareas de traducción de imagen a imagen basado en la teoría de puentes brownianos, permitiendo una transición más directa entre la imagen de origen y la imagen objetivo (Li, Xue, Liu & Lai, 2023).

Arquitectura A diferencia de los modelos de difusión estándar que parten de ruido puro, BBDM utiliza un proceso que vincula dos puntos específicos, creando un "puente" entre la imagen de origen y la imagen objetivo. La arquitectura incluye:

- **Red UNet condicional:** Similar a otros modelos de difusión, pero adaptada para incorporar tanto la imagen de origen como información temporal:
 - **Encoder-Decoder:** Estructura que reduce progresivamente la dimensionalidad espacial para capturar características a múltiples escalas y luego reconstruye la resolución original.
 - **Bloques residuales:** Permiten un mejor flujo de gradientes durante el entrenamiento y preservan información a través de la red.
 - **Mecanismos de atención:** Facilitan que el modelo establezca conexiones entre diferentes regiones de la imagen de origen y la imagen en proceso de generación.
- **Codificador de condición:** Procesa la imagen de origen y cualquier información adicional que condiciona la generación.
- **Módulo de embebido temporal:** Incorpora información sobre el paso temporal actual en el proceso de difusión, crítico para modelar la dinámica del puente browniano.

Puente Browniano El concepto fundamental de BBDM es modelar la transición entre imágenes como un puente browniano, que tiene propiedades matemáticas específicas:

- **Condicionamiento en extremos:** A diferencia de procesos estándar de difusión, un puente browniano está condicionado tanto al punto inicial (imagen de origen) como al punto final (imagen objetivo).
- **Proceso estocástico:** La transición sigue una trayectoria probabilística bien definida que, en promedio, conecta directamente ambos puntos.
- **Derivación sin nueva información:** Toda la dinámica del proceso se puede derivar conociendo los puntos extremos, lo que permite una generación más eficiente.

Función de pérdida La función de pérdida de BBDM se basa en la estimación de la dinámica del puente browniano:

- **Pérdida de Score Matching:** Optimiza el modelo para predecir la dirección y magnitud del gradiente del logaritmo de la densidad condicional.

- **Pérdida de Consistencia Temporal:** Asegura que las predicciones del modelo sean coherentes a lo largo de diferentes pasos de tiempo.

La función de pérdida principal se puede expresar como:

$$\mathcal{L} = \mathbb{E}_{x_0, y, t} [\| s_\theta(x_t, y, t) - \nabla_{x_t} \log p(x_t | x_0, y) \|^2] \quad (3.19)$$

donde x_0 es la imagen objetivo, y es la imagen de origen, t es el paso de tiempo, x_t es la imagen en proceso de difusión en el tiempo t , s_θ es la red que predice el score, y $\nabla_{x_t} \log p(x_t | x_0, y)$ es el verdadero score del puente browniano.

Experiencia Este modelo se entreno sin mayor complicaciones, lo único un poco mas complicado fue la separación del dataset ya que pedía separar las imágenes condicionadas de las reales en carpetas diferentes mientras que en nuestro caso se tenían concatenadas. Fuera de eso, dio buenos resultados con pocos epochs. También requería de una buena cantidad de poder computacional, cabe destacar que este modelo fue entrenado en un servidor más viejo que con los que normalmente se ocuparon.

Palette con DDIM (Denoising Diffusion Implicit Models)

El modelo **Palette** implementa una variante de los modelos de difusión conocida como DDIM (Denoising Diffusion Implicit Models), proporcionando un enfoque más eficiente y controlable para la generación de imágenes condicionadas por diferentes tipos de señales (Saharia, Chan, Chang, Lee, Ho, Salimans, Fleet & Norouzi, 2022).

Arquitectura Palette extiende el marco de los modelos de difusión a una amplia gama de tareas de traducción de imagen a imagen, incluyendo la conversión de bocetos a imágenes realistas. Su arquitectura comprende:

- **Modelo DDIM:** A diferencia de los modelos de difusión probabilística estándar (DDPM), DDIM define un proceso de muestreo implícito determinista que permite:
 - **Muestreo acelerado:** Reducir significativamente el número de pasos de inferencia necesarios para generar imágenes de alta calidad.
 - **Trayectorias deterministas:** Establecer rutas consistentes entre el espacio de ruido y el espacio de imagen, facilitando el control preciso sobre el proceso generativo.

- **Red UNet condicional:** Similar a Stable Diffusion, pero adaptada para las tareas específicas de Palette:
 - **Canales condicionales:** Incorpora directamente los bocetos u otras condiciones en canales de entrada adicionales.
 - **Bloques residuales:** Mantiene la estructura típica de UNet con conexiones residuales para preservar información a múltiples escalas.
 - **Bloques de atención:** Facilita la correlación entre partes distantes del boceto y la imagen generada.
- **Embebido de condición:** Procesa las condiciones de entrada (bocetos) para integrarlas efectivamente en la predicción de ruido.
- **Codificación temporal:** Incorpora la información sobre el paso temporal actual mediante embeddings sinusoidales.

Proceso generativo: DDIM, implementado en Palette, permite el muestreo con muchos menos pasos (típicamente 50-100) que los DDPM originales (1000+), sin sacrificar significativamente la calidad de imagen.

Proceso DDIM El proceso DDIM reformula la etapa de muestreo de los modelos de difusión:

- **Proceso no markoviano:** A diferencia del proceso markoviano de los DDPM, donde cada paso depende únicamente del anterior, DDIM establece dependencias entre estados más distantes.
- **Muestreo acelerado:** La ecuación de muestreo de DDIM permite saltar múltiples pasos:

$$x_{t-1} = \sqrt{\alpha_{t-1}} \left(\frac{x_t - \sqrt{1 - \alpha_t} \epsilon_\theta(x_t, t)}{\sqrt{\alpha_t}} \right) + \sqrt{1 - \alpha_{t-1}} \cdot \epsilon_\theta(x_t, t) \quad (3.20)$$

donde α_t son los parámetros de escala acumulativos, ϵ_θ es la red que predice el ruido y x_t es la imagen con ruido en el tiempo t .

Funciones de pérdida Palette evalúa dos enfoques alternativos para la función de pérdida, comparando sus efectos en la calidad y características de las imágenes generadas:

- **Pérdida L1 (MAE)**: Utiliza la norma L1 para medir la diferencia entre valores predichos y objetivo:

$$\mathcal{L}_1 = \mathbb{E}_{x_0, \epsilon, c, t} [||\epsilon - \epsilon_\theta(x_t, c, t)||_1] \quad (3.21)$$

donde ϵ es el ruido real agregado durante el proceso de difusión y ϵ_θ es el ruido predicho por la red.

- **Pérdida L2 (MSE)**: Utiliza la norma L2 (error cuadrático) para la misma medición:

$$\mathcal{L}_2 = \mathbb{E}_{x_0, \epsilon, c, t} [||\epsilon - \epsilon_\theta(x_t, c, t)||^2] \quad (3.22)$$

Es importante destacar que Palette no combina ambas pérdidas simultáneamente, sino que realiza experimentos comparativos utilizando una u otra. Los autores observan que, aunque ambas funciones de pérdida producen puntuaciones similares en métricas de calidad de imagen, generan comportamientos distintos en el modelo:

- Los modelos entrenados con pérdida **L2** tienden a producir mayor diversidad en las muestras generadas.
- Los modelos entrenados con pérdida **L1** generan resultados más conservadores y consistentes con la condición de entrada.

Esta diferencia resulta particularmente relevante para la traducción de bocetos a imágenes realistas, ya que permite seleccionar la función de pérdida según se priorice la fidelidad estructural al boceto original (L1) o la diversidad creativa en las interpretaciones (L2).

Experiencia En este modelo si se tuvo más complicaciones, en sentido de preprocesamiento y poder computacional. Por lo que se hicieron pruebas cambiando las dimensiones de entrada de 64x64 a 96x96, no se pudo con mayores dimensiones debido a sus requerimientos de computo y el colapso de memoria que generaba. lo mismo ocurría con los batches, se tenía que tener en un número específicos como 258 ya

que con 256, 257 y 259 se caía el entrenamiento. Aunque se apreciaron mejoras en la generación de imágenes sobre todo en el ámbito de testeo, también hubieron perdidas como una peor generación de imágenes en la validación y mayor tiempo de entrenamiento.

3.2.4. Configuración de los entrenamientos

Para entrenar los modelos, se utilizaron distintos servidores con características de hardware variables. Inicialmente, el entrenamiento comenzó en el servidor **Turing**, cuyas especificaciones son:

- **CPU:** Intel Core i7-7820X @ 3.60GHz
- **RAM:** 62GB
- **GPU:** 2x NVIDIA TITAN X (12GB cada una)

Sin embargo, este servidor presentó constantes caídas, lo que dificultó la estabilidad del entrenamiento. Debido a estos problemas, se migró el proceso a dos servidores más robustos, **Optimus** y **Bumblebee**, que comparten la siguiente configuración:

- **CPU:** AMD EPYC 9554P 64-Core Processor
- **RAM:** 251GB
- **GPU:** 3x NVIDIA RTX 6000 Ada Generation (49GB cada una)

El cambio de infraestructura resultó en una notable mejora en los tiempos de entrenamiento. Además, a diferencia de **Turing**, estos servidores no presentaron caídas durante el proceso, asegurando estabilidad hasta la finalización del entrenamiento. Al final del proyecto, ya se contaba con respaldo de los modelos generados.

Métricas de Evaluación

Para evaluar la calidad de las imágenes generadas, se utilizaron dos métricas ampliamente reconocidas en el ámbito de modelos generativos:

- **Fréchet Inception Distance (FID):** Mide la distancia entre la distribución de características de las imágenes reales y generadas. Valores más bajos indican mejor calidad y mayor similitud con los datos reales.

- **Learned Perceptual Image Patch Similarity (LPIPS)** (Zhang, Isola, Efros, Shechtman & Wang, 2018): Calcula la similitud perceptual entre imágenes utilizando redes preentrenadas. Valores más bajos reflejan mayor similitud visual desde el punto de vista humano.
- **Structural Similarity Index Measure(SSIM)** Nilsson & Akenine-Möller (2020): SSIM mide la similitud estructural entre dos imágenes, evaluando diferencias en iluminación, contraste y estructura. Es una métrica perceptual que intenta imitar cómo los humanos perciben la calidad de una imagen, valores más altos indican una mayor similitud.

Ambas métricas permiten evaluar tanto la calidad visual como la diversidad de las imágenes generadas, proporcionando una comparación objetiva entre los modelos entrenados. Pero también se considera la calificación humana, y aunque tal vez el modelo no recrea imágenes iguales, si no más diversas, estas también pueden tener una buen nivel de detalle y calidad.

Capítulo 4

Resultados

4.1. Validación del set de datos

Antes de adentrarnos en los resultados obtenidos por la métricas y analizar las fotos, evaluemos primero el dataset generado. Como dicho anteriormente, las imágenes de los vestidos fueron obtenidos gracias a un proceso de *Webscraping* junto a detección de objetos y segmentación, como se pueden ver la figura A.1 Con las imágenes objetivos lista falta su contraparte, los bocetos para que los modelos aprendan. Con la combinación de técnicas mencionadas en la sección 3.1.3.

Podemos ver cómo InstrucPix2Pix 4.2 tiene un estilo un poco mas abstracto manipulando un poco las características del vestido como tal además de la deformación entregada por Liquify. Los bocetos como tal siguen una estructura clara, fondo negro y el vestido en blanco y negro. Esto se mantiene para todos los sketches, manteniendo el ruido extra al mínimo para el momento del entrenamiento. Aunque se ocuparon *prompts* negativos para eliminar los ”modelos.” partes humanas, igual se pueden apreciar figuras humanoides dentro del dibujo y aunque no es ideal para el entrenamiento, si le da cierta robustez y diversidad a los modelos.

Continuando con las generación de dibujos para los vestidos, sigue el modelo PiDi-Net, enfocado en encontrar los contornos de las imágenes, dándonos lo más parecido a un boceto dibujado como se aprecian en la figura 4.3.

Ahora, tenemos la mezcla de ambas técnicas. Como se ve en la figura 4.4. Este set de datos en especial se utilizó para la validación de los modelos, siendo un punto intermedio entre los estilos de estos. Con su mezcla se logra evaluar ambas formas de generación dentro de su marge. Con esto dicho, las primeras dos técnicas se utilizaron especialmente para el entrenamiento.

Ya para terminar, el set de datos para testeo, el cual fue realizado a mano por

varias persona mediante el software paint, tiene mucha variación de estilos y distintos detalles lo cual lleva al límite lo aprendido por los modelos. Estos los podemos ver en la siguiente figura 4.5.

Con los bosquejos realizados, falta concatenar(dependiendo del modelo) o entrelazar los bosquejos con los vestidos dependiendo de cómo lo pida el modelo. Pero para la realización de la gran mayoría de entrenamientos, se utilizaron el dataset concatenando, así solo se tiene que llamar a la imagen una vez y luego hacer un prerocesamiento adecuado como *crop* o división de la imágenes. Se pueden ver en la figura 4.6

4.2. Modelos Generativos y métricas

Una vez entrenado los modelos, generamos las imágenes, todas las del set de testeo y unas cuantas del set de validación, estas van entre 10 y 20 imagenes. Para la obtención de métricas se concatenaron varias imágenes generadas del mismo modelo, al igual se juntaron las imágenes de los vestidos reales correspondientes de las generadas. Esto se realizó para obtener resultados más fácilmente. Dicho esto, para la obtención de métricas de testeo, ya que no se tiene imagenes reales de estas, se ocupan los bocetos a forma de comparar qué tan fieles son a los condicionales, esto podría influir en los valores obtenidos, pero se analizaron de igual manera teniendo en cuenta esto. Se analizaron las métricas en la tabla 4.1 junto con las imágenes generadas.

Modelo	FID		LPIPS		SSIM	
	Validación	Test	Validación	Test	Validación	Test
Pix2Pix	117.59	329.02	-	0.81	-	0.03286
CycleNet (ResNet)	308.30	374.80	0.18	0.71	0.87116	0.08224
CycleNet (UNet)	372.82	336.45	0.73	0.76	-0.03789	0.00023
VQGan (Det)	123.36	415.16	0.16	0.81	0.85073	0.00736
VQGan (NoPix)	139.81	416.37	0.11	0.86	0.87537	0.00788
BBDM	130.79	333.34	0.26	0.63	0.74275	0.00653
DDIM (96x96)	316.31	478.37	0.30	0.81	0.75454	0.07437
DDIM (64x64)	391.92	474.55	0.34	0.89	0.80537	0.06445
ControlNet	441.51	386.95	0.86	0.77	0.09313	0.22307

Cuadro 4.1: Tabla comparativa de FID, LPIPS y SSIM por modelo en diferentes datasets.

4.2.1. Pix2Pix

El modelo Pix2Pix, una implementación clásica de traducción condicional de imagen a imagen basada en GANs, mostró resultados notables en el conjunto de validación con un FID de 117.59, posicionándose como uno de los modelos con mejor rendimiento en términos de fidelidad visual. Sin embargo, su desempeño se degradó significativamente en el conjunto de prueba con dibujos hechos a mano, alcanzando un FID de 329.02. En términos de diversidad perceptual (LPIPS), Pix2Pix obtuvo un valor de 0.81 en el conjunto de prueba, indicando una alta diversidad en las imágenes generadas en comparación con las imágenes de referencia. El bajo valor de SSIM (0.03286) confirma esta tendencia, sugiriendo una limitada preservación estructural. Hay que destacar que no se obtiene resultados en el conjunto de validación dado que se cayo un servidor y con eso se perdieron cierto orden de datos, la nomenclatura de estas por lo que volver a generarlas con el mismo nombre fue my complicado, siendo la única gran perdida por factores externos. Estos resultados indican que Pix2Pix, aunque efectivo con datos similares a los de entrenamiento,cuando se cambia de estilos significativamente este pierde la coherencia de crear imágenes más realistas, como los dibujados a mano. Se pueden ver su generación en el entorno de validación en esta figura 4.7 .

4.2.2. CycleGan

Se evaluaron dos generadores variantes de CycleGAN:

CycleGAN (ResNet)

Utilizando ResNet como arquitectura base, este modelo mostró un desempeño modesto con un FID de 308.30 en validación y 374.80 en prueba. Sin embargo, demostró una buena preservación estructural en el conjunto de validación con un SSIM de 0.87116, aunque este valor cayó drásticamente a 0.08224 en el conjunto de prueba. Su LPIPS relativamente bajo en validación (0.18) y alto en prueba (0.71) sugiere una tendencia a generar imágenes conservadoras en datos similares al entrenamiento, pero menos predecibles con bocetos manuales. 4.9

CycleGAN (UNet)

La variante con UNet mostró un rendimiento inferior en términos de FID (372.82 en validación y 336.45 en prueba), con un comportamiento inusual en SSIM, pre-

sentando valores negativos en validación (-0.03789) y prácticamente nulos en prueba (0.00023). Esto sugiere problemas significativos en la preservación de la estructura original del boceto. Su alto valor de LPIPS en ambos conjuntos (0.73 y 0.76) indica una alta diversidad pero potencialmente desconectada de las condiciones de entrada.

4.8

4.2.3. Taming Transformers (VQGan)

El modelo VQGan se evaluó en dos configuraciones:

VQGan (Det)

Esta variante mostró un buen rendimiento en validación con un FID de 123.36, aunque su desempeño se deterioró considerablemente en prueba (415.16). Presenta valores de SSIM altos en validación (0.85073) pero extremadamente bajos en prueba (0.00736), indicando una fuerte dependencia del estilo de los datos de entrenamiento.

VQGan (NoPix)

Con resultados similares a la variante Det, obtuvo un FID de 139.81 en validación y 416.37 en prueba. Su comportamiento en términos de SSIM sigue el mismo patrón (0.87537 en validación y 0.00788 en prueba). Es importante destacar que esta variante mostró el mayor valor de LPIPS en prueba (0.86), sugiriendo la mayor diversidad perceptual entre todos los modelos evaluados.

Un aspecto destacable de los modelos VQGan fue la importancia crítica del pre-entrenamiento. Los experimentos mostraron que sin preentrenamiento, estos modelos eran incapaces de producir resultados coherentes en la tarea condicionada^{4.13}. Este hallazgo subraya la importancia de la inicialización adecuada para modelos generativos complejos basados en transformers. En las siguientes figuras se pueden ver el desempeño del modelo y sus técnicas 4.10, 4.12, 4.11

4.2.4. BBDM (Brownian Bridge Diffusion Model)

BBDM, basado en la teoría de puentes brownianos, mostró un rendimiento intermedio con un FID de 130.79 en validación y 333.34 en prueba. Su SSIM de 0.74275 en validación indica una buena preservación estructural, aunque este valor cae drásticamente a 0.00653 en prueba. Con un LPIPS de 0.26 en validación y 0.63 en prueba,

BBDM parece ofrecer un equilibrio entre fidelidad y diversidad, particularmente en datos similares a los de entrenamiento.^{4.14,4.15}

4.2.5. DDIM (Palette)

El modelo DDIM implementado en Palette se evaluó en dos resoluciones como se puede ver en la figura 4.16:

DDIM (96x96)

Con un FID de 316.31 en validación y 478.37 en prueba, esta configuración mostró dificultades para producir imágenes de alta fidelidad. Sin embargo, mantuvo un SSIM respetable de 0.75454 en validación, aunque descendió a 0.07437 en prueba. Su LPIPS de 0.3 en validación y 0.81 en prueba sugiere una capacidad creciente para generar diversidad a medida que se aleja de los patrones de entrenamiento.

DDIM (64x64)

La versión de menor resolución obtuvo resultados mixtos con un FID de 391.92 en validación y 474.55 en prueba. Curiosamente, presentó el mejor SSIM de validación entre los modelos de difusión (0.80537) y un alto LPIPS en prueba (0.89), el más alto entre todos los modelos. Esto sugiere que, aunque con menor fidelidad visual general, esta configuración preserva mejor ciertas características estructurales y ofrece mayor diversidad perceptual. La comparación entre ambas resoluciones de DDIM revela un compromiso interesante: la configuración de menor resolución (64x64) parece preservar mejor la estructura global y ofrecer mayor diversidad, mientras que la resolución más alta (96x96) proporciona algunos detalles adicionales a costa de estabilidad estructural.

4.2.6. ControlNet con Stable Diffusion 1.5

ControlNet, a diferencia de los demás modelos, no fue entrenado específicamente para este dataset, sino que se utilizaron pesos preentrenados para la transformación de "scribble to image". Este enfoque resultó en el FID más alto en validación (441.51) y un valor elevado en prueba (386.95). Su LPIPS fue el más alto en validación (0.86) y alto en prueba (0.77), indicando una consistente generación de alta diversidad. Sin embargo, lo más distintivo fue su SSIM en prueba (0.22307), significativamente superior

al resto de modelos, sugiriendo una mejor transferencia de características estructurales a bocetos hechos a mano. Las imágenes generadas por ControlNet mostraron una naturaleza dual: algunas resultaron realistas mientras otras presentaban cualidades abstractas, reflejando la tendencia del modelo a explorar el espacio de soluciones con mayor libertad creativa como se aprecia en la figura 4.17.

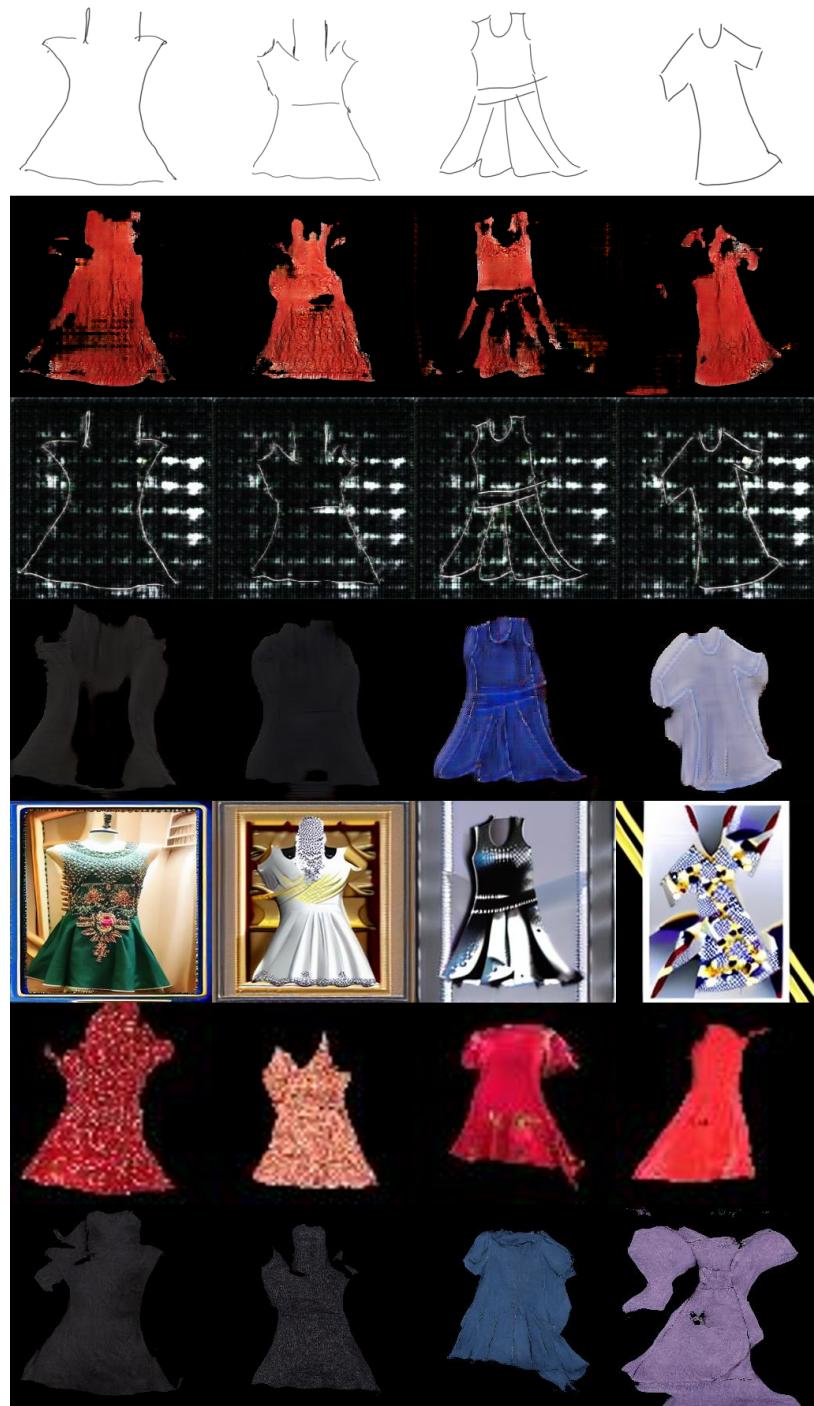


Figura 4.1: En la primera fila se tiene los dibujos de testo, lo siguen de Pix2Pix, después ambas CycleGan(Pix y ResNet respectivamente), más abajo está ControlNaet y después ambos Pallet(64x64 y 96x96 respectivamente)



Figura 4.2: Generación de bocetos mediante IP2P



Figura 4.3: Generación de bocetos mediante PiDiNet

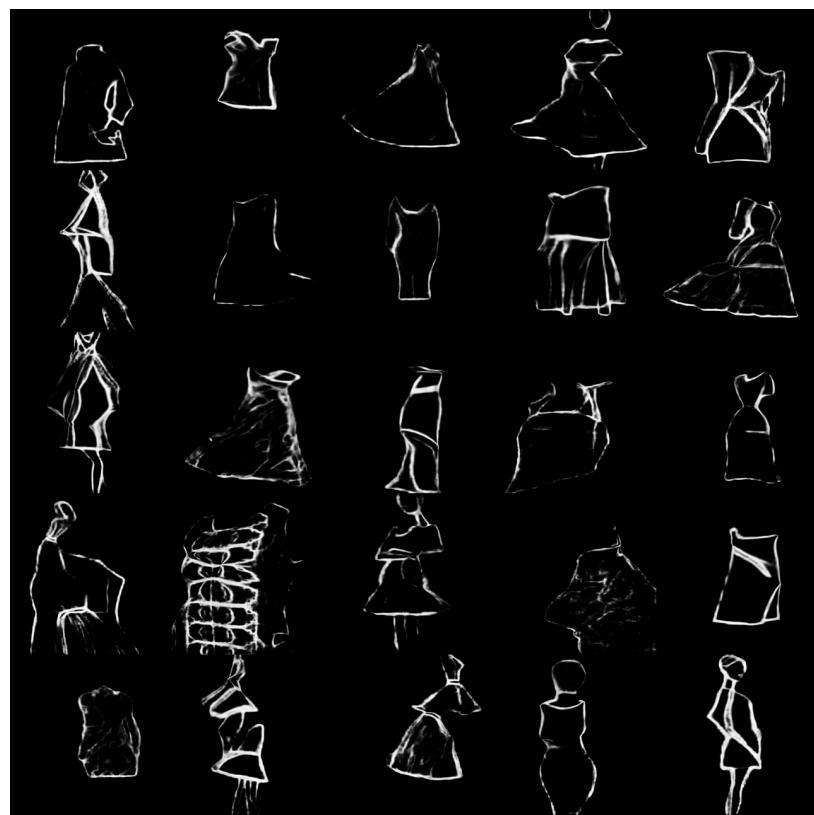


Figura 4.4: Generación de bocetos mediante combinación de ambas



Figura 4.5: Bocetos hechos a mano



Figura 4.6: Datos del entrenamiento y validación

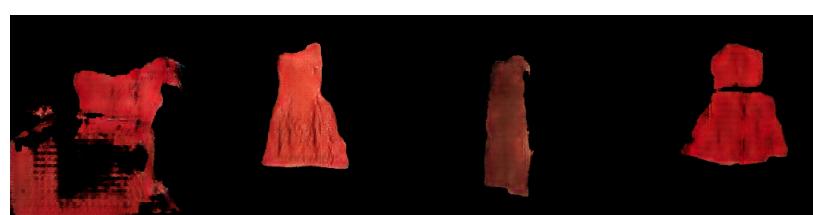


Figura 4.7: Imágenes generadas por Pix2Pix del data set de validación



Figura 4.8: Imágenes generadas por CyclePix del data set de validación

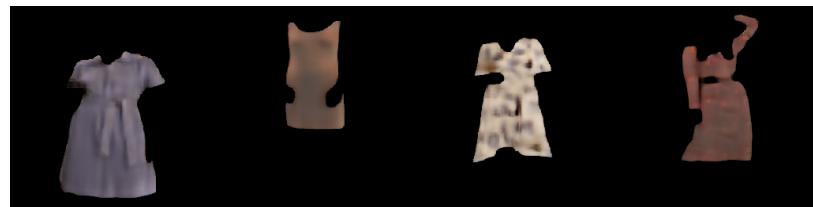


Figura 4.9: Imágenes generadas por CycleResnet del data set de validación

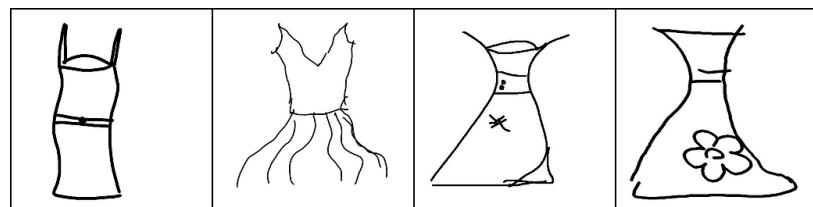


Figura 4.10: Imágenes de testeo para VQGAN



Figura 4.11: Imágenes generadas por VQGAN Nopixa en testeo



Figura 4.12: Imágenes generadas por VQGAN Det en testeo

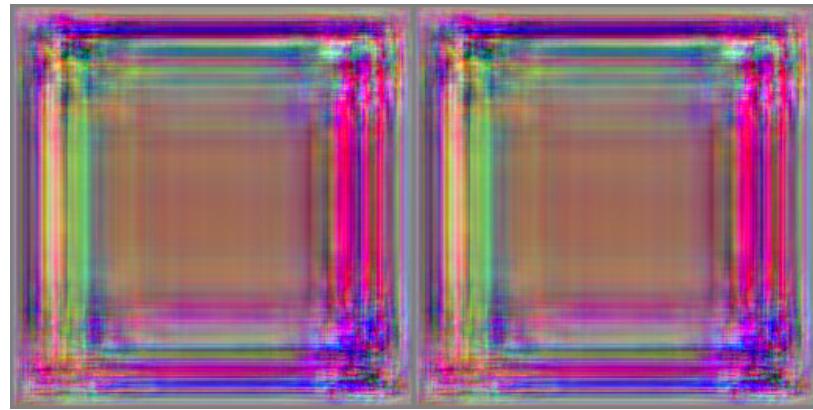


Figura 4.13: Imágenes generadas por VQGAN sin preentrenamiento despues de 30 epochs

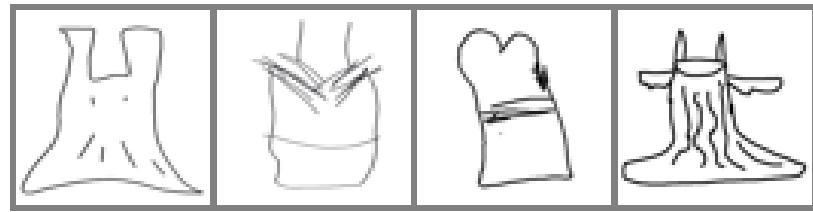


Figura 4.14: Imágenes de input a modo testeo en BBDM



Figura 4.15: Imágenes generadas por BBDM en los datos detesteo



Figura 4.16: En la primera fila se encuentra los vestidos reales, abajo de estos los sketches y las dos filas de más abajo corresponden a las imágenes generadas por Pallet 64x64 y 96x96 respectivamente



Figura 4.17: Imágenes generadas por ControlNet en testeo

Capítulo 5

Conclusiones del Análisis

Los resultados obtenidos destacan tanto las fortalezas como las limitaciones de las distintas arquitecturas generativas para la traducción de bocetos a imágenes realistas. Estos hallazgos tienen implicaciones clave para futuras mejoras y aplicaciones en el diseño de moda asistido por IA. Algunos de estos *insights* son:

- a) **Generalización a bocetos manuales:** Todos los modelos presentan dificultades significativas para generalizar a bocetos hechos a mano, como lo reflejan las métricas degradadas en el conjunto de prueba.
- b) **Compromiso fidelidad-diversidad:** Se observa un patrón donde los modelos que producen imágenes con mayor fidelidad visual (FID bajo) tienden a ofrecer menor diversidad perceptual (LPIPS bajo) y viceversa, ilustrando el clásico compromiso entre calidad y diversidad en modelos generativos.
- c) **Preservación estructural:** La preservación de la estructura del boceto original representa el mayor desafío, con ControlNet destacándose sorprendentemente en esta área a pesar de no haber sido entrenado específicamente para el dataset.
- d) **Impacto del preentrenamiento:** Los experimentos con Taming Transformers resaltaron la importancia del preentrenamiento para obtener resultados coherentes en tareas de generación condicionada.

Alcance y limitaciones

Se lograron entrenar múltiples modelos con variantes específicas, como el uso de diferentes generadores manteniendo la misma estructura base. Sin embargo, algunas limitaciones afectaron la capacidad de explorar otros modelos de interés. Estas limitaciones incluyeron la necesidad de manejar entradas multimodales complejas (más allá

de imágenes y texto), la preparación especializada o estructura del dataset, y los altos requerimientos de memoria. Además, ciertos modelos no estaban completamente disponibles en sus repositorios, lo que imposibilitó la transferencia de preentrenamiento a nuestro dominio. Por último, algunos modelos requerían capacidades computacionales fuera del alcance de los recursos disponibles.

Bibliografía

- Dhariwal, P. & Nichol, A. Q. (2021). Diffusion models beat gans on image synthesis.
- Esser, P., Rombach, R., & Ommer, B. (2021). Taming transformers for high-resolution image synthesis.
- Goodfellow, I. et al. (2014). Generative adversarial networks.
- Ho, J., Jain, A., & Abbeel, P. (2020). Denoising diffusion probabilistic models.
- Isola, P., Zhu, J.-Y., Zhou, T., & Efros, A. A. (2018). Image-to-image translation with conditional adversarial networks.
- Karras, T., Laine, S., & Aila, T. (2019). A style-based generator architecture for generative adversarial networks.
- Li, B., Xue, K., Liu, B., & Lai, Y.-K. (2023). Bbdm: Image-to-image translation with brownian bridge diffusion models.
- Liu, S. et al. (2023). Compensation sampling for improved convergence in diffusion models.
- Nilsson, J. & Akenine-Möller, T. (2020). Understanding ssim.
- Saharia, C., Chan, W., Chang, H., Lee, C. A., Ho, J., Salimans, T., Fleet, D. J., & Norouzi, M. (2022). Palette: Image-to-image diffusion models.
- Takida, Y., Imaizumi, M., Shibuya, T., Lai, C.-H., Uesaka, T., Murata, N., & Mitsu-fuji, Y. (2023). San: Inducing metrizability of gan with discriminative normalized linear layer.
- Tang, Y., Sun, P., Cheng, Z., & Lin, T. (2025). Gmem: A modular approach for ultra-efficient generative models.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2023). Attention is all you need.
- Wikipedia (2024). Red generativa adversativa. [En línea; consultado el 24 de diciembre de 2024].
- Zhang, L., Rao, A., & Agrawala, M. (2023). Adding conditional control to text-to-image diffusion models.

Zhang, R., Isola, P., Efros, A. A., Shechtman, E., & Wang, O. (2018). The unreasonable effectiveness of deep features as a perceptual metric.

Zheng, B. & Yang, T. (2024). Diffusion models are innate one-step generators.

Zhu, J.-Y., Park, T., Isola, P., & Efros, A. A. (2020). Unpaired image-to-image translation using cycle-consistent adversarial networks.

Apéndice A

Tablas y ranking



Figura A.1: Collage de vestidos obtenidos online

Rank	Model	FID	NFE	Paper	Code	Result	Year	Tags
1	GMem	1.22		Generative Modeling with Explicit Memory			2024	
2	EDM+DDO	1.30		Direct Discriminative Optimization: Your Likelihood-Based Visual Generative Model is Secretly a GAN Discriminator			2025	
3	StyleSAN-XL	1.36		SAN: Inducing Metrizability of GAN with Discriminative Normalized Linear Layer			2023	
4	SiDA-EDM	1.396	1	Adversarial Score identity Distillation: Rapidly Surpassing the Teacher in One Step			2024	
5	PFGM++ +CS	1.50		Compensation Sampling for Improved Convergence in Diffusion Models			2023	
6	GDD-I	1.54	1	Diffusion Models Are Innate One-Step Generators			2024	

Figura A.2: Ranking FID en el dataset Cifar-10