Universidad
Rey Juan Carlos

DOCTORAL THESIS

Deep Learning for Photographic Enhancement

Author

Pedro Manuel Cuenca Jiménez

Supervisors

José María Cañas Plaza

Jesús Fernández Conde

Doctoral Program in Information and Communications Technologies

International Doctoral School

2025

"You don't take a photograph, you make it."

Ansel Adams

Abstract

This thesis presents original research on deep learning applied to photo editing. Our work focuses on several goals we use as guiding lights, and that distinguish it from rich, contemporaneous research in the same field.

We focus on the *democratization* of photo editing, not only because the methods don't require steep learning curves to be adopted by end users, but also because they can run on consumer hardware. We strive to make this requirement compatible with *arbitrary resolution* editing, so users don't have to sacrifice quality in exchange for speed or accessibility. Furthermore, we are respectful of the work of photographers as artists and professionals, and develop *lossless parametric methods* whenever possible: they act like traditional filters, rather than replacing captured pixels with generated ones.

Our first contribution is **FilterNet**, an automatic enhancement method that predicts the values of filters that should be applied to a photo in order to improve it, rather than predicting the edited result. Predicted filters can be applied to arbitrary resolution photos, are easily explainable because they map to photography concepts such as *exposure* or *color temperature*, and can be tweaked by the user at will. However, they act globally because the system is trained with whole images.

As an intermediate step to extend FilterNet to local edits, we explore how segmentation methods can be conditioned using text instructions, so users can intuitively indicate the subject they are interested in. Linking visual and textual representations is being extensively researched these days; in particular, generative methods such as diffusion models achieve impressive creative results in this area. Unlike these methods, we don't attempt to produce final versions of edited photos; instead, we demonstrate that they can be leveraged to predict segmentation masks. Our prototype, **CocoGold**, is our second contribution.

Combining FilterNet with CocoGold, we create **CoCoNet**, a pipeline that predicts filter parameters like FilterNet, but applies them locally to the segmentation masks created with CocoGold.

No research is ever complete. Our progress in the areas we set out to explore opens new directions for the future, which we are excited to pursue. We end this thesis recognizing the limitations of our prototypes and looking forward to the next steps that can get us closer to our vision.

Resumen

La edición fotográfica ha acompañado a la fotografía desde sus inicios. Hoy en día, millones de usuarios ajustan sus fotos digitalmente, pero las herramientas profesionales siguen exigiendo una curva de aprendizaje pronunciada y flujos de trabajo complejos. Esta situación ha motivado la búsqueda de métodos automáticos y accesibles que mejoren la calidad técnica de las fotografías sin comprometer la intención artística ni la fidelidad de la captura original.

Esta tesis versa sobre la aplicación del aprendizaje profundo a la mejora fotográfica siguiendo tres principios rectores: (i) independencia de resolución, para que los resultados puedan aplicarse a imágenes de cualquier tamaño sin pérdida de calidad; (ii) interpretabilidad y control, favoreciendo métodos paramétricos no destructivos cuyos resultados puedan ajustarse tras la predicción; y (iii) viabilidad práctica en dispositivos de consumo, haciendo especial hincapié en la ejecución local por razones de privacidad, latencia y sostenibilidad.

Sobre estos cimientos, la tesis presenta dos sistemas principales y una integración de ambos. En primer lugar, desarrollamos FilterNet, un sistema de mejora automática que no genera píxeles editados, sino que aprende a predecir valores de filtros fotográficos estándar (exposición, brillo, contraste, sombras, altas luces y temperatura de color). Este diseño hace que el proceso sea interpretable, editable por el usuario y aplicable a resolución arbitraria: el modelo infiere parámetros a partir de una miniatura y la aplicación los aplica sobre la fotografía original a máxima resolución.

En segundo lugar, exploramos la segmentación guiada por texto como paso clave hacia ediciones locales. Para ello proponemos CocoGold, que reutiliza un modelo de difusión entrenado con millones de pares $(texto, imagen)$, modificándolo para identificar y segmentar objetos a partir de descripciones sencillas (por ejemplo, “gato”, “coche” o “reloj”). A diferencia de los *pipelines* de detección y segmentación por etapas, CocoGold formula la tarea como generación condicionada: el modelo aprende a “copiar” la imagen de entrada resaltando en blanco las regiones que coinciden con el término expresado, de donde extraemos la máscara binaria con un posprocesado ligero.

Por último, combinamos ambos enfoques en una prueba de concepto llamada COCONET: las máscaras de CocoGold permiten aplicar los parámetros de FilterNet de forma localizada. De este modo demostramos que la edición técnica basada en filtros puede adaptarse a regiones especificadas con lenguaje,

manteniendo interpretabilidad e independencia de resolución en todos los componentes salvo la máscara de segmentación.

El objetivo final de nuestro trabajo es contribuir a democratizar la edición fotográfica con soluciones prácticas, explicables y eficientes: un sistema capaz de proponer ajustes técnicos expresados con el mismo vocabulario utilizado en la edición fotográfica, capaz también de seleccionar áreas de interés mediante instrucciones naturales, y de ejecutarse en dispositivos móviles sin exigir infraestructuras especializadas ni comprometer la privacidad del usuario.

La mejora automática de imágenes ha evolucionado desde técnicas algorítmicas clásicas (como los métodos de ecualización de histograma) hasta métodos de aprendizaje profundo capaces de aprender transformaciones a partir de datos. Las primeras propuestas basadas en redes convolucionales abordaron tareas específicas (corrección de exposición, por ejemplo) o estilos locales guiados por un proceso de segmentación previo. Aunque eficaces, estos métodos suelen generar la imagen final en píxeles, lo que complica la escalabilidad a altas resoluciones y produce resultados poco interpretables o difíciles de ajustar.

Una línea interesante ha sido la de *aprender ediciones* en lugar de producir píxeles finales. Este planteamiento refuerza la independencia de resolución y la interpretabilidad, al representar la salida como parámetros de ajustes fotográficos y no como mapas de píxeles. Sin embargo, la disponibilidad de conjuntos de datos con parejas de imágenes $(original, editada)$ es limitada y a veces restrictiva en términos de licencia. Para superar esa limitación han surgido aproximaciones generativas (GANs) y de aprendizaje por refuerzo que aprenden secuencias de ajustes a partir de colecciones de imágenes "buenas" y "malas", sin mapear una a una, lo que presenta retos de estabilidad en el entrenamiento.

El aprendizaje por transferencia se ha consolidado como una estrategia preferente en visión artificial: un extractor de características preentrenado (ResNet, por ejemplo) puede utilizarse como base para tareas de mejora. Junto a ello, las funciones de pérdida perceptuales basadas en redes neuronales (tales como VGG, entre otras) han demostrado mejor correlación con la calidad visual percibida que las distancias píxel a píxel. A nivel de ingeniería, la diferenciabilidad de las operaciones de procesado de imagen resulta crucial para integrar filtros en arquitecturas entrenables extremo a extremo.

En cuanto a los datos, la *degradación sintética* ha resultado ser una receta efectiva de supervisión automática: a partir de fotos de alta calidad se generan variantes empeoradas mediante combinaciones plausibles de filtros. El modelo aprende a revertir esa degradación sin necesidad de pares editados por expertos, ampliando considerablemente la cantidad de muestras disponibles.

En paralelo, los últimos años han visto el auge de los modelos basados en la arquitectura *transformer*, los modelos de difusión y los modelos multimodales que combinan visión y lenguaje (VLMs). Los métodos basados en *transformers* han mejorado la modelización de dependencias de largo alcance en restauración y super-resolución; los modelos de difusión, concebidos para generación de imágenes, se han adaptado con éxito a tareas de traducción imagen a imagen (colorización, rellenado de zonas, estimación de profundidad) y se benefician de avances drásticos en muestreo rápido (DDIM, familias DPM-Solver, destilación). En el terreno lenguaje-visión, el alineamiento entre texto e imagen ha habilitado segmentación de vocabulario abierto y basada en expresiones identificativas, con pipelines modulares que combinan detectores con arquitecturas fundacionales de segmentación (como SAM). Al mismo tiempo, algunos VLMs han comenzado a producir salidas espaciales mediante la utilización de *tokens* especiales en el vocabulario o el uso de cabezas auxiliares de segmentación.

Finalmente, una tendencia emergente es la reutilización de grandes modelos de difusión, ya entrenados a escala masiva com millones de pares texto-imagen, para tareas de comprensión espacial no generativa. Reformulando el objetivo y retocando mínimamente la arquitectura se aprovechan su capacidad de representación y conocimiento espacial. Este es el enfoque que adoptamos para CocoGold, incorporando además condicionamiento textual explícito.

Nuestra primera contribución es FilterNet, que aborda el aprendizaje autosupervisado de parámetros de filtros.

FilterNet parte de un extractor ResNet preentrenado y añade una *cabeza* predictiva que produce, de una sola pasada, seis valores en el rango normalizado $(-1, 1)$: exposición, brillo, contraste, sombras, altas luces y temperatura de color. Para entrenar de extremo a extremo con *backpropagation* es necesario renderizar, dentro del propio modelo, una versión mejorada de la imagen usando **filtros diferenciables**. Implementamos la mayoría mediante curvas de tonos paramétricas (pares alto/bajo para cada ajuste) aplicadas en espacio RGB **lineal** con 10 bits de precisión, y combinadas con la imagen original con ajustes de transparencia. La temperatura de color, en cambio, se resuelve

con matrices de ganancia 3×3 previamente calculadas, y para la exposición aplicamos su definición fotométrica. Este diseño emula el comportamiento de aplicaciones profesionales, evitando saturaciones o artefactos que se observan con métodos simplistas de contraste o brillo.

El entrenamiento es auto-supervisado: a partir de un conjunto de fotos "buenas" generamos muestras degradadas mediante combinaciones controladas de los seis filtros. El objetivo del modelo es aproximar la foto de referencia a partir de la imagen degradada, aplicando los filtros mencionados. La función de pérdida combina: (i) término píxel a píxel en sRGB; (ii) pérdida perceptual con características VGG de varios niveles; y (iii) pérdida de estilo (matrices de Gram), con el objetivo de captar estructura, textura y color en mayor consonancia con la percepción humana.

Estudiamos diversas variantes que favorecen la estabilidad y la generalización. Por ejemplo, inicializamos los pesos de la cabeza de regresión con valores cercanos a cero (en el centro del rango predictivo de los filtros) para evitar predicciones extremas en las etapas iniciales de entrenamiento. Utilizamos también una capa que denominamos *maskout*, análoga a *dropout*, que utilizamos para desactivar aleatoriamente alguno de los seis canales con objeto de fomentar combinaciones alternas. En cuanto a la preparación de datos, utilizamos regímenes de degradación moderados, reduciendo la probabilidad de saturación irrecuperable (*clipping*), y acercando la distribución de datos de entrenamiento a escenarios reales.

Para evaluar FilterNet (Tabla 5.1) usamos métricas de referencia (PSNR, SSIM, FSIMc) y dos métricas con mayor correlación perceptual: DISTS y una pérdida de contenido basada en VGG. Los resultados muestran mejoras consistentes al añadir componentes perceptuales a la pérdida y al operar en RGB lineal. En cuanto a la arquitectura, verificamos que ResNet-50 supera a ResNet-34, y un breve entrenamiento del backbone completo mejora frente a mantenerlo congelado por completo.

En cuanto a la implantación del sistema, exportamos el modelo a Core ML con cuantización a 16 bits, alcanzando tiempos de inferencia bastante inferiores a un segundo, incluso en móviles de 2020 y anteriores. Esto nos permitió incorporar el modelo a una aplicación popular de fotografía, validando la viabilidad de nuestro enfoque paramétrico, interpretable y optimizado para ejecución local (Fig. 1).

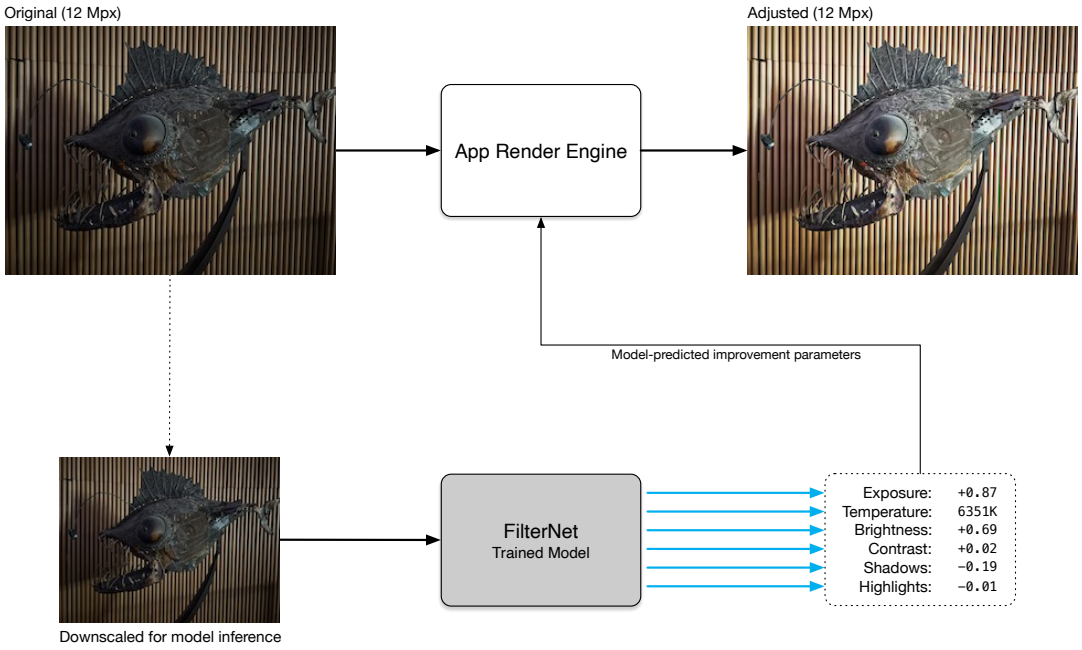


FIGURA 1: Esquema de inferencia local con FilterNet (copia de Fig. 3.7). El modelo predice parámetros de filtros a partir de una imagen en miniatura. Dichos parámetros se aplican a la imagen original a resolución completa, manteniendo la edición no destructiva e interpretable.

Como segunda contribución, CocoGold aborda la tarea de segmentación utilizando técnicas de difusión guiada por texto.

La edición local requiere identificar regiones de interés. CocoGold reutiliza Stable Diffusion 2 y reformula la tarea de segmentación como generación condicionada por imagen y texto: dada una imagen y un término expresado con texto (por ejemplo, "gato"), el modelo aprende a producir una copia de la imagen donde las regiones asociadas al término aparecen destacadas en blanco. Para ello concatenamos la representación latente de la imagen original y de la imagen objetivo, e introducimos ambas como entradas al sistema, adaptando la primera capa de la U-Net para aceptar 8 canales (correspondientes a 2 imágenes) en espacio latente. El resto de la arquitectura del modelo de difusión no necesita cambio alguno. El condicionamiento con texto utiliza el modelo CLIP, al igual que el modelo Stable Diffusion en su versión estándar. El entrenamiento se desarrolla sobre la U-Net; tanto CLIP como el codificador de imágenes (VAE) permanecen congelados.

Nuestro *dataset* de entrenamiento se basa en COCO 2017. Creamos un iterador que, para cada muestra solicitada, proporciona un recorte aleatorio que contiene algún área perteneciente a una de las categorías de objetos de

COCO. Como imagen objetivo, preparamos ese recorte con dicho área en color blanco. Puesto que COCO presenta fuertes desequilibrios entre categorías, seleccionamos para nuestro prototipo un subconjunto de 14 categorías con similar representación. El proceso de inferencia sigue el patrón estándar en Stable Diffusion: DDIM, con 50 pasos de reducción de ruido. El modelo, como se ha indicado, genera una copia aproximada de la imagen objetivo, con la máscara de segmentación sobrepuesta en color blanco. Gracias a un proceso de post-procesado (selección de umbrales de color, seguido de operaciones morfológicas para eliminar falsos positivos de pequeño tamaño), obtenemos la máscara binaria de segmentación. Para conseguir compatibilidad con escenas que contienen áreas blancas de forma natural (cielos sobresaturados, por ejemplo), aplicamos a la imagen original una desaturación selectiva, de modo que la copia generada no confunda áreas realmente saturadas con el color blanco a generar en las zonas de segmentación. Este pequeño truco permite al modelo trabajar con todo tipo de imágenes.

Para la evaluación de resultados medimos IoU (*Intersection over Union*) y precisión por píxel sobre el dataset de validación de COCO, bajo un protocolo acorde con la tarea formulada (clase única condicionada por texto, sin existencia de muestras negativas al utilizar siempre recortes que contienen la categoría objetivo). Observamos una mejora de generalización sobre clases no vistas durante el entrenamiento al aumentar la duración del mismo, aunque el rendimiento en este caso es manifiestamente inferior al de clases entrenadas. La agregación de varias predicciones aporta ganancias modestas. Sin embargo, observamos que métodos de inferencia de pocos pasos, tales como *trailing DDIM*, funcionan sin cambios sobre el modelo previamente entrenado, lo que nos permite utilizar inferencia de tan sólo 1 a 4 pasos para obtener resultados satisfactorios. Esto reduce enormemente el tiempo de latencia y la aplicabilidad práctica del método en hardware de consumo.

La integración de FilterNet y CocoGold da lugar a COCONET, una prueba de concepto para edición local basada en filtros técnicos.

Partiendo de las máscaras de CocoGold exploramos dos integraciones simples de FilterNet: (A) predecir parámetros globales sobre la imagen completa, y aplicarlos en la zona segmentada; (B) aplicar FilterNet sobre un recorte alrededor del área segmentada para obtener parámetros más ajustados a dicha región, y aplicarlos igualmente sobre la máscara de segmentación. En ambos casos, un difuminado suave del contorno facilita transiciones naturales. A nivel práctico, este pipeline combinado respeta los píxeles fuera de la máscara

FIGURA 2: COCONET: edición local aplicando los parámetros de FilterNet (predichos globalmente o utilizando un recorte alrededor del área de interés) sobre la máscara de CocoGold.

(se toman del original) y preserva el carácter no destructivo e interpretable del proceso. La limitación principal es la resolución de la máscara de segmentación: mientras FilterNet es completamente agnóstico a la resolución, la segmentación produce una máscara binaria de tamaño fijo. Esta restricción puede mitigarse con técnicas de aumento de resolución guiadas por bordes, tales como las utilizadas en DiffDIS o en aplicaciones reales demostradas en la industria: es el caso del tratamiento de imágenes de profundidad en el iPhone, donde la máscara (de profundidad, en este caso) es de baja resolución pero se adapta a tamaño real mediante procesos especializados que respetan los bordes.

La Figura 2 muestra los resultados de COCONET utilizando cada uno de los dos métodos indicados.

Para concluir, ofrecemos un breve resumen de resultados, limitaciones identificadas y líneas de trabajo que pretendemos abordar en el futuro.

En esta tesis pretendemos demostrar que es posible acercar la calidad y flexibilidad de la edición profesional a experiencias cotidianas y accesibles, sin renunciar a la interpretabilidad ni a la eficiencia en dispositivos de consumo. Las contribuciones centrales son:

- Un enfoque de mejora fotográfica basado en la predicción de parámetros de filtros fotográficos, en lugar de generar píxeles editados. Esta decisión hace al sistema independiente de la resolución, controlable por el usuario y fácil de implantar, incluso localmente.
- Una arquitectura de filtros diferenciables que replica de forma realista operaciones fotográficas en espacio RGB lineal con 10 bits de precisión,

con pérdidas perceptuales que conducen a resultados más acordes con la evaluación humana.

- Una estrategia auto-supervisada de generación de datos que evita depender de pares editados, permitiendo entrenar con colecciones de fotos de alta calidad degradadas de forma plausible.
- Validación, a través de CocoGold, de que los modelos de difusión preentrenados pueden reutilizarse eficazmente para segmentación guiada por texto con mínimas modificaciones, formulando la tarea como generación *imagen a imagen*.
- Una integración (Coconet) que aplica los parámetros de FilterNet de forma localizada sobre las máscaras de CocoGold, habilitando ediciones no destructivas y semánticamente informadas a partir de instrucciones simples.

Entre las limitaciones señalamos: (i) la resolución fija de las máscaras, que puede producir imprecisiones en detalles finos como el cabello y otras estructuras detalladas; (ii) la restricción, en esta primera versión, a términos de una sola palabra, lejos de descripciones ricas y expresiones identificativas que abundan en casos reales; y (iii) la latencia propia de los modelos de difusión, mitigable con procesos de eliminación de ruido como los indicados, destilación, o técnicas de entrenamiento diseñadas para pocos pasos de inferencia.

Como líneas de trabajo futuras proponemos dos direcciones complementarias. La primera consiste en extender CocoGold a un verdadero escenario de vocabulario abierto y lenguaje natural, incorporando expresiones identificativas ("la persona de la izquierda con camisa roja"). Dado el fuerte alineamiento multimodal de los modelos de difusión, postulamos que esta ampliación depende más del currículo de datos que de cambios profundos de arquitectura. Paralelamente, explorar pérdidas y señales auxiliares sensibles a contornos podría mejorar la fidelidad de bordes sin sacrificar la simplicidad del sistema.

La segunda dirección es complementar el enfoque con VLMs como predictores de parámetros de edición basados en intenciones expresadas por el usuario, en lugar de estimar parámetros automáticos tan sólo. Un VLM podría recibir una imagen y una instrucción en lenguaje natural ("haz el cielo más dramático", "ilumina la cara manteniendo el fondo") y devolver un vector interpretable de parámetros para conseguir tal efecto. Para entrenar un sistema de estas características planteamos la construcción de un dataset sintético utilizando LLMs para generar variabilidad y riqueza en las descripciones.

En conjunto, FilterNet, CocoGold y su integración inicial en Coconet, marcan un camino pragmático hacia sistemas de edición fotográfica más intuitivos: automáticos cuando el usuario lo desea, ajustables cuando la intención artística requiere control fino, y siempre respetuosos con los datos originales. El equilibrio entre independencia de resolución, interpretabilidad y viabilidad en móviles convierte a estas propuestas en una base sólida sobre la que seguir construyendo experiencias de edición centradas en las personas y habilitadas por IA.

Acknowledgements

Working on your passion is a privilege, but it unfairly exacts sacrifices from the people you love. I'm incredibly lucky to have had the support of María José, my partner in life. She made it easy for me to spend time on this, and when I was stuck, she helped with common-sense reasoning that, frankly, is anything but common. I apologize to my Mom and Dad for always bringing my laptop when I visit, to my son Pablo for not exploring Hyrule or Eorzea as much as we'd have liked, and to my son Javier for sometimes talking too much about work and too little about life. They are the best.

I'm deeply grateful to my co-directors. They didn't get in the way when times were good, but they were there for me when I most needed to believe that it was still worth it. All projects have ups and downs; all good things require some fighting for. It takes wisdom to know when to push, when to inspire, and when to stand by. I couldn't have done this without you.

I'm forever indebted to my coworkers and to the open exchange of ideas in the ML community. I can't imagine how people got anything done in the before times. I love every minute of this future we're so lucky to live in.

Thank you.

Contents

List of Figures

List of Tables

List of Abbreviations

ADD	**A**dversarial **D**iffusion **D**istillation
AI	**A**rtificial **I**ntelligence
API	**A**pplication **P**rogramming **I**nterface
CLIP	**C**ontrastive **L**anguage-**I**mage **P**re-training
CNN	**C**onvolutional **N**eural **N**etwork
COCO	**C**ommon **O**bjects in **C**ontext
CPU	**C**entral **P**rocessing **U**nit
DDIM	**D**enoising **D**iffusion **I**mplicit **M**odels
DISTS	**D**eep **I**mage **S**tructure and **T**exture **S**imilarity
FSIM	**F**eature **S**imilarity **I**ndex
FSIMc	**F**eature **S**imilarity **I**ndex (color)
GAN	**G**enerative **A**dversarial **N**etwork
GPU	**G**raphics **P**rocessing **U**nit
IoU	**I**ntersection **o**ver **U**nion
LADD	**L**atent **A**dversarial **D**iffusion **D**istillation
LCM	**L**atent **C**onsistency **M**odels
LLM	**L**arge **L**anguage **M**odel
LoRA	**L**ow-**R**ank **A**daptation
mIoU	mean **I**ntersection **o**ver **U**nion
ML	**M**achine **L**earning
MSE	**M**ean **S**quared **E**rror
PSNR	**P**eak **S**ignal-to-**N**oise **R**atio
RefCOCO	**Ref**erring Expressions in **COCO**
RefCOCO+	**Ref**erring Expressions in **COCO P**lus
RefCOCOg	**Ref**erring Expressions in **COCO** (**G**oogle collection method)
ResNet	**Res**idual **Net**work
RGB	**R**ed **G**reen **B**lue
RL	**R**einforcement **L**earning
SAM	**S**egment **A**nything **M**odel
sRGB	**s**tandard **RGB** color space
SSIM	**S**tructural **S**imilarity **I**ndex
UI	**U**ser **I**nterface
U-Net	**U**-shaped **Net**work
VAE	**V**ariational **A**utoencoder
ViT	**Vi**sion **T**ransformer
VLM	**V**ision-**L**anguage **M**odel

For my Mom and Dad.

Chapter 1

Introduction

The editing of photographs is as old as photography itself. Analog photographers applied darkroom techniques, such as *Dodge and Burn* [1], to adapt the light and dark areas of an image captured in film to the much narrower dynamic range of printed media. Technical, artistic, and communications reasons persist to this day.

Digital photography made photography more approachable, and made it easier to apply traditional editing processes and derive new ones. Even though millions of people take photographs every day and modify them on computers and mobile phones, professional editing tools still require a steep learning curve to master. This complexity is the reason why most users rely on simple, one-touch filters that don't require any technical specialization or familiarity with the theoretical foundations of digital image processing. Very frequently, these filters are algorithmically programmed and follow a set of fixed recipes, whose rules are applied to any target photo. Professional editors, on the other hand, are able to apply dozens of edits on a photo to achieve the look and style they are after, attract attention to a subject, adjust areas too light or too dark, and many other technical and artistic reasons.

The democratization of photography through mobile devices has created an unprecedented demand for automatic photo enhancement tools. Modern smartphones capture billions of photos daily, yet the vast majority of these images never undergo any form of enhancement beyond the automatic processing applied by the camera software. This gap between the capabilities of professional editing tools and the needs of casual photographers presents both a challenge and an opportunity for applying machine learning techniques to photo enhancement.

The goal of this work is to explore how deep learning methods can be applied to photo editing. In particular, we are primarily interested in the topic of automatic improvement, where a deep learning system is able to determine a set of operations that would increase the technical quality of a photo, based on the contents of that photo or the way it looks. By leveraging the accumulated experience of deep learning on computer vision, we aim to make *technical photo editing* more approachable to end users, on the devices they use every day. To achieve this goal, we also need to pay attention to practical deployment considerations and technical constraints, such as the impact on memory consumption, latency, power, or maximum resolution supported. If we succeed, these methods will allow users to enjoy more flexibility in their photo editing needs, without requiring a high level of technical specialization.

Let's stop a second on the term *technical photo editing*, which we used in the previous paragraph, and clarify what we mean. We use this term to refer to edits that do not impose stylistic choices. Our goal is to fix issues that naturally occur when capturing images, such as *underexposure* when shooting in the dark, *color casts* motivated by artificial light sources, etc. We aim to revert these imperfections in a sensible way, but also neutrally in terms of aesthetic preference. Once photos have been *technically recovered*, they can be used as starting points for users to apply styles of their choice.

1.1 The Evolution of Photo Enhancement

Photo enhancement has evolved through several distinct phases. In the analog era, photographers relied on chemical processes and physical manipulations in the darkroom. The transition to digital photography introduced pixel-based manipulations and mathematical transformations. The current era is characterized by the application of machine learning techniques that can learn complex enhancement strategies from data rather than relying on hand-crafted algorithms.

Each phase has brought its own advantages and challenges. Digital processing made edits non-destructive and infinitely repeatable, while machine learning promises to make sophisticated edits accessible to users without technical expertise. However, these advances also introduce new challenges: ensuring that automated enhancements respect artistic intent, maintaining computational efficiency for real-time applications, and providing users with meaningful control over the enhancement process.

In recent years, deep learning has had a huge impact on computer vision tasks, such as image classification, object detection, semantic segmentation, and many others. Generative methods have also been applied to some creative tasks, such as style transfer, or the ability to create plausible visual content based on different types of inputs. These methods have also been explored for photo editing, but it's rare that they can work on high-resolution photos, such as those taken by any modern mobile phone. Some recent efforts attempt to solve this problem by learning the parameters of editing functions that can be applied to the full-resolution photo, but limitations still exist: editing functions are heavily simplified versions of tools available in professional editing tools [2, 3]; learning is difficult or relatively unstable [4]; final results are frozen and the user cannot tweak the applied parameters (black boxes) [5].

1.2 From Automatic to Interactive Enhancement

Professional photo editing tools have evolved significantly in recent years. Adobe Lightroom, Photoshop, and similar software now incorporate sophisticated AI-powered features including automatic subject selection, sky replacement, and intelligent masking. These tools demonstrate that localized, content-aware editing is technically feasible and highly valued by photographers. However, these capabilities remain largely confined to desktop applications that require significant technical expertise, commercial subscriptions, and a complex multi-step workflow: select, refine, adjust, blend. Moreover, most automatic enhancement systems – particularly those designed for mobile devices – still operate globally, applying uniform adjustments across the entire image without considering local variations in content or lighting.

The spectacular advances in Large Language Models (LLMs) and their integration with vision systems opens new research directions for democratizing and simplifying these professional capabilities. We can envision how users could potentially express their editing intent through natural language: "brighten the shadows," "make the sky more dramatic," or "enhance the person's face while keeping the background soft." The challenge lies not just in achieving localized edits – which professional tools can already do – but in making this process intuitive, conversational, and accessible on the devices people use every day, while maintaining photographic quality. These interfaces, in fact, could become natural UIs to extend our definition of *technical editing* towards ways to capture user intent and apply stylistic and artistic edits.

To bridge this gap between professional capabilities and everyday accessibility, we need systems that can: (1) understand and predict high-quality enhancements automatically without requiring technical expertise, (2) identify and segment specific regions based on natural language descriptions rather than manual selection tools, and (3) apply different treatments to different parts of an image while maintaining coherence – all within the computational constraints of mobile devices. This requires combining the strengths of automatic enhancement systems with the spatial understanding capabilities needed for localized editing, while maintaining the simplicity and efficiency required for consumer applications.

As a first step toward this vision, diffusion models – originally designed for image generation – show promise for being repurposed for text-grounded understanding tasks. By leveraging their pre-trained visual-linguistic representations, we can potentially identify and segment objects specified through natural language without requiring complex architectural modifications or extensive task-specific training. This capability could form a crucial building block for localized, instruction-based photo enhancement.

1.3 The Promise and Challenges of Generative AI for Photography

Generative techniques can create images based on natural-language text descriptions supplied by the user. This can be achieved using autoregressive methods, such as DALL·E [6] or Parti [7]; or diffusion models, like Stable Diffusion [8], Imagen [9], or DALL·E 2 [10]. Even though these methods were designed to interpret textual descriptions of images and generate new images that match those descriptions, later research showed that they can be adapted to accept images (instead of text) to guide the generation process [11]. It is also possible to use other conditioning signals – such as depth maps or canny edges [12] – with the same purpose. Progress has been fast, helped by open models like Stable Diffusion that make experimentation easy. Quality has advanced dramatically, as evidenced by some *closed* models like Google's Gemini 2.5 Flash Image (aka "nano-banana") [13], state of the art for image generation and transformation. Yet for photographic editing these approaches remain challenging: they often lack arbitrary-resolution support, behave as black boxes (results cannot be tweaked, except by iteration), and generate

images anew rather than actually applying *edits* to them. These constraints conflict with many professional, artistic, or documentary use cases.[1]

Given these limitations, we focus on edits rather than generation: improving exposure, color, or brightness, and doing so in a parametric way that can be understood, modified by the user, and applied to images of any resolution. When we rely on generative models, we do so for their learned representations and spatial understanding, but not to synthesize edited results.

Recent work has begun to explore how generative models can be adapted for more conservative editing tasks. Marigold [14] demonstrated that diffusion models pre-trained for image generation can be effectively repurposed for depth estimation by reformulating the task as image-to-image translation. This insight – that generative models can be adapted for non-generative vision tasks with minimal architectural changes – opens new possibilities for leveraging their powerful representations while maintaining the integrity of the original image (see Section 2.7).

Furthermore, the computational requirements of current generative models pose significant challenges for deployment on consumer hardware and mobile devices. Most state-of-the-art diffusion models require powerful GPUs, making them challenging to use in real-time mobile applications. This motivates the development of lighter-weight approaches that can deliver meaningful enhancements within the constraints of mobile hardware, and encourages exploration of fast inference methods when diffusion models are used.

Happening in parallel with the evolution of text-to-image models, large language models (LLMs) have gone through tremendous progress. The possibility to use natural language as a UI has been anticipated by the community for many years, but LLMs have just crossed a quality threshold that has allowed the development of general-purpose chat assistants, such as ChatGPT [15, 16] or Claude [17]. These state-of-the-art models are *closed*, in the sense that they are only accessible through private web services and APIs and can only be examined by researchers through these methods. Fortunately, models with varying levels of openness are also being published, whose quality is approaching that of the best closed models [18, 19, 20, 21, 22, 23]. The increased accessibility makes it much easier to use this powerful technology for tasks such as synthetic dataset generation, or integration with computer vision models in complex pipelines. This is a promising research direction

[1] Nevertheless, we expect advances in this area to continue, and they may soon close the gap between *editing* and *generation* for most practical purposes.

that could facilitate the creation of intuitive editing pipelines, and the tasks we propose as future work in Chapter 6.2.

The capability of LLMs for synthetic data generation has been explored to enrich generative text-to-image tasks in the context of image editing. This is the case of InstructPix2Pix [24], which is able to generate a modified image based on an existing image used as a reference, combined with editing instructions supplied in natural language. A big part of the success of the method is the clever use of an LLM to produce a rich dataset of image editing instructions, bootstrapped with a small set of human-supplied data. As pointed out before, this is a very promising area where we expect lots of progress. However, it still suffers from the same drawbacks we outlined: it does not support arbitrary resolution, modifications are generative so new data is generated rather than performing image transformations on the original photo, results are opaque and cannot be easily tweaked by the user, and the process is stochastic in nature. We will further explore these methods in this work.

A recent approach that combines the strong capabilities of LLMs with strong computer vision models is the integration of both into combined Vision-Language Models, or VLMs [25, 26, 27, 28, 29]. These models encode an image (or several ones), plus natural language text inputs, and process the combined representations through an LLM which produces text responses. This simple and unified architecture has been successfully used to tackle traditional computer vision tasks. Tasks such as OCR, captioning, or visual-question answering were clear candidates to be approached by models that *understand* images and text, and produce text as output. But notably, VLMs have also been successfully applied to not-so-obvious tasks like object detection and semantic segmentation, whose outputs are bounding boxes and segmentation masks. This is achieved by training the LLM portion of the VLM to produce some sort of structured text, like formatted output (json, for instance), or through the use of geometry-aware language *tokens* added to the model's vocabulary, and then translating those outputs to boxes or masks. This was first shown by the PaliGemma models [26, 27], whose pretraining scheme used a great variety of computer vision tasks and demonstrated the ability for these models to learn complex representations of the image and text inputs, allowing their use for spatial-aware tasks.

Recent VLMs have further advanced these capabilities. The Qwen-VL family [28, 29] has demonstrated state-of-the-art performance on referring expression comprehension benchmarks like RefCOCO, RefCOCO+, and RefCOCOg,

achieving over 90% accuracy through sophisticated spatial position embeddings and multimodal rotary position encoding (M-RoPE). Similarly, Moondream [30] has introduced *grounded reasoning*, where the model can pause to explicitly reason about spatial relationships and point at objects in images during inference, dramatically improving performance on counting and localization tasks. These advances show that VLMs are becoming increasingly capable of precise spatial understanding – a crucial requirement for localized photo editing applications.

However, it's an open question whether text generation (the output of LLMs and VLMs) is sufficient to model and solve complex geometric tasks where precision and detail are important. Recent work confirms that VLMs can be reasonable generalists for vision tasks, but lag behind specialist vision models for specific tasks, in particular geometric ones [31].

1.4 Goals

The convergence of these technologies – neural photo enhancement, representational capabilities of diffusion models, and vision-language understanding – presents an opportunity to create more flexible and intuitive photo editing systems. By combining automatic enhancement capabilities with text-grounded segmentation, we can work toward a system that applies interpretable edits to specific regions based on text instructions.

This vision requires addressing several technical challenges, that are our motivating goals for this work:

- Create a system capable of predicting technical enhancements that improve photos taken in unfavorable conditions
- Identify regions of interest from text descriptions
- Apply technical enhancements on local regions, blending seamlessly within the image
- Maintain real-time performance suitable for interactive editing
- Preserve the non-destructive and interpretable nature of edits so users can refine results

1.5 Contributions and Thesis Organization

Aligned with the goals stated in the previous section, this dissertation makes several contributions to the field of neural photo enhancement:

1. We introduce FilterNet (Chapter 3), a neural network that predicts parameters for traditional photographic filters rather than directly generating enhanced pixels. This design choice enables resolution-independent editing and produces interpretable results that users can further adjust.
2. We develop a self-supervised training approach for Filternet that eliminates the need for paired training data. By automatically generating degraded versions of high-quality images and using a multi-component loss function (combining pixel, perceptual, and style losses), we achieve effective photo enhancement that runs on modest hardware and can be deployed to mobile devices (Chapter 5.1.1).
3. We demonstrate how models trained to bridge visual and textual modalities can be adapted for localized photo editing tasks. We show that diffusion models, despite being trained for generation, contain rich representations suitable for dense prediction tasks, like segmentation, when properly fine-tuned.
4. We present CocoGold (Chapter 4), which repurposes pre-trained diffusion models for text-grounded segmentation. By reformulating segmentation as a generative task, we leverage the visual-linguistic understanding from billions of image-text training pairs while requiring only minimal architectural changes and modest computational resources for fine-tuning.
5. We combine FilterNet with CocoGold to create CoCoNet (Chapter 5.3), a unified pipeline for text-guided local photo enhancement. Users can select objects using simple text terms, and the system automatically predicts appropriate photographic adjustments (exposure, contrast, etc.) for those regions. While the segmentation operates at a fixed resolution, the predicted filter parameters remain interpretable and can be applied at any image resolution.

Figure 1.1 displays a visual conceptualization of the systems contributed by this work.

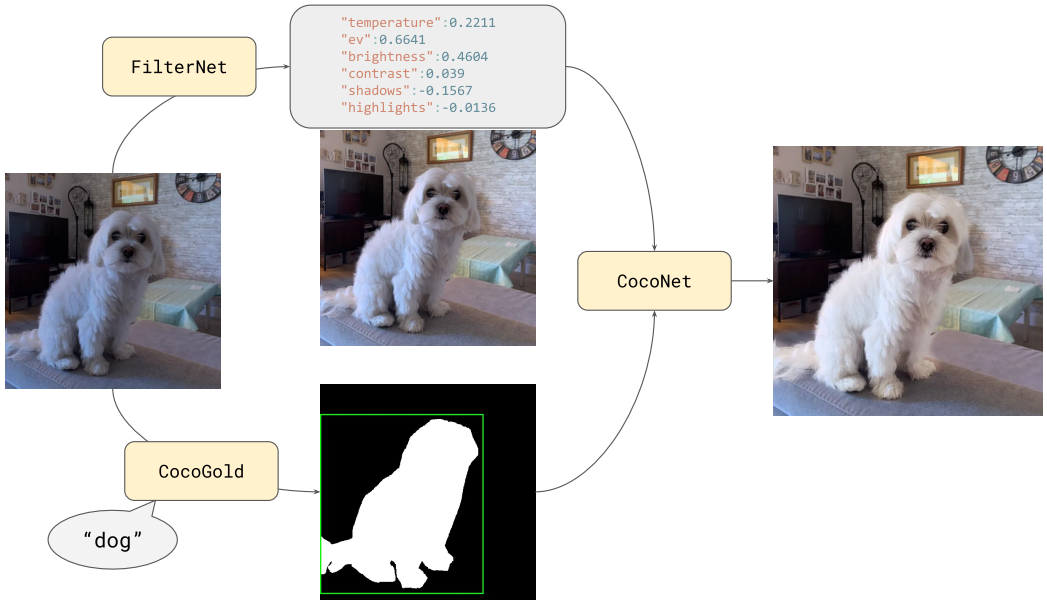


FIGURE 1.1: Conceptual pipeline. FilterNet predicts global filter parameters that can be used to render at full resolution. CocoGold uses text inputs to generate a segmentation mask. Coconet extends FilterNet with CocoGold masks to achieve local edits based on parametric filters.

These methods establish a foundation for future interactive photo editing systems driven by complex text descriptions where users can express editing intents verbally.

The remainder of this dissertation is organized as follows:

- Chapter 2 provides a comprehensive review of the state of the art in neural photo enhancement, adjacent, and supporting areas. It walks through various research topics related to image and text representations, computer vision, or image generation; and discusses how these methods relate or can be applied to photo editing.
- Chapter 3 presents the FilterNet architecture and methodology in detail. FilterNet is a novel method to estimate global automatic edits that supports resolution independence, interpretability, and is efficient in the use of resources.
- Chapter 4 discusses CocoGold, our exploration of diffusion models for text-grounded image segmentation. It demonstrates a simple way to leverage the rich text-image relationships encoded in pre-trained diffusion models, repurposing them for segmentation.

- Chapter 5 describes our experimental setup and results, including ablation studies and performance analysis. The chapter ends with experiments that integrate FilterNet with CocoGold, showing a proof-of-concept method to extend FilterNet for local edits, where the subject areas are expressed using simple text terms.
- Chapter 6 concludes with a summary of our contributions and outlines future research directions that arise as natural continuations of this work.

- Capítulo 7 es un resumen en castellano de las ideas principales de nuestra investigación, incluyendo antecedentes, metodología y resultados.

Chapter 2

State of the Art

2.1 Overview and Scope

This chapter contextualizes methods for photo enhancement, contrasting classical heuristics with learning-based approaches that predict either pixels or interpretable parameters. We then review enabling components (transfer learning, perceptual losses, differentiability) and supervision regimes (self-supervision, synthetic degradation) that make these systems practical. Finally, we move from global enhancement to localized, text-grounded manipulation by surveying segmentation approaches (open-vocabulary, referring expressions, foundation models), and show how diffusion models can be adapted from generation to dense prediction tasks that serve photo editing needs.

2.2 Photo Retouching and Enhancement

2.2.1 Classical and Heuristic Photo Enhancement

Classical automatic correction methods, such as global histogram equalization [32] and CLAHE [33], remain strong baselines for exposure and contrast adjustment. However, these approaches operate on summary luminance statistics rather than image semantics: they can amplify noise, push highlights into saturation, or distort skin tones, and they do not coordinate color and tone in a way that preserves photographic intent. These limitations motivate learning-based methods that infer content-aware, coordinated adjustments.

2.2.2 Deep Learning Approaches to Photo Enhancement

The application of deep learning techniques to the problem of image editing and retouching has been previously addressed by diverse authors. Yan et

al. [2] use a combination of features to enrich each pixel with information about its vicinity, and they also consider global image attributes. In order to provide local context, they perform two previous segmentation steps —scene parsing and object detection— and then select the most representative mask categorization for a group of related pixels. Their focus is to learn artistic styles and how to apply them in a local-sensitive manner. Even though their method can be configured to work with global adjustments, their results require a considerable amount of consistency, which is enforced by various means: a single photographer is in charge of creating all the styled image pairs; the photographer knows what the categories of the segmentation step are (and treats those objects accordingly); finally, the effects are applied using Photoshop Actions to ensure that all filter amounts are the same when applied to different images. Furthermore, the results are obtained as a rendered version of the retouched image, so scalability to high-resolution images is complex. The nature of edits is opaque and unknown to the final user.

A fundamental limitation of Yan et al.'s approach is the requirement for strict consistency in the training data creation process. This constraint significantly limits the scalability of the method and makes it challenging to adapt to diverse photographic styles or multiple photographers' preferences. Additionally, the pixel-based output representation means that the system must process images at their final resolution during both training and inference, which poses computational challenges for modern high-resolution photography where images typically contain tens of megapixels.

Wang et al. [3] focus their attention on exposure correction. They obtain good results by learning an illumination mapping as an intermediate result, and then leverage that mask to perform the exposure correction. One of their contributions is the use of a sensible loss function that combines several aspects, such as the smoothness of the illumination mask, the color vector angle difference in RGB space, and the L2 pixel difference in the CIE Lab color space. Their method seems to generalize well in the context of exposure correction, but it also produces final pixel representations of the input images. The lack of availability of suitable datasets is a frequent limitation of these supervised training methods, so the authors resort to creating their own dataset consisting of 3,000 underexposed images with their correctly exposed counterparts.

The illumination-based approach of Wang et al. represents an interesting middle ground between purely pixel-based methods and parameter-based

approaches. By explicitly modeling illumination as an intermediate representation, they achieve better interpretability than end-to-end pixel generation methods. However, the method is specialized for exposure correction and does not generalize to other common photographic adjustments such as color temperature, contrast, or highlight/shadow management.

Beyond models, suitable datasets and supervision choices strongly influence feasibility; we summarize key datasets next.

2.2.3 Datasets for Photo Enhancement

One significant effort in producing a dataset suitable for supervised learning of image edits was undertaken by Bychkovsky et al., who created the MIT-Adobe FiveK Dataset [34]. It consists of 5,000 photographs in RAW format, plus five sets of edits performed by individual photographers. This dataset has been influential in the field, as it provides ground truth edits from multiple experts, allowing researchers to study the subjective nature of photo enhancement and the diversity of editing styles. Unfortunately, the dataset license does not support commercial use, which limits adoption in applications that require permissive licensing.

In addition to the dataset itself, Bychkovsky et al. [34] also introduce the concept of *learning edits* instead of producing final image representations. They do so by focusing on just tone (luminance) adjustment and applying various regression methods on global image features.

The concept of learning edits rather than pixel outputs represents a paradigm shift in photo enhancement research. This approach offers several advantages: (1) the learned parameters are resolution-independent and can be applied to images of any size; (2) the edits are interpretable and can be further adjusted by users; and (3) the computational requirements during inference are significantly reduced since only parameters need to be predicted rather than full-resolution images generated.

We next consider generative and reinforcement learning approaches that likewise aim for resolution-agnostic edits but differ in training dynamics and inference.

2.2.4 Generative and Reinforcement Learning Approaches

Another effort in the direction of learning edits was undertaken by Hu et al. [4]. They create a resolution-agnostic image improvement model by applying a sequence of piecewise-linear filters in a GAN-like reinforcement-learning (RL) environment. One advantage of their approach is that they do not need prepared image pairs, just a set of badly-exposed pictures and a different set of pictures considered to be of high quality. One disadvantage is that convergence of these models, like in other generative networks, is challenging and fragile [35], so the authors resort to several training strategies to facilitate convergence. Despite the relative instability in training, RL/GAN approaches are often considered alternatives to purely generative translation models such as CycleGAN [36] when preserving input resolution and avoiding hallucinated content are primary objectives.

The use of reinforcement learning for photo enhancement introduces an interesting sequential decision-making framework to the problem. In Hu et al.'s approach, the agent learns to apply a sequence of edits, mimicking the workflow of professional photo editors who typically apply multiple adjustments in a specific order. However, the sequential nature of the approach can lead to longer inference times and the GAN-based training can be unstable, requiring careful hyperparameter tuning and training strategies.

2.2.5 Advances in Learnable Filters

Moran et al. [5] combine a U-Net-like architecture [37] with a set of learnable piecewise-linear differentiable filters applied in three different colorspaces. Filters are inspired by curve tools available in photo editing tools. While the shapes of these filters are learned, they are never exposed to the end user because the system uses them internally to generate the enhanced image pixels instead of the edits. In addition, learned curves, while being interpretable, cannot be easily decomposed as image editing primitives like brightness, contrast, or color temperature.

The same authors also employ a loss function composed of several components, which has been identified as a key factor to quality improvement (Section 2.3.1).

These method choices are enabled by widely useful components: transfer learning, perceptual losses, and differentiable operators, which we cover next.

2.3 Transfer Learning and Deep Features

Instead of attempting to train a deep network from scratch, many systems rely on well-known transfer learning techniques that leverage previously-trained networks and fine-tune them to a different domain [38, 39, 40]. Convolutional networks such as ResNet [41] are frequently used as feature extractors, with custom predictive layers on top to adapt to particular tasks.

Transfer learning has become a fundamental technique in deep learning, particularly for computer vision tasks where large-scale annotated datasets may not be available. The success of transfer learning can be attributed to the hierarchical nature of features learned by convolutional neural networks: lower layers learn generic features like edges and textures that are useful across different visual tasks, while higher layers learn more task-specific features. This hierarchical feature learning makes it possible to adapt pre-trained models to new tasks with relatively small amounts of task-specific data.

Previous work has shown that the choice of pre-trained model and the layer at which features are extracted can significantly impact transfer learning performance. Models pre-trained on large-scale datasets like ImageNet [42] have been shown to provide excellent initialization for a wide variety of downstream tasks, including medical imaging [43], scene classification [44], or malware detection [45].

Feature reuse is only part of the story; the loss we optimize also shapes perceived quality, as we discuss next.

2.3.1 Perceptual Loss Functions

A key component of these architectures is the loss function that measures the deviation between the output and the desired result. As Wang et al. describe [3], the use of an appropriate loss function can be a significant factor in the system's quality. Johnson et al. [46] describe the use of a separate network to extract feature vectors from both the network output and the target, using a metric between those vectors as the basis for a loss function. We follow that approach and combine it with additional loss components to achieve a simple training strategy based on backpropagation.

The evolution of loss functions for image enhancement and generation tasks represents a significant advancement in the field. Traditional pixel-wise losses

like L1 or L2 distance fail to capture perceptual similarity because they operate on individual pixels without considering spatial relationships or semantic content. The introduction of perceptual losses by Johnson et al. [46] marked a paradigm shift by leveraging the feature representations learned by pre-trained networks as a proxy for human perception.

Further developments in perceptual losses include the use of style losses based on Gram matrices [47], which capture texture and style information, combined with metrics like LPIPS [48] that are explicitly designed to match human perceptual judgments. The combination of multiple loss components, each capturing different aspects of image quality, has become standard practice in image enhancement and generation tasks.

In order to train end-to-end, the image processing primitives themselves must be differentiable.

2.3.2 Differentiable Image Processing

Another key component of this strategy is the requirement for all operations to be differentiable so that the loss error can be propagated across the network and the weights accordingly adjusted. This is an active area of research in many applications. For example, systems such as LIIF [49] learn an implicit, differentiable and continuous image function to achieve 3D reconstruction or 2D super-resolution. NeRF [50] takes advantage of the differentiability of volume rendering operations to build a volumetric scene function and generate new 3D views of complex scenes. In other scenarios, the use of differentiable primitives is allowing the use of deep learning techniques to fields such as physics simulation [51].

The requirement for differentiability has led to innovative reformulations of traditional image processing operations. Classical operations like histogram equalization, bilateral filtering, and even complex geometric transformations have been reformulated in differentiable forms, enabling their integration into end-to-end learning frameworks. This trend toward differentiable programming has opened new possibilities for combining domain knowledge from traditional image processing with the learning capabilities of neural networks.

With these enablers established, we turn to supervision strategies that reduce reliance on paired data through self-supervision and synthetic degradation.

2.4 Self-Supervised Learning for Image Enhancement

2.4.1 Dataset Augmentation and Synthetic Degradation

Supervised learning, and in particular, supervised *deep learning* of image enhancements, requires the use of rich and comprehensive datasets that can provide the network with a vast variety of image pairs during the training process. Dataset construction is a slow and labor-intensive process whose results are sometimes difficult to extend beyond the scope of their original contributions. In practice, licensing constraints and dataset scope can limit direct use of resources such as [34, 3]. Some authors overcome the difficulty of sourcing high-quality datasets by using other training methods instead of supervised learning. For example, Hu et al. [4] use two sets of images rather than a stricter dataset of exact image pairs.

Even in cases where existing datasets can be readily applied, the high amount of weights to be trained in a deep learning network demands augmentation techniques to increase the variability of the original dataset and try to reduce overfitting. The use of ad-hoc augmentation methods for image-based networks is becoming more and more sophisticated. Consequently, specialized software libraries such as Albumentations [52] are now being created for that very purpose.

The evolution of data augmentation techniques has been crucial for the success of deep learning in computer vision. Beyond traditional augmentations like rotation, scaling, and color jittering, recent approaches have explored learned augmentation policies [53], adversarial augmentations, and even using generative models to create synthetic training data. The key insight is that augmentations should preserve the semantic content while introducing variations that help the model generalize better.

A frequent setup in supervised approaches involves the preparation of image pairs that combine images to be improved with reference ones [3, 34]. Because of the difficulties just mentioned, some authors have moved to use transformations or augmentation-only approaches, in what can be considered a type of self-supervised learning. For example, Alvarez et al. [54] create a performant blur segmentation model by algorithmically blurring certain areas of otherwise unblurred reference images. Antic et al. [55] also transform reference images using methods appropriate to the problem at hand, and use

that method for tasks such as super-resolution from microscopy videos or colorization of black & white photographs and videos.

The concept of "crappification" introduced by Antic et al. represents a powerful paradigm for self-supervised learning in image enhancement. By systematically degrading high-quality images in ways that mimic real-world degradations, researchers can create virtually unlimited training data without the need for manual annotation. This approach has been successfully applied to various restoration tasks including denoising, deblurring, super-resolution, and colorization. The key to success lies in accurately modeling the degradation process to match real-world scenarios.

Equipped with robust supervision and training practices, we now review recent modeling trends that improve quality and efficiency for enhancement.

2.5 Recent Modeling Trends for Enhancement

2.5.1 Transformer-Based Approaches

The success of Vision Transformers[1] (ViT) [56] has led to their application in image enhancement tasks. The self-attention mechanism in transformers allows for better modeling of long-range dependencies in images, which is particularly beneficial for global adjustments. SwinIR [57] demonstrated state-of-the-art performance in image restoration tasks using a hierarchical transformer architecture. The Swin Transformer's shifted window approach [58] provides an efficient way to handle high-resolution images while maintaining global context.

2.5.2 Diffusion Models for Image Enhancement

Diffusion models have emerged as a powerful generative approach for image enhancement. Unlike traditional enhancement methods that directly predict enhanced images, diffusion models learn to reverse a gradual noising process. SDEdit [11] showed how diffusion models can be used for image editing by adding noise to an input image and then denoising it with guidance. This approach allows for controlled editing while preserving the overall structure of the image. Palette [59] also demonstrated that diffusion models can be

[1] In this section, *transformer-based* refers specifically to vision backbones (e.g., ViT, Swin) applied to image restoration; general Transformer/LLM background is covered later in the Chapter.

used for various image-to-image translation tasks, including colorization, inpainting, uncropping, and JPEG restoration. The key advantage is their ability to generate high-quality, diverse outputs while maintaining fidelity to the input image.

2.5.3 Real-World Image Enhancement Datasets

The challenge of obtaining paired training data for image enhancement has led to innovative dataset creation methods. Recent work has focused on capturing real-world degradations rather than synthetic ones. For example, RealSR [60] introduced a dataset of real-world super-resolution pairs captured using different focal lengths, providing more realistic training data than traditional bicubic downsampling.

In the previous sections we have been implicitly considering methods for global enhancement. For better interactive workflows we also need to consider localized edits. In particular, we focus on segmentation as a way to bridge this gap, and consider text inputs as a suitable and intuitive way to control this process.

2.6 From Global to Local: Text-Grounded Segmentation for Editing

The emergence of vision-language models has opened new possibilities for text-driven image understanding and manipulation. This section reviews approaches that enable interaction with visual content through natural language, which forms the foundation for more intuitive photo editing interfaces.

2.6.1 Language-Driven Segmentation Tasks

Powerful and precise object segmentation can be leveraged for photo editing tasks, where the user selects the object they want to manipulate and the system applies the desired changes to it. The integration of language and vision has revolutionized semantic segmentation by enabling more flexible and intuitive ways to specify what to segment. This has led to several distinct but related tasks:

Open-Vocabulary Segmentation

CLIP [61] demonstrated that contrastive learning on large-scale image-text pairs could produce powerful visual representations aligned with natural language. This breakthrough enabled zero-shot classification and laid the groundwork for open-vocabulary segmentation – the ability to segment object categories not seen during training.

Building on CLIP's success, several approaches have emerged for open vocabulary segmentation. LSeg [62] pioneered language-driven semantic segmentation by training a transformer-based image encoder to align pixel embeddings with text embeddings from CLIP. This approach enables zero-shot generalization to unseen categories by leveraging the semantic structure of the text embedding space. CLIPSeg [63] extends this paradigm by supporting both text and image prompts for segmentation, adapting CLIP by adding a decoder that generates segmentation masks conditioned on multimodal queries.

More recent approaches like OpenSeg [64] and OVSeg [65] improve open vocabulary segmentation by better adapting frozen CLIP models. FC-CLIP [66] shows that even a single frozen convolutional CLIP model can achieve strong open-vocabulary segmentation with proper architectural design. These methods demonstrate the importance of preserving CLIP's zero-shot capabilities while adapting it for dense prediction tasks.

Referring Expression Segmentation

While open-vocabulary segmentation focuses on recognizing novel categories, referring expression segmentation tackles a different challenge: identifying specific objects based on natural language descriptions that may include attributes, spatial relationships, and context. This task is evaluated on benchmarks like RefCOCO / RefCOCO+ [67, 68], and RefCOCOg [69, 70], which contain images paired with referring expressions like "person in red shirt on the left" or "the smaller of the two dogs."

Early approaches to referring expression segmentation often used modular pipelines combining language parsing, visual feature extraction, and reasoning modules. More recent end-to-end approaches leverage transformer architectures to jointly process language and visual features. LAVT [71] achieves strong performance by progressively fusing language and visual features at multiple scales. RefFormer [72] uses a unified transformer architecture that treats referring expression segmentation as a direct set prediction problem.

The distinction between these tasks is important: open-vocabulary segmentation asks "can you segment any zebra?" while referring expression segmentation asks "can you segment the zebra on the left eating grass?" For interactive photo editing applications, referring expression capabilities are particularly valuable as they align with how users naturally describe their editing intentions.

It's worth noting that referring expression benchmarks like RefCOCO have an inherent limitation: while they evaluate the ability to understand complex descriptions, they're built on datasets with fixed object categories (RefCOCO uses COCO's 80 categories). This creates an interesting tension: models are tested on their ability to understand referring expressions, but only for a predefined set of object types. True open-vocabulary referring expression segmentation would require both understanding complex descriptions **and** generalizing to novel object categories, a challenge that current benchmarks don't fully address. Recent work has begun exploring this intersection, often by combining open-vocabulary detection models with referring expression understanding.

2.6.2 Foundation Models for Segmentation

The Segment Anything Model (SAM) [73] represents a paradigm shift in segmentation by introducing a promptable segmentation model trained on over 1 billion masks. SAM's class-agnostic approach and strong zero-shot generalization have made it a foundation for many subsequent works. However, SAM lacks semantic understanding and cannot directly handle text-based queries.

To address this limitation, several works combine SAM with language models. LISA [74] connects LLaVA with SAM through an embedding prompt, enabling reasoning-based segmentation. SEEM [75] proposes a more unified approach that segments "everything everywhere all at once" by jointly modeling visual and textual prompts. X-Decoder [76] takes a different approach by proposing a generalized decoding framework that unifies pixel-level, image-level, and language tasks.

2.6.3 Detection-Based Approaches for Text-Grounded Localization

Grounded language-image pretraining has shown remarkable success in open-vocabulary object detection. GLIP [77] unifies object detection and phrase

grounding by reformulating detection as a grounding problem, enabling strong zero-shot transfer to new object categories and domains. Grounding DINO [78] extends DINO with grounded pre-training, achieving state-of-the-art performance on open-set object detection.

Recent benchmarking efforts have revealed important insights about text-grounded localization. The Ref-L4 benchmark [79] demonstrates that while models like Qwen-VL [28, 29] and CogVLM [80] achieve impressive accuracy on RefCOCO benchmarks, significant labeling errors in these datasets (14-24%) have masked true model capabilities. When evaluated on cleaned datasets, these models show even stronger performance, with Qwen2-VL achieving 93.2% accuracy on RefCOCO. This suggests that current VLMs are more capable at spatial understanding than previously thought, particularly when given high-quality supervision.

These detection models can be combined with SAM for open-vocabulary segmentation through a two-stage pipeline: first detecting objects based on text descriptions, then segmenting them using SAM. While effective, this pipeline approach can be computationally expensive and may struggle with ambiguous object boundaries where detection and segmentation disagree.

2.6.4 The Rise of LLMs as Universal UIs

Large language models (LLMs) have had a rapid evolution since the Transformer architecture was introduced [81]. They have progressed through masked-language encoders (such as BERT [82]), sequence-to-sequence encoder-decoder models (T5 [83]), and auto-regressive decoders like the GPT family [15]. Today, they are primarily built as auto-regressive decoders and have culminated in general-purpose *closed* systems such as GPT-5 [84], GPT-4 [16], as well as *open* families like Llama [85, 18], Qwen [19], Gemma [20, 21, 22], or OpenAI's gpt-oss [86].

Progress has been driven by scale (in number of parameters, data, and compute), together with improved optimization and regularization.

Modern LLMs typically follow a two-stage training pipeline: (1) web-scale pre-training, and (2) post-training for usefulness and alignment via supervised fine-tuning and preference optimization. Public reports describe these stages for frontier models [16, 17, 18], and recent work explores reinforcement-learning post-training to boost reasoning ability [23]. These steps are designed

to improve instruction following, factuality, and robustness, allowing their use in interactive conversational applications.

LLMs have thus emerged as "universal UIs", where they can interpret natural-language goals, plan, and coordinate tools through standardized function-calling interfaces[2], turning free-form requests into structured and concerted actions.

Deployment has advanced rapidly as well. The use of sparsity through mixture-of-experts (MoEs) routing [87] allows for fast inference even at high scale. Parallel inference and attention-focused optimizations [88] alleviate the quadratic complexity on sequence length, and quantization methods make it possible to run LLMs on consumer hardware [89, 90].

LLMs can also be used for visual understanding. Combining a powerful image feature extractor with an auto-regressive LLM stage allows the language model to refer to visual and text inputs, and generate responses that take both into account. This is the foundation of VLMs.

2.6.5 VLMs as General-Purpose Tools

Vision-Language Models, or VLMs, incorporate a vision component to LLMs, providing them with the capability to accept both visual and text inputs for auto-regressive LLM text-generation phase. Thanks to the demonstrated generalization capabilities of LLMs, VLMs can be used for tasks such as visual question answering, captioning, synthetic data generation, detection, segmentation, and many more.

VLMs have shown impressive visual understanding capabilities. Models like LLaVA [25], BLIP-2 [91], and Flamingo [92] can answer complex questions about images but initially lacked pixel-level output capabilities.

Several works attempt to bridge this gap. PaliGemma [26] and its successors demonstrated that VLMs can be adapted for spatial tasks like object detection and segmentation by adding geometry-aware tokens to the vocabulary and training on diverse vision tasks. This approach requires extensive pre-training on millions of examples to learn spatial reasoning and careful architectural modifications to output spatial information.

[2]For example, the Model Context Protocol: https://huggingface.co/docs/hub/en/hf-mcp-server.

More recent VLMs have made significant advances in geometry-aware understanding. Qwen-VL and Qwen2-VL [28, 29] achieve state-of-the-art performance on referring expression comprehension benchmarks (RefCOCO, RefCOCO+, RefCOCOg), surpassing 90% accuracy through sophisticated architectural innovations. Qwen2-VL introduces Multimodal Rotary Position Embedding (M-RoPE), which decomposes positional information into temporal, height, and width components, enabling precise spatial reasoning for both images and videos. The model processes images at arbitrary resolutions dynamically, converting them into variable numbers of visual tokens, which is crucial for handling high-resolution photographs in editing applications.

Qwen3-VL [93] achieves even better spatial understanding, thanks to an improved version of M-RoPE called Interleaved-MRoPE. It is capable of judging object positions, viewpoint changes, and occlusion relationships.

Moondream [30] introduces *grounded reasoning*, a paradigm where the model can pause during inference to think step-by-step about spatial relationships and explicitly point at objects in images. This capability dramatically improves performance on tasks requiring accurate visual interpretation, such as counting objects or understanding complex spatial arrangements. By allowing the model to "ground" its reasoning with spatial positions when needed, Moondream addresses the confirmation bias issues that plague many VLMs [94], and provides an auditable reasoning trace – valuable properties for photo editing applications where accuracy and interpretability are crucial.

However, adapting VLMs for dense prediction presents challenges: significant architectural modifications are required, extensive spatial reasoning data is needed, and there's a risk of catastrophic forgetting of language capabilities. Our approach with CocoGold sidesteps these issues by leveraging diffusion models' inherent spatial understanding rather than teaching language models to reason about space.

Despite these challenges, VLMs are a fruitful and promising area of research because they demonstrate extraordinary generalization capabilities and can be adapted to multiple downstream tasks. Large language models (LLMs) are initially trained for *just* the auto-regressive text generation task, but they can be post-trained, fine-tuned, prompted and adapted to tackle a vast number of problems, including winning gold medals at the International Mathematical

Olympiad[3], assist in coding tasks[4], manipulate external tools with a formal API[5], and many more.

We now turn to diffusion models, highlighting how they can be repurposed from generation to dense prediction in service of photo editing.

2.7 Diffusion Models: From Generation to Dense Prediction and Segmentation

While diffusion models were initially developed for image generation, recent work has demonstrated their versatility for other vision tasks. This section explores how the rich representations learned by diffusion models can be repurposed for non-generative applications.

2.7.1 Diffusion Models for Dense Prediction

Marigold [14] pioneered the adaptation of pre-trained diffusion models for dense prediction tasks, specifically depth estimation. By reformulating depth prediction as an image-to-image translation task in the latent space of Stable Diffusion, Marigold demonstrated that minimal architectural modifications could unlock the model's spatial understanding capabilities. The key insight was to leverage the dual image conditioning mechanism, where the model learns to generate depth maps conditioned on RGB images.

This approach offers several advantages: it preserves the pre-trained model's rich visual representations, requires minimal architectural changes (only modifying the first convolutional layer), and achieves strong performance with relatively little task-specific training data. The success of Marigold suggests that diffusion models could be similarly adapted for other dense prediction tasks.

2.7.2 Leveraging Diffusion Features for Segmentation

Segdiff [95] showed that the diffusion process can be trained for tasks other than generation, and they achieved state-of-the-art results, at the time, for

[3] https://deepmind.google/discover/blog/advanced-version-of-gemini-with-deep-think-officially-achieves-gold-medal-standard-at-the-international-mathematical-olympiad/, https://x.com/alexwei_/status/1946477742855532918

[4] https://github.com/hesreallyhim/awesome-claude-code

[5] https://huggingface.co/docs/hub/en/hf-mcp-server

segmentation tasks. Their approach was to train an end-to-end specialized segmentation model from scratch, instead of relying on pre-trained backbones. While this is an early effort that showed the versatility of the diffusion training process, later works have focused on leveraging the rich representations of pre-trained models and their effective use af foundation models that can be adapted to computer vision tasks, despite being trained for generation.

ODISE [96], for instance, uses internal representations from text-to-image diffusion models for open-vocabulary panoptic segmentation, by cleverly combining a frozen diffusion model, a frozen CLIP discriminative model, and training an *implicit* captioner and a mask generator from the frozen representations. They show impressive generalization to unseen classes and open vocabulary queries through this carefully architected pipeline.

A particularly notable recent advancement is DiffDIS [97], which demonstrates that diffusion models can achieve state-of-the-art results in binary dichotomous image segmentation (DIS) [98], which is the task to identify category-agnostic foreground objects in images. Unlike methods that extract features from pre-trained models, DiffDIS trains the U-Net architecture within Stable Diffusion V2.1 for high-resolution, fine-grained segmentation. One of their key contributions is the use of a one-step denoising approach that dramatically reduces inference time while maintaining exceptional segmentation quality. In addition, they use an auxiliary edge detection step (used as an additional conditioning signal) that improves quality and helps overcome the probabilistic nature of diffusion without the need for ensembling. This work provides strong validation for using diffusion models in segmentation tasks, though it focuses on the dichotomous segmentation problem rather than on multi-class, text-grounded segmentation.

2.7.3 Fast Diffusion Sampling

While diffusion models traditionally require hundreds or thousands of denoising steps for high-quality generation, recent research has dramatically reduced this computational burden. These advances are particularly relevant for adapting diffusion models to real-time applications like photo editing.

Deterministic Sampling Methods

DDIM (Denoising Diffusion Implicit Models) [99] pioneered fast sampling by reformulating the diffusion process as a deterministic ODE rather than a

stochastic SDE. This allows for:

- High-quality generation with 50-100 steps instead of 1000
- Deterministic outputs given the same random seed
- Meaningful interpolation in latent space

Building on DDIM, DPM-Solver [100] and DPM-Solver++ [101] use higher-order ODE solvers to achieve comparable quality with just 20-25 steps. These methods have become standard in practical diffusion model deployments.

Trailing DDIM timesteps [102] modify timestep spacing and zero-terminal SNR at inference, improving performance in the low-step regime and enabling single-step inference when coupled with appropriate training.

Consistency and Distillation Acceleration

Recent approaches aim to reduce sampling to just 1-4 steps:

Progressive Distillation [103] trains a student model to match the output of a teacher model using twice as many steps, progressively reducing the required steps through multiple distillation rounds.

Consistency Models [104] learn to map any point along the diffusion trajectory directly to the final output, enabling single-step generation while maintaining the ability to trade off speed for quality with multiple steps.

LCM (Latent Consistency Models) [105] adapt consistency training to latent diffusion models like Stable Diffusion, achieving high-quality results in just 4-8 steps, demonstrating that complex vision tasks can be performed with minimal denoising steps.

LCM-LoRA [106] showed that LCM distillation can be performed using LoRA [107] adapters instead of fine-tuning the pre-trained model weights. Furthermore, LCM-LoRA adapters excel at generalization and can be used as universal accelerators for arbitrarily fine-tuned models, without additional training.

The universal nature of LCM-LoRA suggests similar acceleration techniques can be applied to fine-tuned models without full retraining.

Adversarial Diffusion Distillation (ADD) [108] combines teacher-student distillation with an adversarial loss to produce few-step (even single-step) generators with strong prompt adherence, e.g., SD-Turbo [109].

Latent Adversarial Diffusion Distillation (LADD) [110] performs ADD fully in latent space, enabling faster training and higher resolutions; recent single-step models such as FLUX.1 [schnell] [111] exemplify this recipe.

Task-Specific Optimizations

Non-generative tasks like depth estimation or segmentation can enjoy specific optimizations enabled by the nature of the task.

Marigold [14, 112] demonstrates that depth estimation can work effectively with just 4 denoising steps when the model is specifically fine-tuned for this setting. The key insight is that depth prediction requires less stochasticity than image generation.

DiffDIS [97], which uses a different approach for dichotomous segmentation, demonstrates that careful training based on the latest distillation techniques can enable single-step diffusion for segmentation tasks. DiffDIS, for example, builds upon Adversarial Diffusion Distillation, but without requiring a post-training distillation process. Instead, 1-step performance is baked in as part of the DiffDIS training recipe. The authors initialize the U-Net weights from the SD-Turbo model [109] (which was distilled from Stable Diffusion using ADD), instead of using the original Stable Diffusion weights. Furthermore, timesteps are hardcoded to 999 during training, forcing the model to always predict in a single step. The diffusion noise schedule, following previous literature, uses 1,000 steps, but the model is only shown samples from the last, fully-noised step.

Bringing these threads together, we synthesize their implications for practical photographic editing.

2.8 Implications for Photo Enhancement

The convergence of these technologies – open-vocabulary understanding, adaptable diffusion models, and combined visual and textual representations – creates new opportunities for photo enhancement systems. Text-grounded segmentation capabilities enable users to specify regions of interest naturally, while preserving non-destructive, parameter-based approaches for interpretable editing.

The key insight from our review is that different model architectures excel at different aspects of the problem:

- Parameter-prediction models demonstrate efficient, interpretable global enhancement
- Vision-language models like CLIP provide strong text-image alignment but require adaptation for spatial tasks
- Foundation models like SAM excel at segmentation but lack semantic understanding
- Diffusion models offer rich visual-linguistic representations that can be efficiently adapted for spatial tasks

Next Chapters. The next chapters detail two complementary directions aligned with the conclusions from this review and our research goals.

- Chapter 3: a parameter-prediction approach to automatic global enhancement that emphasizes resolution independence and interpretability.
- Chapter 4: adapting diffusion models for text-grounded segmentation, enabling localized edits based on natural-language descriptions.

Section 5.3 discusses COCONET, a combined pipeline that applies interpretable parameters locally using text-grounded segmentation.

Chapter 3

FilterNet: Self-Supervised Learning for Photo Enhancement

3.1 Introduction

This chapter introduces FilterNet, a model that predicts parameters of photographic operations to perform technically meaningful global edits on arbitrary-resolution images.

3.1.1 Overview

Our design is guided by the requirements established in Chapter 1 for *technical photo editing*:

- **Resolution independence**: predicted parameters can be applied to the full-resolution image, minimizing prediction memory consumption, avoiding tiling artifacts and resizing limits.
- **Interpretability and control**: outputs correspond to familiar photographic adjustments that users can inspect and refine, rather than black-box finalized images.
- **Efficiency**: inference is lightweight enough for interactive use and mobile deployment.
- **Robustness**: edits improve common capture issues (exposure, color, contrast) while preserving natural appearance, especially for skin tones.

Method at a glance. FilterNet maps an input image to a compact vector of edit parameters that control a differentiable set of photographic operators (exposure, contrast, brightness, white balance, highlights, and shadows). A

differentiable renderer applies these operators during training so the model learns enhancement parameters by backpropagation. At inference time, the predicted parameters can be applied to images of arbitrary size using non-differentiable counterparts of the same filters.

Training without paired data. To avoid the need for expert-edited pairs, we adopt a self-supervised strategy. We synthesize degradations from high-quality images and train FilterNet to predict edit parameters that undo those degradations. A multi-component objective combines pixel-level fidelity with perceptual and style loss components to stabilize learning and encourage natural-looking results. This chapter focuses on the method while Chapter 5 summarizes quantitative and qualitative results, metrics and ablations.

Contextualization in our research. FilterNet addresses the first step towards our vision: automatic, global, interpretable enhancements based on parameterized *edits* instead of generation. Chapter 4 introduces CocoGold, which addresses segmentation guided by text inputs. We combine both components in Section 5.3 to demonstrate text-guided local enhancement: FilterNet supplies the adjustments, while CocoGold indicates where to apply them.

3.1.2 From Heuristics to Interpretable Edits

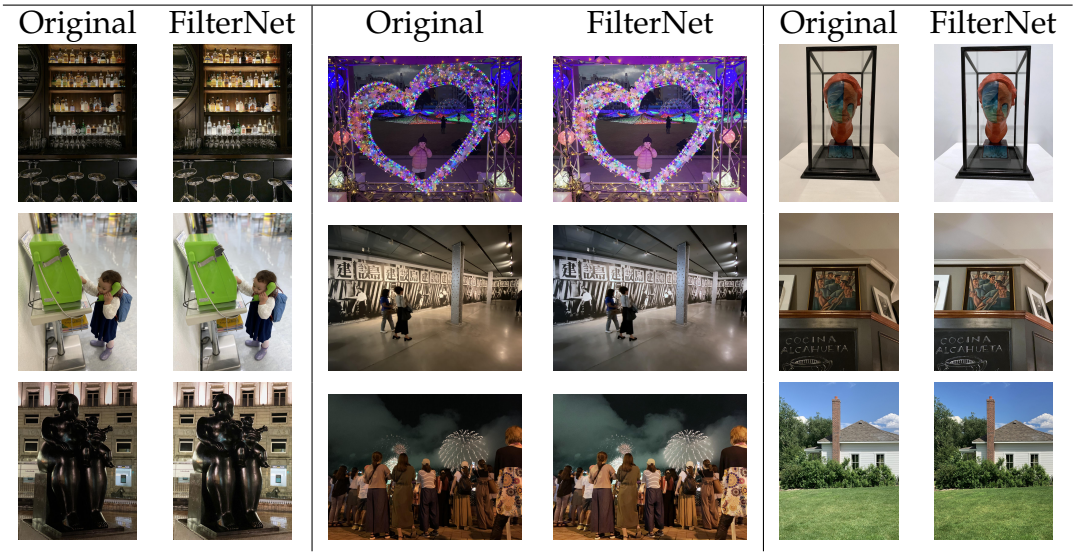
Classical baselines are summarized in Section 2.2. In brief, heuristic methods like histogram equalization and CLAHE operate on luminance statistics and can introduce artifacts in highlights and skin tones. FilterNet takes a different stance: it predicts a compact set of photographic parameters that jointly govern tone, color, and contrast, producing edits that are both interpretable and robust across content.

Deep learning methods use a different approach: they try to capture the essence of professional photographers' work to achieve one-touch adjustments that take into account the characteristics of the image instead of applying fixed rules or compensate with heuristics. Usually, these deep learning methods are trained on relatively low-resolution images and are difficult to scale to larger resolutions, which is especially important for photography applications. Resolution-agnostic approaches started to emerge to address this problem. This is the case of LIIF [49], which learns an implicit function representation that can be rendered at any resolution. LIIF is intended for single-image super-resolution and not editing; in addition, inference is costly at high resolution.

Exposure [4] uses adversarial components that typically complicate training and tuning; more importantly, it trains in a multi-step sequence and relies on paired RAW photos retouched by experts, while FilterNet is self-supervised and predicts all parameters in one forward pass, as we'll see.

To better ground this discussion, Figure 3.1 shows representative input/output pairs from real-life unmodified photos captured with mobile phones and consumer DSLRs, with various subjects and lighting conditions. FilterNet is able to improve tonality, exposure, and color temperature while being respectful of human features and avoiding opinionated stylistic modifications.

FIGURE 3.1: Original straight-out-of-camera photos (odd columns) and optical adjustments proposed by FilterNet (even columns). Results improve tonality, exposure, and color temperature while being respectful of human features and avoiding extreme stylistic changes.[1]



FilterNet proposes a method to learn editing parameters that can improve the *technical* quality of digital photos, achieving the following contributions:

- An end-to-end system to enhance photographs of arbitrary resolutions. The model is able to propose enhancements to six different factors, without imposing a particular visual style and avoiding artifacts or unnatural results. The six areas improved are: color temperature, exposure, brightness, contrast, shadows and highlights. High-resolution enhancement is made possible by the fact that our model is configured to learn the intensities of the filters that best improve the images presented as inputs to the network. Training is performed using downscaled images, but the results are applied to full-resolution photos when the model is deployed.

- The validation that transfer learning can be applied to the photography enhancement task in a differentiable backpropagation setting. Training stability is improved with respect to alternative generative approaches based on reinforcement learning or adversarial networks. Furthermore, training requires modest computing resources and deployment is efficient even in constrained environments, such as mobile phones.
- A self-supervised data preparation method where training pairs are synthetically generated from high-quality photos using a degradation process. This provides great variety and generalization capabilities.
- The development of high-quality differential image processing filters that can reproduce the operations of image editing primitives implemented in commercial applications. These differentiable filters can be applied to image batches in parallel, during the forward training pass, in order to produce rendered versions of the inputs. Because filters are differentiable, their gradients can be computed for loss attribution using backpropagation.
- The analysis of factors and methods that contribute to improved enhancement quality and more stable training. Loss components computed from feature-extraction CNNs, the use of a 10-bit linear sRGB colorspace for image processing, sensible weight initialization schemes, and image preparation methods with varying degrees of transformation are measured. Experimental results and metrics are provided in Chapter 5.

Our research on FilterNet has also been validated by its use in a well-known iPhone photography application [2]. Users can instantly get automatic enhancement results by simply tapping a button. Furthermore, they are able to see how the result is composed of various amounts of the filters we mentioned, so they can learn from the suggestions or tweak those values to their liking. This feature uses an on-device FilterNet model to minimize latency and ensure privacy. It supports all current iPhones and takes less than half a second to run, including rendering at full-resolution, on all devices.

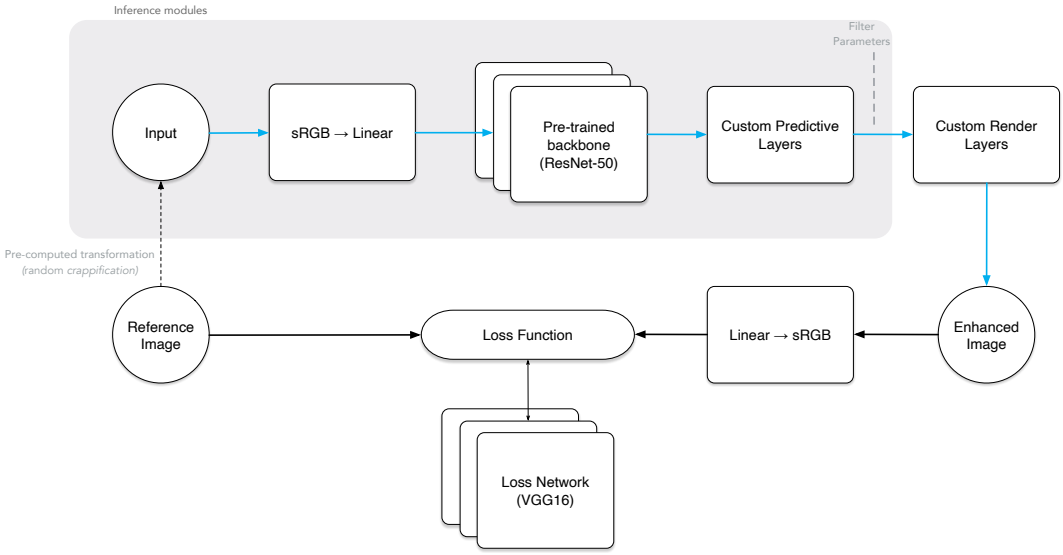


FIGURE 3.2: FilterNet overview diagram. The gray background area shows the modules used at inference time, where the network estimates the values or intensities of the enhancement filters to apply, as discussed in Section 3.7. In order to train the model, custom filter rendering layers are used to apply the enhancements (3.5), and the results are compared against the reference high-quality images (3.4, 3.6). The training process is based on a single dataset of high-quality photographs from which bad pictures are created by applying random transformations using the process described in Section 3.3.

3.2 Task Definition and Base Architecture

FilterNet is designed to learn the task of enhancing photos by applying a fixed number of global filters with varying degrees of intensity. The outcome we are interested in is the set of filter intensities that improve an input image. The improvement is measured as to how close the result is with respect to a reference, high-quality image. In order to train the model, the network generates improved versions of the input images and compares them against their high-quality counterparts. That comparison yields a loss figure used in a backward pass to update the weights of the trainable network layers. As Figure 3.2 shows, the main components of the architecture are:

- A method to obtain as many *bad* quality images as we need, from the high-quality dataset we prepared. This allows the use of the high-quality images in a self-supervised fashion. Section 3.3 provides more details about data preparation.
- A set of differentiable parallel filters carefully crafted to replicate typical operations performed in professional photo editing tools, as explained in Section 3.5. Filters are applied in a linear RGB color space using 10-bit precision.
- A loss function that compares the images enhanced by the network with the high-quality reference images. As discussed in Section 3.6, we use a combination of metrics to create a robust loss function.
- A backbone network, previously trained on ImageNet, used in a transfer-learning fashion to leverage its feature-extraction capabilities on our relatively small dataset.

Figure 3.2 also depicts the modules that are used at *inference* time, as opposed to the ones used during *training*. In training mode, the network needs to render the images in order to compare them against the high-quality references. However, when the model is deployed, the rendering operation is not required: the model produces the values (or intensities) of the filters to be applied. This allows the model to enhance images of any resolution, despite training it with downscaled samples. At inference time, the filter parameters are estimated from a low-resolution version of the input image. Then the editing application uses those values to render the high-resolution original, making the process suitable for end-user photography applications.

[2] https://9to5mac.com/2020/06/03/camera-iphone-ipad-update-magic-ml/

The task was modeled after a transfer-learning architecture. We chose to focus on residual networks of the ResNet family because their generalization to other tasks and domains has been well established [45, 43, 44]. FilterNet is built on the ResNet-50 variant, which in addition to having shown excellent transfer-learning performance, also represents a reasonable compromise between quality, inference time, and model size. The ResNet backbone we used was pre-trained on ImageNet [42] on a classification task. As other authors have shown, ResNet provides excellent generalized feature extraction capabilities without having to undergo a very lengthy and data-intensive training from scratch.

With FilterNet, we validate self-supervised learning in a transfer-learning setting for high-resolution image enhancement. We concentrate on the ResNet networks for our experiments, in order to attain some homogeneity and consistency that would allow us to extract general conclusions about the overall effectiveness of the solution and the impact of its constituent components. Other backbone architectures are possible, but we didn't thoroughly explore them. As a comparison data point, Section 5.1.5 shows the use of a much smaller SqueezeNet network in the context of studying different size-performance trade-offs.

As Section 5.1 describes, our experiments show that both ResNet-34 and ResNet-50 have a similar conceptual training behaviour. The main differences are that ResNet-50 produces better quality (all other conditions being equal), but trains slower. We used ResNet-34 to iterate and experiment faster, and then validated on the final ResNet-50 models we chose for deployment. ResNet-101 was discarded because of size and latency considerations, as our model was intended to run in mobile devices.

All code was written using PyTorch [113] and fastai [114].

3.3 Dataset and Data Preparation

Since our ultimate goal was to create a general-purpose photo enhancement model, we needed appropriate good quality data with sufficient photographic merit. We also needed it to cover most of the topics users would take pictures of in their daily lives. In order to comply with those requirements, we compiled a new dataset from photos downloaded from public sources.

One important reason to curate a new dataset was that many datasets frequently used for computer vision tasks lack a focus on technical photographic merit. Instead, they are meant for tasks such as object detection or segmentation, where photographic quality or color accuracy are less critical than subject variety. This is the case even for Google's extensive Open Images Dataset[3]. Despite its vast scale, it was excluded because of the difficulty of searching for high-quality images from the point of view of technical photographic merit. As it was briefly touched-upon in Chapter 2, we did not consider the use of the MIT-Adobe FiveK Dataset [34] either, because of its restrictive license. Modern datasets such as LAION-5B [115] could potentially be used, but our work predates this collection, and it would require extensive filtering and culling of images that are representative of real photographic conditions.

In order to gather a representative set of pictures, we wrote a script to download 7,200 images across 16 different categories using keyword-based search. After a deduplication step (some images were valid results for different search queries), we wrote an app to aid us in culling through the images to create a relatively consistent set. For example, some images had been converted to black and white or adopted a heavily desaturated look, while others used vibrant and very saturated colors. While significant desaturation is a perfectly valid artistic choice, our purpose was to focus on technical enhancement and avoid creating an opinionated model that tried to enforce a particular style vision. In addition, having similar photos edited in radically different directions could make the model harder to train or achieve a dull, intermediate look as a compromise among all edit styles. We used the app to review all photos and manually exclude extreme edits and unrepresentative images such as illustrations or vector art, ending up with a dataset of 4,885 high-quality photos, correctly exposed and with natural-looking colors. Those images are representative of 16 particular topics, including subjects such as landscape, architecture, portraits, or group photos, but the topics themselves were not used after collection.

All images were downloaded at a resolution of 1280 pixels in the longest dimension, which is enough to apply editing filters and obtain results visually similar to the ones that would have been obtained on higher-resolution photographs.

Training images were obtained by applying random transformations to downloaded photos, making it possible to create a virtually unlimited supply of

[3] https://storage.googleapis.com/openimages/web/index.html

TABLE 3.1: Custom predictive layers, appended after the ResNet-50 backbone.

Layer Type	Hyperparameters	Output
AdaptiveAvgPool2d		$512 \times 1 \times 1$
AdaptiveMaxPool2d		$512 \times 1 \times 1$
Flatten		1024
BatchNorm1d	$\epsilon = 10^{-5}$	1024
Dropout	$p = 0.25$	1024
Linear		512
Activation	ReLU	512
BatchNorm1d	$\epsilon = 10^{-5}$	512
Dropout	$p = 0.25$	512
Linear		512
Activation	Sigmoid in $(-1, 1)$	6
Maskout	$p = 0.1$	6

training data from the relatively small dataset we collected. In general, we performed the transformation process prior to training for performance and reproducibility considerations. Factors that affect the characteristics of the training dataset and therefore have an impact on training include:

- The set of filters used.
- The range of variation allowed for each filter.
- The total range of variation allowed, considering all filters simultaneously applied to an image.
- The number of transformation runs applied to the original set, which translates to the number of training samples to use.

For simplicity, we did not explore combinations of filters that tend to co-occur naturally; or conversely, we did not prevent transformations that may engage changes that go in opposite directions. We plan to address this line of work as part of our future work on user-intent predictions (Section 6.2).

Experiments that explore these factors will be further discussed in Section 5.1.

3.4 Custom Predictive Head

Table 3.1 expands on the *Custom Predictive Layers* depicted in Figure 3.2, showing the layers appended after the ResNet-50 backbone. The purpose of this custom model *head*, which replaces the standard classification head the ResNet architecture was designed for, is to learn the parameters of the 6 filters that

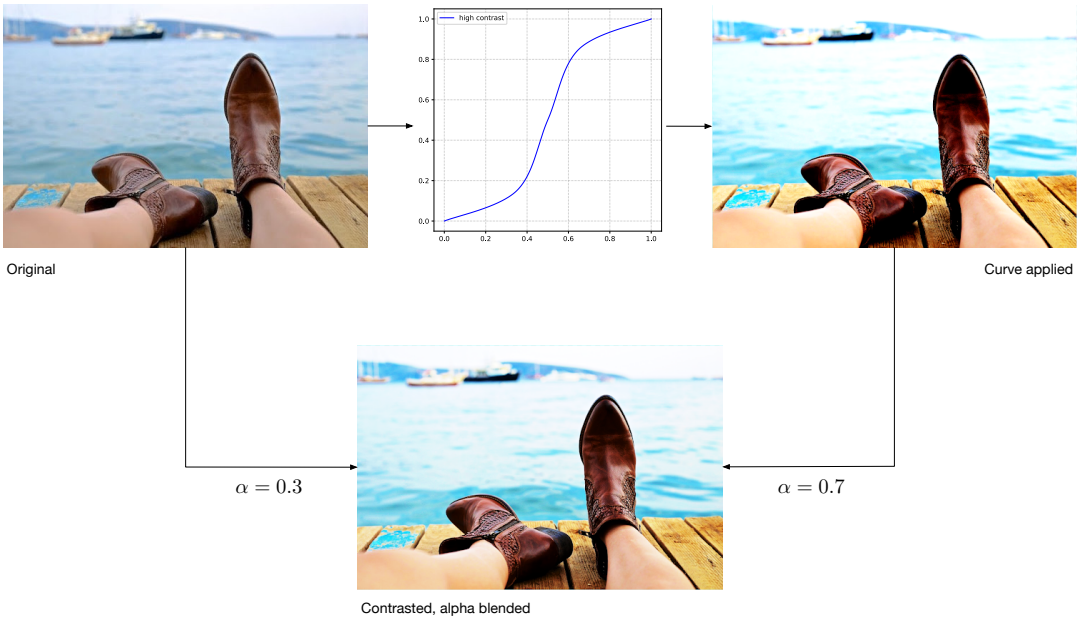


FIGURE 3.3: Applying contrast with 0.7 intensity. A contrasted version of the image is produced by filtering through a high-contrast tone curve, and the result is interpolated (or *alpha-blended*) with the original.

are rendered immediately afterwards. All the filters are designed to operate in the normalized $(-1, 1)$ range, so we chose a sigmoid curve (translated to that range) as our final activation layer. The $(-1, 1)$ range is denormalized where necessary to the natural filter units when it makes sense. Such is the case of the color temperature filter, as we will comment in Section 5.1.2.

The *Maskout* layer refers to the use of a regularization technique we devised that is very similar to dropout [116], [117]. The goal is to randomly disable the output of some of the predicted filters, forcing the network to compensate by using other filter combinations. The difference with respect to the usual dropout implementation is that we do not scale the outputs by factor $1/(1-p)$, with p being the probability of dropout. We want to randomly exclude filters to encourage the model to explore others, but we don't want to increase the intensity of the filters we didn't exclude. The output of the predictive layer is interpreted as a filter intensity amount, and those semantics would be invalidated if we scaled the values unaffected by the dropout.

3.5 Filter Rendering

The filters our model learns are exposure, brightness, contrast, shadows, highlights, and color temperature. All filters are applied globally, but shadows and

highlights are designed to operate on the dark and light areas of the histogram. These filters represent an adequate set of operations that photographers may want to apply to improve the optical characteristics of a photo before they attempt any further artistic-oriented edits. Even though there is some overlap in the nature of the changes they perform (a high brightness, for example, can be correlated with a high exposure value), the selection is still representative of the basic toolbox most editing applications provide. Different filter combinations may lead to satisfactory results. Since the training process does not use information about the transformations applied to originals, we do not impose a particular preference to the system on the combinations that it can learn.

Image enhancement operations can be implemented in very different ways, producing different results. For example, contrast can be computed by interpolating the original image with a gray image with the same mean luminosity as the original. Interpolation-based methods can be a convenient abstraction to represent multiple enhancement operations, and they can be found in implementations such as Python's Pillow library[4]. On the other hand, OpenCV uses a multiplication factor to compute contrast[5], and this is the same approach inherited by the Albumentations library [52]. Unfortunately, these simple methods are *not* usually found in professional tools, because they tend to produce extreme and unpleasant results for high-intensity variations. Photography applications model their editing tools in such a way that results are pleasing and can be controlled using fine-grained UI controls. Adobe Photoshop, for instance, decided years ago to move away from a mathematical formula to a method better suited for photographers[6].

In our case, most of our filter implementations are based on the concept of a *tone curve*. As Figure 3.3 shows, a high-contrast tone curve is applied to the original image to generate a contrasted version, and the result is alpha-blended with the original to modulate the intensity of the transformation. Tone curves are generated from a short number of control points using Catmull-Rom splines [118]. Usually, a single filter consists of two different tone curves, as depicted in Figure 3.4: one of the curves is applied to positive values of the transformation (high contrast, for instance), while the other curve in the pair is applied to negative values of the transformation. These curve

[4]https://python-pillow.org
[5]https://docs.opencv.org/3.4/d3/dc1/tutorial_basic_linear_transform.html
[6]https://www.photoshopessentials.com/photo-editing/brightness-contrast/

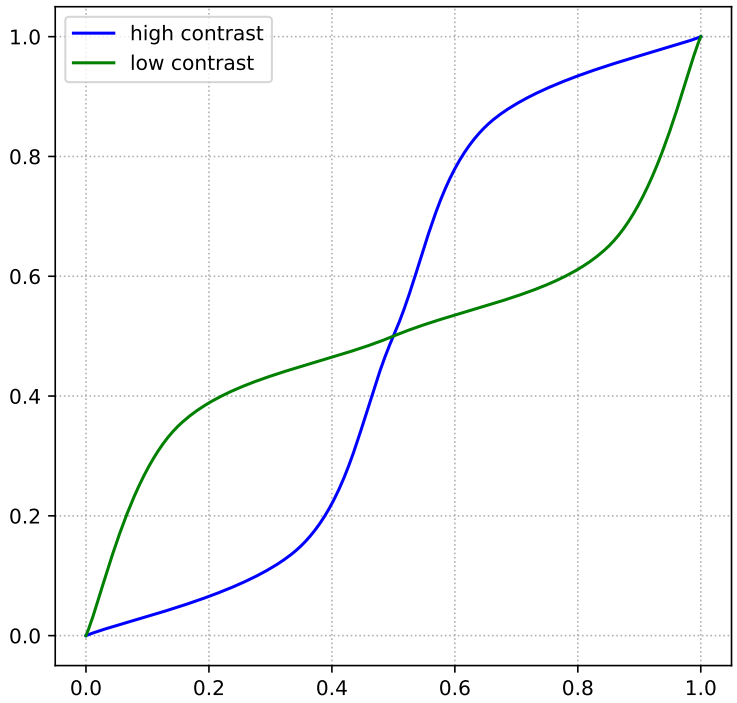


FIGURE 3.4: Curve pairs to increase / decrease contrast.

pairs are sometimes symmetrical, but they do not need to. Because tone-transformed images are blended with the originals, these transformations are essentially not invertible, in the sense that the original image cannot be easily recovered from the transformed version even if we knew the intensity of the transformation. A naïve regression-only approach trained to learn applied transformations would not suffice to recover the untransformed image, as we illustrate in Fig. 5.5.

For optimization and parallelization reasons, we pre-computed lookup tables for all the curve pairs used in our filters, using 10-bit precision per channel instead of the usual 8-bit precision that is commonplace in computer vision tasks. We perform all computations in *linear RGB* color space, instead of sRGB. The use of 10-bit linear RGB color exactly matches the format used in professional rendering pipelines, like the one used in our iOS app. This ensures that filters rendered while training match their non-differentiable production counterparts.

More importantly, colors in sRGB are encoded using a non-linear transfer function, whereas linear RGB encodes pixels using values proportional to physical light intensity and are therefore suitable for mathematical operations. Note, however, that despite using 10-bit precision, our method does not require input images in the RAW format. Both the FiveK paper [34] and

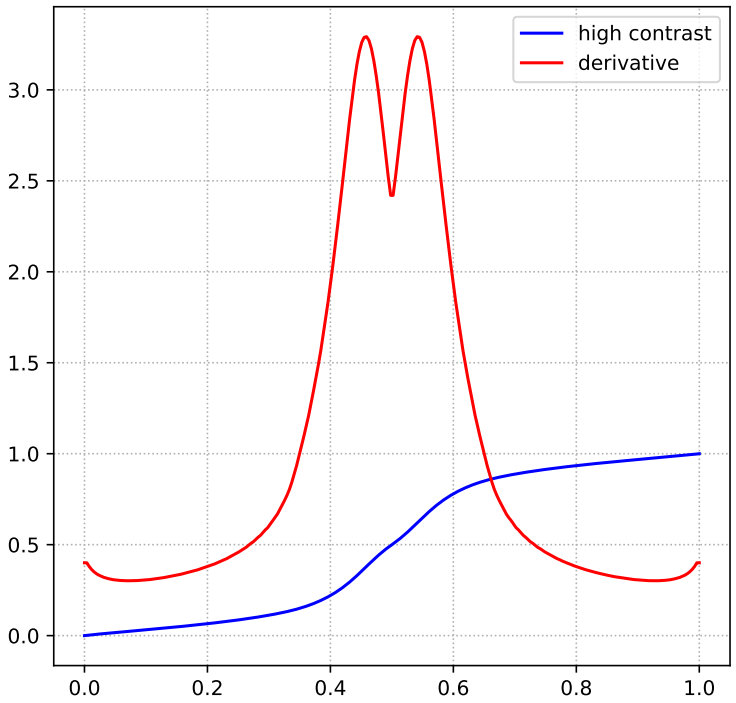


FIGURE 3.5: Tone curve and corresponding numerical derivative.

Exposure [4] use RAW files for input, but this limits the applicability of the methods. We use standard sRGB JPEG files as our input because that is the most usual real-life scenario, but perform all calculations with a precision of 10-bit per channel after converting colors to the linear colorspace. Linear to sRGB and sRGB to linear conversions are performed following the piecewise transfer function described in the specification [119].

Because tone curves are applied in custom network layers, the transformations need to be differentiable in order for the loss figure to be propagated and attributed correctly. Just as the curves are applied using lookup tables, the same approach was adopted for their gradients. Figure 3.5 shows the representation of a tone curve and its numerically-computed differentiation table.

Not all filters are implemented using the tone curve method. In the case of *exposure*, we simply apply the mathematical definition, computing $2^{ev}c$, where ev is the amount of increase or decrease in exposure and c is the original color intensity, per channel.

All filters in the network are allowed to vary in the $(-1, 1)$ interval. In the case of tone-curve filters (brightness, contrast, highlights, shadows), the absolute value is interpreted as the opacity of the transformed layer. Negative values

use the *low* curve (decreasing the corresponding parameter), while positive values use the *high* curve of the curve pair. In the case of color temperature, we pre-compute a 3×3 gain matrix for each 10K step between 2700K and 15000K, and map the filter variation range $(-1, 1)$ to the kelvin range $(15000, 2700)$. The gain matrix associated with the neutral temperature, considered to be 6500K, is the identity matrix I_3. Gains for different temperatures transform the original (R, G, B) triplet into a new triplet by matrix multiplication.

3.6 Loss Function

FilterNet is designed to produce enhanced images starting from randomly-transformed pictures derived from a set of high-quality references. To compare the effectiveness of the model, we need to measure the difference between FilterNet's output and the reference pictures. Following previous research [46, 55], we created a loss function that uses a pre-trained image network to extract features for comparison, instead of just computing the pixel difference between the images. As shown in Figure 3.6, the loss function is made of a combination of the following factors:

- A measure of the difference between the image pixels, ℓ_P.
- The difference between features extracted from both images by a VGG-16 model, ℓ_F.
- The difference between Gram matrices computed from the features extracted by the VGG in the previous step, ℓ_G.

The Gram matrix loss component, ℓ_G, captures style and texture information following the formulation of Gatys et al. [47]. Given feature maps with C channels and spatial dimensions $H \times W$, the Gram matrix $G \in \mathbb{R}^{C \times C}$ is obtained by reshaping the feature tensor into $F \in \mathbb{R}^{C \times HW}$ and computing $G = FF^\top$. Each entry Gij represents the correlation between feature channels i and j across all spatial positions. By discarding explicit spatial structure while preserving inter-channel correlations, the Gram matrix encodes textural patterns, color statistics, and stylistic characteristics that are invariant to spatial arrangement. Minimizing the discrepancy between Gram matrices encourages the enhanced images to preserve natural texture characteristics and to suppress artifacts that could appear visually artificial. Computing Gram matrices from multiple VGG layers (`relu3_3`, `relu4_3`, and `relu5_3`)

allows the model to capture style information at progressively higher levels of abstraction.

In all cases, the base difference metric was mean L1 distance, which was applied to each of the items above. Therefore, the individual components of the loss function can be conceptually summarized as follows:

$$\ell_P = \ell_{BASE}(\hat{\boldsymbol{x}}, \boldsymbol{y})$$

$$\ell_F = \ell_{BASE}(F(\hat{\boldsymbol{x}}), F(\boldsymbol{y}))$$

$$\ell_G = \ell_{BASE}(Gram(F(\hat{\boldsymbol{x}})), Gram(F(\boldsymbol{y})))$$

Where ℓ_{BASE} is the L1 distance between the corresponding matrices, averaged, $F(\cdot)$ is the output of the last three activation layers in the VGG network (usually denoted as `relu3_3`, `relu4_3`, and `relu5_3` in VGG diagrams), and $Gram(\cdot)$ is the gram matrix of the supplied matrix argument.

The combined loss function can be expressed as:

$$\mathcal{L} = w_p \ell_p + w_f \ell_f + w_g \ell_g$$

or simply

$$\mathcal{L} = \boldsymbol{W_L} \cdot \boldsymbol{L}^{\boldsymbol{T}}$$

where $\boldsymbol{W_L} = (w_p, w_f, w_g)$ are training hyperparameters that represent the contribution of each loss component, and $\boldsymbol{L} = (\ell_p, \ell_f, \ell_g)$.

Note that the loss function is computed between sRGB images, and not in linear mode. There are two reasons for this. First, the input to the loss network is expected to be an sRGB image, because that is how the original VGG model was pre-trained. Second, quality metrics and loss measures are, to the best of our knowledge, always computed between sRGB images in the literature, so we do the same for consistency. That is why Figure 3.2 shows a linear to sRGB transformation block before the loss function is applied.

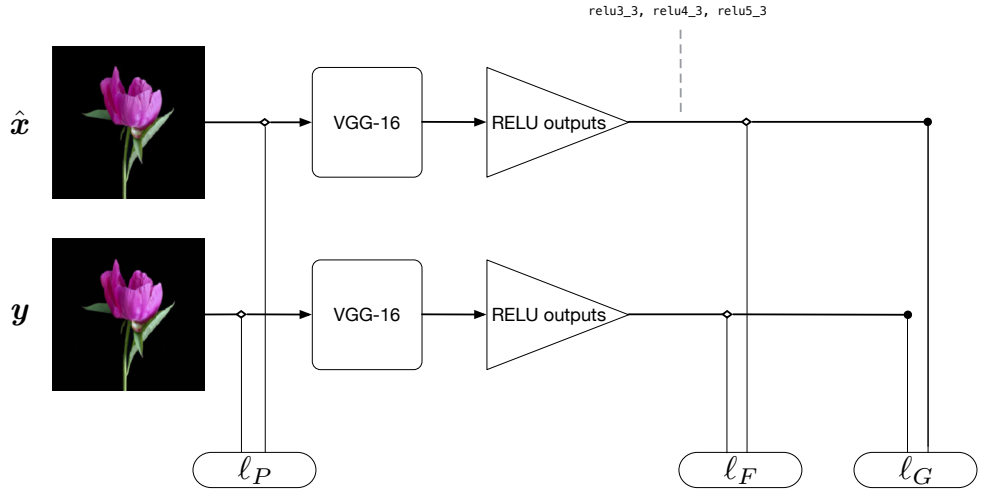


FIGURE 3.6: Components of the loss function. Outputs from the VGG network are sampled from the last three activation layers that precede maxpooling layers.

3.7 Final System Inference

Figure 3.7 shows a high-level overview of how the model works when deployed for inference; for example, inside a mobile app. The user captures a photo that may have been taken in less than ideal lighting or exposure situations, and uses the enhancement feature to try to improve it. The model was trained using images of just 299×299, so the first step in order to get a prediction is to downscale the image to the same size. Based on the input thumbnail, the model estimates the values of the parameters of the 6 editing filters we trained the system on. The six filters are estimated at once in a single inference run (and not incrementally on top of previous results), and the predicted values represent the best combination – as estimated by the model – that produces a pleasing enhancement, without imposing opinionated styles or unnatural-looking variations. The application gets the values of the six filter parameters, and renders the original, full-resolution image using the in-app filters, whose implementations match the differentiable implementations used for training. The image displayed in Figure 3.7 was taken indoors in a museum in dim light. We can observe that the model recommends increasing the brightness and exposure, decreasing the shadows and adjusting the white balance; while contrast and highlights remain essentially untouched in this case.

Figure 3.2 shows the inference pipeline in relationship with the training pipeline.

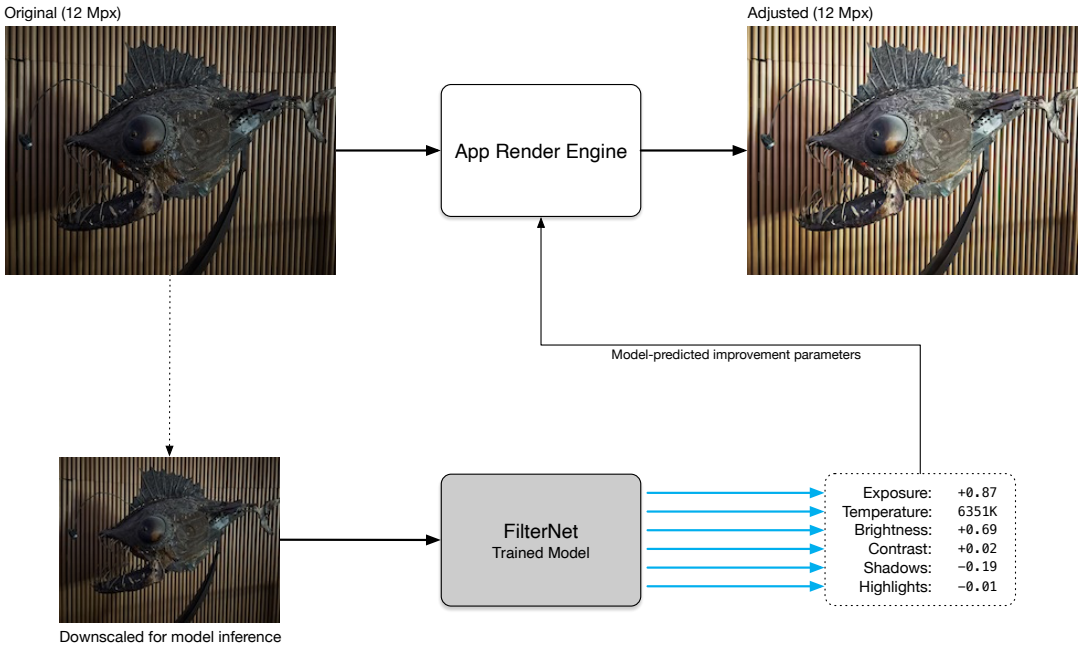


FIGURE 3.7: On-device FilterNet inference, as deployed in a mobile app. The trained model predicts 6 different improvement parameters using a small thumbnail as input. The predicted filter parameters are applied to the original high-resolution photo in a non-destructive way, producing an adjusted version of the image that the user can keep editing. In this example, the photo was taken indoors in a museum with dim lighting, and FilterNet created a combination of filter parameters to increase exposure and decrease shadows.

As noted in Section 3.4, multiple parameter combinations can produce similar looks and filter effects overlap and correlate. We train without supervision on the applied transformations, minimizing the distance to high-quality references so the network learns to use the six filters jointly to bring degraded inputs closer to them, irrespective of the final combination.

Sections 3.2 through 3.6 provided details on the dataset, model architecture and training process that were used to train the system whose behaviour we just described.

3.8 Summary

This chapter presented FilterNet, a self-supervised model that predicts parameters of photographic operators to perform technically meaningful global edits at arbitrary resolution. By learning to output interpretable controls (exposure, brightness, contrast, shadows, highlights, and color temperature)

and applying them through production-matched, differentiable operators in 10-bit linear RGB, FilterNet delivers resolution-independent, non-destructive enhancements that users can inspect and refine. Training is framed as a degrade-and-restore problem with pixel, perceptual, and style (Gram) losses, yielding stable learning and natural-looking results. We also described a lightweight inference path that predicts all parameters in a single pass on a thumbnail, and applies them to the original image, enabling efficient mobile deployment.

Key contributions of this work include:

1. A parameter-prediction approach to photo enhancement that preserves interpretability and supports resolution-independent application at full image size.
2. A differentiable filter stack that mirrors professional editing tools, implemented in 10-bit linear RGB to closely match production rendering while remaining trainable end-to-end.
3. A self-supervised training recipe that synthesizes degradations from high-quality images and combines pixel, perceptual, and style losses for stable optimization without paired expert edits or RAW inputs.
4. A compact, single-pass inference procedure suitable for mobile deployment, demonstrated with on-device application and practical latency.
5. An analysis of design choices (operator parameterization, loss composition, precision, and color space) and ablations (with details in Chapter 5) that inform robust, high-quality enhancement.

FilterNet is designed to operate globally on images. The next chapter (Chapter 4) introduces CocoGold, a system that repurposes diffusion models for text-grounded segmentation. Combined with FilterNet's interpretable parameters, this enables the localized, instruction-driven editing pipeline evaluated later in Section 5.3.

Chapter 4

CocoGold: Text-Grounded Segmentation with Diffusion

4.1 Introduction

In the previous chapters, we presented FilterNet (Chapter 3), a system that successfully predicts global filter parameters for arbitrary-resolution photo enhancement. While FilterNet demonstrates the potential of neural networks to automate complex photo editing tasks, it operates as a fully automatic system that applies uniform adjustments across the entire image. This global approach, though powerful, does not allow users to express specific editing intentions or to apply different treatments to different regions of an image.

To start addressing these limitations, we will focus on the segmentation task, as a stepping stone towards our vision of text-driven, localized photo enhancement, where users can specify editing intentions like "brighten the person's face" or "increase contrast in the sky".

In order for segmentation to become part of a photo editing system with such a natural, text-based UI, it needs to be grounded on text inputs. Our State of the Art review (Chapter 2) revealed a convergence of technologies that could enable more flexible, instruction-based photo editing: vision-language models [61, 25, 26] provide text-image alignment, foundation models like SAM [73] excel at segmentation, and diffusion models [8] offer rich visual-linguistic representations. However, these approaches face challenges when adapted for text-grounded segmentation tasks. Detection-based pipelines, like Grounding DINO [78], are computationally expensive, require various components to be combined and can be error-prone. In addition, they must be further combined with segmentation components (as Grounded SAM does) to extract detailed segmentation masks to act upon. Vision-Language Models

[25], unless already pre-trained for segmentation and spatial-aware tasks, require careful fine-tuning to be adapted for the segmentation task, and have intrinsic expressive limitations in terms of the restorative capacity that can be achieved with discrete output tokens.

In this chapter we present CocoGold, a novel approach that repurposes pre-trained diffusion models for text-grounded object segmentation. By leveraging the insights from Marigold's successful adaptation of diffusion models for depth estimation [14], we demonstrate that these models can be efficiently adapted for identifying and segmenting objects specified through natural language. This capability represents a crucial building block toward our vision of photo enhancement based on natural-language instructions.

4.2 Motivation and Key Insights

The development of CocoGold is motivated by several observations about the current landscape of computer vision and photo editing:

4.2.1 The Segmentation Bottleneck

While FilterNet can predict filter parameters for enhancement, applying these edits selectively requires accurate identification of image regions. Traditional semantic segmentation approaches are limited to predefined categories and struggle with the flexible nature of user instructions. Users don't think in terms of fixed class vocabularies – they express intentions using natural, varied language that may refer to objects, attributes, or even abstract concepts.

Understanding the Task Landscape

The attempt to connect segmentation with language has originated various segmentation tasks that involve language in some capacity. Let's briefly summarize the major areas and contextualize CocoGold in this landscape.

- **Traditional Semantic Segmentation**: Assigns each pixel to one of a fixed set of predefined categories (e.g., person, car, tree). This is the classical semantic segmentation task, and is limited to classes seen during training.

- **Open-Vocabulary Segmentation**: Extends semantic segmentation to handle novel categories not seen during training by leveraging vision-language models, such as the foundational CLIP. Can segment "zebra" even if only trained on "horse".
- **Referring Expression Segmentation**: Identifies and segments specific objects based on natural language descriptions that may include attributes, spatial relationships, and context (e.g., "the red car on the left," "the tallest person wearing glasses").

For the purposes of free-form, natural-language instructions used for photo editing, it may appear that referrring expression segmentation would be the best fit. However, current referring expression segmentation models and benchmarks are usually based on relatively restricted datasets such as RefCOCO / RefCOCO+ [67, 68], and RefCOCOg [69, 70]. These datasets extend COCO with annotated expressions that refer to specific object instances, but are inherently limited to the original 80 COCO object categories. Referring expression segmentation systems based on these datasets are able to disambiguate references and select specific object instances, as long as these instances and expressions refer to the set of known categories. Models trained on these datasets are not truly open with respect to novel object categories, or to complex descriptions out of the domain of the object-centric expressions covered in the datasets. More importantly, benchmarks that use these datasets do not evaluate generalization to novel object categories and expressions.

CocoGold, in its initial implementation, is akin to open-vocabulary segmentation systems, because we purposefully focus on single-word prompts (e.g., "cat" or "chair") as a practical simplification to establish the fundamental capability of leveraging diffusion models' text-image understanding for segmentation. This simplified approach allows us to demonstrate that pre-trained diffusion models are excellent candidates to perform text-guided segmentation, and show generalization to object categories not seen during training.

However, the fact that diffusion models have been trained on billions of image-text pairs and *understand* nuanced text references, makes us optimistic about extending the system – using the same simple training methodology – to rich natural-language expressions, such as "the person wearing red on the left". We are hopeful that such a system would be successful on referring expression segmentation tasks, while being capable of generalizing to arbitrary objects and complex descriptions.

4.2.2 The Diffusion Model Opportunity

Recent work has shown that diffusion models trained for text-to-image generation possess rich internal representations that can be repurposed for other vision tasks, with or without relating visual information with language. For example, ODISE [96] unifies pre-trained text-image diffusion and discriminative models to perform open-vocabulary panoptic segmentation. It uses features extracted from both a CLIP model and a Stable Diffusion one to train additional components: an *implicit* captioner that produces text embeddings rather than natural language captions, and a mask generator.

While ODISE shows impressive generalization to unseen classes, it requires a complex, carefully orchestrated training and inference pipeline based on feature selection from the frozen models. Marigold [14], on the other hand, does not require feature extraction and selection. Instead, it repurposes the generative diffusion process and shows that it can be successfully fine-tuned for a variety of computer vision tasks [112]. Fine-tuning Marigold is a relatively light-weight process, because it's based on diffusion models that have been trained on billions of images and, consequently, have learned powerful representations. Marigold, however, ignores the rich text information present in the pre-trained text-to-image model, and has not been adapted for segmentation.

DiffDIS [97] is similar to Marigold in that it leverages pre-trained text-to-image models for segmentation, but it's strongly optimized for the specific task of binary dichotomous image segmentation. Like Marigold, it doesn't use the rich text information built into state-of-the-art text-to-image models.

CocoGold builds on the Marigold approach, improves it with language conditioning and adapts it to segmentation. This has several advantages for our application:

- **Pre-trained Visual-Linguistic Understanding**: Because diffusion models like Stable Diffusion have been trained on billions of image-text pairs, they already understand the relationship between text descriptions and visual content.
- **Spatial Awareness**: The U-Net architecture at the heart of diffusion models naturally preserves spatial information thanks to the use of skip connections [37]. It also preserves locality, typically through the use of convolutional processing.

- **Minimal Adaptation Required**: Following the Marigold approach, only minor changes in architecture are required to repurpose these models, making training simple, fast, and efficient.

4.2.3 Reformulating Segmentation as Generation

Our key insight is to reformulate the segmentation task as an image generation problem, as conceptually illustrated in Fig. 4.1. Rather than predicting discrete class labels or binary masks, we train the model to generate a modified version of the input image where target objects are highlighted in white while everything else remains unchanged. This formulation has several advantages:

1. It leverages the model's existing image generation capabilities without requiring new output heads or classification layers.
2. It naturally handles multiple instances – when asked to segment "cat", the model highlights all cats in the image.
3. It maintains the continuous nature of diffusion models, avoiding the challenges of discrete optimization.
4. It produces interpretable intermediate results that users can understand and verify.

4.2.4 Why We Call It Text-Grounded Segmentation

In its current incarnation, CocoGold is similar to open-vocabulary segmentation, because it uses single-word inputs, segments all objects that match the specified class, and is capable of generalization to unseen classes. However, it was not trained to fulfil the open-vocabulary segmentation task per se – the reason we use single-word inputs is a deliberate simplification that allows us to validate our assumptions, and is a starting point for subsequent iterations.

CocoGold was created to demonstrate that diffusion models trained on billions of images can be easily leveraged to combine text and image inputs for segmentation tasks. Marigold and other methods had already established the rich feature representation capabilities of diffusion models, and their ability to be adapted to new tasks. However, these previous methods do not leverage the – equally rich – text representation capabilities of diffusion models. Our work demonstrates that conditioning on text inputs is feasible and practical, and opens a new direction towards text-based segmentation.

FIGURE 4.1: The CocoGold generative segmentation task. Given an input image and a text description, the diffusion model generates a *copy* of the input, except that the objects that match the text are highlighted in white. It works with small, distant, and partially occluded subjects.

The next iterations of CocoGold will use the same training approach, but we'll provide richer text expressions as inputs, instead of just single words. This will be similar to referring expression segmentation insomuch as the system will be able to use complex phrases for target disambiguation. Unlike usual referring segmentation training, however, our method will not necessarily be based on a limited set of known classes, like the ones used in the traditional RefCOCO, RefCOCO+ and RegCOCOg datasets [67, 68, 69]. Instead, we plan to leverage extensive image datasets combined with synthetic segmentation and captioning data.

Our key insight – we can leverage the representational capacity of heavily-trained diffusion models for novel tasks – suggests the feasibility to create a truly open-category, text-based segmentation model that is easy to train and use (it will be a single model, not a complex pipeline). This is what we refer to as *text-grounded segmentation*, where CocoGold is the first step in that direction.

4.3 Method

Our approach builds on the Marigold architecture, which adapted Stable Diffusion to dense prediction tasks. We extend their dual image conditioning mechanism to incorporate text prompts, enabling text-grounded object segmentation.

4.3.1 Problem Formulation

Given an input RGB image $\mathbf{x} \in \mathbb{R}^{3\times H\times W}$ and a text description $\mathbf{t}$ of target objects (e.g., "cat", "clock", "chair"), our goal is to generate a modified image $\mathbf{y} \in \mathbb{R}^{3\times H\times W}$ where:

- Pixels belonging to objects matching the text description are set to white (RGB value $[255, 255, 255]$)
- All other pixels retain their original values from $\mathbf{x}$

This formulation transforms segmentation into an image-to-image translation task guided by text, allowing us to leverage the full power of pre-trained text-to-image diffusion models.

Why Not Generate Binary Masks

The usual output of segmentation models consists of a per-pixel category classification. In the particular case of binary segmentation, such as the one we train CocoGold for (we ultimately want to retrieve the segmentation mask for objects that match the specified class), output consists therefore of classifying pixels among two categories: pixels that belong to the mask, and pixels that don't. However, we observed that training using this binary data as the ground truth was not successful. The reason, we believe, is that this type of output is out of distribution for the diffusion model, which was trained to generate images or image-like outputs and not binary data.

We ran early experiments that stayed close to Marigold's formulation, and asked the diffusion model to reconstruct a binary mask decoded by the VAE. Even though the autoencoder can faithfully represent two-level images as reported in Table 1 of DiffDIS [97], the denoiser rapidly collapsed to the majority class: after a brief period of seemingly good predictions, the loss plateaued at solutions that reproduced uniform backgrounds and ignored foreground structure. This is illustrated in Fig. 4.2. The failure, as mentioned, can be traced to a domain mismatch. Stable Diffusion was pre-trained to synthesize natural images; forcing it to regress to piecewise-constant silhouettes pushes the model far outside the distribution it models well, so gradients provide little guidance once the latent-space MSE objective is minimized by predicting the dominant value.

We attempted several countermeasures, including class-balanced MSE [120, 121], Dice-inspired penalties [122], and focal losses [123]. We implemented these by operating on the predicted latent residuals, weighting errors to emphasise minority foreground pixels while avoiding a full decode to pixel space, but none consistently restored learning. The imbalance inherent to the COCO crops we use for training – background pixels vastly outnumber foreground ones – kept rewarding trivial predictions even with these adjustments. One potential approach could have been to compute MSE on decoded samples in pixel space, but this would have incurred prohibitive memory costs and was not a sure avenue for success.

The successful strategy reframed the target as an image-to-image translation task: the supervision signal is the original photograph with the requested objects overpainted in white. This aligns the optimization problem with the generator's training data, letting the network focus on drawing localized

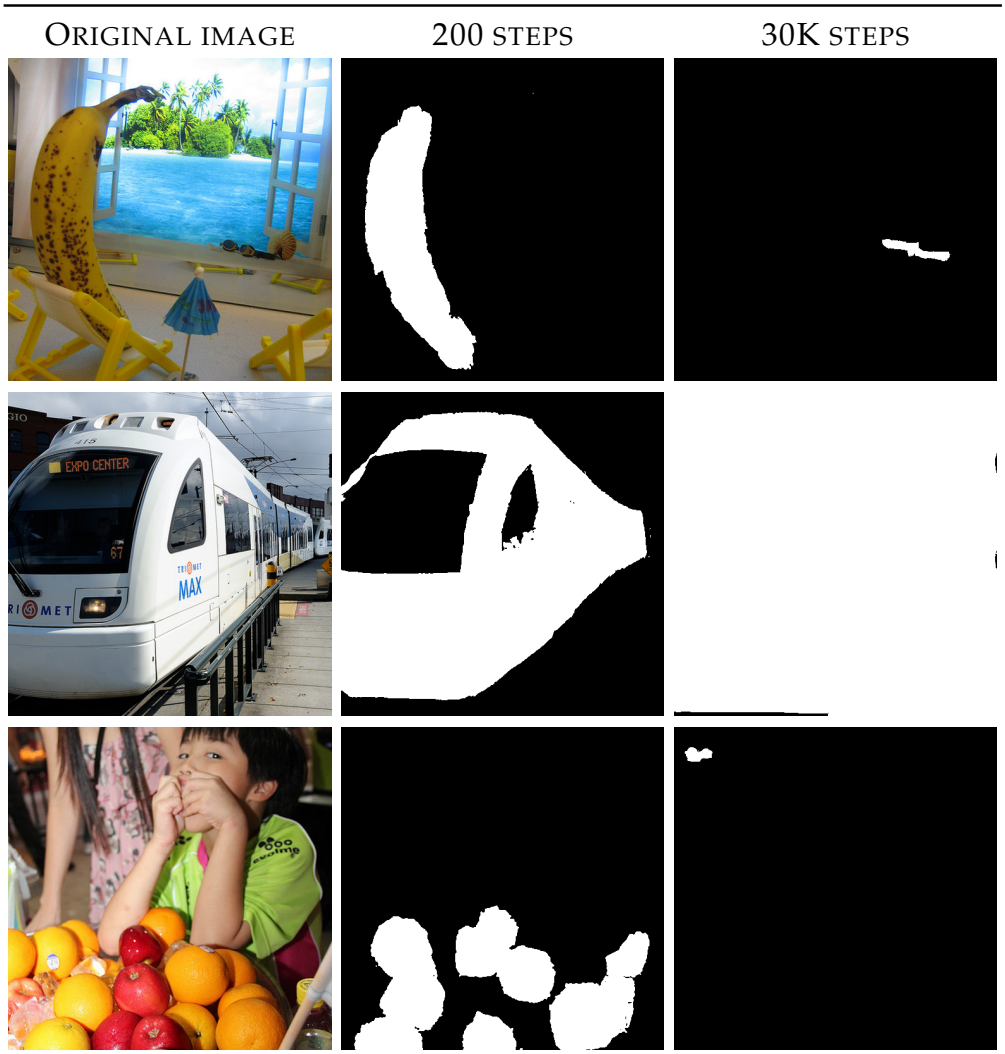


FIGURE 4.2: Failed attempt to train generation of binary masks. The model learns quickly to identify the target objects, but training collapses and degenerates.

highlights on top of textures it already knows how to reproduce. In this regime the diffusion process naturally treats every instance that matches the prompt identically, so all detected *cat* regions inherit the same highlight without additional bookkeeping.

Recovering a binary mask then becomes a lightweight post-processing step. As we'll see in more detail later, we threshold the generated image to retain near-white pixels, apply a morphological opening (erosion followed by dilation) to suppress isolated false positives, and, when necessary, desaturate bright regions of the input before inference so genuine highlights are not mistaken for mask pixels. This pipeline keeps the diffusion model within its comfort zone while still producing crisp masks for downstream editing, and it can be extended with confidence maps or alternative highlight colors in future iterations.

Note that this approach, despite being a generative task, strictly respects regions in the image not affected by the segmentation mask. Unlike other image translation tasks that affect all pixels in the target image, we use generation as an intermediate step to build the segmentation mask, so subsequent editing operations that act on this mask will preserve other image regions untouched.

Characterizing Text Inputs

To isolate the contribution of language-grounded conditioning, we decided to use the COCO dataset and restrict prompts to the canonical COCO object categories. Each training sample pairs a single noun (e.g., *cat*, *clock*) with an image crop that contains that class, providing unambiguous supervision and enabling clean zero-shot evaluations on withheld labels. Although the underlying CLIP encoder accepts arbitrary prompts, keeping the vocabulary compact makes dataset preparation straightforward and avoids conflating segmentation performance with referring-expression parsing during this first study.

Class imbalance required additional curation. As Fig. 4.3 illustrates, the dataset is dominated by the *person* category, and several classes frequently appear as tiny instances. We removed *person* entirely and kept the fourteen next-most-common categories spanning vehicles, animals, and household objects ("car", "dining table", "chair", "train", "airplane", "giraffe", "clock", "toilet", "bed", "bird", "truck", "cat", "horse", "dog"). Our custom dataset tooling samples crops around the chosen annotation, rejects views where the

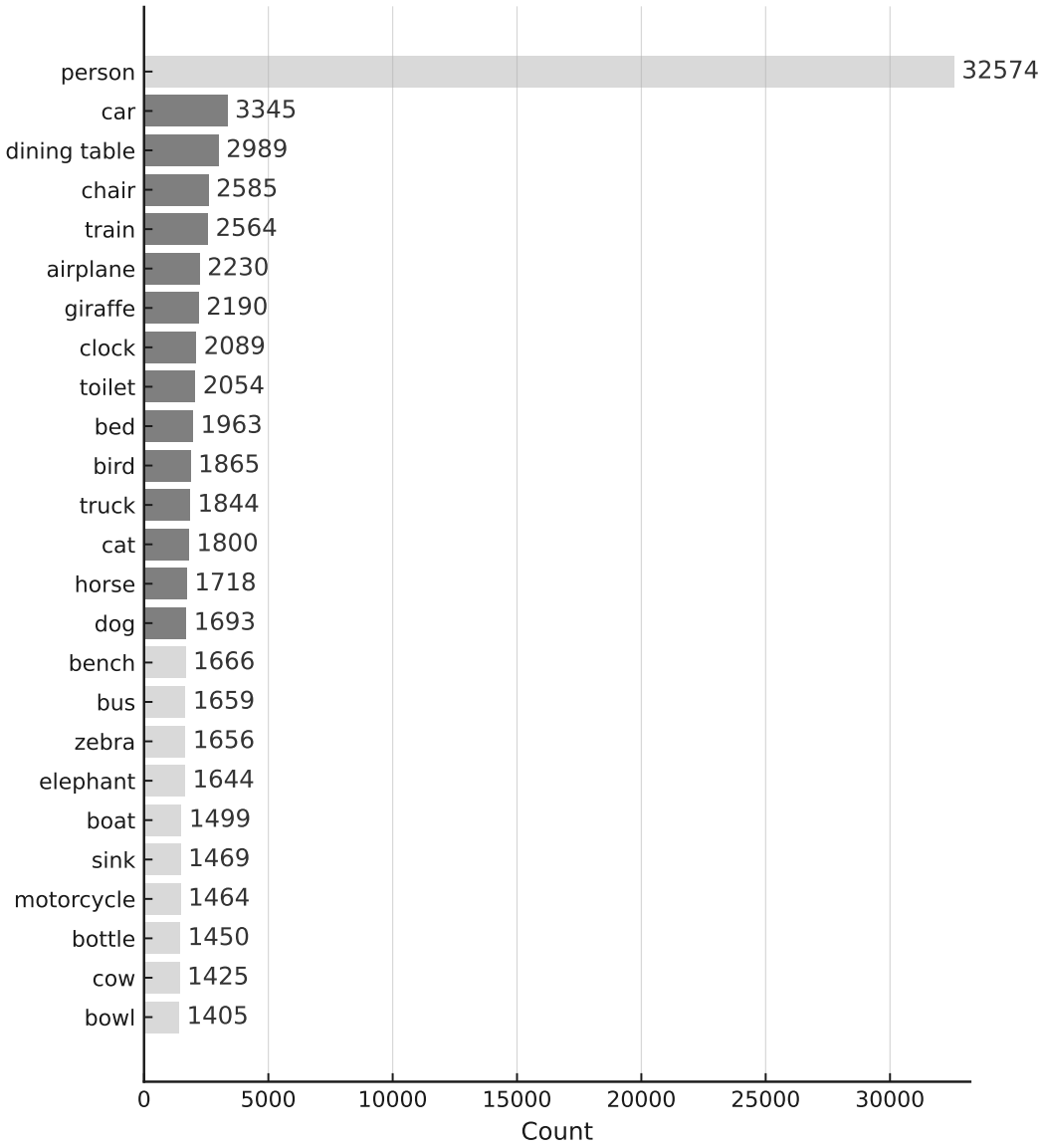


FIGURE 4.3: Number of class instances for the top 25 classes after iterating once through the COCO training dataset. Each iteration selects a new image, then a random category and square crop, so results vary per epoch. For training, we excluded *"person"* and used the next top 14 classes.

subject would be clipped, and biases selection toward masks occupying a meaningful portion of the frame. This guarantees that the text tokens refers to visually salient regions in the conditioning image.

Despite their simplicity, these prompts already benefit from the rich alignment learned during Stable Diffusion's pre-training: the fine-tuned model correctly segments categories never included in the training subset, such as "elephant", purely from the shared embedding space. Future iterations will keep the same architecture while expanding the linguistic space with synonyms, multi-word descriptions, or captions synthesized by vision-language models. This will allow us to study attribute binding and disambiguation without introducing a new training recipe.

4.3.2 Architecture

We adopt the architectural modifications introduced by Marigold, extending them with text conditioning to enable our text-grounded segmentation task.

Base Model

We build upon Stable Diffusion 2[1] [8], part of the family of *latent diffusion models* whose general architecture is depicted in Fig. 4.4. Stable Diffusion models consist of three main components:

- A CLIP text encoder [61] that converts text prompts into embeddings for conditioning.
- A pre-trained Variational Autoencoder (VAE) that encodes images from pixel space ($512 \times 512 \times 3$) to a more compact latent space ($64 \times 64 \times 4$), and decodes latent representations back to pixel space.
- A U-Net [37] that performs the denoising process in latent space, conditioned on text embeddings.

The VAE used in Stable Diffusion inherits the conceptual framework laid out by seminal works on latent variable autoencoders [125, 126], but it's implemented using a deeper convolutional architecture optimized for compressing images into a latent space where the diffusion process takes place. As a sanity check, we verified that its restorative capabilities are more than adequate to faithfully decode the outputs from our trained model. We conducted a test where we generate ground truth images with segmented objects highlighted

[1] https://huggingface.co/stabilityai/stable-diffusion-2

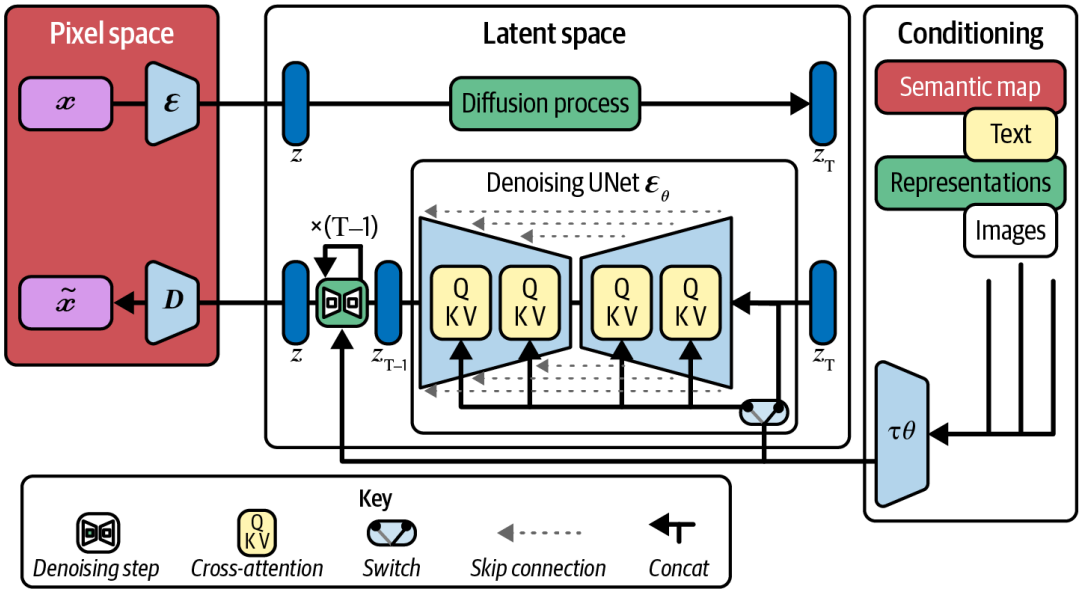


FIGURE 4.4: Latent Diffusion schematic architecture. Different types of conditioning signals can be potentially supplied to the image generation process. Stable Diffusion uses **text inputs** for conditioning. *Source: [124], based on Figure 3 from [8]*

in white, pass them through the VAE encoder and then the decoder, and results were visually indistinguishable from the ground truths. This confirms the observation in the DiffDIS paper [97] that the Stable Diffusion VAE can achieve near perfect reconstruction of even full binary masks.

Dual Image Input Mechanism

Following Marigold's approach, we modify the U-Net to accept two image inputs in latent space:

1. **Original image latent $\mathbf{z}_o \in \mathbb{R}^{4\times64\times64}$**: The VAE-encoded representation of the input RGB image, providing spatial context and appearance information.
2. **Target (masked) image latent $\mathbf{z}_m \in \mathbb{R}^{4\times64\times64}$**: The noised latent representation of the target image (original with objects highlighted in white) that the model learns to denoise.

These two sets of latents are concatenated along the channel dimension to create the combined latent representation sent to the model for training:

$$\mathbf{z} = \text{concat}(\mathbf{z}_o, \mathbf{z}_m) \in \mathbb{R}^{8\times64\times64}$$

Architectural Modifications

We retain Marigold's adaptation strategy, altering only the components needed to ingest the paired latents. The first convolution of the U-Net is widened to eight channels so it can process $\mathbf{z} = \text{concat}(\mathbf{z}_o, \mathbf{z}_m)$. Following Marigold, we duplicate the pre-trained weights along the channel axis and scale them by one half, yielding $\mathbf{W}_{\text{new}} = \text{concat}(\mathbf{W}_{\text{orig}}/2, \mathbf{W}_{\text{orig}}/2)$. This simple initialization maintains the magnitude of activations, preventing sudden distribution shifts that could destabilize training.

No further architectural edits are required beyond the first layer. Keeping the subsequent convolutions and attention blocks untouched preserves the spatial priors, multi-scale reasoning, and text-aligned features that were learned during the original Stable Diffusion training.

[Figure: Detailed architecture diagram showing the modified Stable Diffusion architecture. Include VAE encoder/decoder, modified U-Net with dual image inputs, text encoder, and cross-attention connections. Highlight the minimal changes in red.]

Text Conditioning

Unlike Marigold, which conditions on empty prompts, CocoGold uses the full text pathway already present in Stable Diffusion. Prompts are encoded by the frozen CLIP text encoder and injected through the existing cross-attention layers, so the denoising process attends to tokens that describe the desired objects. During denoising, the model receives simultaneous cues from the latent pair and from the token embeddings that specify which concepts to isolate, letting the attention maps focus on regions whose semantics match the prompt.

Leaving the encoder untouched preserves the language-vision alignment obtained during large-scale pre-training. As a result, the model keeps its zero-shot behaviour on categories that are missing from the fine-tuning set, while still grounding the segmentation outcome on the input text.

4.3.3 Training Strategy

Our training approach focuses on enabling the model to learn the segmentation task while preserving its pre-trained capabilities.

Dataset Preparation

For training, we require triplets of (input image, text prompt, target image with highlighted objects). We assemble these triplets directly from the MS COCO 2017 instance segmentation annotations, using the category names themselves as prompts, as discussed above. Because the sampler constructs a fresh crop on each iteration, diversity comes from the dataset rather than from synthetic augmentations. We restricted our initial training run to the top 14 categories after *"person"*, as represented in Fig. 4.3.

To create each training example we:

1. Retrieve and image and its segmentation mask annotations from COCO's training split.
2. Select a random square crop of the original image.
3. Select a segmentation mask for one random category of an instance inside the crop. We apply heuristics to avoid masks that would be heavily cut out after applying the random crop, and to assing larger priorities to larger masks.
4. Paint the selected region white to form the target image, and use the category name as the text prompt.

Iterating through the dataset yields different crops, categories and masks across epochs, providing the variation normally expected from data augmentation without the need to apply augmentation transforms.

Training Objective

We optimise the standard latent-space diffusion loss used in Stable Diffusion:

$$\mathcal{L} = \mathbb{E}_{t,\tilde{\mathbf{z}}_0,\boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{z}_t, t, \mathbf{c}_{\text{txt}})\right\|^2\right], \tag{4.1}$$

with $t \sim \mathcal{U}([0,T])$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0},\mathbf{I})$. The target latent $\tilde{\mathbf{z}}_0 = \text{VAE}_{\text{enc}}(\mathbf{y})$ is diffused according to $\alpha_t\,\tilde{\mathbf{z}}_0 + \sigma_t\,\boldsymbol{\epsilon}$, which we concatenate with the frozen image latent $\mathbf{z}_o = \text{VAE}_{\text{enc}}(\mathbf{x})$ to obtain $\mathbf{z}_t = \text{concat}(\mathbf{z}_o, \alpha_t\,\tilde{\mathbf{z}}_0 + \sigma_t\,\boldsymbol{\epsilon})$. The conditioning input $\mathbf{c}_{\text{txt}} = \text{CLIP}(\mathbf{t})$ provides the text prompt, and $\boldsymbol{\epsilon}_\theta$ denotes the noise-prediction U-Net.

Fine-Tuning Details

Key hyperparameters and implementation choices:

- **Learning rate**: 3×10^{-5} with the IterExponential schedule (same as used in Marigold), preceded by 100 warmup steps.
- **Batching**: Physical batch size 16 with gradient accumulation yielding an effective batch of 32 examples.
- **Resolution**: 512×512 images in latent space. Stable Diffusion 2 was pre-trained on this resolution and then further trained on 768×768, but we choose 512 to save memory and compute resources.
- **Data sampling**: No synthetic augmentations; variety arises from the randomness in the loading dataset, as described above.
- **Trainable modules**: Only the U-Net is fine-tuned, while the VAE and CLIP text encoder remain frozen. We don't use LoRA or other parameter-efficient fine-tuning techniques.

Similar to Stable Diffusion, training is performed fully in latent space, so we don't need to decode latents to go back to pixel space throughout the process. As a demonstration of the efficiency gains we get from staying in latent space, we highlight that training can be peformed on a single GPU. A training run of 18,000 steps took less than two days on a NVIDIA RTX A6000 Ada Generation GPU.

4.3.4 Inference

During inference, we use the same architecture, with the noised masked latent $\mathbf{z}_m$ initialized to pure Gaussian noise. Using the conditioning signals from the reference image $\mathbf{z}_o$ and the text embeddings, the diffusion process progressively generates $\mathbf{z}_m$ the same way it learned to do during training: it (conceptually) *copies* the pixels from the reference image, but highlights in white the object of interest that was specified with the text prompt.

Standard Inference Pipeline

1. Encode the input image $\mathbf{x}$ to latent space: $\mathbf{z}_o = \text{VAE}_{\text{enc}}(\mathbf{x})$
2. Encode the text prompt: $\mathbf{c_{txt}} = \text{CLIP}(\mathbf{t})$
3. Initialize target masked latent with random noise: $\mathbf{z}_m \sim \mathcal{N}(0, \mathbf{I})$

4. Iteratively denoise for T steps, conditioned on $\mathbf{z}^{(x)}$ and $\mathbf{c}$
5. Decode the final latent: $\mathbf{y} = \text{VAE}_{\text{dec}}(\mathbf{z}_0^{(y)})$
6. Extract the binary mask using the post-processing pipeline described below

Sampling Strategy

Following the Stable Diffusion 2 inference configuration [2], we use the DDIM scheduler [99] for 50 steps by default. We observe good results with this regime. Due to the stochastic nature of diffusion-based generation, we optionally ensemble various prediction runs and average results, as evaluated in Section 5.2.2.

Post-Processing

The diffusion model predicts an RGB image that mimics the original, with highlighted regions on the segmentation targets. We apply a lightweight post-processing stage to extract a binary mask for downstream applications. In line with our goals to be faithful to the original images, this also ensures that the pixels outside the mask are untouched – when we need to use the areas outside the mask, we'll revert to the original image, not the one generated during the diffusion process.

To isolate the white overlay with minimal artefacts, we apply the strategy outlined in this section.

We threshold the output RGB image to retain pixels that fall within a tolerance window around pure white. We empirically set the tolerance to accommodate the slight variation observed in the model outputs, with mask colors close to white, but not quite pure white. A single threshold, however, cannot distinguish between predicted highlights and genuinely bright content that was already present in the input image, such as over-exposed skies.

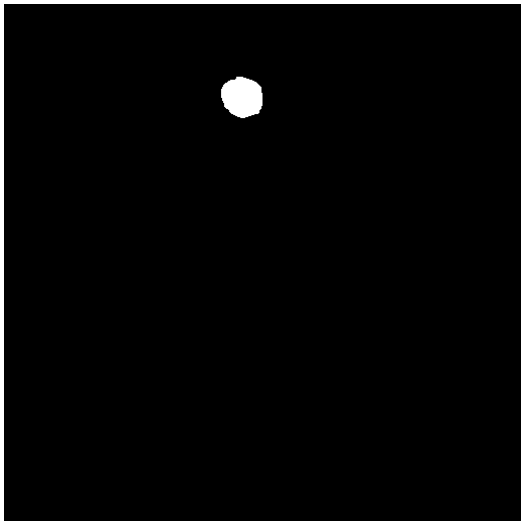
These highly saturated highlights in the source image require an additional safeguard, demonstrated in Fig. 4.5. Before running inference, we desaturate pixels whose luminance exceeds a fixed threshold, effectively neutralising almost-white regions while leaving the rest of the image untouched. Because our fine-tuned Stable Diffusion model strives to reproduce the conditioning

[2] https://huggingface.co/stabilityai/stable-diffusion-2/blob/main/model_index.json#L15

(A) Original

(B) Naïve threshold mask

(C) Desaturated highlights



(D) Fixed threshold mask

FIGURE 4.5: Desaturation trick. Panel B shows naïve thresholding for binary mask extraction from an image A that contains legitimate highlight colors, close to white. We fix the problem by desaturating highlights in the input image (C), so the model also predicts the desaturated colors and the mask can be successfully thresholded (D).

image, the model propagates this desaturation to its output, ensuring that white color appears only where the mask should be.

After the thresholding process is applied, we observe very small isolated false positives throughout the image – see panel C of Fig. 4.6. These false positives are pixel-sized regions where the model generates a color close to white, despite the safeguard in place. We mitigate this effect through morphological opening [127]: an erosion followed by dilation using a 3×3 convolution kernel. This removes small predicted white speckles while preserving the overall shape of the predicted mask, as can be seen in panel D of the same figure.

In practice, this combination—desaturation pre-processing, tolerant thresholding, and morphological opening—yields clean binary masks without retraining the model. Future work could eliminate the pre-processing step altogether by training with a chroma-key colour instead of white, or by supervising an auxiliary mask head during fine-tuning.

Future Speed Optimizations

Interactive photo editing on consumer devices benefits from fast inference, to provide the user with a responsive feedback loop. Diffusion models, such as CocoGold's segmentation approach, typically require tens of steps to ensure good quality. Furthermore, the use of test-time ensembling is usually beneficial for computer vision tasks because of the stochastic nature of predictions in diffusion models.

The acceleration of diffusion models is an area of very active research. CocoGold could potentially benefit from a number of advances originating in the field, as well as techniques that are appropriate when diffusion models are used for computer-vision tasks. We refer the reader to the State of the Art Section 2.7.3 for a compact review of fast sampling methods (e.g., DDIM/DPM-Solver, consistency models, LCM/LCM-LoRA, ADD/LADD, and trailing DDIM).

In the experiments chapter we also show that trailing-DDIM is a viable configuration for CocoGold, enabling few-step inference without fine-tuning for this regime (Section 5.2.4).

(A) Original

(B) Model Generation

(C) Extracted Mask

(D) Speckles Removed

FIGURE 4.6: Fixing false positive speckles with morphological opening. Panel C shows the extracted mask contains some speckles caused by false positives (near-white pixels in the generated image outside the mask region). Applying erosion and dilation eliminates the speckles (D).

Resolution Handling

While our experiments use 512×512 resolution for training efficiency, Stable Diffusion 2 natively supports resolutions up to 768×768. For practical photo editing applications at arbitrary resolutions, we can adopt a strategy inspired by computational photography techniques:

1. **Inference Resolution**: Generate masks at 768×768 for better detail
2. **Smart Upsampling**: Use guided filtering or learned upsampling to match the original image resolution
3. **Edge Refinement**: Apply morphological operations and progressive gradients at mask boundaries for smooth blending

This approach, successfully used in applications like Apple's Portrait Mode for depth-based segmentation [128], provides a practical path to high-resolution masks while maintaining computational efficiency.

4.4 Analysis: Why Diffusion Models Excel at This Task

To understand why our approach is effective, it's instructive to compare with alternative architectures and analyze the unique advantages of diffusion models for text-grounded segmentation.

4.4.1 Comparison with Vision-Language Models

Recent VLMs like PaliGemma [27, 26] demonstrate that a single model can handle captioning, question answering, and segmentation. Nevertheless, repurposing a general-purpose VLM for dense prediction still requires non-trivial architectural and training work, especially when the original model was never exposed to spatial supervision.

Architectural Complexity

Instruction-tuned VLMs emit autoregressive token sequences. Producing dense masks requires extra machinery on top of the core text decoder:

- Dedicated spatial vocabularies or output tokens to represent mask structure (e.g., PaliGemma's learned mask tokens, Qwen2-VL's high-resolution spatial embeddings [29]).

- Auxiliary decoders are required to decode those tokens into pixel maps. PaliGemma trains a mask autoencoder whose latent codes feed the language model; LISA adds a <SEG> token that conditions SAM's mask decoder [27, 74].
- Integrating the additional modules without destabilising the pre-trained backbone often calls for staged optimisation and carefully chosen initialisation.

Recent releases bake these components into the base model—Qwen2-VL couples a NaViT-style vision tower [129] with multimodal rotary embeddings so spatial detail is available to the decoder at arbitrary resolutions [29], while lightweight retrofits such as LISA modify LLaVA post-hoc. Both lines of work highlight that segmentation demands explicit architectural augmentation rather than simple re-prompting.

Restorative Capacity

Because most VLMs operate in token space, their segmentation heads must quantise masks before the language model can consume them. PaliGemma therefore trains a discrete mask autoencoder so that the decoder can reconstruct masks from a short sequence of learned tokens [27]. The published evaluations report strong downstream results on referring expression segmentation, but they do not analyse the reconstruction error introduced by the quantised codebook, leaving the ultimate fidelity of fine structures unclear. Diffusion models also rely on a VAE, yet the continuous latent formulation combined with Stable Diffusion's decoder has been shown to reproduce binary masks with negligible loss [97].

Data Requirements

PaliGemma's segmentation ability stems from a geometry-rich curriculum: tens of millions of annotated examples covering panoptic, referring, and grounding tasks are mixed with text instructions [27]. Qwen2-VL follows a similar approach, scaling to trillions of multimodal tokens that include high-resolution segmentation labels [29]. Even post-hoc adaptations such as LISA rely on curated mask datasets and SAM-generated pseudo-labels to teach the new <SEG> token [74]. Balancing this spatial supervision with conversational quality remains an active area of research.

Works such as LISA and Moondream demonstrate that post-training or fine-tuning an existing VLM for segmentation is possible, but each introduces additional modules and supervision. LISA operates on top of LLaVA while freezing SAM, and Moondream adds grounded reasoning heads after its initial release [74, 30]. These adaptations confirm the feasibility while underscoring the engineering effort required to keep the original language capabilities intact.

Optimization Challenges

Autoregressive decoding imposes optimisation constraints that differ markedly from diffusion fine-tuning:

- Masks are supervised through cross-entropy over a discrete codebook, so metrics such as mIoU or boundary quality cannot be optimised directly. Exposure bias remains because training relies on teacher forcing while inference relies on past tokens generated autoregressively.
- Co-training language and spatial objectives can potentially introduce interference, so practitioners often resort to staged curricula or reduced learning rates to preserve dialogue quality while learning dense prediction.
- Maintaining high-resolution features while decoding long token sequences is memory-intensive, slowing iteration compared with updating a single diffusion U-Net.

4.4.2 Comparison with Detection-Based Pipelines

Detection-driven pipelines approach text-conditioned segmentation by chaining an open-vocabulary detector with a powerful mask generator. Systems such as GLIP or Grounding DINO translate a natural-language query into bounding boxes that localise candidate objects [77, 78]. Those boxes are then forwarded to a general segmentation backbone—typically SAM or one of its derivatives—to obtain high-resolution masks [73]. The Grounded SAM family popularised this recipe because it inherits zero-shot detection from the language-aware detector while leveraging SAM's strong boundary accuracy. Variants further cascade multiple prompts (points, boxes, scribbles) or run iterative prompt refinement to disambiguate similar categories.

This modularity is appealing: each component can be swapped for the best available specialist, training data can be reused, and practitioners can upgrade a stage without retraining the entire system. The approach also scales well to large vocabularies, since the text encoder in the detector already supports open-vocabulary queries.

However, the decomposition introduces practical drawbacks. Detection recall becomes the upper bound for segmentation quality: missed or misaligned boxes leave the mask head with nothing to refine. Hand-off between modules is inherently lossy: rectangular boxes rarely align with object silhouettes, so SAM must carve out the foreground inside a coarse box and can miss thin structures or fine boundaries. Running two heavy backbones sequentially increases latency and memory footprint, complicating deployment, particularly on consumer devices for interactive use. Finally, joint optimisation is difficult because the detector and the mask head are typically trained separately, which limits end-to-end fine-tuning.

4.4.3 Comparison with DiffDIS

A concurrent development, DiffDIS [97], provides valuable validation of using diffusion models for segmentation tasks while highlighting different design choices:

Task Focus

- **DiffDIS**: Binary dichotomous segmentation with extreme precision for foreground/background separation
- **CocoGold**: Text-grounded multi-class segmentation for identifying specific objects based on language

Technical Innovations

DiffDIS introduces a stable one-step denoising approach that achieves high-quality results with a single diffusion step, compared to our 4-50 step range. They also introduce an auxiliary edge generation task that provides an additional conditioning signal and loss term. This method is designed to optimize for precise boundaries, addressing the soft edge problem inherent in diffusion-based segmentation. These innovations could be adapted to improve CocoGold's efficiency and boundary precision.

Complementary Approaches

Rather than competing approaches, DiffDIS and CocoGold demonstrate complementary uses of diffusion models:

- DiffDIS proves that diffusion models can achieve state-of-the-art segmentation quality
- CocoGold shows that these models can understand complex language-vision relationships even when adapted for other downstream tasks interactive editing tasks

These different methods reinforce the idea that diffusion models offer a powerful foundation for computer vision tasks such as segmentation, where design choices can be made to adapt to different requirements and use cases.

4.4.4 Why Diffusion Fits CocoGold

Together, the preceding comparisons highlight why we adopt a diffusion backbone: a single U-Net absorbs both visual and textual conditioning without introducing discrete mask vocabularies or multi-stage pipelines; the continuous latent space preserves fine detail when decoding masks; and pre-training provides the zero-shot grounding necessary for open category prompts. This balance of simplicity and expressiveness makes diffusion particularly well-suited to the text-driven segmentation setting we target.

4.5 Implications for Photo Enhancement

CocoGold represents an important step toward our vision of text-driven, localized photo enhancement. In its current form the system responds to single-word category prompts, yet the underlying architecture lays the groundwork for richer interactions as the textual interface evolves. We discuss some of the capabilities that will be unlocked from a combination of what the prototype already enables and what upcoming iterations can provide.

4.5.1 Localized Filter Application

Even with single-word prompts, CocoGold provides class-specific masks that can modulate FilterNet's global adjustments. Early integrations focus on straightforward scenarios where the vocabulary aligns with the curated

training set. As we broaden the prompt space, the same mechanism will enable more granular parameter routing and softer blends across regions.

- Different filter parameters for different regions (e.g., brightening faces while darkening backgrounds)
- Gradual transitions between regions using soft masks derived from confidence estimates
- Preservation of the non-destructive editing paradigm through mask-guided adjustment layers

4.5.2 Natural User Interaction

The proof-of-concept requires single tokens (“cat”, “clock”), so free-form language remains future work. Nonetheless, the diffusion backbone already accepts CLIP embeddings, making sentence-level prompts a question of data rather than architecture.

4.5.3 Future Extensions

The success of CocoGold opens several research directions:

- **Richer prompts**: Moving beyond single objects to complex descriptions (“the person on the left wearing sunglasses”)
- **Soft segmentation**: Generating confidence maps instead of binary masks for smoother blending
- **Interactive refinement**: Allowing users to refine masks through additional prompts or lightweight edits

4.6 Summary

This chapter presented CocoGold, our novel approach to text-grounded segmentation using diffusion models. By reformulating segmentation as an image generation task and leveraging the Marigold architecture, we demonstrated that pre-trained diffusion models can be efficiently adapted for text-grounded spatial understanding tasks with minimal modifications. While our current implementation focuses on single-word category prompts, the approach establishes a foundation for future extension to complext text inputs and open segmentation categories.

Key contributions of this work include:

1. A novel formulation of segmentation as generating images with highlighted objects, avoiding complex architectural changes
2. Demonstration that simple MSE loss in latent space suffices for training, without need for specialized losses
3. Evidence that diffusion models offer practical advantages over VLMs and detection pipelines for this task
4. A foundation for future text-driven, localized photo enhancement systems

In the next chapter, we will present experimental results showing CocoGold's performance and limitations. We will also explore how the integration of CocoGold with FilterNet enables new possibilities for intuitive photo editing that demonstrate that even generative models like CocoGold can be used for non-destructive editing.

Chapter 5

Experiments and Results

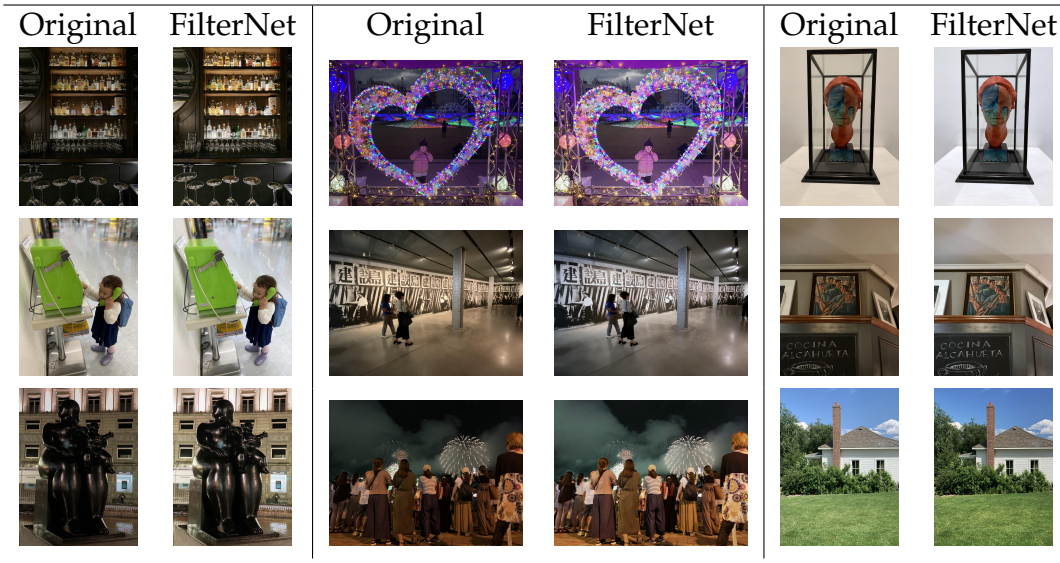
This chapter presents experimental results from our two main contributions: FilterNet for automatic global photo enhancement, and CocoGold for text-grounded object segmentation. These systems represent complementary approaches to the broader goal of intuitive, professional-quality photo editing. FilterNet demonstrates that neural networks can successfully predict enhancement parameters that rival professional adjustments, while CocoGold shows how pre-trained diffusion models can be efficiently adapted to understand and segment objects based on natural language descriptions.

We begin with a comprehensive evaluation of FilterNet, including quantitative metrics, ablation studies, and qualitative analysis of its performance on real photographs. We then present results from CocoGold, demonstrating its ability to segment objects from simple text prompts and exploring few steps inference. Finally, we describe the integration of both in the COCONET prototype, a combined pipeline for local enhancements based on text-driven segmentation. The experiments reveal both the strengths of our approaches and the challenges that remain in creating a fully integrated text-driven photo editing system.

5.1 FilterNet Experiments

Figure 5.1, duplicated from Fig. 3.1 for convenience, demonstrates the use of FilterNet on real-life photographs taken with cameras or mobile phones. FilterNet is able to improve exposure, tone, and color temperature without imposing dramatic style changes and while being respectful with human features such as skin tones. This section discusses, in both quantitative and qualitative terms, the results from the experiments we carried out, and it also analyzes the individual contributions of the components we used. We

FIGURE 5.1: Original straight-out-of-camera photos (odd columns) and optical adjustments proposed by FilterNet (even columns). Results improve tonality, exposure, and color temperature while being respectful of human features and avoiding extreme stylistic changes.



will start by presenting the set of metrics that will be used as a reference for comparisons.

5.1.1 Metrics

The selection of a suitable set of quantitative metrics to compare the quality of images is a difficult problem, particularly when the desired measure seeks to reproduce the human judgement of quality [48]. Part of the problem is that some mistakes are worse than others, even if they yield similar metrics. For example, a photography system that occasionally proposed unnatural skin colors would be automatically rejected by human users, whereas the same system producing a slightly too vivid foliage tone could be tolerated, or even preferred by some.

Despite the difficulties, it is undoubtedly important to measure results quantitatively, in order to produce objective—even if not perfect—figures that can be used for comparability.

For this work, we chose several image quality metrics which are briefly presented below.

- **PSNR** (Peak signal-to-noise ratio) and **SSIM** [130] (Structural Similarity) were selected because they are frequently used as baseline references.

TABLE 5.1: FilterNet Experiments: performance metrics and ablation study.

		PSNR	SSIM	FSIMc	Content	DISTS
		score ↑	*score* ↑	*score* ↑	*error* ↓	*error* ↓
ebcwh ResNet-34	baseline	23.421	0.912	0.956	402	0.0885
	+feat_loss	22.852	0.915	0.960	348	0.0843
	+feat_loss strong	23.696	0.924	0.968	294	0.0746
	+fp16	23.164	0.917	0.963	327	0.0804
	+linear_RGB	23.771	0.925	0.967	308	0.0759
tebcwh ResNet-34	baseline	23.394	0.919	0.967	329	0.0782
	+small_init	23.415	0.923	0.969	316	0.0760
	+maskout	23.836	0.927	0.970	302	0.0744
	+moderate	24.206	0.928	0.970	292	0.0712
tebcwh ResNet-50	baseline	*25.160*	*0.934*	*0.973*	*254*	*0.0614*
FilterNet	+fine_tuning	**25.483**	**0.936**	**0.974**	**249**	**0.0598**

Unfortunately, PSNR does not match perceived image quality well [131, 132]. SSIM is better in that regard, but the rest of the metrics in our selection provide additional insight.

- **FSIMc** is the per-color-channel variant of FSIM [133] (Feature Similarity Index for Image Quality Assessment), which has a higher correlation with perceived quality than PSNR or SSIM [134].
- **Content** refers to the so-called content-based loss introduced by Gatys et al. (2015) [47]. It uses the outputs from layers of a VGG network as features to compare, a method that highly correlates with components of the loss function we used to train our system. It has also been recently shown [48] that features extracted from deep networks can be effectively used as a perceptual metric.
- **DISTS** [135] (Deep Image Structure and Texture Similarity) is a recent index that correlates well with human perception of image quality.

Because of their higher correlation with human-perceived image quality, we consider both *DISTS* and *Content* the reference metrics for this research.

In order to evaluate the models, we created a test set consisting of 256 images that were selected following the same process described in Section 3.3. The reference test images underwent a transformation (to make them worse) that applied the same 6 filters mentioned in Section 3.5: temperature, exposure, brightness, contrast, highlights, and shadows. The transformation strategy for the test images was the one we refer to as *moderate*, as discussed later. The evaluation procedure applied each version of the model (in inference

mode) to the 256 transformed images, and compared model results against the reference images using the 5 metrics under consideration. Images in the test set were not part of the training or validation sets in any of the experiments performed.

The next section presents the results we achieved and discusses the techniques and parameters that most contributed to the quality of the system.

5.1.2 Results, Ablation Study

Our proposed system is designed to learn how to improve images using 6 filters or variables. It leverages transfer learning to take advantage of the feature extraction capabilities of well-known computer vision architectures like ResNet. As previously discussed in Section 3.2, we focused on the ResNet architecture for homogeneity and easier analysis, and considering their success in multitude transfer-learning projects across various tasks and domains. Our goal was to create a working system to enhance full-resolution photographs by leveraging transfer-learning in a self-supervised setting, and to extract general conclusions from the process. The quest for state of the art performance in this scenario was out of the scope of this work.

Among the ResNet variants, we found that ResNet-50 offered the best quality, while still being of an *affordable* (as discussed in Section 5.1.5) size. We also performed extensive testing on ResNet-34, because this shallower architecture allowed us to complete a large number of experiments and variations in a reasonable amount of time. This made it possible to quickly draw conclusions about the contribution of each of the components of the solution. Our experiments show that results attained in ResNet-34 extrapolate to the deeper ResNet-50 architecture, and quality improves as a consequence of the increased number of training samples, parameters, and training time.

Shallower architectures such as ResNet-18 were discarded because they were not able to faithfully capture the nuances of the transformations we applied, yielding worse quality. Deeper variants such as ResNet-101 were discarded for practical reasons: their size was deemed impractical for use in a mobile application because of space and speed requirements. We expect our training approach and methods to achieve better quality in those deeper architectures, given sufficient training samples.

Table 5.1 summarizes results from our most relevant experiments, measured on the test set previously discussed. It shows the contribution of the main

components of the solution, as well as the improvement of ResNet-50 over ResNet-34.

The first few rows describe results on models trained to predict 5 variables (*ebcwh*, for *e*xposure, *b*rightness, *c*ontrast, shado*w*s, and *h*ighlights) instead of 6 (same, plus *t*emperature). This is a simpler version of our task that allows us to assess the impact of increasing the complexity of the model and dataset. The color temperature filter (t) was excluded because its implementation is the most different from the rest of the filters: exposure (e) uses a mathematical equation, and all of brightness (b), contrast (c), shadows (w), and highlights (h) are based on differential curves with alpha-blending, as described in Section 3.5. In addition, the gain factors used in the implementation of the color temperature filter were calculated in linear color space using 10-bit precision, so the filter can only be applied if the network supports the same environment.

These first few rows, therefore, are useful to establish the baseline performance of the system and introduce the impact of some significant components of our proposed solution. The use of *feat_loss* refers to the combination of loss components ℓ_f and ℓ_g, as described in Section 3.6. Rows 2 and 3 show a clear improvement over the model in row 1, which was trained using ℓ_p only. Furthermore, increasing the weights of both ℓ_f and ℓ_g translates into a better model performance. The weights used for *feat_loss base* were:

$$\boldsymbol{w}_f = \boldsymbol{w}_g = (5, 15, 2)$$

whereas the ones used in the *strong* version were:

$$\boldsymbol{w}_f = (15, 30, 12)$$
$$\boldsymbol{w}_g = (6, 15, 3)$$

The use of mixed-precision training [136] takes advantage of specialized computing cores optimized for 16-bit floating-point operations, while at the same time decreasing the requirements of GPU memory (or, alternatively, allowing the use of larger mini-batches). In our experiments we observed slight out-of-range issues in our custom rendering layers when mixed-precision training was in use: pixel values were sometimes smaller than 0 or larger than 1. We interpreted this problem as a consequence of the accumulation of rounding

errors caused by precision loss, and worked around it by simply clamping the values. The *+fp16* row in Table 5.1 shows a performance decrease in this training regime.

The incorporation of filter computation in a *linear RGB* color space has a positive impact on quality, for the reasons anticipated in Section 3.5.

The introduction of the temperature filter, which was held out in the experiments previously mentioned, again shows a decrease in quality. This was expected, for the following reasons:

- The additional complexity of the model, despite using the same number of parameters.
- The training dataset was changed with respect to the experiments involving 5 variables. A color temperature transformation was added to the set of modifications applied to originals, so the resulting training images are, in general, *worse* than the training images used in the 5-var experiments. The number of training images remained equal. In both cases, we ran 10 transformation processes using either 5 or 6 filters, generating a total of 48850 training images.
- Training hyperparameters like batch size, learning rate, or training epochs were adjusted to account for the characteristic of the new, more complex model. Our general rule for these experiments was to use a batch size that maximizes GPU memory usage, and heuristically select an appropriate learning schedule based on the evolution of validation loss.

The experiment referred to as *+small_init* measures the contribution of a custom initialization method applied to the parameters of the linear layers that predict the values of filter parameters. The go-to initialization method for this type of architecture would have been the use of the Kaiming Normal distribution (also known as He initialization) [137]. However, we decided to initialize layer weights with values randomly close to zero, by drawing from a normal distribution with mean 0 and a small standard deviation of 0.01.

The motivation behind these changes was the observation that initialization using the Kaiming Normal method tended to produce extreme values for the predicted filters during the initial training epochs. The new initialization method produced values close to the mean of the filters, causing the training process to converge faster.

As we discussed in Section 3.4, *maskout* is a technique similar to dropout, which we use to randomly disable the output of some of the predicted filters. The method was introduced after observing that the model's predictions showed a smaller amount of variance in some of the filters.

The row *+maskout* in Table 5.1 shows the effect of adding a maskout layer following the predictive layer, with a 0.1 probability of maskout.

The dataset preparation tasks also play a role in the ability of the model to learn and generalize successfully. As discussed in Section 3.3, we create our training images by applying random transformations to good quality photographs. The magnitude of those transformations has a direct impact on the intensity of the filters that the model needs to apply in order to revert the modifications. But there is an additional consideration: if transformations are too strong, we may reach the point where changes are not recoverable because we incur in significant information loss. For example, increasing exposure in a light area may cause all colors in that area to clip, turning completely white. This means we lose the texture and subtle gradation previously present, because all the transformed pixels are represented as pure white. Beyond this point, it does not matter how much is the amount of exposure increase, because the transformed result will be the same.

We experimented with two transformation regimes: a *strong* and a *moderate* one. In the *strong* transformation approach, the ranges for the 6 filters are wider and, furthermore, the effects of all 6 filters are combined together. This is useful for experimentation because the network is forced to find high-magnitude answers and we can easily compare the impact of the changes we perform. The training dataset, however, is not representative of actual photos users may take, because the ones in our dataset have a random chance of having been excessively transformed, sometimes to the point of not being recognizable.

The *moderate* training set was created by decreasing the probability to apply excessive or unnatural transformations. Our approach was as follows:

- The ranges of some filters were reduced. For example, temperature was uniformly chosen between $[4000K, 10000K]$, whereas the corresponding range in the *strong* method was $[2700K, 15000K]$.
- The randomly-selected filter values were scaled by values drawn from $\mathcal{N}(0.7, 0.3^2)$, the normal distribution with mean $\mu = 0.7$ and standard

deviation $\sigma = 0.3$. Scaling makes high-magnitude values possible but less frequent.

We saw a clear increase in quality when the *moderate* preparation scheme was applied. This is partly due to the fact that information loss is largely prevented, and also because the filter values predicted by the network have more latitude to adapt to the combination of changes seen in the input samples. We also need to keep in mind that the test set was created using the *moderate* approach, because it is closer to the type of photos we are expected to find in the wild.

Our proposed system takes advantage of the components and techniques discussed in the previous paragraphs and applies them to a ResNet-50 architecture. The increased depth and larger number of trainable parameters translate into a quality increase, as shown in the last 2 rows of Table 5.1. To account for the increased number of parameters, we used a larger training set, too. Most of our ResNet-34 experiments used 10 transformation runs over the $4,885$ original photographs, yielding a training set close to 50K image pairs. In the case of ResNet-50, we applied 30 transformation runs, creating a training set three times as big as the previous one. In both cases, 20% of the image pairs were reserved for validation.

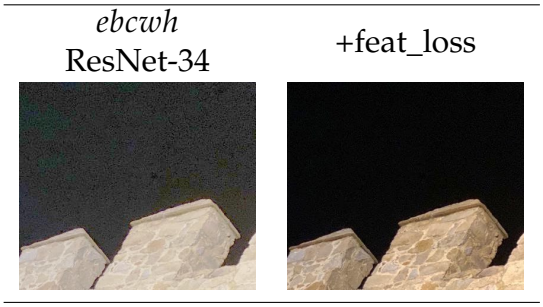
We also compared the performance of the model with and without a full fine-tuning phase. In the absence of full fine-tuning, we only train the prediction head while the parameters (weights and biases) of the backbone network remain *frozen*, meaning they are not allowed to change. Training in such a way for a total of 10 epochs results in a model that is noticeably better than any of the fine-tuned ResNet-34 models. Adding a fine-tuning phase of 5 epochs improves quality, as the backbone adapts to the characteristics of our dataset. Even though the fine-tuned model is better, the use of a standard, unmodified backbone offers some advantages when a space/performance trade-off is considered, as briefly presented in Section 5.1.5.

5.1.3 Qualitative Examples

As briefly mentioned in the previous subsection, numbers do not tell the whole story. Looking at the figures in Table 5.1, it would appear that each component did not make a dramatic contribution to the overall quality. Even when all aspects of the solution are taken into account, the relative differences between the best model and the baselines are moderate. Looking at just PSNR, for example, the difference between the best and worst scores is just about

FIGURE 5.2: Results from two models (with and without feature loss) on underexposed images from the validation set. Both models produce acceptable results.

FIGURE 5.3: Qualitative example of the benefits of adding a feature-based component to the loss function, on a detail extracted from a test photo. A model without feature-loss attempted to make the whole image brighter, while the second model successfully prevents artifacts in the sky.



11%. However, looking at the images transformed by the parameters learned by the models, we see important differences and artifacts that humans would deem as unacceptable.

For example, Figure 5.2 shows a comparison of the very first two systems in Table 5.1 on two underexposed images taken from the validation set. The first column is a basic 5-var model with pixel-based loss, while the second column is the result predicted by the same model after introducing a feature-based or *perceptual* component to the loss function, as described in previous sections. Both models successfully understand that the images are underexposed, and apply filters to compensate for it.

However, Figure 5.3 shows what happens when the same models are applied to a real photo taken with an iPhone, where a stone wall is illuminated against a dark sky. The first model blindly attempts to increase exposure and brightness, while the second model refrains from doing so. The artifacts

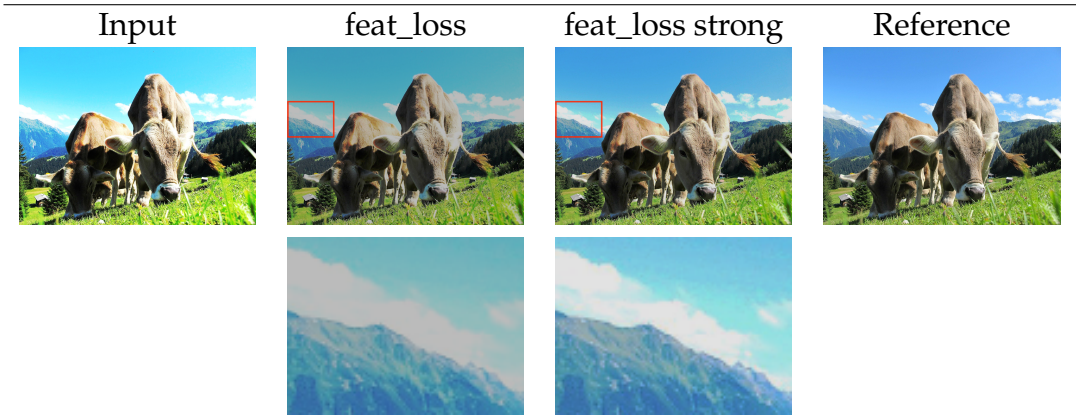
introduced in the sky texture by the first model are unrealistic and distracting, detracting from the experience. The feature-loss model is more conservative in this case, but it does not destroy an otherwise acceptable photograph. Looking again at the first two rows in Table 5.1 we see, however, that the quantitative metrics show a similar performance when averaged over the complete test set.

This exemplifies why during the course of our work we resorted to frequent qualitative analysis of results. We held internal team meetings to analyze the merits and shortcomings of various models when applied to a common set of images. The sessions involved six persons with different profiles in engineering, marketing, and photography.

Figure 5.4 elaborates further on the impact of feature-based loss, showing results when trying to improve an image that underwent an extreme overexposure transformation. Adding a strong feature-based component to the loss function results in much better color and contrast, compared to the version when just a moderate feature-based component was used. However, in order to achieve that goal the model had to sacrifice texture fidelity in the area where the sky joins the clouds and the mountain. Due to the excessive overexposure that was applied, there was information loss that can never be recovered, which produces banding artifacts when attempting to improve tone and contrast. The original picture, shown in the last column of the table, is no longer recoverable from the transformed input. Different forms of the loss function drive the network to learn different approaches: an overall dull look that is possibly attempting to reach a compromise across the transformation range, or a better approximation to color despite introducing artifacts.

The treatment of skin tones and faces is crucially important for human-perceived quality. No matter what the quality metrics say, unnatural skin colors or unrealistic face textures would render the whole image unacceptable, even if the rest of the subjects look great. Figure 5.5 shows how different models attempt to improve a difficult photo, in which the main subject is lit and vibrant but the overall scene was taken in the dark. A naïve regression model tries to overcompensate for tones and colors and results are completely unacceptable. Our proposed system, FilterNet, is perhaps the most conservative of all, but it does not attempt to replace the dramatic mood of the original with a different style and, notably, attempts to reduce the excessive vividness of the main subject's skin color — the original was shot using Fujifilm's Velvia film emulation style, which resulted in accentuated red tones.

FIGURE 5.4: Model trade-offs attempting to correct an image with extreme overexposure. The bottom row shows texture details on the area highlighted in the top row.



5.1.4 Mobile Performance

FilterNet has been incorporated as a feature in Camera+ 2, a photography application for iOS. As mentioned in previous sections, the model runs locally in the device to reduce latency, increase inference performance and guarantee privacy. In order to optimize the model for both performance and space, we converted the trained PyTorch model to the native Core ML format used in iOS, and quantized all weights to 16-bit floats.

Table 5.2 shows inference figures measured in various iPhone models, using the CPU, GPU or dedicated Neural Engine where available. The time for the first prediction takes into account model loading from the filesystem, model preparation, and copying to the appropriate inference engine. After these steps have been performed, subsequent inference times are significantly improved. As the table shows, we achieve sub-second inference times in permanent mode for all devices.

We decided not to apply a pre-warm strategy to minimize the time to the first prediction, in order not to incur in unnecessary resource usage. Instead, we use tasteful animations to let the user know that predictions are being computed.

These performance figures are adequate for synchronous use and user exploration. The enhancement feature is available in the two most prominent sections of the app: the camera system, and the photo editor. When used in the camera, all photos captured are automatically adjusted with the predicted filter values, but the changes are not destructive so the user can later tweak

FIGURE 5.5: Results from different models on a human-centered subject. The input is the original captured photo, with no transformations applied.

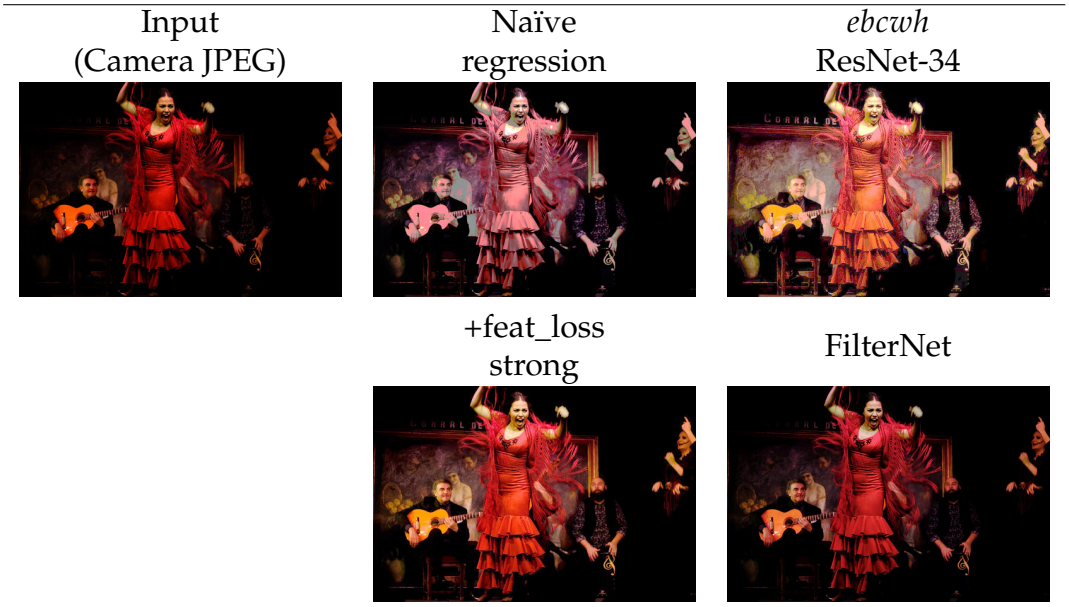


TABLE 5.2: Inference time measured in devices from 2020 and 2021, using the GPU / Neural Engine for inference. All values in seconds.

	iPhone 12 Pro 2020	**iPhone 13 Pro** 2021
First prediction	2.9	2.8
Subsequent predictions	0.47	0.37

or revert them. When accessing the feature from the editor, users can easily compare the before / after versions of the applied filters.

5.1.5 Space-Performance Trade-offs

Figure 5.6 shows the quality achieved by our system when using various backbone architectures. Quality is represented as a percentage of the DISTS score achieved by the best-performing model. In addition to the backbones already discussed, we also introduce results from a much smaller SqueezeNet 1.1 backbone [138], optimized for environments with space or download constraints, such as mobile devices.

In our case, we wanted to deploy the final model to the users' mobile devices (rather than hosting it in a cloud service), in order to comply with our requirements regarding user privacy and prediction latency. In such a scenario,

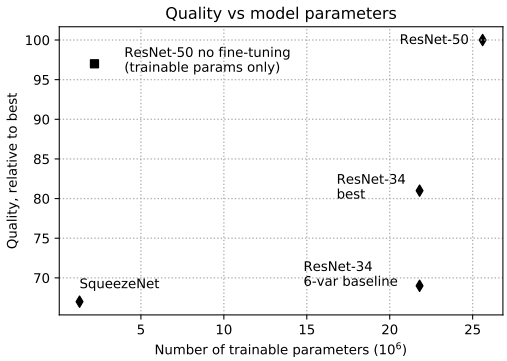


FIGURE 5.6: Quality vs number of trainable parameters for different architectures. ResNet-50 with no fine-tuning requires a fraction of the trainable parameters of the full architecture.

model size has an impact on several aspects that may affect user experience: download size, storage space, and computation time.

Figure 5.6, therefore, represents different trade-offs in the quality versus size space. Note that SqueezeNet is comparable in quality with our *6-var tebcwh ResNet-34* model, but is much lighter. Also note that, in some situations, the appropriate metric to consider is the number of *trainable* parameters, because some model backbones may already be available in some devices, or the same architecture may have been previously downloaded for other purposes. If a ResNet-50 backbone is already available, then the incremental cost of deploying a FilterNet system could be dramatically reduced if no fine-tune is performed on the backbone parameters. This version is slightly worse than the fine-tuned one, but could be more convenient in scenarios like that.

5.2 CocoGold Experiments

The following subsections present experimental results for CocoGold, our text-grounded segmentation approach based on generative diffusion models, and demonstrate the viability of repurposing pre-trained diffusion models for this task. Focusing on simplicity, our current proof-of-concept implementation uses single-word category prompts rather than full referrring expressions, but it establishes a framework that could be used for more complex subject descriptions.

We also combine FilterNet, our global filter-predicting model, with CocoGold, and show that a simple integration approach can apply localized automatic edits on regions of interest specified by segmentation masks that were extracted

from text instructions. This method, despite using generative models, is respectful of the original image data – results are not *generated*, even in the segmented area –, and can be applied to large resolution images on constrained devices.

5.2.1 Experimental Setup

5.2.2 Segmentation Evaluation

The CocoGold task differs from the well-described instance, semantic, and panoptic segmentation tasks [139, 140, 141] in some relevant ways.

This is how the reference COCOeval framework evaluates instance and semantic segmentation:

- For instance segmentation, models are asked to predict each of the objects belonging to the set of 80 known classes, assigning a class and confidence value to each mask. Multiple objects that belong to the same class are identified as separate instances.
- For semantic segmentation, the model must assign a class to every pixel in the image, including "stuff" classes for background items (like *sky*). Objects that belong to the same instance are not identified separately.

COCOeval judges models that see the full scene on how completely and precisely they segment it.

CocoGold, in its current implementation, uses a simplified setup where a text prompt specifies a class, and produces a binary mask of all objects that belong to that class. In this sense, it's similar to semantic segmentation in that individual instances are not separated, but it does not attempt to yield a full-image per-pixel classification. More importantly, segmentation is conditioned on text inputs: we ask the model to give results for the prompt we specify, and therefore the model knows to pay attention to regions that may contain objects of that class. In addition to that, we train the model on crops known to contain (at least partially) objects of the class we condition the model on, and we follow the same dataset iteration procedure for the evaluations presented in this section.

The combined impact of these factors—text conditioning, single-class results, no false negative inputs—makes this task, we believe, easier than the usual COCOeval process; or at least not comparable in terms of numerical results.

Training Steps	IoU	Pixel Accuracy (%)
8000	0.662	91.2
18000	0.664	90.4

TABLE 5.3: Performance on trained classes (no ensembling).

Training Steps	IoU	Pixel Accuracy (%)
8000	0.165	79.4
18000	0.241	77.2

TABLE 5.4: Zero-shot performance on unseen classes.

We believe, nevertheless, that the task is sufficiently different to instance or semantic segmentation to undergo a separate evaluation procedure that measures the adherence to the prompt, the quality of the masks, and the capability to generalize to unseen classes.

In its current implementation, CocoGold is more similar to text-to-mask open-vocabulary models such as CLIPSeg [63] or the GroundingDINO + SAM pipeline combination we've mentioned before.

We report IoU [142] (Jaccard index [143]) and pixel accuracy using the following protocol:

- We evaluate on the full COCO 'val' dataset, including the 14 categories we used for training, as well as the ones we excluded.
- We evaluate two different models: one after 8,000 training steps, and another one after 18,000 steps, with the goal to verify whether longer training helps with zero-shot generalization.
- We accummulate confusion metrics (TP, FP, FN) across all instances, and then compute IoU and pixel accuracy for:
- – Instances that belong to classes seen during training.
  – Instances belonging to *Other* classes, to measure zero-shot generalization.

Following this protocol, larger detected masks will carry more weight into the final IoU metric. Similarly, large foreground or background areas will contribute more to pixel accuracy.

Table 5.3 shows IoU and pixel accuracy on the set of trained classes, while table 5.4 shows zero-shot performance on the rest of the COCO classes the model never saw examples for. Generalization performance lags behind trained classes, but we can see that longer training improves zero-shot performance. We hypothesize that varied text inputs, not constrained to single-world category names, combined with longer training runs and more exhaustive datasets, could achieve closer results on zero-shot performance.

While CLIPSeg is conceptually similar, it's evaluated on more challenging conditions: it follows a multi-object setup, includes negative examples and, using CLIP, is capable of multi-word descriptions. It also reports mIoU rather than global IoU. By construction, our evaluation protocol selects crops known to contain the desired class in a relatively prominent way. To make fair comparisons against CLIPSeg we could evaluate it on our dataset using the same protocol; perhaps after fine-tuning to ensure it *sees* the same classes we are using.

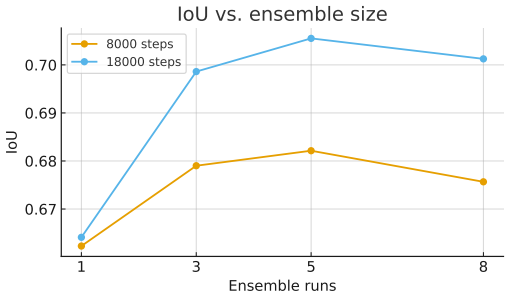
The metrics we obtain are meant as baselines for additional experiments and future work. For example, following Marigold, we measure the impact of ensembling.

5.2.3 The Impact of Ensembling

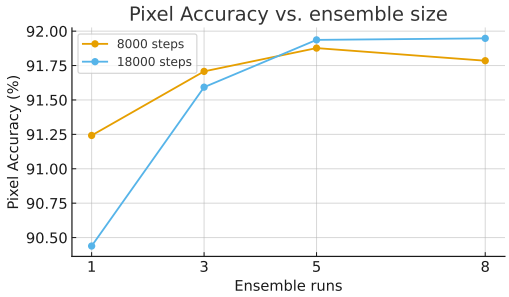
Following Marigold, we evaluate whether ensembling helps achieve better quality. The reasoning behind ensembling is that using a generative diffusion model to predict segmentation masks (or depth maps, in their case) is inherently stochastic. By running the diffusion process several times, we can choose a majority vote and see if those pixels match the ground truth better.

We ran ensembling experiments for the two models we mentioned in the previous section, considering ensembling sizes of 1 (no ensembling), 3, 5, and 8. Fig. 5.7 shows that ensembling only provides modest quality improvements.

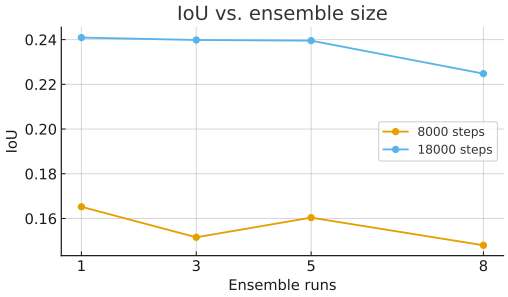
The reason, we believe, is that errors in binary mask segmentation results are more forgiving than errors in depth estimation, where each pixel must be assigned a float value. In our case, we think that small errors fixed in certain areas have low impact on the total score, and cancel out with new small errors introduced elsewhere.



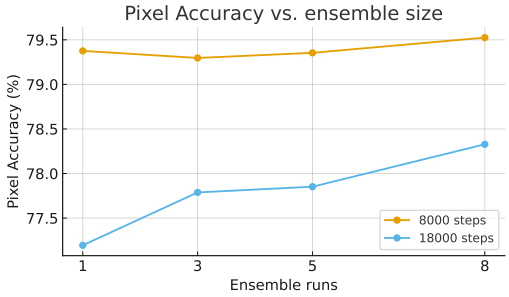
(A) Trained classes IoU



(B) Trained classes Pixel Accuracy (%)



(C) Zero-shot IoU



(D) Zero-shot Pixel Accuracy (%)

FIGURE 5.7: Performance vs. ensemble size (1, 3, 5, 8) for models after *8000 steps* and *18000 steps*.

5.2.4 Few Steps Regime

Diffusion models require multiple iterations to progressively refine their predictions. Despite training being typically based on 1,000 noise prediction steps, research on noise schedulers allows inference to be performed in 50 or fewer steps. This represents a huge gain for inference, but is still a bottleneck for interactive, on-device use cases.

Recent research pointed out that diffusion inference usually presents a mismatch with the conditions the model saw during training [102]. In particular, inference is not always guaranteed to start from $t = T$ (pure noise); and even so, the SNR at $t = T$ is not 0 (i.e., the input signal at the start of the diffusion process is not exactly *pure* noise).

Following these observations, we experimented using the so-called *trailing DDIM* scheduler to run CocoGold inference in the low-step regime. To give precise context about the change, we changed the default scheduler at inference from standard DDIM to trailing DDIM with zero terminal snr, using the following code:

LISTING 5.1: Quick example

```
# Load cocogold pipeline
```

```
pipe = load_cocogold_pipeline(unet_checkpoint,
    sd_checkpoint)

# Replace scheduler with trailing DDIM
pipe.scheduler = DDIMScheduler.from_config(
    pipe.scheduler.config,
    rescale_betas_zero_snr=True,
    timestep_spacing="trailing",
)
```

This configuration worked out of the box with our pre-trained CocoGold models, with no additional fine-tuning.

We evaluated CocoGold using this setup and following the same protocol described in Section 5.2.2, without ensembling. We run inference for as few as 1, 2, and 4 steps, and observed:

- Few-steps inference does not work at all with the standard DDIM scheduler, but it produces reasonable results when using trailing DDIM. See Fig. 5.8 for example visualizations.
- On the 18,000 steps model, IoU for trained classes is as good as standard DDIM with **just a single** trailing DDIM denoising step, as shown in the left panel of Fig. 5.9. IoU is even better using 4 trailing DDIM steps. Zero-shot performance, however, is slightly worse than DDIM, as shown in the right panel of Fig. 5.9.

5.3 COCONET: Integrating FilterNet with CocoGold

We explore the integration of FilterNet (our global, automatic, resolution-agnostic photo enhancement system) with CocoGold. We call the combined system COCONET, whose task is to automatically enhance subjects identified by segmentation masks that were extracted from text instructions. This is a proof-of-concept prototype that demonstrates how the filters predicted by a global enhancement filter can be applied locally to regions identified semantically.

5.3.1 Integration and Qualitative Results

We consider two simple, straightforward implementations:

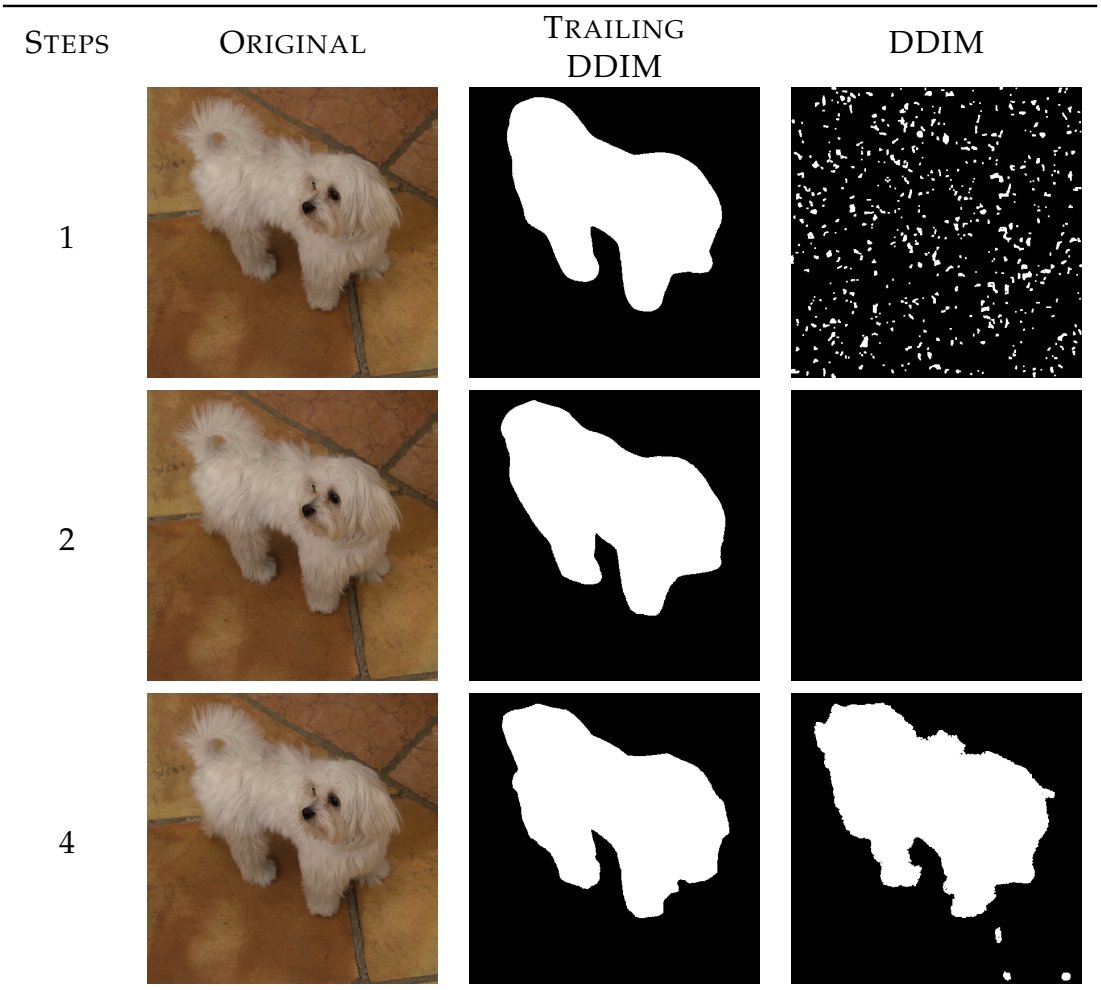
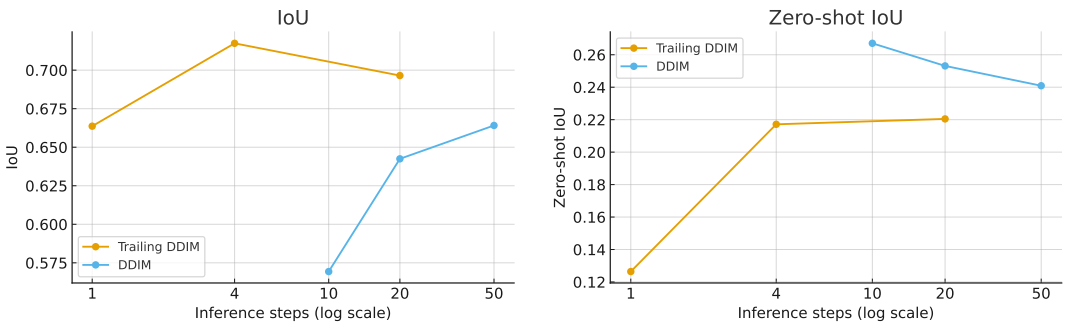


FIGURE 5.8: Low-step regime for trailing DDIM and DDIM denoising schedules. Initial random seed not fixed across runs.



(A) IoU (trained classes)

(B) Zero-shot IoU

FIGURE 5.9: IoU vs. denoising steps (log scale) for model trained after 18,000 steps. No ensembling. Legends show *Trailing DDIM* at 1, 4, 20 denoising steps, and *DDIM* at 10, 20, 50 denoising steps.

A) We apply FilterNet to the whole image, extract the global enhancement filters and apply them. From this edited version of the photo, we extract the area that matches the segmentation mask, and blend this region on top of the original image.

B) We cut out the bounding box defined by the segmentation mask, apply some padding to provide more context, and run FilterNet on this area. The filter parameters it predicts, therefore, will be different than the ones it would choose for the whole image, and will focus on the area that surrounds the subject of interest. We apply these filters to the photo, and like in the previous case, cut the segmentation mask area and blend it on top of the original image.

In both cases, we feather around the edges of the segmentation mask with a Gaussian blur, in order to make transitions less abrupt. These helps the edits blend better in the original context of the photo.

Note that we didn't attempt to cut out the subject delineated by the segmentation mask and present it in isolation (or against a neutral background) to FilterNet. This is because this type of data would be too much out of domain of the training samples, whereas FilterNet is used to look at complete, real-looking photographs.

Fig. 5.10 shows some examples from COCONET using both methods (FULL and BBOX) we just outlined. The combined pipeline successfully corrects problems such as underexposed subjects or color temperature, leaving the background untouched. When examining the edited images at 100% size, we observe the edited filters sometime extend beyond fine textures such as hair strands.

5.3.2 Demo for Experimentation

To quickly test our method on real-life input images, and to share with the research community, we wrote an interactive demo to explore results easily. We used gradio for rapid prototyping [144], and included options that allow the user to configure the most relevant knobs that influence editing. As Fig. 5.11 shows, the user can:

1. Upload an image, enter a prompt, and run inference with the two methods, seeing the results side by side.

2. Examine the segmentation masks, and the filter values predicted by FilterNet in each case.

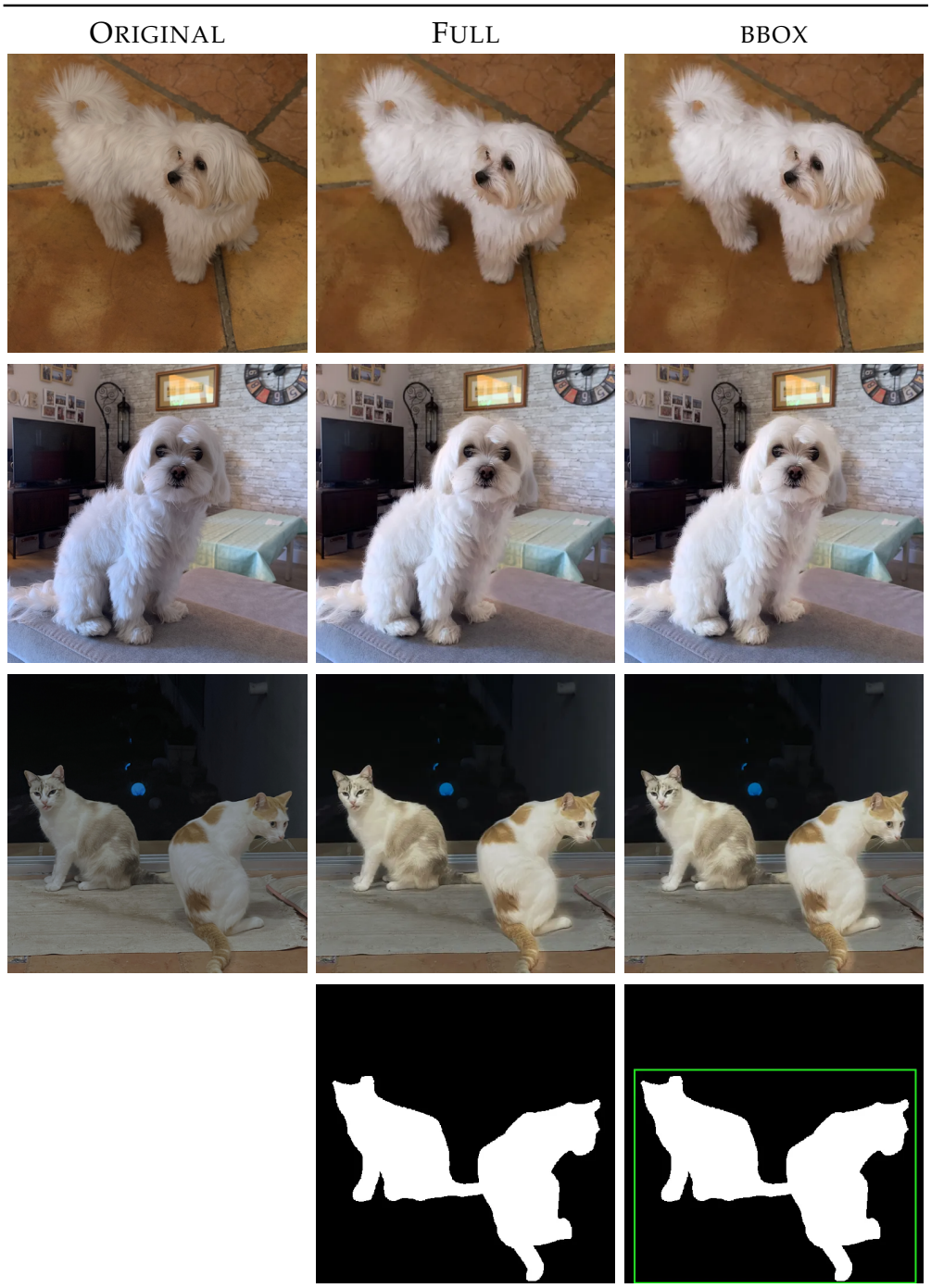


FIGURE 5.10: A few COCONET examples, using integration methods FULL and BBOX. Segmentation masks are shown for one of the examples. With method FULL, FilterNet looks at the whole image to predict filter enhancement values. With BBOX, it only looks at the bounding box outlined in green.

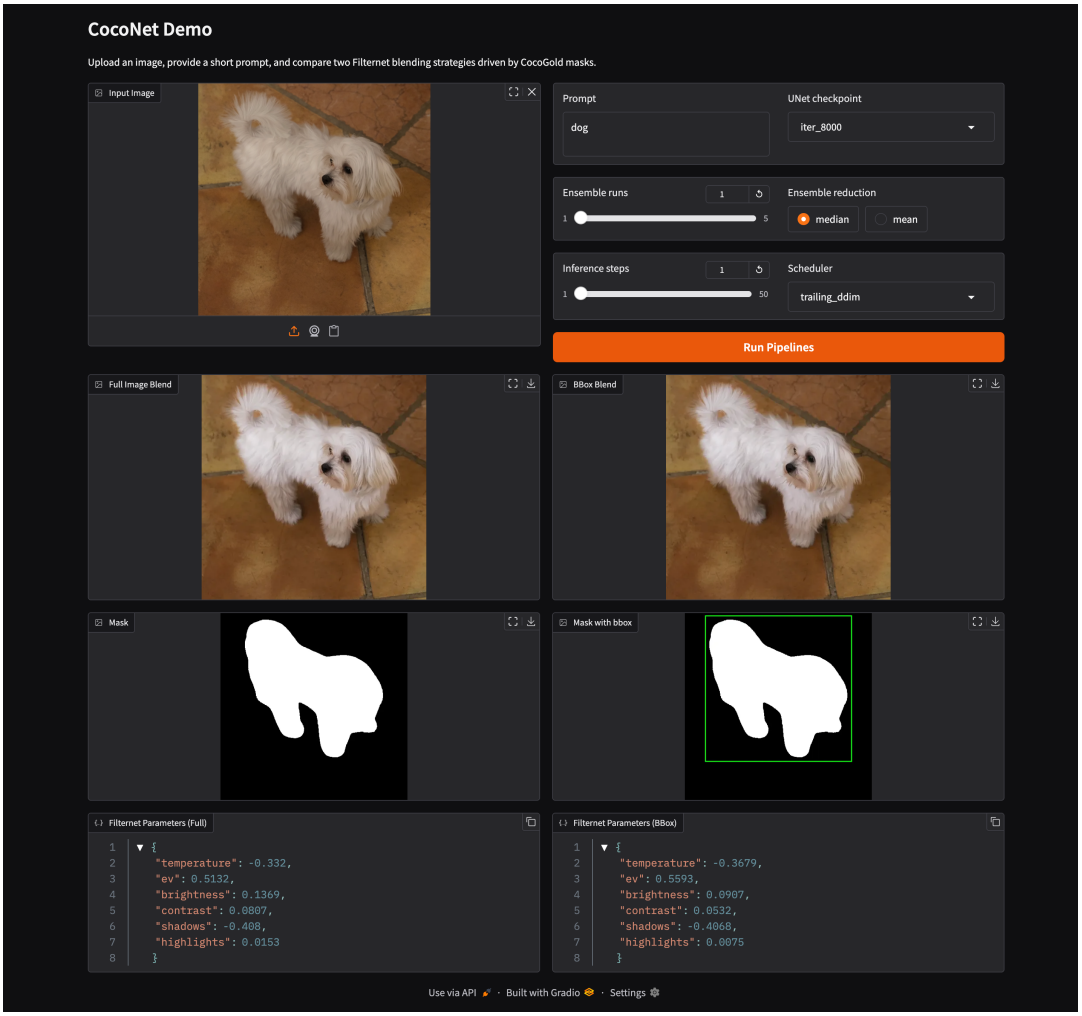


FIGURE 5.11: Interactive demo for COCONET.

3. Select the model checkpoint to use.
4. Configure the number of ensemble runs to use, the number of inference steps and the noise scheduling strategy.

5.3.3 COCONET and arbitrary resolution editing

All the components in the COCONET pipeline, except the segmentation mask, are designed for arbitrary-resolution editing. FilterNet, for example, predicts filter values from a small image thumbnail. Those predicted values can be easily applied to images of any size, and the same thing is true for COCONET. CocoGold, however, generates a segmentation mask as part of a generative diffusion process, and is not resolution independent. This is a limitation of the global editing pipeline, but it can be alleviated through various mechanisms:

1. We used 512×512 input images for faster training. Stable Diffusion 2 natively supports images of 768×768, and can be fine-tuned to larger sizes as subsequent models demonstrated.

2. The particular case of segmentation masks has been studied, and there exist algorithms for high-quality mask upsampling. Some of these methods have been implicitly demonstrated in real-world applications. Apple frameworks, for example, contain a 'CIEdgePreserveUpsample' filter that is capable of upscaling a low resolution depth map captured by the camera, to full-size photo resolution for editing. We don't know the specifics of this filter, but it may involve image-guided, edge-aware methods.

3. In any case, segmentation masks obtained by CocoGold are not perfect, and cannot faithfully represent fine details such as hair strands and other textures. No upscaling mechanism will be able to overcome these limitations. As DiffDIS showed [97], the introduction of edge aware components in the loss function could greatly improve the representation of these fine details. Given a theoretically *perfect*, but low resolution segmentation mask, the use of edge-aware upscaling, combined with feathering around the edges, may suffice to produce excellent visual results with negligible artifacts.

Even though COCONET can work well at reasonable sizes, it doesn't support arbitrary resolution rendering as the rest of the pipeline does. The generation of truly resolution agnostic segmentation mask is an open challenge, and could potentially be solved using tiling or models capable of vector output. These are lines that remain open for future work.

Chapter 6

Conclusions and Future Work

This chapter presents the main conclusions from our work and reviews them against our goals to advance photo editing with intuitive, interpretable, and resolution-independent methods. We also share the main artifacts derived from our work in these topics, which include peer-reviewed research papers, use in industry, and other materials. We finalize with an outline of research directions that arise as natural continuations of the work done so far.

6.1 Conclusions

The first part of this thesis contextualizes our research efforts within the field of deep learning for photo editing. Our goals to achieve resolution independence, low compute budget for training and inference, and lossless interpretable edits that are familiar to photographers, culminated in the creation of FilterNet. FilterNet is a deep learning model that predicts photographic filters that enhance the appearance of a photo, rather than generating edited pixels at once. It examines the photo globally and the filters it predicts are meant to be applied globally as well. The system focuses on neutral, technical photographic improvements, and while it is capable of fixing characteristics such as exposure, color, or contrast, it cannot be used to apply arbitrary stylistic choices, or to act locally on image regions the user is interested in.

The main contributions of FilterNet center on two innovations.

First, we adapted the self-supervised training paradigm informally known as "crappification" to the domain of photo enhancement, where high-quality images are deliberately degraded using random combinations of photographic filters to generate training pairs. This approach eliminates the need for manually curated datasets of enhanced photos, enabling the model to learn enhancement by reversing the degradation process. Training employs a composite

loss function that integrates pixel-level reconstruction, perceptual similarity based on VGG features, and style consistency to ensure that the enhanced outputs preserve both technical fidelity and aesthetic coherence.

Second, we developed a set of six differentiable filters corresponding to fundamental photographic adjustments familiar to practitioners: exposure, color temperature, brightness, contrast, highlights, and shadows. Unlike most machine learning approaches to image manipulation, which typically operate on gamma-corrected sRGB values, our filters operate in linear RGB, where pixel values are proportional to physical light intensity and thus suitable for mathematically and physically meaningful operations. The differentiability of these filters is essential, as it enables the neural network to learn optimal parameter values through standard backpropagation while maintaining full interpretability: users can directly inspect and adjust the filter parameters predicted by the model.

Two major research directions blossomed during our work in FilterNet, crystallizing after years of research. The first one is the success of auto-regressive language models (LLMs) as viable methods for UI interaction. Scale – model size, but most significantly, Internet-scale training datasets – and post-training methods that align responses to user preferences, contributed to an impressive jump in capabilities that allowed the use of these models as general-purpose tools for the public.

The second research breakthrough is also related to language, and includes vision as well. The idea to link visual and text representations was approached from various directions, and originated two major lines of research with immense practical applications. On one hand, combining a strong visual feature extractor with an LLM produced so-called vision-language models (VLMs). VLMs work like LLMs but can also understand images or videos provided as inputs. Suddenly, traditional computer vision tasks could be attempted via a universal UI: ask the VLM about it, and get text responses with the result. This setup is surprisingly flexible, and can even be adapted for geometry tasks (such as segmentation), by preparing or fine-tuning the LLM to respond with structured text or special tokens.

The second major direction we were referring to that combines vision with language, is the prolific development of generative models. They come in various forms and families, but we are considering their general capability to create a new image (or even video), based on a description of the desired output. These techniques are flexible enough to be adapted to related tasks,

including image modification, with an obvious impact on the general field of computer-assisted photo editing.

The idea to use language as a natural and intuitive UI, as well as the capabilities unlocked by generative models whose output can be controlled with text descriptions, are both very compelling towards our vision of photo editing democratization. We explored how these methods could fit within the self-imposed boundaries of our framework: limited compute, interpretability, tunability, and high-resolution output.

To make our research manageable, we focused on the use of generative diffusion models for text-grounded segmentation, where the user specifies the subject they are interested in and the model produces the associated segmentation mask. We worked under the hypothesis that diffusion models, having been trained on billions of image-text pairs, possess a rich understanding of the relationships between vision and language – including geometry and locality awareness – and can be adapted to other tasks. Our prototype, CocoGold, can be trained by fine-tuning Stable Diffusion 2 on a single GPU, and demonstrates that segmentation is a viable task for diffusion models to solve. One of the main insights from this effort is the reformulation of segmentation as an image-to-image generation task, where the model is asked to reproduce the original input image, but highlighting the areas that match the text description of the subject we want to segment. Attempting to generate a binary segmentation mask (rather than an image with mask overlays) was unsuccessful, as the task takes the model outside its training domain.

By combining FilterNet with CocoGold using simple methods, we demonstrate great progress in the extension of the filter-predicting method to local edits, addressing the global-use limitation of FilterNet. Coconet is our unified pipeline for text-guided local photo enhancement. It takes an image and a simple text term as inputs, estimates the segmentation mask that matches the text, and applies appropriate photographic adjustments (exposure, contrast, color temperature, etc.) to those regions. The predicted filter parameters remain interpretable, can be further tuned by the user, and can be applied at any image resolution. One limitation, however, is that the segmentation mask is a binary image at fixed resolution, so the end-to-end pipeline is not fully resolution independent. Our discussion in Chapter 4 outlines methods known to alleviate this issue for practical use cases.

Another limitation of CocoGold (and Coconet) is the use of simple text terms, derived from a subset of COCO categories, to condition the diffusion model

for segmentation. One obvious area of improvement is to expand this method to richer and more varied descriptions. We outline this approach in the next section about future work.

6.2 Future Work

We briefly describe two important research directions that could greatly impact the applicability of our method and bring it closer to our vision of intuitive photo editing: how to better identify the subjects to operate on, and how to capture user intent (including stylistic choices) rather than predicting technically "correct" filter values.

6.2.1 CocoGold: Expand for Open and Natural Descriptions

Future work on CocoGold should focus on expanding the segmentation component beyond the limited set of categorical text terms used during training. Achieving open-vocabulary segmentation requires training on larger and more diverse datasets that include a wide range of visual concepts, and adapting the model to associate visual regions with arbitrary textual descriptions rather than fixed class labels. To obtain natural and varied supervision, we can generate synthetic captions or referring expressions that describe the object or region to be segmented (for example, "the person on the left", "the red car", or "the shadow on the wall"), either by prompting a vision-language model or by templating over object metadata. This would expose the model to richer linguistic variation and contextual phrasing, enabling it to generalize beyond predefined category names and respond to natural user instructions in free-form language. Given the strong multimodal understanding demonstrated by modern diffusion models trained on billions of image-text pairs, we are optimistic that such an approach can extend CocoGold's segmentation capabilities to open, descriptive prompts without requiring fundamental architectural changes.

6.2.2 VLMs as Filter Predictors

Another promising continuation of this work is the exploration of vision-language models (VLMs) as predictors of photographic filter parameters, capturing user intent expressed with natural language. Following our existing approach to predict filter parameters rather than generating edited pixels, a VLM could take an input image and a natural-language instruction, and

predict seven interpretable parameters corresponding to exposure, brightness, contrast, color temperature, tint, highlights, and shadows. These parameters would then be applied in the same fashion as FilterNet, preserving interpretability and full-resolution output while enabling intuitive, text-based interaction.

Note that we consider the introduction of "Tint" as an additional filter to the FilterNet set. The reason is that it complements color temperature by compensating for green-magenta imbalances that temperature alone cannot correct, allowing the model to represent a wider range of white balance adjustments found in real photographs.

To get started with this approach, we would initially consider a global editing solution similar to FilterNet, but driven by natural language instructions.

Developing such a model requires a dataset linking natural-language editing intents to numeric filter values. Existing instruction-based editing datasets, such as MagicBrush [145], GIER [146], and OmniEdit [147], contain natural and diverse text prompts but focus on pixel-level modifications, often local or semantic. To bridge this gap, we propose combining several complementary strategies:

(i) Adapting the self-supervised "crappification" approach introduced in FilterNet, but applying it to natural combinations of global adjustments rather than fully random degradations. In this case, training pairs would be generated by simulating realistic photographic conditions (such as underexposure with low contrast, or warm color temperature shifts under tungsten lighting) so that each degraded variant remains plausible and can be described naturally in language.

(ii) Recovering approximate parameter values from existing edited image datasets, using differentiable filter fitting. This data could be used to ground the model on what the effect of editing parameters looks like, but would need to be complemented with semantics based on natural language instructions.

(iii) Selecting and labeling global edits from existing instruction-based datasets by combining a language-based classifier that filters text prompts describing whole-image adjustments with an inverse-rendering stage that estimates the seven filter parameters through differentiable optimization. The estimated parameters allow each accepted example to be expressed

in the same numeric form as the synthetic data, providing consistent supervision for training.

To obtain natural and varied language supervision, we could generate synthetic instructions by prompting an LLM to describe the intent behind each parameter change, following a strategy similar to InstructPix2Pix. The resulting dataset would pair realistic, diverse language with accurate parameter supervision, enabling efficient fine-tuning of lightweight VLM architectures that map textual editing instructions to interpretable global adjustments.

6.3 Contributions

We collect the research contributions completed during the elaboration of this thesis, detailing our involvement and contributions. We also enumerate a few non-peer-reviewed contributions to the community.

6.3.1 Research Contributions

- P. M. Cuenca-Jiménez, J. Fernández-Conde and J. M. Cañas-Plaza, *FilterNet: Self-Supervised Learning for High-Resolution Photo Enhancement*, in IEEE Access, vol. 10, pp. 2669-2685, 2021, doi: 10.1109/ACCESS.2021.3139778. [148]

  Impact Factor: 3.6 (2024).

  Contextualization: FilterNet research.

  Author contributions: Research approach, dataset collection, experiments design and execution, evaluations, paper preparation.

- M. Rodríguez-Ibáñez, F. J. Gimeno-Blanes, P. M. Cuenca-Jiménez, C. Soguero-Ruiz, J. L. Rojo-Alvarez, *Sentiment Analysis of Political Tweets From the 2019 Spanish Elections*, in IEEE Access, vol. 9, pp. 101847-101862, 2021, doi: 10.1109/ACCESS.2021.3097492. [149]

  Impact Factor: 3.6 (2024).

  Contextualization: Evaluation of autoencoder methods for semantic interpretation of short texts. We approached NLP representation learning methods to later explore their suitability to capture semantic descriptions of editing instructions or object selection, but we decided for deep learning methods and LLMs instead.

Author contributions: Evaluation of autoencoders for text analysis, contributions to dataset collection and pre-processing, execution of experiments, writing contributions.

- M. Rodríguez-Ibáñez, A. Casánez-Ventura, F. Castejón-Mateos, P. M. Cuenca-Jiménez, *A review on sentiment analysis from social media platforms,* in Expert Systems with Applications, vol. 223, pp. 119862, 2023, doi: 10.1016/j.eswa.2023.119862. [150]

  Impact Factor: 7.5 (2023).

  Contextualization: Systematic evaluation of representational performance of language models and in-depth literature review of progress until this moment in time. This was instrumental to gain intuition on LLM capabilities and limitations, which in turn carries over to VLMs (they combine a vision model with an LLM auto-regressive generator), and to diffusion models as well (they reuse pre-trained language models as text encoders).

  Author contributions: Methods analysis (Section 6), including design and execution of experiments for language models. Writing contributions to other sections, paper review.

- A. Marafioti, O. Zohar, M. Farré, M. Noyan, E. Bakouch, P. M. Cuenca-Jiménez, C. Zakka, L. Ben-Allal, A. Lozhkov, N. Tazi, V. Srivastav, J. Lochner, H. Larcher, M. Morlon, L. Tunstall, L. von Werra, T. Wolf, *SmolVLM: Redefining small and efficient multimodal models,* in Second Conference on Language Modeling (COLM 2025) [151].

  Conference Paper: COLM 2025.

  Contextualization: Analysis and design of VLMs for efficient execution on consumer hardware such as mobile devices.

  Author contributions: Consumer application for video and image understanding on iPhone, MLX modeling code for VLM architecture, contributions to related open source packages.

6.3.2 Use in Industry

We applied the learnings from FilterNet to the development of a feature we called Magic ML, incorporated to the popular Camera+ iPhone application. References:

- Example press coverage on 9to5mac.
- Blog post explaining how it worked (archived).

6.3.3 Other Contributions

- *Hands-On Generative AI with Transformers and Diffusion Models*, book for ML developers and practitioners covering the state-of-the-art generative AI techniques, such as diffusion models, LLMs, variational autoencoders, and more. O'Reilly Media 2024, ISBN 978-1098149246 [124].
- Dozens of blog posts and talks for developers on deep learning topics and generative AI.

Capítulo 7

Resumen en Castellano

7.1 Introducción y Objetivos

La edición fotográfica ha acompañado a la fotografía desde sus inicios. Hoy en día, millones de usuarios ajustan sus fotos digitalmente, pero las herramientas profesionales siguen exigiendo una curva de aprendizaje pronunciada y flujos de trabajo complejos. Esta situación ha motivado la búsqueda de métodos automáticos y accesibles que mejoren la calidad técnica de las fotografías sin comprometer la intención artística ni la fidelidad de la captura original.

Esta tesis versa sobre la aplicación del aprendizaje profundo a la mejora fotográfica siguiendo tres principios rectores: (i) independencia de resolución, para que los resultados puedan aplicarse a imágenes de cualquier tamaño sin pérdida de calidad; (ii) interpretabilidad y control, favoreciendo métodos paramétricos no destructivos cuyos resultados puedan ajustarse tras la predicción; y (iii) viabilidad práctica en dispositivos de consumo, haciendo especial hincapié en la ejecución local por razones de privacidad, latencia y sostenibilidad.

Sobre estos cimientos, la tesis presenta dos sistemas principales y una integración de ambos. En primer lugar, desarrollamos FilterNet, un sistema de mejora automática que no genera píxeles editados, sino que aprende a predecir valores de filtros fotográficos estándar (exposición, brillo, contraste, sombras, altas luces y temperatura de color). Este diseño hace que el proceso sea interpretable, editable por el usuario y aplicable a resolución arbitraria: el modelo infiere parámetros a partir de una miniatura y la aplicación los aplica sobre la fotografía original a máxima resolución.

En segundo lugar, exploramos la segmentación guiada por texto como paso

clave hacia ediciones locales. Para ello proponemos CocoGold, que reutiliza un modelo de difusión entrenado con millones de pares $(texto, imagen)$, modificándolo para identificar y segmentar objetos a partir de descripciones sencillas (por ejemplo, "gato", "coche" o "reloj"). A diferencia de los *pipelines* de detección y segmentación por etapas, CocoGold formula la tarea como generación condicionada: el modelo aprende a "copiar" la imagen de entrada resaltando en blanco las regiones que coinciden con el término expresado, de donde extraemos la máscara binaria con un posprocesado ligero.

Por último, combinamos ambos enfoques en una prueba de concepto llamada COCONET: las máscaras de CocoGold permiten aplicar los parámetros de FilterNet de forma localizada. De este modo demostramos que la edición técnica basada en filtros puede adaptarse a regiones especificadas con lenguaje, manteniendo interpretabilidad e independencia de resolución en todos los componentes salvo la máscara de segmentación.

El objetivo final de nuestro trabajo es contribuir a democratizar la edición fotográfica con soluciones prácticas, explicables y eficientes: un sistema capaz de proponer ajustes técnicos expresados con el mismo vocabulario utilizado en la edición fotográfica, capaz también de seleccionar áreas de interés mediante instrucciones naturales, y de ejecutarse en dispositivos móviles sin exigir infraestructuras especializadas ni comprometer la privacidad del usuario.

7.2 Antecedentes

La mejora automática de imágenes ha evolucionado desde técnicas algorítmicas clásicas (como los métodos de ecualización de histograma) hasta métodos de aprendizaje profundo capaces de aprender transformaciones a partir de datos. Las primeras propuestas basadas en redes convolucionales abordaron tareas específicas (corrección de exposición, por ejemplo) o estilos locales guiados por un proceso de segmentación previo. Aunque eficaces, estos métodos suelen generar la imagen final en píxeles, lo que complica la escalabilidad a altas resoluciones y produce resultados poco interpretables o difíciles de ajustar.

Una línea interesante ha sido la de *aprender ediciones* en lugar de producir píxeles finales. Este planteamiento refuerza la independencia de resolución y la interpretabilidad, al representar la salida como parámetros de ajustes fotográficos y no como mapas de píxeles. Sin embargo, la disponibilidad de conjuntos de datos con parejas de imágenes $(original, editada)$ es limitada y a

veces restrictiva en términos de licencia. Para superar esa limitación han surgido aproximaciones generativas (GANs) y de aprendizaje por refuerzo que aprenden secuencias de ajustes a partir de colecciones de imágenes "buenas" y "malas", sin mapear una a una, lo que presenta retos de estabilidad en el entrenamiento.

El aprendizaje por transferencia se ha consolidado como una estrategia preferente en visión artificial: un extractor de características preentrenado (ResNet, por ejemplo) puede utilizarse como base para tareas de mejora. Junto a ello, las funciones de pérdida perceptuales basadas en redes neuronales (tales como VGG, entre otras) han demostrado mejor correlación con la calidad visual percibida que las distancias píxel a píxel. A nivel de ingeniería, la diferenciabilidad de las operaciones de procesado de imagen resulta crucial para integrar filtros en arquitecturas entrenables extremo a extremo.

En cuanto a los datos, la *degradación sintética* ha resultado ser una receta efectiva de supervisión automática: a partir de fotos de alta calidad se generan variantes empeoradas mediante combinaciones plausibles de filtros. El modelo aprende a revertir esa degradación sin necesidad de pares editados por expertos, ampliando considerablemente la cantidad de muestras disponibles.

En paralelo, los últimos años han visto el auge de los modelos basados en la arquitectura *transformer*, los modelos de difusión y los modelos multimodales que combinan visión y lenguaje (VLMs). Los métodos basados en *transformers* han mejorado la modelización de dependencias de largo alcance en restauración y super-resolución; los modelos de difusión, concebidos para generación de imágenes, se han adaptado con éxito a tareas de traducción imagen a imagen (colorización, rellenado de zonas, estimación de profundidad) y se benefician de avances drásticos en muestreo rápido (DDIM, familias DPM-Solver, destilación). En el terreno lenguaje-visión, el alineamiento entre texto e imagen ha habilitado segmentación de vocabulario abierto y basada en expresiones identificativas, con pipelines modulares que combinan detectores con arquitecturas fundacionales de segmentación (como SAM). Al mismo tiempo, algunos VLMs han comenzado a producir salidas espaciales mediante la utilización de *tokens* especiales en el vocabulario o el uso de cabezas auxiliares de segmentación.

Finalmente, una tendencia emergente es la reutilización de grandes modelos de difusión, ya entrenados a escala masiva com millones de pares texto-imagen, para tareas de comprensión espacial no generativa. Reformulando el objetivo y retocando mínimamente la arquitectura se aprovechan su capacidad

de representación y conocimiento espacial. Este es el enfoque que adoptamos para CocoGold, incorporando además condicionamiento textual explícito.

7.3 Metodología y Resultados

7.3.1 FilterNet: aprendizaje auto-supervisado de parámetros de filtros

FilterNet parte de un extractor ResNet preentrenado y añade una *cabeza* predictiva que produce, de una sola pasada, seis valores en el rango normalizado $(-1, 1)$: exposición, brillo, contraste, sombras, altas luces y temperatura de color. Para entrenar de extremo a extremo con *backpropagation* es necesario renderizar, dentro del propio modelo, una versión mejorada de la imagen usando **filtros diferenciables**. Implementamos la mayoría mediante curvas de tonos paramétricas (pares alto/bajo para cada ajuste) aplicadas en espacio RGB **lineal** con 10 bits de precisión, y combinadas con la imagen original con ajustes de transparencia. La temperatura de color, en cambio, se resuelve con matrices de ganancia 3×3 previamente calculadas, y para la exposición aplicamos su definición fotométrica. Este diseño emula el comportamiento de aplicaciones profesionales, evitando saturaciones o artefactos que se observan con métodos simplistas de contraste o brillo.

El entrenamiento es auto-supervisado: a partir de un conjunto de fotos "buenas" generamos muestras degradadas mediante combinaciones controladas de los seis filtros. El objetivo del modelo es aproximar la foto de referencia a partir de la imagen degradada, aplicando los filtros mencionados. La función de pérdida combina: (i) término píxel a píxel en sRGB; (ii) pérdida perceptual con características VGG de varios niveles; y (iii) pérdida de estilo (matrices de Gram), con el objetivo de captar estructura, textura y color en mayor consonancia con la percepción humana.

Estudiamos diversas variantes que favorecen la estabilidad y la generalización. Por ejemplo, inicializamos los pesos de la cabeza de regresión con valores cercanos a cero (en el centro del rango predictivo de los filtros) para evitar predicciones extremas en las etapas iniciales de entrenamiento. Utilizamos también una capa que denominamos *maskout*, análoga a *dropout*, que utilizamos para desactivar aleatoriamente alguno de los seis canales con objeto de fomentar combinaciones alternas. En cuanto a la preparación de datos, utilizamos regímenes de degradación moderados, reduciendo la probabilidad

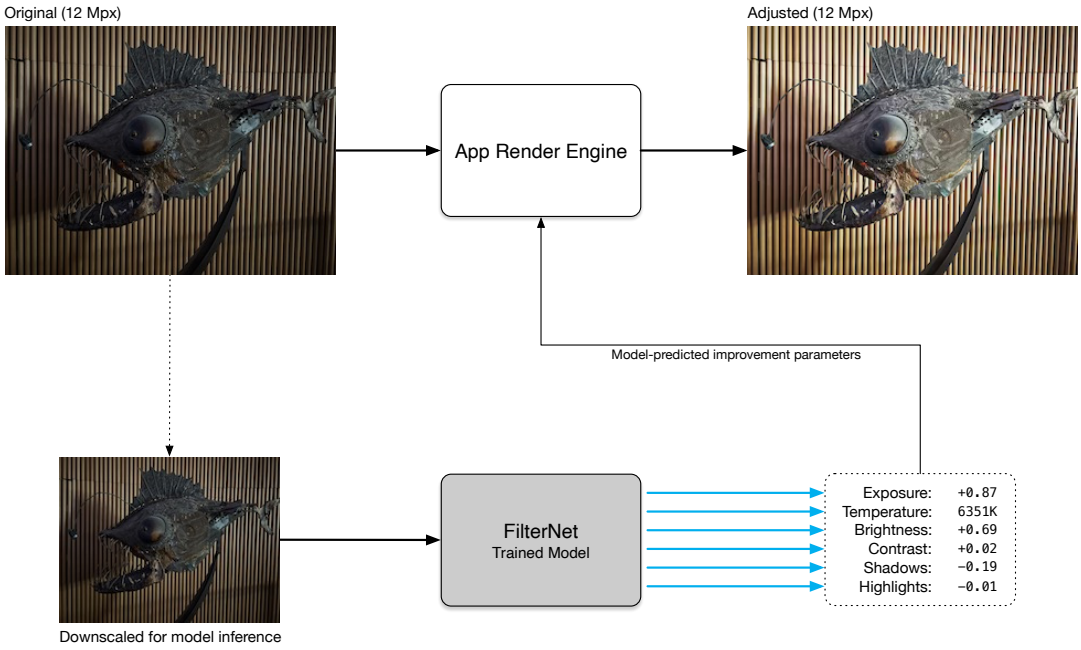


FIGURA 7.1: Esquema de inferencia local con FilterNet (copia de Fig. 3.7). El modelo predice parámetros de filtros a partir de una imagen en miniatura. Dichos parámetros se aplican a la imagen original a resolución completa, manteniendo la edición no destructiva e interpretable.

de saturación irrecuperable (*clipping*), y acercando la distribución de datos de entrenamiento a escenarios reales.

Para evaluar FilterNet (Tabla 5.1) usamos métricas de referencia (PSNR, SSIM, FSIMc) y dos métricas con mayor correlación perceptual: DISTS y una pérdida de contenido basada en VGG. Los resultados muestran mejoras consistentes al añadir componentes perceptuales a la pérdida y al operar en RGB lineal. En cuanto a la arquitectura, verificamos que ResNet-50 supera a ResNet-34, y un breve entrenamiento del backbone completo mejora frente a mantenerlo congelado por completo.

En cuanto a la implantación del sistema, exportamos el modelo a Core ML con cuantización a 16 bits, alcanzando tiempos de inferencia bastante inferiores a un segundo, incluso en móviles de 2020 y anteriores. Esto nos permitió incorporar el modelo a una aplicación popular de fotografía, validando la viabilidad de nuestro enfoque paramétrico, interpretable y optimizado para ejecución local (Fig. 7.1).

7.3.2 CocoGold: segmentación basada en difusión guiada por texto

La edición local requiere identificar regiones de interés. CocoGold reutiliza Stable Diffusion 2 y reformula la tarea de segmentación como generación condicionada por imagen y texto: dada una imagen y un término expresado con texto (por ejemplo, "gato"), el modelo aprende a producir una copia de la imagen donde las regiones asociadas al término aparecen destacadas en blanco. Para ello concatenamos la representación latente de la imagen original y de la imagen objetivo, e introducimos ambas como entradas al sistema, adaptando la primera capa de la U-Net para aceptar 8 canales (correspondientes a 2 imágenes) en espacio latente. El resto de la arquitectura del modelo de difusión no necesita cambio alguno. El condicionamiento con texto utiliza el modelo CLIP, al igual que el modelo Stable Diffusion en su versión estándar. El entrenamiento se desarrolla sobre la U-Net; tanto CLIP como el codificador de imágenes (VAE) permanecen congelados.

Nuestro *dataset* de entrenamiento se basa en COCO 2017. Creamos un iterador que, para cada muestra solicitada, proporciona un recorte aleatorio que contiene algún área perteneciente a una de las categorías de objetos de COCO. Como imagen objetivo, preparamos ese recorte con dicho área en color blanco. Puesto que COCO presenta fuertes desequilibrios entre categorías, seleccionamos para nuestro prototipo un subconjunto de 14 categorías con similar representación. El proceso de inferencia sigue el patrón estándar en Stable Diffusion: DDIM, con 50 pasos de reducción de ruido. El modelo, como se ha indicado, genera una copia aproximada de la imagen objetivo, con la máscara de segmentación sobrepuesta en color blanco. Gracias a un proceso de post-procesado (selección de umbrales de color, seguido de operaciones morfológicas para eliminar falsos positivos de pequeño tamaño), obtenemos la máscara binaria de segmentación. Para conseguir compatibilidad con escenas que contienen áreas blancas de forma natural (cielos sobresaturados, por ejemplo), aplicamos a la imagen original una desaturación selectiva, de modo que la copia generada no confunda áreas realmente saturadas con el color blanco a generar en las zonas de segmentación. Este pequeño truco permite al modelo trabajar con todo tipo de imágenes.

Para la evaluación de resultados medimos IoU (*Intersection over Union*) y precisión por píxel sobre el dataset de validación de COCO, bajo un protocolo acorde con la tarea formulada (clase única condicionada por texto, sin existencia de muestras negativas al utilizar siempre recortes que contienen la

categoría objetivo). Observamos una mejora de generalización sobre clases no vistas durante el entrenamiento al aumentar la duración del mismo, aunque el rendimiento en este caso es manifiestamente inferior al de clases entrenadas. La agregación de varias predicciones aporta ganancias modestas. Sin embargo, observamos que métodos de inferencia de pocos pasos, tales como *trailing DDIM*, funcionan sin cambios sobre el modelo previamente entrenado, lo que nos permite utilizar inferencia de tan sólo 1 a 4 pasos para obtener resultados satisfactorios. Esto reduce enormemente el tiempo de latencia y la aplicabilidad práctica del método en hardware de consumo.

7.3.3 COCONET: integración de FilterNet y CocoGold para edición local

Partiendo de las máscaras de CocoGold exploramos dos integraciones simples de FilterNet: (A) predecir parámetros globales sobre la imagen completa, y aplicarlos en la zona segmentada; (B) aplicar FilterNet sobre un recorte alrededor del área segmentada para obtener parámetros más ajustados a dicha región, y aplicarlos igualmente sobre la máscara de segmentación. En ambos casos, un difuminado suave del contorno facilita transiciones naturales. A nivel práctico, este pipeline combinado respeta los píxeles fuera de la máscara (se toman del original) y preserva el carácter no destructivo e interpretable del proceso. La limitación principal es la resolución de la máscara de segmentación: mientras FilterNet es completamente agnóstico a la resolución, la segmentación produce una máscara binaria de tamaño fijo. Esta restricción puede mitigarse con técnicas de aumento de resolución guiadas por bordes, tales como las utilizadas en DiffDIS o en aplicaciones reales demostradas en la industria: es el caso del tratamiento de imágenes de profundidad en el iPhone, donde la máscara (de profundidad, en este caso) es de baja resolución pero se adapta a tamaño real mediante procesos especializados que respetan los bordes.

La Figura 7.2 muestra los resultados de COCONET utilizando cada uno de los dos métodos indicados.

7.4 Conclusiones

En esta tesis pretendemos demostrar que es posible acercar la calidad y flexibilidad de la edición profesional a experiencias cotidianas y accesibles, sin

FIGURA 7.2: COCONET: edición local aplicando los parámetros de FilterNet (predichos globalmente o utilizando un recorte alrededor del área de interés) sobre la máscara de CocoGold.

renunciar a la interpretabilidad ni a la eficiencia en dispositivos de consumo. Las contribuciones centrales son:

- Un enfoque de mejora fotográfica basado en la predicción de parámetros de filtros fotográficos, en lugar de generar píxeles editados. Esta decisión hace al sistema independiente de la resolución, controlable por el usuario y fácil de implantar, incluso localmente.
- Una arquitectura de filtros diferenciables que replica de forma realista operaciones fotográficas en espacio RGB lineal con 10 bits de precisión, con pérdidas perceptuales que conducen a resultados más acordes con la evaluación humana.
- Una estrategia auto-supervisada de generación de datos que evita depender de pares editados, permitiendo entrenar con colecciones de fotos de alta calidad degradadas de forma plausible.
- Validación, a través de CocoGold, de que los modelos de difusión preentrenados pueden reutilizarse eficazmente para segmentación guiada por texto con mínimas modificaciones, formulando la tarea como generación *imagen a imagen*.
- Una integración (Coconet) que aplica los parámetros de FilterNet de forma localizada sobre las máscaras de CocoGold, habilitando ediciones no destructivas y semánticamente informadas a partir de instrucciones simples.

Entre las limitaciones señalamos: (i) la resolución fija de las máscaras, que puede producir imprecisiones en detalles finos como el cabello y otras estructuras detalladas; (ii) la restricción, en esta primera versión, a términos de una sola

palabra, lejos de descripciones ricas y expresiones identificativas que abundan en casos reales; y (iii) la latencia propia de los modelos de difusión, mitigable con procesos de eliminación de ruido como los indicados, destilación, o técnicas de entrenamiento diseñadas para pocos pasos de inferencia.

Como líneas de trabajo futuras proponemos dos direcciones complementarias. La primera consiste en extender CocoGold a un verdadero escenario de vocabulario abierto y lenguaje natural, incorporando expresiones identificativas ("la persona de la izquierda con camisa roja"). Dado el fuerte alineamiento multimodal de los modelos de difusión, postulamos que esta ampliación depende más del currículo de datos que de cambios profundos de arquitectura. Paralelamente, explorar pérdidas y señales auxiliares sensibles a contornos podría mejorar la fidelidad de bordes sin sacrificar la simplicidad del sistema.

La segunda dirección es complementar el enfoque con VLMs como predictores de parámetros de edición basados en intenciones expresadas por el usuario, en lugar de estimar parámetros automáticos tan sólo. Un VLM podría recibir una imagen y una instrucción en lenguaje natural ("haz el cielo más dramático", "ilumina la cara manteniendo el fondo") y devolver un vector interpretable de parámetros para conseguir tal efecto. Para entrenar un sistema de estas características planteamos la construcción de un dataset sintético utilizando LLMs para generar variabilidad y riqueza en las descripciones.

En conjunto, FilterNet, CocoGold y su integración inicial en Coconet, marcan un camino pragmático hacia sistemas de edición fotográfica más intuitivos: automáticos cuando el usuario lo desea, ajustables cuando la intención artística requiere control fino, y siempre respetuosos con los datos originales. El equilibrio entre independencia de resolución, interpretabilidad y viabilidad en móviles convierte a estas propuestas en una base sólida sobre la que seguir construyendo experiencias de edición centradas en las personas y habilitadas por IA.

Bibliography

[1] Ansel Adams. *The Print*. New York Graphic Society, 1950. ISBN: 9780821207185.

[2] Zhicheng Yan et al. "Automatic Photo Adjustment Using Deep Neural Networks". In: *ACM Trans. Graph.* 35.2 (Feb. 2016). ISSN: 0730-0301. DOI: 10.1145/2790296. URL: https://doi.org/10.1145/2790296.

[3] Ruixing Wang et al. "Underexposed Photo Enhancement Using Deep Illumination Estimation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019.

[4] Yuanming Hu et al. "Exposure: A White-Box Photo Post-Processing Framework". In: *ACM Trans. Graph.* 37.2 (May 2018). ISSN: 0730-0301. DOI: 10.1145/3181974. URL: https://doi.org/10.1145/3181974.

[5] Sean Moran, Steven McDonagh, and Gregory Slabaugh. "CURL: Neural Curve Layers for Global Image Enhancement". In: *2020 25th International Conference on Pattern Recognition (ICPR)*. 2021, pp. 9796–9803. DOI: 10.1109/ICPR48806.2021.9412677.

[6] Aditya Ramesh et al. "Zero-shot text-to-image generation". In: *International conference on machine learning*. Pmlr. 2021, pp. 8821–8831.

[7] Jiahui Yu et al. *Scaling Autoregressive Models for Content-Rich Text-to-Image Generation*. 2022. arXiv: 2206.10789 [cs.CV]. URL: https://arxiv.org/abs/2206.10789.

[8] Robin Rombach et al. "High-resolution image synthesis with latent diffusion models". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 10684–10695.

[9] Chitwan Saharia et al. "Photorealistic text-to-image diffusion models with deep language understanding". In: *Advances in neural information processing systems* 35 (2022), pp. 36479–36494.

[10] Aditya Ramesh et al. "Hierarchical text-conditional image generation with clip latents". In: *arXiv preprint arXiv:2204.06125* 1.2 (2022), p. 3.

[11] Chenlin Meng et al. *SDEdit: Guided Image Synthesis and Editing with Stochastic Differential Equations*. 2022. arXiv: 2108.01073 [cs.CV]. URL: https://arxiv.org/abs/2108.01073.

[12] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. "Adding conditional control to text-to-image diffusion models". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2023, pp. 3836–3847.

[13] OpenAI. *GPT-5*. 2025. URL: https://openai.com/index/introducing-gpt-5/.

[14] Bingxin Ke et al. "Repurposing diffusion-based image generators for monocular depth estimation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 9492–9502.

[15] Tom Brown et al. "Language models are few-shot learners". In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901.

[16] OpenAI et al. *GPT-4 Technical Report*. 2024. arXiv: 2303.08774 [cs.CL]. URL: https://arxiv.org/abs/2303.08774.

[17] Anthropic. *The Claude 3 Model Family: Opus, Sonnet, Haiku*. 2024. URL: https://assets.anthropic.com/m/61e7d27f8c8f5919/original/Claude-3-Model-Card.pdf (visited on 05/27/2025).

[18] Aaron Grattafiori et al. *The Llama 3 Herd of Models*. 2024. arXiv: 2407.21783 [cs.AI]. URL: https://arxiv.org/abs/2407.21783.

[19] An Yang et al. *Qwen2 Technical Report*. 2024. arXiv: 2407.10671 [cs.CL]. URL: https://arxiv.org/abs/2407.10671.

[20] Gemma Team et al. *Gemma: Open Models Based on Gemini Research and Technology*. 2024. arXiv: 2403.08295 [cs.CL]. URL: https://arxiv.org/abs/2403.08295.

[21] Gemma Team et al. *Gemma 2: Improving Open Language Models at a Practical Size*. 2024. arXiv: 2408.00118 [cs.CL]. URL: https://arxiv.org/abs/2408.00118.

[22] Gemma Team et al. *Gemma 3 Technical Report*. 2025. arXiv: 2503.19786 [cs.CL]. URL: https://arxiv.org/abs/2503.19786.

[23] DeepSeek-AI et al. *DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning*. 2025. arXiv: 2501.12948 [cs.CL]. URL: https://arxiv.org/abs/2501.12948.

[24] Tim Brooks, Aleksander Holynski, and Alexei A Efros. "Instructpix2pix: Learning to follow image editing instructions". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2023, pp. 18392–18402.

[25] Haotian Liu et al. "Visual instruction tuning". In: *Advances in neural information processing systems* 36 (2023), pp. 34892–34916.

[26] Lucas Beyer et al. *PaliGemma: A versatile 3B VLM for transfer*. 2024. arXiv: 2407.07726 [cs.CV]. URL: https://arxiv.org/abs/2407.07726.

[27] Andreas Steiner et al. *PaliGemma 2: A Family of Versatile VLMs for Transfer*. 2024. arXiv: 2412.03555 [cs.CV]. URL: https://arxiv.org/abs/2412.03555.

[28] Jinze Bai et al. *Qwen-vl: A frontier large vision-language model with versatile abilities*. 2023. arXiv: 2308.12966 [cs.CV].

[29] QwenTeam. "Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution". In: (2025). URL: https://qwen.ai/blog?id=99f0335c4ad9ff6153e517418d48535ab6d8afef&from=research.latest-advancements-list.

[30] Moondream Team. *Moondream: Grounded Reasoning and Visual Understanding for Small Vision-Language Models*. https://moondream.ai/blog/moondream-2025-06-21-release. Accessed: 2025-07-01. 2025.

[31] Rahul Ramachandran et al. *How Well Does GPT-4o Understand Vision? Evaluating Multimodal Foundation Models on Standard Computer Vision Tasks*. 2025. arXiv: 2507.01955 [cs.CV]. URL: https://arxiv.org/abs/2507.01955.

[32] Stephen M. Pizer et al. "Adaptive histogram equalization and its variations". In: *Computer Vision, Graphics, and Image Processing* 39.3 (1987), pp. 355 –368. ISSN: 0734-189X. DOI: https://doi.org/10.1016/S0734-189X(87)80186-X. URL: http://www.sciencedirect.com/science/article/pii/S0734189X8780186X.

[33] Karel Zuiderveld. "Contrast Limited Adaptive Histogram Equalization". In: *Graphics Gems IV*. USA: Academic Press Professional, Inc., 1994, 474–485. ISBN: 0123361559.

[34] V. Bychkovsky et al. "Learning photographic global tonal adjustment with a database of input/output image pairs". In: *CVPR 2011*. 2011. DOI: 10.1109/CVPR.2011.5995413.

[35] Ian Goodfellow et al. "Generative Adversarial Networks". In: *Commun. ACM* 63.11 (Oct. 2020), 139–144. ISSN: 0001-0782. DOI: 10.1145/3422622. URL: https://doi.org/10.1145/3422622.

[36] Jun-Yan Zhu et al. "Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2017.

[37] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation". In: *International Conference on Medical image computing and computer-assisted intervention*.

Springer. 2015, pp. 234–241. arXiv: 1505.04597. URL: https://arxiv.org/abs/1505.04597.

[38] Jonathan Baxter et al. *Learning to Learn: Knowledge Consolidation and Transfer in Inductive Systems*. http://socrates.acadiau.ca/courses/comp/dsilver/NIPS95_LTL/transfer.workshop.1995.html. Accessed: 2021-04-21. 1995.

[39] Sinno Jialin Pan and Qiang Yang. "A Survey on Transfer Learning". In: *IEEE Transactions on Knowledge and Data Engineering* 22.10 (2010), pp. 1345–1359. DOI: 10.1109/TKDE.2009.191.

[40] Fuzhen Zhuang et al. "A Comprehensive Survey on Transfer Learning". In: *Proceedings of the IEEE* 109.1 (2021), pp. 43–76. DOI: 10.1109/JPROC.2020.3004555.

[41] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[42] Jia Deng et al. "ImageNet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. IEEE. 2009, pp. 248–255.

[43] A. Sai Bharadwaj Reddy and D. Sujitha Juliet. "Transfer Learning with ResNet-50 for Malaria Cell-Image Classification". In: *2019 International Conference on Communication and Signal Processing (ICCSP)*. 2019, pp. 0945–0949. DOI: 10.1109/ICCSP.2019.8697909.

[44] Shaopeng Liu, Guohui Tian, and Yuan Xu. "A novel scene classification model combining ResNet based transfer learning and data augmentation with a filter". In: *Neurocomputing* 338 (2019), pp. 191–206. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2019.01.090. URL: https://www.sciencedirect.com/science/article/pii/S0925231219301833.

[45] Edmar Rezende et al. "Malicious Software Classification Using Transfer Learning of ResNet-50 Deep Neural Network". In: *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*. 2017, pp. 1011–1014. DOI: 10.1109/ICMLA.2017.00-19.

[46] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. "Perceptual Losses for Real-Time Style Transfer and Super-Resolution". In: *Computer Vision – ECCV 2016*. Ed. by Bastian Leibe et al. Cham: Springer International Publishing, 2016, pp. 694–711. ISBN: 978-3-319-46475-6.

[47] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. "A neural algorithm of artistic style". In: *arXiv preprint arXiv:1508.06576* (2015).

[48] Richard Zhang et al. "The unreasonable effectiveness of deep features as a perceptual metric". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 586–595.

[49] Yinbo Chen, Sifei Liu, and Xiaolong Wang. *Learning Continuous Image Representation with Local Implicit Image Function*. 2021. arXiv: 2012.09161 [cs.CV].

[50] Ben Mildenhall et al. "NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis". In: *CoRR* abs/2003.08934 (2020). arXiv: 2003.08934. URL: https://arxiv.org/abs/2003.08934.

[51] Yuanming Hu et al. "DiffTaichi: Differentiable Programming for Physical Simulation". In: *International Conference on Learning Representations*. 2020.

[52] Alexander Buslaev et al. "Albumentations: Fast and Flexible Image Augmentations". In: *Information* 11.2 (2020), p. 125. ISSN: 2078-2489. DOI: 10.3390/info11020125. URL: http://dx.doi.org/10.3390/info11020125.

[53] Ekin D. Cubuk et al. *AutoAugment: Learning Augmentation Strategies from Data*. 2019. arXiv: 1805.09501 [cs.CV].

[54] Aitor Alvarez-Gila et al. "Self-supervised blur detection from synthetically blurred scenes". In: *Image and Vision Computing* 92 (2019), p. 103804.

[55] Jason Antic, Jeremy Howard, and Uri Manor. *Decrappification, DeOldification, and Super Resolution*. https://www.fast.ai/2019/05/03/decrappify/. Accessed: 2021-01-01. 2019.

[56] Alexey Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021. arXiv: 2010.11929 [cs.CV]. URL: https://arxiv.org/abs/2010.11929.

[57] Jingyun Liang et al. "Swinir: Image restoration using swin transformer". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 1833–1844.

[58] Ze Liu et al. "Swin transformer: Hierarchical vision transformer using shifted windows". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 10012–10022.

[59] Chitwan Saharia et al. "Palette: Image-to-image diffusion models". In: *ACM SIGGRAPH 2022 conference proceedings*. 2022, pp. 1–10.

[60] Kai Zhang et al. "Designing a practical degradation model for deep blind image super-resolution". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 4791–4800.

[61] Alec Radford et al. "Learning transferable visual models from natural language supervision". In: *International conference on machine learning*. PmLR. 2021, pp. 8748–8763.

[62] Boyi Li et al. "Language-driven semantic segmentation". In: *arXiv preprint arXiv:2201.03546* (2022).

[63] Timo Lüddecke and Alexander Ecker. "Image segmentation using text and image prompts". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 7086–7096.

[64] Golnaz Ghiasi et al. "Scaling open-vocabulary image segmentation with image-level labels". In: *European conference on computer vision*. Springer. 2022, pp. 540–557.

[65] Feng Liang et al. "Open-vocabulary semantic segmentation with mask-adapted clip". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2023, pp. 7061–7070.

[66] Qihang Yu et al. "Convolutions die hard: Open-vocabulary segmentation with single frozen convolutional clip". In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 32215–32234.

[67] Licheng Yu et al. "Modeling context in referring expressions". In: *European conference on computer vision*. Springer. 2016, pp. 69–85. URL: https://arxiv.org/abs/1608.00272.

[68] Sahar Kazemzadeh et al. "Referitgame: Referring to objects in photographs of natural scenes". In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014, pp. 787–798.

[69] Junhua Mao et al. "Generation and comprehension of unambiguous object descriptions". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 11–20. URL: https://arxiv.org/abs/1511.02283.

[70] Varun K Nagaraja, Vlad I Morariu, and Larry S Davis. "Modeling context between objects for referring expression understanding". In: *European Conference on Computer Vision*. Springer. 2016, pp. 792–807.

[71] Zhao Yang et al. "Lavt: Language-aware vision transformer for referring image segmentation". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 18155–18165. URL: https://arxiv.org/abs/2112.02244.

[72] Jiannan Wu et al. "Language as queries for referring video object segmentation". In: *Proceedings of the IEEE/CVF Conference on Computer*

Vision and Pattern Recognition. 2022, pp. 4974–4984. URL: https://arxiv.org/abs/2201.00487.

[73] Alexander Kirillov et al. "Segment anything". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2023, pp. 4015–4026. URL: https://arxiv.org/abs/2304.02643.

[74] Xin Lai et al. "Lisa: Reasoning segmentation via large language model". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 9579–9589.

[75] Xueyan Zou et al. "Segment everything everywhere all at once". In: *Advances in neural information processing systems* 36 (2023), pp. 19769–19782.

[76] Xueyan Zou et al. "Generalized decoding for pixel, image, and language". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2023, pp. 15116–15127.

[77] Liunian Harold Li et al. "Grounded language-image pre-training". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 10965–10975.

[78] Shilong Liu et al. "Grounding dino: Marrying dino with grounded pre-training for open-set object detection". In: *European Conference on Computer Vision*. Springer. 2024, pp. 38–55.

[79] Jierun Chen et al. *Revisiting Referring Expression Comprehension Evaluation in the Era of Large Multimodal Models*. 2024. arXiv: 2406.16866 [cs.CV].

[80] Weihan Wang et al. *CogVLM: Visual Expert for Pretrained Language Models*. 2023. arXiv: 2311.03079 [cs.CV].

[81] Ashish Vaswani et al. "Attention is all you need". In: *Advances in neural information processing systems* 30 (2017).

[82] Jacob Devlin et al. "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*. 2019, pp. 4171–4186. URL: https://arxiv.org/abs/1810.04805.

[83] Colin Raffel et al. "Exploring the limits of transfer learning with a unified text-to-text transformer". In: *Journal of machine learning research* 21.140 (2020), pp. 1–67. URL: https://arxiv.org/abs/1910.10683.

[84] Google. *Introducing Gemini 2.5 Flash Image, our state-of-the-art image model*. 2025. URL: https://developers.googleblog.com/en/introducing-gemini-2-5-flash-image/.

[85] Meta AI. *The Llama 4 herd: The beginning of a new era of natively multimodal AI innovation*. 2025. URL: https://ai.meta.com/blog/llama-4-multimodal-intelligence/.

[86] OpenAI: Sandhini Agarwal et al. *gpt-oss-120b & gpt-oss-20b Model Card*. 2025. arXiv: 2508.10925 [cs.CL]. URL: https://arxiv.org/abs/2508.10925.

[87] Noam Shazeer et al. *Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer*. 2017. arXiv: 1701.06538 [cs.LG]. URL: https://arxiv.org/abs/1701.06538.

[88] Tri Dao et al. "Flashattention: Fast and memory-efficient exact attention with io-awareness". In: *Advances in neural information processing systems* 35 (2022), pp. 16344–16359. URL: https://arxiv.org/abs/2205.14135.

[89] Georgi Gerganov and multiple authors. *Llama C++: LLM inference in C/C++*. Version 0.0. 2023. URL: https://github.com/ggml-org/llama.cpp.

[90] Awni Hannun et al. *MLX: Efficient and flexible machine learning on Apple silicon*. Version 0.0. 2023. URL: https://github.com/ml-explore.

[91] Junnan Li et al. "Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models". In: *International conference on machine learning*. PMLR. 2023, pp. 19730–19742.

[92] Jean-Baptiste Alayrac et al. "Flamingo: a visual language model for few-shot learning". In: *Advances in neural information processing systems* 35 (2022), pp. 23716–23736.

[93] Peng Wang et al. *Qwen3-VL: Sharper Vision, Deeper Thought, Broader Action*. 2024. (Visited on 10/08/2025).

[94] An Vo et al. *Vision Language Models are Biased*. 2025. arXiv: 2505.23941 [cs.LG]. URL: https://arxiv.org/abs/2505.23941.

[95] Tomer Amit et al. "Segdiff: Image segmentation with diffusion probabilistic models". In: *arXiv preprint arXiv:2112.00390* (2021).

[96] Jiarui Xu et al. "Open-vocabulary panoptic segmentation with text-to-image diffusion models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 2955–2966.

[97] Qian Yu et al. *High-Precision Dichotomous Image Segmentation via Probing Diffusion Capacity*. 2024. arXiv: 2410.10105 [cs.CV]. URL: https://arxiv.org/abs/2410.10105.

[98] Xuebin Qin et al. "Highly accurate dichotomous image segmentation". In: *European Conference on Computer Vision*. Springer. 2022, pp. 38–56.

[99] Jiaming Song, Chenlin Meng, and Stefano Ermon. "Denoising diffusion implicit models". In: 2020.

[100] Cheng Lu et al. "Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 5775–5787.

[101] Cheng Lu et al. "Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models". In: *Machine Intelligence Research* (2025), pp. 1–22.

[102] Shanchuan Lin et al. *Common Diffusion Noise Schedules and Sample Steps are Flawed*. 2024. arXiv: 2305.08891 [cs.CV]. URL: https://arxiv.org/abs/2305.08891.

[103] Tim Salimans and Jonathan Ho. "Progressive distillation for fast sampling of diffusion models". In: *arXiv preprint arXiv:2202.00512* (2022).

[104] Yang Song et al. "Consistency Models". In: *Proceedings of the 40th International Conference on Machine Learning*. Ed. by Andreas Krause et al. Vol. 202. Proceedings of Machine Learning Research. PMLR, 2023, pp. 32211–32252. URL: https://proceedings.mlr.press/v202/song23a.html.

[105] Simian Luo et al. "Latent consistency models: Synthesizing high-resolution images with few-step inference". In: *arXiv preprint arXiv:2310.04378* (2023).

[106] Simian Luo et al. "LCM-LoRA: A universal stable-diffusion acceleration module". In: *arXiv preprint arXiv:2311.05556* (2023).

[107] Edward J. Hu et al. *LoRA: Low-Rank Adaptation of Large Language Models*. 2021. arXiv: 2106.09685 [cs.CL]. URL: https://arxiv.org/abs/2106.09685.

[108] Axel Sauer et al. *Adversarial Diffusion Distillation*. 2023. arXiv: 2311.17042 [cs.CV]. URL: https://arxiv.org/abs/2311.17042.

[109] Stability AI. *SD-Turbo Model Card*. 2023. URL: https://huggingface.co/stabilityai/sd-turbo (visited on 08/24/2025).

[110] Axel Sauer et al. *Fast High-Resolution Image Synthesis with Latent Adversarial Diffusion Distillation*. 2024. arXiv: 2403.12015 [cs.CV]. URL: https://arxiv.org/abs/2403.12015.

[111] Black Forest Labs. *FLUX.1 [schnell] Model Card*. 2024. URL: https://huggingface.co/black-forest-labs/FLUX.1-schnell (visited on 08/24/2025).

[112] Bingxin Ke et al. "Marigold: Affordable Adaptation of Diffusion-Based Image Generators for Image Analysis". In: *IEEE Transactions on Pattern*

Analysis and Machine Intelligence (2025), pp. 1–18. DOI: 10.1109/TPAMI.2025.3591076.

[113] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach et al. Curran Associates, Inc., 2019, pp. 8024–8035.

[114] Jeremy Howard and Sylvain Gugger. "Fastai: A Layered API for Deep Learning". In: *Information* 11.2 (2020), p. 108. ISSN: 2078-2489. DOI: 10.3390/info11020108. URL: http://dx.doi.org/10.3390/info11020108.

[115] Christoph Schuhmann et al. "Laion-5b: An open large-scale dataset for training next generation image-text models". In: *Advances in neural information processing systems* 35 (2022), pp. 25278–25294. URL: https://arxiv.org/abs/2210.08402.

[116] Geoffrey E Hinton et al. "Improving neural networks by preventing co-adaptation of feature detectors". In: *arXiv preprint arXiv:1207.0580* (2012).

[117] Nitish Srivastava et al. "Dropout: a simple way to prevent neural networks from overfitting". In: *The journal of machine learning research* 15.1 (2014), pp. 1929–1958.

[118] Edwin Catmull and Raphael Rom. "A class of local interpolating splines". In: *Computer aided geometric design*. Elsevier, 1974, pp. 317–326.

[119] International Electrotechnical Commission et al. "Multimedia systems and equipment-Colour measurement and management-Part 2-1: Colour management-Default RGB colour space-sRGB". In: *IEC 61966-2-1* (1999).

[120] Yin Cui et al. "Class-balanced loss based on effective number of samples". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 9268–9277.

[121] Sota Kato and Kazuhiro Hotta. "Mse loss with outlying label for imbalanced classification". In: *arXiv preprint arXiv:2107.02393* (2021).

[122] Xiaoya Li et al. "Dice loss for data-imbalanced NLP tasks". In: *arXiv preprint arXiv:1911.02855* (2019).

[123] Tsung-Yi Lin et al. "Focal loss for dense object detection". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2980–2988.

[124] Omar Sanseviero et al. *Hands-On Generative AI with Transformers and Diffusion Models*. Sebastopol, CA: O'Reilly Media, Inc., 2024. ISBN: 978-1098149246.

[125] Diederik P Kingma and Max Welling. "Auto-encoding variational bayes". In: *arXiv preprint arXiv:1312.6114* (2013). arXiv: 1312.6114.

[126] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. "Stochastic backpropagation and approximate inference in deep generative models". In: *International conference on machine learning*. PMLR. 2014, pp. 1278–1286. arXiv: 1401.4082. URL: https://arxiv.org/abs/1401.4082.

[127] Robert M Haralick, Stanley R Sternberg, and Xinhua Zhuang. "Image analysis using mathematical morphology". In: *IEEE transactions on pattern analysis and machine intelligence* 4 (1987), pp. 532–550.

[128] Apple Machine Learning Research. "On-device Panoptic Segmentation for Camera Using Transformers". In: *Apple Machine Learning Research* (2021). URL: https://machinelearning.apple.com/research/panoptic-segmentation.

[129] Mostafa Dehghani et al. "Patch n' Pack: NaViT, a Vision Transformer for any Aspect Ratio and Resolution". In: *Advances in Neural Information Processing Systems*. Ed. by A. Oh et al. Vol. 36. Curran Associates, Inc., 2023, pp. 2252–2274. URL: https://proceedings.neurips.cc/paper_files/paper/2023/file/06ea400b9b7cfce6428ec27a371632eb-Paper-Conference.pdf.

[130] Zhou Wang et al. "Image quality assessment: from error visibility to structural similarity". In: *IEEE Transactions on Image Processing* 13.4 (2004), pp. 600–612. DOI: 10.1109/TIP.2003.819861.

[131] Bernd Girod. "What's wrong with mean-squared error?" In: *Digital images and human vision* (1993), pp. 207–220.

[132] Zhou Wang, Alan C Bovik, and Ligang Lu. "Why is image quality assessment so difficult?" In: *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*. Vol. 4. IEEE. 2002, pp. IV–3313.

[133] L. Zhang et al. "FSIM: A Feature Similarity Index for Image Quality Assessment". In: *IEEE Transactions on Image Processing* 20.8 (2011), pp. 2378–2386. DOI: 10.1109/TIP.2011.2109730.

[134] Nikolay Ponomarenko et al. "Image database TID2013: Peculiarities, results and perspectives". In: *Signal processing: Image communication* 30 (2015), pp. 57–77.

[135] Keyan Ding et al. "Image Quality Assessment: Unifying Structure and Texture Similarity". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2020), 1–1. ISSN: 1939-3539. DOI: 10.1109/tpami.2020.3045810. URL: http://dx.doi.org/10.1109/TPAMI.2020.3045810.

[136] Paulius Micikevicius et al. "Mixed precision training". In: *arXiv preprint arXiv:1710.03740* (2017).

[137] Kaiming He et al. "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1026–1034.

[138] Forrest N Iandola et al. "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and $<$ 0.5 MB model size". In: *arXiv preprint arXiv:1602.07360* (2016).

[139] Tsung-Yi Lin et al. "Microsoft coco: Common objects in context". In: *European conference on computer vision*. Springer. 2014, pp. 740–755. URL: https://arxiv.org/abs/1405.0312.

[140] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. "Coco-stuff: Thing and stuff classes in context". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 1209–1218. URL: https://arxiv.org/abs/1612.03716.

[141] Alexander Kirillov et al. "Panoptic segmentation". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 9404–9413. URL: https://arxiv.org/abs/1801.00868.

[142] Mark Everingham et al. "The pascal visual object classes (voc) challenge". In: *International journal of computer vision* 88.2 (2010), pp. 303–338.

[143] Paul Jaccard. "Distribution de la flore alpine dans le bassin des Dranses et dans quelques régions voisines". In: *Bull Soc Vaudoise Sci Nat* 37 (1901), pp. 241–272.

[144] Abubakar Abid et al. "Gradio: Hassle-free sharing and testing of ml models in the wild". In: *arXiv preprint arXiv:1906.02569* (2019).

[145] Kai Zhang et al. "Magicbrush: A manually annotated dataset for instruction-guided image editing". In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 31428–31449. URL: https://arxiv.org/abs/2306.10012.

[146] Jing Shi et al. "A benchmark and baseline for language-driven image editing". In: *Proceedings of the Asian Conference on Computer Vision*. 2020. URL: https://arxiv.org/abs/2010.02330.

[147] Cong Wei et al. "Omniedit: Building image editing generalist models through specialist supervision". In: *The Thirteenth International Conference on Learning Representations*. 2024. URL: https://arxiv.org/abs/2411.07199.

[148] Pedro-Manuel Cuenca-Jiménez, Jesús Fernández-Conde, and José-María Cañas-Plaza. "FilterNet: Self-Supervised Learning for High-Resolution Photo Enhancement". In: *IEEE Access* 10 (2021), pp. 2669–2685. DOI: 10.1109/ACCESS.2021.3139778.

[149] Margarita Rodríguez-Ibáñez et al. "Sentiment Analysis of Political Tweets From the 2019 Spanish Elections". In: *IEEE Access* 9 (2021), pp. 101847–101862. DOI: 10.1109/ACCESS.2021.3097492.

[150] Margarita Rodríguez-Ibánez et al. "A review on sentiment analysis from social media platforms". In: *Expert Systems with Applications* 223 (2023), p. 119862. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2023.119862. URL: https://www.sciencedirect.com/science/article/pii/S0957417423003639.

[151] Andrés Marafioti et al. "SmolVLM: Redefining small and efficient multimodal models". In: *Second Conference on Language Modeling*. 2025. URL: https://openreview.net/forum?id=qMUbhGUFUb.