Title of the Final Project

First Author
firstauthor@i1.org

Second Author
secondauthor@i2.org

Third Author
thirdauthor@i2.org

Fourth Author
fourthauthor@i2.org

Abstract

Brief summary of the work developed as well as the main results and contributions.

1. Introduction

Here, the problem to be solved is described (what do we want to do?), the motivation (why is it relevant to do it?), and the goals (what specific objectives are we going to address in order to solve the problem?).

We use the LaTeX format of the CVPR conference. This document can be written either in English or Spanish.

What follows is a tentative approximation of the sections the document should have. If students consider that they need others, as well as subdividing the sections into different subsections, they can do it without problem.

Students can take inspiration from the cs231n final projects website: http://cs231n.stanford.edu/project.html, where different projects are presented and developed.

This whole final report can have from 6 to 8 pages (no more and no less).

2. Background

This section presents the fundamental concepts necessary to understand the work.

3. Related Works

It presents what has been done in the field previously, and what the best methods are currently. It is very important, in general, not only in this section, to adequately document the relevant literature. To do this, you must use the *.bib* file, in the way I show here: [1–3]

4. Methods

4.1. Modelos de Clasificación

En nuestro problema tenemos dos etapas de clasificación de imágenes. Primero, una clasificación de hojas en tipos de plantas, y luego una clasificación de las enfermedades presentes en las hojas de cada tipo de planta.

Estos son dos problemas bastante diferentes en cuanto a la complejidad de la tarea, principalmente porque en la primera etapa las diferencias entre clases son más notorias (muy diferentes tipos de plantas), mientras que en la segunda etapa las diferencias son más sutiles (diferentes enfermedades que pueden afectar a la misma planta, y que a veces pueden parecerse mucho entre sí). Además, nos enfretamos al problema de la diferencia de tamaños de las imágenes de las hojas y la de sus enfermedades.

Con esto en mente, tiene sentido estudiar una amplia gama de modelos de clasificación, y elegir los más adecuados para cada etapa del problema.

He dividido el estudio inicial de los modelos en dos partes: arquitecturas clásicas y arquitecturas modernas basadas en Transformers. En las tablas 1 y 2 se presentan las comparaciones detalladas de las diferentes arquitecturas estudiadas.

5. Experiments

The data used, the experimental validation protocol, the metrics used, the experiments carried out, the results obtained, and their discussion are presented here.

5.1. Dataset

6. Conclusions

Section that presents, briefly and as a summary, the main conclusions of the work carried out. It also usually includes future possible works. That is, what are the most promising lines to continue with this work, as well as possible proposals for improvement. IMPORTANT: these are the scientific conclusions reached in

Característica	GoogLeNet (Inception)	ResNet	EfficientNet (V1)	EfficientNetV2
Primera Publicación	2014 (Szegedy et al., Google)	2015 (He et al.)	2019 (Tan & Le, Google)	2021 (Tan & Le, Google)
Idea Principal	Módulos Inception con convoluciones paralelas de múltiples escalas	Conexiones residuales (skip) para permitir el entrenamiento de redes muy profundas	Escalado compuesto: escalar conjuntamente profundidad, anchura y resolución usando un coeficiente fijo	Escalado compuesto mejorado + entrenamiento más rápido: arquitectura más inteligente + regularización adaptativa
Bloque Básico	Módulo Inception: convoluciones paralelas 1×1, 3×3, 5×5 + max pooling, concatenadas	• BasicBlock (2× 3×3 convs) para ResNet18/34 • Bottleneck (1×1 → 3×3 → 1×1 convs) para ResNet50+	MBConv (Mobile Inverted Bottleneck): 1×1 exp → 3×3 depthwise → SE → 1×1 proj	Fused-MBConv (etapas iniciales): conv 3×3 regular en lugar de 1×1+depthwise MBConv (etapas posteriores) SE solo en bloques posteriores
Conexiones Skip	× No (pero versiones posteriores como Inception-ResNet sí)	✓ Sí (residual aditivo)	✓ Sí (MBConv usa residual cuando strides=1)	✓ Sí
Atención	× No	× No	✓ Squeeze-and-Excitation (SE) en cada bloque	✓ SE, pero solo en etapas posteriores (para reducir overhead)
Método de Escalado	Manual: aumento de módulos Inception apilados	Aumento manual de profundidad (18 → 34 → 50 → 101 → 152)	Escalado compuesto uniforme: Profundidad $\times\alpha$, Anchura $\times\beta$, Res $\times\gamma$	Escalado compuesto no uniforme + aprendizaje progresivo (imágenes pequeñas → grandes durante entrenamiento)
Resolución de Entrada	Fija (224×224)	Fija (usualmente 224×224), pero flexible	Escala con el tamaño del modelo: B0: 224 → B7: 600	Menor que V1 para la misma precisión: S: 384, M: 480, L: 480–512
Tamaños Típicos	• GoogLeNet (6.8M params) • Inception-v3 (23.8M) • Inception-v4 (42.7M)	• ResNet18 (11M params) • ResNet34 (21M) • ResNet50 (25M) • ResNet101 (44M)	• B0 (5.3M) • B3 (12M) • B5 (30M) • B7 (66M)	• S (21M) • M (54M) • L (120M)
Precisión (ImageNet)	• GoogLeNet: ∼69.8% • Inception-v3: ∼78.8%	• ResNet50: ∼76% • ResNet101: ∼78%	• B0: ∼77% • B3: ∼82% • B7: ∼84.4%	• S: ∼83.9% • M: ∼85.2% • L: ∼85.7%
Velocidad de Entrenamiento	Moderada	Moderada	Lenta (debido a entradas de alta resolución y profundidad)	Mucho más rápida (hasta 11× vs V1 con precisión similar)
Eficiencia de Inferencia	Buena; diseño eficiente en parámetros para su época	Buena para modelos pequeños (18/34); más pesada para 101+	Muy eficiente en parámetros, pero alta resolución afecta latencia	Mejor relación FLOPs/precisión; optimizado para TPU/GPU
Mejor Para	• Comprensión histórica de CNNs • Extracción de características multi-escala	• Modelos base • Transfer learning con datos limitados • Prototipado rápido	• Despliegue con recursos limitados • Cuando se necesita alta precisión con menos parámetros	• Sistemas de producción que necesitan velocidad + precisión • Entrenamiento desde cero o fine-tuning
fastai / PyTorch	✓ Nativo en torchvision (`googlenet`, `inception_v3`)	✓ Nativo en torchvision y fastai	✓ Vía `timm` (ej., `efficientnet_b0`)	✓ Vía `timm` (ej., `efficientnetv2_s`)
Debilidad Principal	Arquitectura obsoleta; menor precisión que modelos modernos	Eficiencia de parámetros subóptima; obsoleto vs modelos modernos	Entrenamiento lento; SE añade overhead computacional; alta resolución = mucha memoria	Ecosistema ligeramente menos maduro; no está en torchvision

Table 1. Comparación de las Arquitecturas GoogLeNet (Inception), ResNet, EfficientNetV1 y EfficientNetV2

the project; not your personal conclusions about the work you have done!

References

[1] Stéphane Lathuilière, Pablo Mesejo, Xavier Alameda-Pineda, and Radu Horaud. A comprehensive analysis of deep regression. *IEEE transactions on pattern analysis and machine intelligence*, 42(9):2065–2081, 2019. 1

[2] Pablo Mesejo, Daniel Pizarro, Armand Abergel, Olivier Rouquette, Sylvain Beorchia, Laurent Poincloux, and Adrien Bartoli. Computer-aided classification of gastrointestinal lesions in regular colonoscopy. *IEEE transactions on medical imaging*, 35(9):2051–2063, 2016. 1

[3] Víctor A Vargas-Pérez, Pablo Mesejo, Manuel Chica, and Oscar Cordón. Deep reinforcement learning in agent-based simulations for optimal media planning. *Information Fusion*, 91:644–664, 2023. 1

Característica	ConvNeXt	Swin Transformer	MaxViT
Primera Publicación	2022 (Liu et al., Meta/FAIR)	2021 (Liu et al., Microsoft)	2022 (Tu et al., Google)
Idea Principal	Modernización de ResNet con técnicas inspiradas en Transformers, demostrando que CNNs puras pueden competir	Transformer jerárquico con atención en ventanas locales y desplazamiento para conexión entre ventanas	Combinación de convoluciones + atención local (block) + atención global diluida (grid) en cada bloque
Tipo de Arquitectura	CNN pura (sin atención)	Transformer puro (sin convoluciones en backbone)	Híbrido CNN + Transformer
Bloque Básico	Inverted bottleneck con depthwise conv 7×7, LayerNorm, GELU	Swin Transformer Block: Window-MSA → MLP → Shifted-Window-MSA → MLP	MBConv → Block Attention → Grid Attention
Mecanismo de Atención	× No usa atención (solo convoluciones)	✓ Multi-head Self-Attention en ventanas 7×7, con shift en capas alternas	✓ Block Attention (local en ventanas) + Grid Attention (global diluida)
Receptive Field Global	Solo en capas profundas (por apilamiento)	Gradual mediante shifted windows	✓ Desde las primeras capas (via Grid Attention)
Complejidad Computacional	O(n) lineal respecto a resolución	O(n) lineal (atención local en ventanas fijas)	O(n) lineal (block + grid attention)
Representación Jerárquica	✓ Sí (4 etapas con downsampling)	✓ Sí (patch merging entre etapas)	✓ Sí (estructura multi-escala)
Resolución de Entrada	224×224 (escalable)	224×224 (escalable a alta resolución)	224×224 a 512×512
Tamaños Típicos	• ConvNeXt-T (29M) • ConvNeXt-S (50M) • ConvNeXt-B (89M) • ConvNeXt-L (198M)	• Swin-T (29M) • Swin-S (50M) • Swin-B (88M) • Swin-L (197M)	• MaxViT-T (31M) • MaxViT-S (69M) • MaxViT-B (120M) • MaxViT-L (212M)
Precisión (ImageNet-1K)	• T: ∼82.1% • B: ∼83.8% • L: ∼84.3% • XL (21K): ∼87.8%	• T: ∼81.3% • B: ∼83.5% • L: ∼86.4% • L (21K): ∼87.3%	• T: ∼83.6% • B: ∼85.0% • L: ∼86.4% • L (21K): ∼89.5%
Velocidad de Entrenamiento	Rápido (operaciones de convolución optimizadas)	Moderado (overhead de atención)	Moderado-lento (múltiples tipos de atención)
Eficiencia de Inferencia	Excelente (convoluciones muy optimizadas en GPU)	Buena (atención local eficiente)	Buena (pero más complejo que los otros)
Backbone Universal	✓ Sí (detección, segmentación)	✓ Sí (muy popular para downstream tasks)	✓ Sí (clasificación, detección, segmentación)
Mejor Para	• Cuando se prefiere simplicidad de CNNs • Inferencia eficiente • Transfer learning	• Backbone para tareas densas • Estado del arte en detección/segmentación • Preentrenamiento a gran escala	• Máxima precisión • Cuando se necesita contexto global desde el inicio • Tareas que requieren relaciones de largo alcance
Disponibilidad	✓ torchvision + `timm`	✓ torchvision + `timm`	✓ `timm` (`maxvit_*`)
Debilidad Principal	Menor capacidad de modelar dependencias globales que Transformers	Conexión entre ventanas limitada por shifted windows	Mayor complejidad de implementación; menos maduro que Swin/ConvNeXt

Table 2. Comparación de Arquitecturas Modernas: ConvNeXt, Swin Transformer y MaxViT