Implementación de algoritmos de aprendizaje automático para la optimización de procesos industriales

Olai Dizy Aranda

Universidad de Oviedo
Grado en Física

Tutores: Santiago Folgueras Gómez e Isidro González Caballero

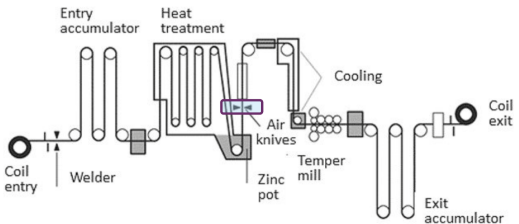
Índice

1. Motivación y contexto
2. Fundamentos del modelo
3. Implementación
4. Resultados
5. Conclusiones
6. Anexo

El problema industrial: galvanizado de acero

- Proceso industrial de ArcelorMittal™: deposición de zinc sobre acero
- Las **cuchillas de aire** regulan el grosor de la capa
- Objetivo: asistir al operario calculando los parámetros óptimos
- Modelo original: MLP en PyTorch entrenado por TheNextPangea SL™
- Variables de entrada (21): de control (17) y de estado (4)

Objetivo del TFG: implementar el algoritmo de optimización en una FPGA

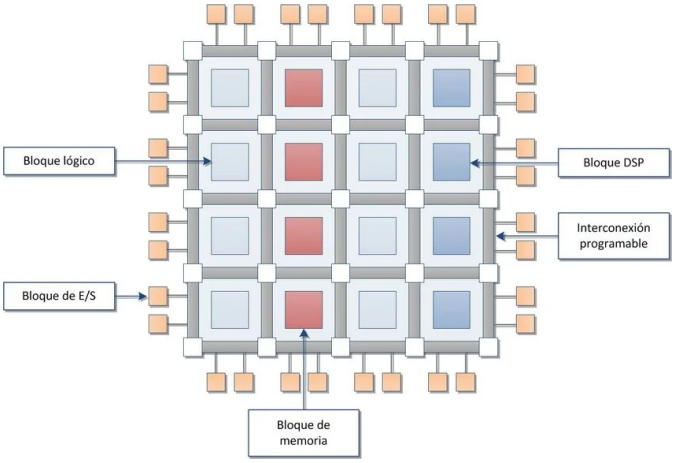


¿Qué es una FPGA?

	Flexib.	Veloc.
CPU	●●●	●
GPU	●●	●●
FPGA	●●	●●●
ASIC	●	●●●

Recursos clave:

- **DSPs**: multiplicaciones
- **LUTs**: lógica + funciones (sigmoide)
- **BRAM**: pesos e intermedios



Características de las FPGAs

Ventajas

- **Latencia determinista** (μs): respuesta predecible y adecuada para sistemas en tiempo real.
- **Procesamiento en paralelo**: permite ejecutar múltiples operaciones simultáneamente.
- **Bajo consumo energético**: mayor eficiencia energética para tareas específicas.

Desventajas

- **Mayor complejidad de diseño**: requiere conocimientos de arquitectura de hardware.
- **Menor flexibilidad en tiempo de ejecución**: la reconfiguración es más costosa que en software.
- **Tiempo de desarrollo elevado**: desarrollo más lento.

Instrumentación científica y FPGAs

Usos de FPGAs en instrumentación e industria

- **CMS (LHC):** colisiones a 40 MHz, decisión de aceptar en $<4\,\mu$s
- **DUNE:** 145 EB/año sin filtrado
- **Trading de alta frecuencia:** transacciones en cientos de ns
- **Redes 5G:** latencia determinista
- **Procesado de imagen:** reconstrucción de imagen al instante
- **Radar y defensa:** muestreo y procesado a muy alta velocidad

En aplicaciones científicas $\Rightarrow$ *Intelligence on detector*: procesar cerca de la fuente

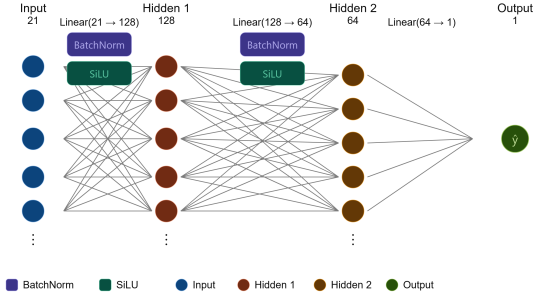
Herramientas de Trabajo

- IDEs: Vitis, Vivado
- Lenguajes: Python, C++, HLS
- Librerías: Pytorch, PYNQ, *hls4ml*
- Hardware/Sistemas: Kria KV260, Sistema de colas del C^3 del grupo

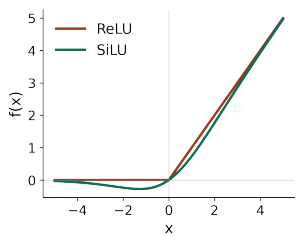
Este trabajo aplica un flujo de trabajo académico a un problema industrial.

El algoritmo de optimización

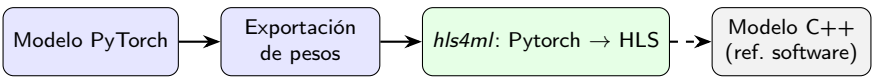
Arquitectura del MLP



- Entrada: 21 variables normalizadas
- Capas ocultas: 128 → 64
- Salida: grosor predicho (escalar)
- Activación: **SiLU** (gradiente suave, sin discontinuidad)
- Normalización: **BatchNorm1d** entre capas

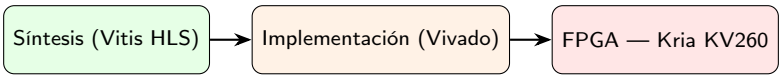


Flujo de trabajo: de PyTorch a HLS



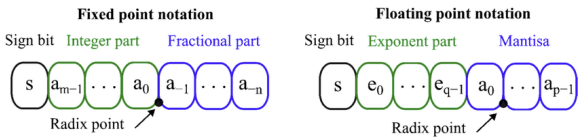
- Pasada hacia adelante y exportación de pesos con *hls4ml*
- Propagación hacia atrás (gradiente) traducido y calculado a mano
- Modelo de C++ obtenido a partir de adaptar y traducir el código de HLS

Flujo de trabajo: de HLS a FPGA



- Cambio en tamaño de bits o disposición de memoria $\rightarrow$ Nueva síntesis ($\sim$1 h)
- Proceso iterativo de síntesis hasta llegar a valores de utilización aceptables
- Implementación (muy costosa) solo si los resultados de la síntesis son buenos
- Vitis y Vivado consumen $>$ 64 GB de RAM $\rightarrow$ Se ejecutan las compilaciones en el C^3
- Tests en hardware real (Kria) con conexión remota

Punto fijo frente a punto flotante



Formato `ap_fixed<W,I>`

W bits totales, *I* para la parte entera

Formatos usados en este trabajo:

- Pesos: `<16,5>`
- Activaciones/entrada: `<24,13>`
- Learning rate: `<24,1>`
- LUTs sigmoide: `<18,8>`

Compromiso: menor área y mayor velocidad, pero pérdida de precisión.

Optimización de recursos: *factores de reutilización*

Problema: $21 \times 128 = 2688$ multiplicaciones por capa

Con *reuse factor* r: se instancian $\lceil N/r \rceil$ multiplicadores, reutilizados r veces

Capa	**Operación**	**RF**	**Mults/ciclo**
Capa 1	Linear $21 \to 128$	168	16
Capa 2	Linear $128 \to 64$	256	32
Capa 3	Linear $64 \to 1$	64	1
Backprop 3	$W_3^T \cdot \delta_3$	1	64
Backprop 2	$W_2^T \cdot \delta_2$	256	32
Backprop 1	$W_1^T \cdot \delta_1$	384	7

Las funciones SiLU/sigmoide se aproximan con **LUTs de 1024 entradas** (DSPs insuficientes).

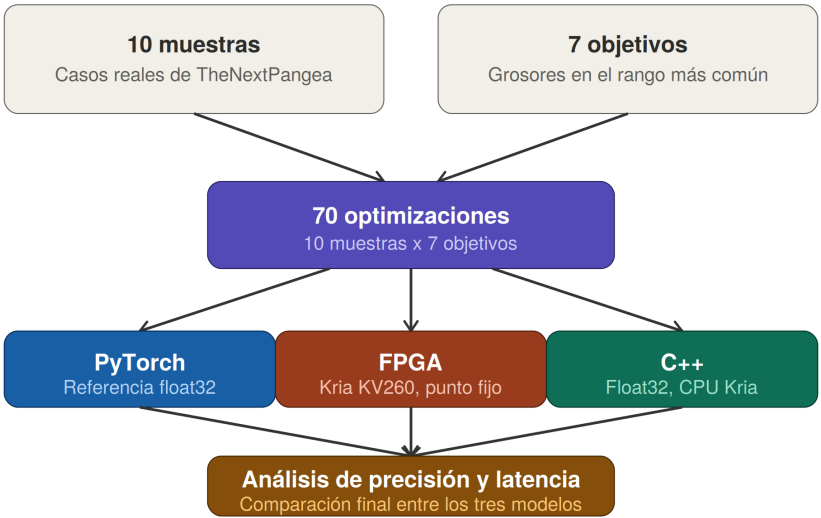
Uso de recursos: síntesis vs. implementación

Recurso	Síntesis (Vitis)	Implementación (Vivado)
LUTs	∼69 %	**40,72 %**
DSPs	∼45 %	45,27 %
BRAM	∼50 %	51,39 %

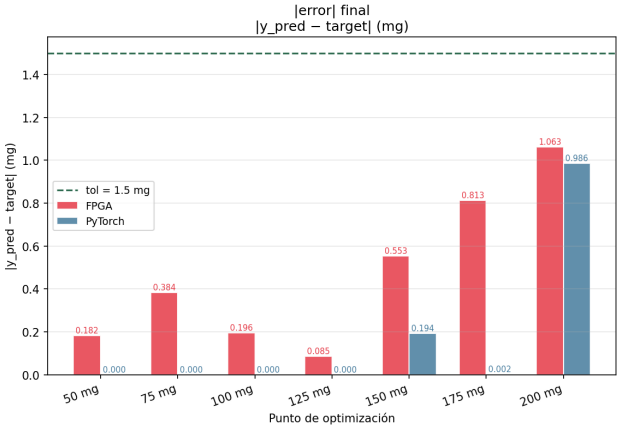
Frecuencia	Latencia total	Potencia
100 MHz	145 ms (1000 iters.)	0,82 W

- Síntesis: estimación
- Implementación: realidad

Metodología de comparación

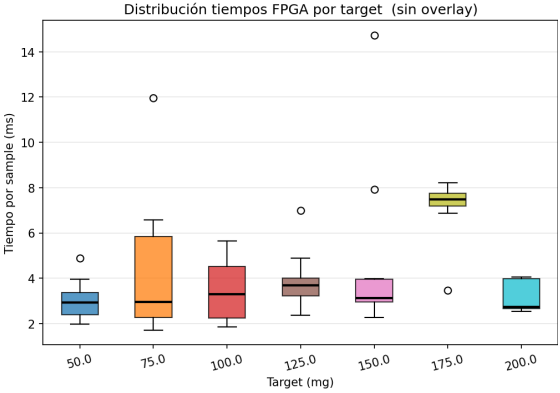


Comparativa de errores: FPGA vs. PyTorch

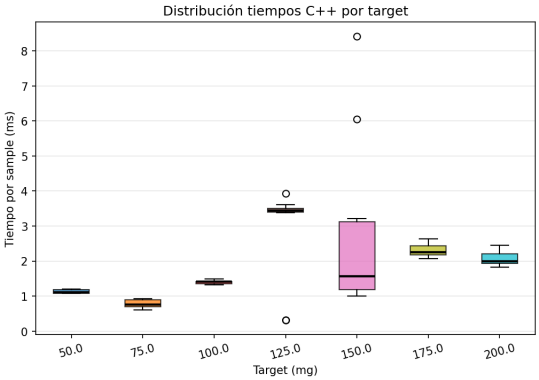


- Ambos modelos convergen dentro de la tolerancia de 1,5 *mg*
- FPGA usa $lr = 5 \cdot 10^{-4}$ (mayor, necesario por la precisión reducida)
- PyTorch usa $lr = 10^{-5}$ (menor, más suave)
- Parada forzada a las 1000 iteraciones
- Mediana de error relativo del MLP de FPGA $\sim 0{,}6\,\%$

Comparativa de tiempos: FPGA vs. C++



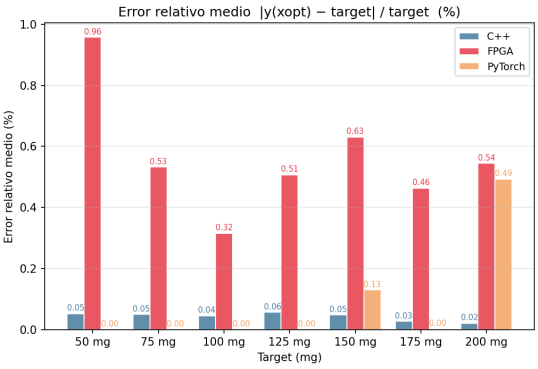
Kernel FPGA

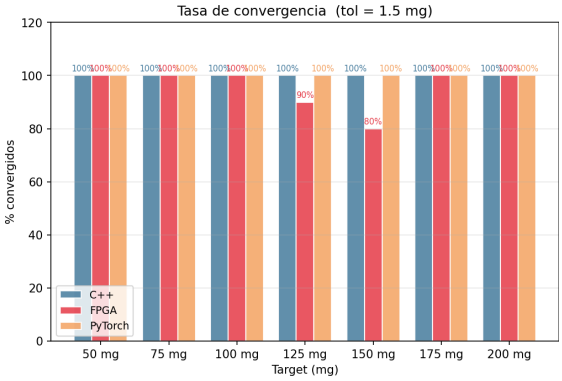


C++

De media, C++ es más rápido y con menor dispersión que el kernel de FPGA.

Comparativa de tiempos y errores: tres modelos





	PyTorch	**FPGA (kernel)**	**C++**
Tiempo	$\sim$14 s	$<$8 ms	$<$4 ms
Precisión	Alta	Media	Alta
Portabilidad	Media	Baja	Alta

Conclusiones

Conclusiones del trabajo realizado:

- Se pudo reproducir el flujo de trabajo exitosamente usando tecnología de estado del arte
- El trabajo sirvió para aprender en profundidad el ciclo completo de desarrollo
- El modelo final cumple con los requerimientos técnicos
- Queda demostrado que el flujo de trabajo es aplicable a problemas industriales
- El modelo sirve de prueba de concepto y práctica para trabajos futuros

Lecciones aprendidas y trabajo futuro:

- El modelo no estaba diseñado para FPGA $\rightarrow$ Entrenar con pesos discretizados (8 bits)
- Se podría diseñar el hardware para que lea directamente datos de sensores
- Aplicar a problemas que exploten mejor las ventajas en latencia

Gracias

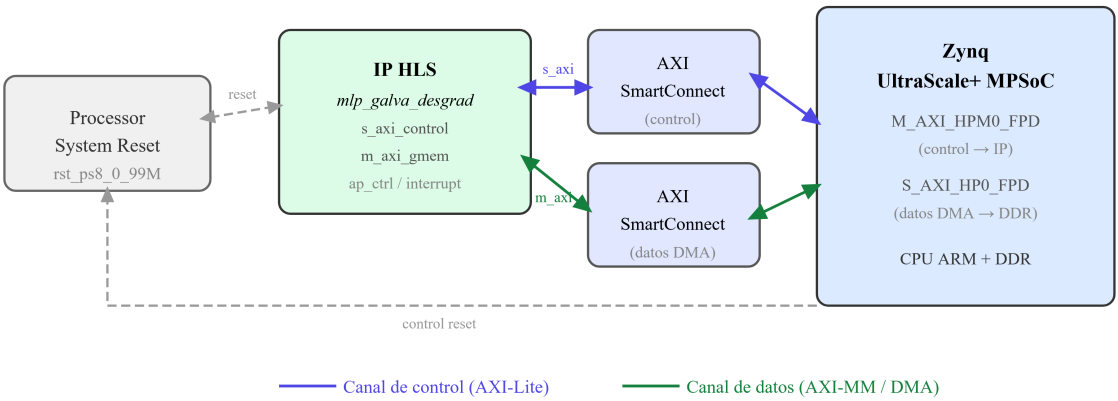
Olai Dizy Aranda

Tutores: Santiago Folgueras Gómez e Isidro González Caballero

Implementación de algoritmos de aprendizaje automático para la optimización de procesos industriales

Universidad de Oviedo — Grado en Física

Diseño de bloques



Report Síntesis resumido

Módulo / Bucle	Latencia (ciclos)	Latencia (ms)	Intervalo	BRAM	DSP	FF	LUT
`mlp_galva_desgrad_complete`	14 536 001	145,4	14 536 002	150 (52 %)	563 (45 %)	76 242 (32 %)	81 166 (69 %)
↪ *Iter_Loop* ×1000	14 536 000	145,4	—	—	—	—	—
Capa 1 — Linear 21 → 128	241	$2{,}41 \times 10^{-3}$	168	—	16 (1 %)	6 531 (2 %)	9 440 (8 %)
↪ *ReuseLoop* ×168	240	$2{,}40 \times 10^{-3}$	1	—	—	—	—
Activación 1 — SiLU 128	0	0	1	—	256 (20 %)	—	5 120 (4 %)
Capa 2 — Linear 128 → 64	257	$2{,}57 \times 10^{-3}$	256	—	32 (2 %)	6 173 (2 %)	13 428 (11 %)
↪ *ReuseLoop* ×256	256	$2{,}56 \times 10^{-3}$	1	—	—	—	—
Activación 2 — SiLU 64	0	0	1	—	128 (10 %)	—	2 560 (2 %)
Capa 3 — Linear 64 → 1	134	$1{,}34 \times 10^{-3}$	64	1 (∼0 %)	1 (∼0 %)	1 798 (∼0 %)	2 343 (2 %)
Derivada MSE (δ_3)	0	0	1	—	—	—	31 (∼0 %)
Capa traspuesta 3 — $W_3^T \cdot \delta_3$	0	0	0	—	64 (5 %)	—	2 560 (2 %)
Capa traspuesta 2 — $W_2^T \cdot \delta_2$	258	$2{,}58 \times 10^{-3}$	256	—	32 (2 %)	7 720 (3 %)	12 584 (10 %)
↪ *ReuseLoop* ×256	257	$2{,}57 \times 10^{-3}$	1	—	—	—	—
Capa traspuesta 1 — $W_1^T \cdot \delta_1{}^*$	4 226	$4{,}226 \times 10^{-2}$	4 224	—	7 (∼0 %)	7 270 (3 %)	6 906 (5 %)
↪ *ReuseLoop* ×384	4 225	$4{,}225 \times 10^{-2}$	11	—	—	—	—
Máscara — *Mask_Loop* ×21	23	$2{,}3 \times 10^{-4}$	—	—	—	7 (∼0 %)	64 (∼0 %)
Descenso del gradiente ×21	3 002	$3{,}002 \times 10^{-2}$	21	—	21 (1 %)	6 469 (2 %)	4 610 (3 %)

Report Implementación resumido

Categoría	Parámetro	Valor
Utilización de recursos	CLB LUTs (total)	47 693 / 117 120 (40,72 %)
	LUT como lógica	44 063 / 117 120 (37,62 %)
	LUT como memoria	3 630 / 57 600 (6,30 %)
	Block RAM Tile	74 / 144 (51,39 %)
	URAM	0 / 64 (0,00 %)
	DSPs	565 / 1 248 (45,27 %)
Potencia (estimada post-implementación)	Relojes	0,096 W
	Lógica CLB	0,116 W
	Señales	0,211 W
	Block RAM	0,029 W
	DSPs	0,077 W
	Potencia dinámica	**0,530 W**
	Potencia estática	0,291 W
	Potencia total estimada	**0,821 W**
	Temperatura de juntura	26,9 °C (máx. ambiente: 83,1 °C)