

**FACULTAD DE INGENIERÍA**  
**Sede Bogotá**

**FORMATO**  
**REGISTRO ASIGNATURA TRABAJO DE GRADO**

Favor diligenciar en computador. Modifique el tamaño de los campos de texto según su necesidad.  
Para cualquier inquietud remítase al Acuerdo 037 de 2017 del Consejo de Facultad de Ingeniería  
sede Bogotá:

<http://www.legal.unal.edu.co/sisjurun/normas/Norma1.jsp?i=88861>

<b>PROGRAMA CURRICULAR</b>	Ingeniería Electrónica	<b>FECHA</b>	<b>12</b>	<b>11</b>	<b>2020</b>
----------------------------	------------------------	--------------	-----------	-----------	-------------

**DATOS DEL ESTUDIANTE**

NOMBRES: <b>Wilson Javier</b>						APELLIDOS: <b>Almarío Rodríguez</b>					
TIPO IDENTIFICACIÓN:	T.I.		C.C	<b>X</b>		C.E.		NÚMERO: 1026572314			
CORREO INSTITUCIONAL: wjalmarior@unal.edu.co								TELÉFONO: 3212666241			

**MODALIDAD DE TRABAJO DE GRADO**

(Seleccione una opción)

<b>I. Trabajos investigativos:</b> <input type="checkbox"/>	<b>II. Prácticas de extensión:</b> <input type="checkbox"/>	<b>III. Asignaturas de postgrado:</b> <input type="checkbox"/>
a. Trabajo monográfico <input type="checkbox"/>	a. Emprendimiento empresarial <input type="checkbox"/>	(Para este caso sólo imprima y diligencie la página 5)
b. Proyecto final <input checked="" type="checkbox"/>	b. Pasantía <input type="checkbox"/>	

c. Participación en proyecto de investigación <input type="checkbox"/>	c. Proyecto social <input type="checkbox"/>	
--	---	--

**AVAL DEL DOCENTE DIRECTOR** (En caso de modalidades I o II):

NOMBRES: Carlos Ivan Camargo Bareño	DEPARTAMENTO: Electrónica
CORREO INSTITUCIONAL: <a href="mailto:cicamargoba@unal.edu.co">cicamargoba@unal.edu.co</a>	TELÉFONO-EXT. : 3004435306

### COMPONENTES

#### 1. TÍTULO DEL TRABAJO DE GRADO

**Acelerador de red neuronal recurrente (RNN) para una aplicación NLP en FPGA**

#### 2. INTRODUCCIÓN Y JUSTIFICACIÓN

La inteligencia artificial (IA) en los últimos años ha tomado gran interés debido a sus aplicaciones y su impacto en la cuarta revolución industrial, en especial encontramos el aprendizaje máquina (*Machine Learning ML*) como una rama de estudio de la IA, en la que un sistema toma un volumen de datos de entrada y es entrenado para identificar patrones y hacer predicciones entre otras aplicaciones. Una de las técnicas de ML es el aprendizaje profundo (*deep learning*), en el cual encontramos redes neuronales con varias capas ocultas las cuales involucran cálculos matriciales y vectoriales, que pueden ser ejecutados de forma paralela y que permite obtener una buena precisión en la predicción [7].

La precisión de la red en la inferencia está condicionada a diversos parámetros del modelo como son el tipo de neuronas y el número de capas a implementar. El proceso de diseño de redes neuronales se divide en dos etapas; el entrenamiento y la inferencia. En el entrenamiento se calculan los pesos del modelo usando un proceso de optimización, y debido a el costo computacional del mismo este se realiza generalmente en GPUs. Mientras que en la inferencia se evalúa la red neuronal con los pesos previamente calculados, por lo que se requieren menos recursos de memoria y cómputo lo que hace viable su inferencia en CPUs.

Uno de los modelos de redes neuronales de mayor interés en el estado del arte son las Redes Neuronales Recurrentes (Recurrent Neural Network RNN). Este interés se atribuye al hecho a que una de las ventajas que tienen este tipo de redes es que estas pueden ser entrenadas a partir de una secuencia de datos y usadas para tomar decisiones en tiempo real [2]. Esta característica permite obtener un buen desempeño en aplicaciones NLP (Natural Language Processing) tales como; sistemas de reconocimiento de voz [3], máquinas traductoras [4] y análisis de escena [5] entre otras. Como desventaja estas redes presentan un alto uso de recursos de memoria y de cómputo, lo cual ha motivado el desarrollo de implementaciones en hardware que tienen como objetivo mejorar el desempeño.

Por lo general, el uso de CPUs o GPUs en el proceso de inferencia ofrece un buen desempeño, sin embargo el consumo de potencia, la precisión y el paralelismo obtenidos están limitados por la arquitectura del hardware

y las librerías de ML disponibles. Por esta razón, en los últimos años las implementaciones de redes neuronales en hardware a través de plataformas reconfigurables como las FPGAs se han incrementado, debido a que se puede lograr un alto desempeño a un bajo consumo de potencia logrando hasta 10 veces menos con relación a las CPUs y las GPUs [1].

Por ejemplo, en [1] se compara una CPU (Xeon E5-2650v2) a 2.6 GHz, una GPU (GTX TITAN X) a 1 GHz y una FPGA (Virtex7-485t) a 150 MHz implementando una RNN-LSTM, con la CPU se obtiene 103 GOP/s con 95 w, la GPU 1828 GOP/s con 250 w y la FPGA 1833 GOP/s con 19.63 w.

En [6] se realiza una comparación de rendimiento entre una FPGA, una CPU y una GPU, llegando a las siguientes observaciones; el rendimiento de la FPGA es superior  $\sim 10x$  rendimiento/Watt que las CPU y GPU. Con respecto a las ASIC la FPGA es  $\sim 7x$  menos eficiente [6], pero las características de programación y reconfiguración de la FPGA hacen que sea posible implementar diseños personalizados [8]. Por esta razón las FPGAs son atractivas para los sistemas embebidos en especial los sistemas móviles en los cuales resulta indispensable optimizar el consumo de energía, pero a su vez aún hay mucho por mejorar por ejemplo para la implementación de RNN está por mejorar la superposición del tiempo de cálculo con el tiempo de transferencia de datos [9]. Además actualmente como principal foco de investigación es la implementación de aceleradores en FPGA para modelos específicos de redes neuronales esto principalmente debido a que cuando se implementa un modelo de red neuronal a un problema específico usualmente solo se deben configurar parámetros específicos en la red para un determinado problema [1].

Así podemos observar de lo anterior que la implementación de aceleradores en FPGA es una necesidad actual y por el cuál falta aún explorar mucho en la implementación de aceleradores en FPGA para modelos específicos de redes neuronales como lo mencionado en el anterior párrafo, por esta razón se plantea una propuesta de proyecto para la implementación de un acelerador en FPGA que permita realizar el proceso de inferencia de una RNN a través de un co-diseño hardware-software.

### **3. OBJETIVO GENERAL** (En caso de modalidades I o II)

Implementar un acelerador de una red neuronal recurrente (RNN) en una FPGA para una aplicación NLP.

### **4. OBJETIVOS ESPECÍFICOS** (En caso de modalidades I o II)

1. Evaluar diferentes modelos de RNN para una aplicación NLP usando TensorFlow en CPU y GPU, y determinar el modelo más viable para su implementación en hardware.
2. Implementar un acelerador de RNNs en FPGA basado en el modelo obtenido previamente en TensorFlow, para una aplicación NLP.
3. Desarrollar un wrapper para un bus de comunicaciones que permita la portabilidad del acelerador.
4. Comparar las métricas obtenidas (e.g. precisión, OP/w) en la CPU, la GPU y el acelerador de RNN implementado en una FPGA.

## 5. ANTECEDENTES

Al momento de implementar redes neuronales generalmente la ruta común es usar GPU's pero como se mencionó en la sección anterior el interés por la implementación en la FPGA ha incrementado debido a que muestra un mejor rendimiento respecto a la energía usada, la primera vez que se usó una FPGA para una red neuronal fue en 1994 por D.S Ray, pero solo hasta el 2012 con el nacimiento de AlexNet en ILSVRC se marcó un punto de referencia del desarrollo de redes neuronales seguida del nacimiento de modelos como VGGNet, GoogleNet, ResNet marcando la tendencia al incremento de la complejidad en las redes neuronales, al mismo tiempo que se notaba el uso de las FPGA como solución a este problema del incremento de la complejidad computacional, hasta el 2018 se encontraban 69 artículos en la IEEE eXplore relacionadas con la implementación de aceleradores para redes neuronales en FPGA [1].

Como antecedente recientes de la implementación de RNN en hardware, se tiene la implementación de una LSTM (Long-Short time memory) publicada en el 2016 usando una Zynq 7020 FPGA, usa el modelo de lenguaje por carácter prediciendo el siguiente carácter dado un carácter previamente. La topología adoptada fueron dos capas de LSTM con 128 capas ocultas, el entrenamiento fue realizado en Torch7, El porcentaje de error de la red fue del 2.8% [10].

En [11] publicado en 2017 se propone el diseño de un acelerador para una red RNN tipo LSTM que optimiza el rendimiento y los requerimientos de comunicación, en donde se logra un pico de rendimiento del acelerador de 7.26 GFLP/s. En este diseño se añade un área de buffer para guardar directamente los parámetros del estado de la LSTM de las capas ocultas, eliminando así la necesidad de cargar parámetros todo el tiempo y con esto reduciendo los requerimientos de ancho de banda de hardware.

En [12] se implementa 3 diferentes diseños de aceleradores para una RNN tipo LSTM, cada diseño se diferenciaba en la forma en que implementan las unidades de puerta (gate unit), para un diseño tenían tres y procesaba los datos desde una DMA, para otro almacenaban los pesos y aumentaban el número de unidades de puerta, y para el último diseño cambiaban la organización de las unidades de puerta. Se implementaron plataformas Zedboard Zynq ZC7020 y ZC706, el porcentaje de error en promedio para las celdas de memoria fue de 3.9 % y para los vectores de las capas de salida fue de 2.8%.

De esta revisión de antecedentes se puede notar que el tipo de red RNN más implementada en una FPGA es LSTM.

## 6. METODOLOGÍA

### ● Evaluación y determinación de un modelo de RNN

La evaluación del modelo RNN se hará basada en la revisión bibliográfica, para determinar el modelo RNN a usar se tendrá como criterio de selección la más usada en el estado del arte para aplicaciones de NLP, y la que mejor rendimiento tenga usando Frameworks como TensorFlow y PyTorch. Después de esta selección se diseñará una aplicación tipo NLP con su respectiva base de datos, para luego implementar la aplicación en una CPU y una GPU por medio de TensorFlow y PyTorch. En este proceso se analizarán los algoritmos que realizan las operaciones implicadas en software para el tipo de red seleccionada (e.g. funciones de activación - recursividad - inferencia), para luego plantear los recursos que se van a usar en hardware como tipo de datos, cantidad de memoria, si se va usar una memoria adicional (off-ship), cantidad de operaciones entre otras características implícitas en la red.

### ● Diseño de un acelerador de RNN para ser implementado en una FPGA

Una vez determinado el modelo de RNN a implementar, se realizará un análisis teniendo en cuenta los procesos y funciones involucradas en esta red para determinar el esquema y arquitectura de la red RNN a implementar en el acelerador. Esta implementación puede ser usando solo módulos que optimicen las funciones de activación o la implementación de las capas ocultas ya sea en diferentes ordenes o números de matrices. Para ello se plantearán las alternativas presentes y como criterio de selección se tendrá en cuenta viabilidad, el uso en cuanto a los recursos globales requeridos y la relación costo/beneficio (i.e., recursos específicos requeridos frente a la eficiencia esperada).

### ● Evaluación del acelerador

Para la evaluación se hará desde lo más básico hasta lo más complicado, seguido de parámetros simplificados de la red hasta completar el modelo seleccionado de acelerador para RNN, proceso iterativo en el cual se definirá la arquitectura apropiada para el acelerador, haciendo una selección de la arquitectura se procede a la evaluación de la misma usando la base de datos seleccionada que se usó en la evaluación del framework, a partir de esto se obtienen las métricas (e.g. OP/s - OP/s/w - tiempo de inferencia - Almacenamiento en memoria) que aporten mayor información en la evaluación de valor de los aceleradores para RNN como alternativa favorable en la optimización de redes neuronales y su implementación. Por último se realizará la comparación con las métricas obtenidas con la CPU y la GPU, y se concluirá.

## 7. RECURSOS

1. FPGA Nexys A7
2. FPGA Zynq 7020
3. GPU NVIDIA GeForce GTX 750 Ti
4. CPU Intel<sup>(R)</sup> Core<sup>(TM)</sup> i7-6700 @ 3.40GHz
5. Base de datos IEE Xplore, ScienceDirect y Scopus
6. Base de Datos: [Yelp Open Dataset](#), [Penn Treebank](#), [Stanford Sentiment Treebank](#)
7. Framework de Machine Learning TensorFlow y PyTorch
8. Software de desarrollo Vivado design suite Xilinx

## 8. CRONOGRAMA DE ACTIVIDADES

ACTIVIDADES A REALIZAR	Semanas de ejecución de cada actividad															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Revisión y actualización del estado del arte.	X															
Determinar modelo RNN favorable para la implementación	X															
Evaluación del modelo seleccionado en tensorflow		X	X													
Diseño de arquitectura del acelerador				X	X	X	X	X								
Implementar acelerador en la FPGA						X	X	X	X							

Validar desempeño de la arquitectura								X	X	X									
Determinar protocolos de comunicación para su portabilidad										X	X								
Implementar la arquitectura de integración											X	X	X						
Obtener métricas de desempeño													X	X					
Pruebas e implementación														X	X				
Realizar documentación del proyecto								X	X	X	X	X	X	X	X	X	X	X	X

## 9. PRESUPUESTO Y FUENTES DE FINANCIACIÓN

N°	Recurso	Costo (COP)
1	FPGA Nexys A7	899.277
2	FPGA Zynq 7020	916.075
3	GPU NVIDIA GeForce GTX 750 Ti	582.825
4	CPU Intel <sup>(R)</sup> Core <sup>(™)</sup> i7-6700 @ 3.40GHz	4'000.000
5	Base de Datos IA NLP	---
6	Framework de Machine Learning TensorFlow	---
7	Software de desarrollo Vivado design suite Xilinx	---
Total		5'483.018

N°	Profesional	hora (COP)	cant. (h)	Total (COP)
1	Asesor Docente	160.000	64	10'240.000
2	Estudiante Pregrado	12.500	640	8'000.000
Total				18'240.000
3	Recursos			5'483.018
Total				23'723.018

Este proyecto se desarrollará como miembro del grupo de física nuclear de la Universidad Nacional de Colombia (GFNUN) quienes poseen recursos de cómputo y sistemas de desarrollo FPGAs. Adicionalmente, se cuenta con las bases de datos proporcionadas por la Universidad Nacional, bases de datos de clasificación y reconocimiento de imágenes para *Machine Learning Open Source*, y software de desarrollo con licencia para estudiantes.

## 10. CRITERIOS DE EVALUACIÓN (En caso de modalidades I o II)

CRITERIOS DE EVALUACIÓN	Porcentaje del criterio
Usabilidad	35%
Correcto funcionamiento	35%
Sustentación	30%

## 11. RESULTADOS ESPERADOS (En caso de modalidades I o II (a.) o II (c.))

Se espera obtener un modelo implementable para redes neuronales RNN que realice procesos de inferencia en plataformas de hardware FPGA, que presente características favorables en el consumo de energía por dato evaluado, para aplicaciones con sistemas de bajo consumo, en comparación con la implementación de la misma red RNN y su modelo equivalente RNN en CPU y GPU.

## 12. BIBLIOGRAFÍA (En caso de modalidad I)

- [1]T. Wang, C. Wang, X. Zhou and H. Chen, "An Overview of FPGA Based Deep Learning Accelerators: Challenges and Opportunities," *2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, Zhangjiajie, China, 2019, pp. 1674-1681.
- [2]E-RNN: Design Optimization for Efficient Recurrent Neural Networks in FPGAs W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent neural network regularization," arXiv preprint arXiv:1409.2329, 2014.
- [3]A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *IEEE Int. Conf. on ICASSP*, 2013, pp. 6645–6649.
- [4]I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.
- [5]W. Byeon, M. Liwicki, and T. M. Breuel, "Scene analysis by mid-level attribute learning using 2d lstm networks and an application to web-image tagging," *Pattern Recognition Letters*, vol. 63, pp. 23–29, 2015.
- [6] E. Nurvitadhi, Jaewoong Sim, D. Sheffield, A. Mishra, S. Krishnan and D. Marr, "Accelerating recurrent neural networks in analytics servers: Comparison of FPGA, CPU, GPU, and ASIC," *2016 26th International Conference on Field Programmable Logic and Applications (FPL)*, Lausanne, 2016, pp. 1-4.
- [7] V. T. Phat, P. H. Tho, H. B. Dat and C. Chou, "Deep Learning Accelerator on FPGA Using Handwritten Digit Recognition for Example," *2018 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)*, Taichung, 2018, pp. 1-2.
- [8]Ahmed Ghazi Blaiech, Khaled Ben Khalifa, Carlos Valderrama, Marcelo A.C. Fernandes, Mohamed Hedi

Bedoui, A Survey and Taxonomy of FPGA-based Deep Learning Accelerators, Journal of Systems Architecture, Volume 98, 2019, Pages 331-345, ISSN 1383-7621, <https://doi.org/10.1016/j.sysarc.2019.01.007>.

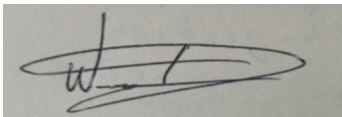

[9] A. X. M. Chang and E. Culurciello, "Hardware accelerators for recurrent neural networks on FPGA," *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, Baltimore, MD, 2017, pp. 1-4.

[10] A.X.M. Chang, B. Martini, E. Culurciello, Recurrent neural networks hardware implementation on FPGA, Int. J. Adv. Res. Electr. Electron. Instrum. Eng. 5 (1) (2016). January

[11] Yijin Guan, Zhihang Yuan, Guangyu Sun, and Jason Cong. 2017. FPGA-based accelerator for long short-term memory recurrent neural networks. In Design Automation Conference (ASP-DAC), 2017 22nd Asia and South Pacific. IEEE, 629634.

[12] A. X. M. Chang and E. Culurciello, "Hardware accelerators for recurrent neural networks on FPGA," *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, Baltimore, MD, 2017, pp. 1-4.

### 13. FIRMAS

FIRMA DEL ESTUDIANTE:	FIRMA DEL DOCENTE DIRECTOR
	

### 14. DATOS DE TRÁMITE COMITÉ ASESOR DE PROGRAMA (Espacio para diligenciar por el Comité Asesor de Programa)

No. CONSECUTIVO	
No. ACTA	
FECHA	



