International Journal of Science, Strategic Management and Technology
Volume **02** Issue **03** March-**2026** | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

Approximate Softmax Architecture for Energy-Efficient Deep Neural Networks

Athul Krishna R,
Department of Electronics and Communication Engineering
Kongunadu College of Engineering and Technology, Trichy.
athulkrish740@gmail.com

Deepak S J,
Department of Electronics and Communication Engineering
Kongunadu College of Engineering and Technology, Trichy.
deepaksj2021@gmail.com

Fayas Mohamed A,
Department of Electronics and Communication Engineering
Kongunadu College of Engineering and Technology, Trichy.
afayasmohamed8@gmail.com

Mr. S. Baskar, M.E., Mba., (Ph.D),
Associate Profeesor,
Department of Electronics and Communication Engineering
Kongunadu College of Engineering and Technology, Trichy.
baskar.sanandh@gmail.com

https://doi.org/10.55041/ijsmt.v2i3.262

Cite this Article: R,, A. K., J, D. S. & A, F. M. (2026). Approximate Softmax Architecture for Energy-Efficient Deep Neural Networks. International Journal of Science, Strategic Management and Technology, *02*(03). https://doi.org/10.55041/ijsmt.v2i3.262

ABSTRACT

Softmax is commonly used in neural network- based classification systems to convert output values into probabilities, but its conventional implementation involves complex operations such as exponentials and division, which are inefficient for FPGA and VLSI-based hardware due to high area, power, and latency requirements. This project presents a hardware- efficient approximate softmax architecture designed for low-power FPGA systems using a simplified Top-1 approximation approach. The proposed design identifies the dominant output class using comparator-based logic and fixed- point arithmetic, thereby eliminating computationally expensive operations while maintaining correct decision-making. The approximate softmax module is implemented using synthesizable SystemVerilog and validated through simulation and synthesis using Xilinx Vivado, with Python used only for basic numerical verification. The developed design is hardware-ready and suitable for integration into FPGA-based neural network accelerators and embedded VLSI systems, with scope for future hardware implementation.

Keywords: Approximate Softmax, FPGA, VLSI, Fixed-Point Arithmetic, System Verilog, Xilinx Vivado, Hardware-Efficient Design.

I) INTRODUCTION

Deep neural networks have become one of the most powerful tools in modern artificial intelligence and are widely used in applications such as image recognition, speech processing, and natural language understanding. These networks often require significant computational resources to perform complex mathematical operations during inference and training. As the demand for deploying intelligent systems on mobile devices, embedded platforms, and edge computing environments continues to grow, reducing power consumption and improving computational efficiency has become an important research challenge in deep learning.

One of the commonly used functions in classification tasks is the Softmax function. It converts the output values of a neural network into probability distributions, allowing the model to determine the most likely class among multiple categories. Although Softmax is mathematically simple, its implementation involves computationally expensive operations such as exponentiation, division, and large summations. These operations increase hardware complexity, energy consumption, and latency, particularly in systems with a large number of output classes.

To address these challenges, researchers have explored approximate computing techniques that simplify the Softmax computation while maintaining acceptable accuracy levels. Approximate Softmax architectures aim to reduce the complexity of exponential and division operations by using simplified mathematical approximations, lookup tables, or hardware-friendly algorithms. By minimizing the number of arithmetic operations required, these approaches significantly improve energy efficiency and processing speed. The objective of this project is to study and design an approximate Softmax architecture that reduces computational cost and power consumption while maintaining reliable performance in deep neural network applications and the proposed approach focuses on optimizing hardware resources.

II) SYSTEMARCHITECTURE FOR APPROXIMATE SOFTMAX NETWORK

An efficient architecture for implementing the Softmax function in deep neural networks with reduced computational complexity and energy consumption. In conventional neural networks, the Softmax layer requires exponential and division operations, which are computationally expensive and consume significant hardware resources. To address this challenge, the proposed architecture introduces an approximate Softmax computation method that simplifies the mathematical operations while maintaining acceptable accuracy. The process begins with input logits or test data, which represent the raw output values generated by the neural network before probability normalization. These values are passed to a maximum value finder, which identifies the largest logit among the inputs. Determining the maximum value helps stabilize the computation and improves numerical efficiency during further processing. After identifying the maximum value, the system proceeds to the index selection logic. This module determines the position or index of the highest logit value among the inputs. The selected index is important for simplifying probability estimation and helps guide the subsequent normalization process. The verified results are further examined through output analysis, where the effectiveness of the approximate method is assessed.

The simplified normalization unit performs an approximation of the Softmax normalization step. Instead of using the traditional exponential- based Softmax calculation, the system employs a simplified mathematical approximation that significantly reduces hardware complexity and power consumption. The data enters the comparator-based processing stage, where input values are compared to generate relative probability estimations.

A] System Overview

The system implements an energy-efficient architecture for computing the Softmax function in deep neural networks using approximation techniques. It begins by processing input logits generated from the neural network and identifying the maximum value to stabilize computation. The index selection and simplified normalization to reduce complex mathematical.

III) BACKGROUNDOF DEEP NEURAL NETWORKS

Deep Neural Networks (DNNs) are an advanced form of artificial neural networks designed to model complex patterns in large datasets. They are inspired by the structure and functioning of the human brain, where interconnected neurons process and transmit information. In artificial neural networks, neurons are represented as computational units that receive input signals, apply mathematical operations, generate output values Traditional machine learning methods rely heavily on manual feature extraction, where important characteristics of the data must be identified and designed by experts. In contrast, deep neural networks automatically learn hierarchical representations of data through multiple processing layers. Each layer extracts more abstract features, enabling the model to recognize complex patterns and relationships within the data. A deep neural network typically consists of three main components: an input layer, multiple hidden layers, and an output layer. The input layer receives raw data such as images, audio signals, or text. Hidden layers perform mathematical transformations on the data using weighted connections and activation functions.

A] Overview of Deep Neural Networks

Deep Neural Networks are computational models inspired by the structure of the human brain. They consist of interconnected layers of artificial neurons that process data and learn patterns through training. Each layer extracts meaningful features from the input, enabling the network to perform tasks such as classification and prediction with high accuracy. DNNs generate output scores that indicate how strongly the input belongs to aparticular category. These scores must be interpreted properly to support reliable decision-making, which is why probability conversion methods like softmax are essential in neural network architectures.

B] Role of Softmax in Classification

The softmax function is widely used in classification models to convert output scores into probabilities. It ensures that each output value lies between 0 and 1 and that the sum of all probabilities equals one. This property makes softmax particularly useful for multi-class classification problems. By assigning higher probabilities to more likely classes, softmax helps systems make clear and interpretable predictions. As a result, it has become a standard component in many neural network models. A deep neural network is made up of several layers. The first layer is called the input layer, which receives the raw data such as images, text, or sound. The data then passes through multiple hidden layers where different mathematical operations are performed. These hidden layers help the network learn important features from the data. Finally, the output layer produces the final prediction or classification result. Deep neural networks learn by adjusting the strength of the connections between neurons.

During training, the network compares its prediction with the correct answer and calculates the error. Then it updates the connection weights using an algorithm called backpropagation. This process repeats many times until the network can make accurate predictions. Because of their powerful learning ability, deep neural networks are widely used in applications such as speech recognition, medical image analysis, natural language processing, and autonomous vehicles.

However, these networks require large computational resources and consume significant energy. Therefore, researchers are working on improving their efficiency by optimizing important components such as the Softmax function used in the output layer.

C] Limitations of Conventional Softmax

The Softmax function is widely used in deep neural networks, especially in the output layer of classification models. It converts the raw output values of a neural network into probabilities so that the model can determine which class is most likely. Although Softmax is effective for classification tasks, the conventional implementation has several limitations, particularly when it is implemented in hardware or energy-constrained systems. One major limitation is high computational complexity. The Softmax function requires exponential calculations for each input value, followed by a normalization step that involves division by the sum of all exponentials. These mathematical operations are computationally expensive and require significant processing resources.

D] Need for Approximate Softmax Techniques

The Softmax function is commonly used in the final layer of deep neural networks to convert output values into probability distributions. Although it plays an important role in classification tasks, the conventional Softmax computation involves complex mathematical operations such as exponentiation division. These operations require significant computational power and can increase the overall processing time of the neural network. In modern applications, deep neural networks are often deployed in devices with limited hardware resources, such as mobile phones, embedded systems, and Internet of Things (IoT) devices. In such environments, computational complexity leads to increased energy consumption and slower system performance. Therefore, it becomes necessary to find more efficient ways to implement the Softmax function. Approximate Softmax techniques are introduced to address this problem by simplifying the mathematical calculations involved in the Softmax operation.

IV) System Analysis Introduction

System analysis is an important stage in the development of any project, as it helps in understanding the requirements, limitations, and overall functioning of the system. In this project, system analysis focuses on studying the traditional Softmax computation used in deep neural networks and identifying the challenges associated with its implementation. The goal of this analysis is to determine how the system can be improved to achieve better energy efficiency and reduced computational complexity. The conventional Softmax function is widely used in classification tasks because it converts the output values of a neural network into probability distributions. However, implementing the exact Softmax function requires complex mathematical operations such as exponential calculations and normalization using division.

During system analysis, the performance of the existing Softmax architecture is carefully examined. Factors such as power consumption, hardware complexity, processing speed, and memory usage are evaluated. This helps in identifying the major limitations of the current approach, particularly when the system is implemented in hardware platforms or embedded devices with limited resources. The analysis also considers the requirements for designing a more efficient system. The proposed solution should reduce computational complexity while maintaining acceptable accuracy in classification results. Approximation techniques are therefore explored as a potential approach to simplify the Softmax computation.

A] Limitations of Conventional Softmax Computation

The conventional Softmax function requires exponential calculations for each input value. Exponential operations are mathematically complex and require more processing time compared to basic arithmetic operations such as addition or multiplication. As the number of output classes increases, the amount of computation also increases significantly. Division operations are relatively slow and consume more hardware resources, which further increases system complexity.

B] Existing System

The Softmax function is implemented using the conventional mathematical formulation in deep neural networks. This function is mainly used in the final layer of classification models to transform the output scores into probability values. Each output value is processed using an exponential function and then normalized by dividing it by the sum of all exponential values. This ensures that the final outputs represent a valid probability distribution where the total sum equals one. Although the traditional Softmax implementation provides accurate probability results, it requires complex mathematical operations such as exponentiation and division. These operations are computationally intensive and demand significant processing resources. When deep neural networks contain a large number of output classes, the number of calculations required by the Softmax function increases considerably.

C] Approximate Softmax Logic

Approximate Softmax logic is a technique used to simplify the computation of the Softmax function in deep neural networks. The traditional Softmax operation involves complex mathematical functions such as exponentiation and division, which require significant computational resources. These operations increase hardware complexity, power consumption, and processing time, especially when the neural network has a large number of output classes. To overcome these challenges, approximate Softmax logic is introduced to reduce the complexity of these calculations. These approximations require fewer arithmetic operations and therefore reduce the computational workload of the system.

V) METHODOLOGY

Introduction

The methodology describes the systematic approach used to design and implement the approximate Softmax architecture for improving energy efficiency in deep neural networks. The main objective of the methodology is reduce the

conventional Softmax function while maintaining acceptable accuracy in classification tasks. The first step in the methodology is to study the working principle of the traditional Softmax function used in neural networks. This includes analyzing the mathematical operations involved in converting the network output values into probability distributions. The conventional Softmax function requires exponential calculations and normalization through division, which increases computational cost and power consumption. After understanding the existing system, the next step is to identify the parts of the Softmax computation that contribute most to complexity. In particular, the exponential function and the normalization process require significant hardware resources. These operations are therefore targeted for optimization using approximation techniques.

The proposed method introduces simplified mathematical models to replace the exact exponential calculations. Instead of computing precise exponential values, the system uses approximation techniques that generate results close to the original values with fewer arithmetic operations. This helps reduce the overall computational load of the system. In addition, the normalization step is optimized to minimize the number of division operations required. Efficient accumulation and scaling techniques are applied to produce probability values while reducing processing time and hardware usage.

VI) RESULT AND DISCUSSION

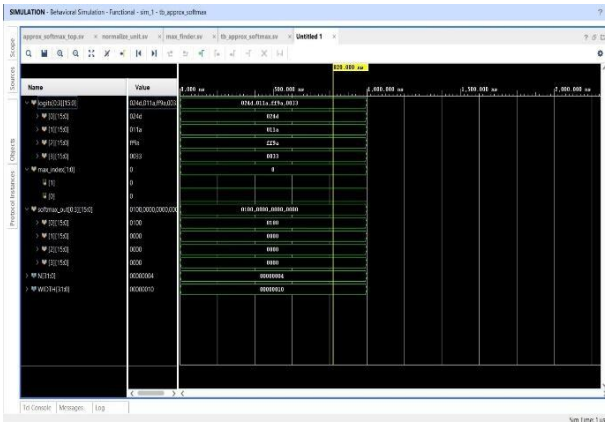
The proposed approximate Softmax architecture was implemented and evaluated through simulation in order to analyze its functional correctness and efficiency. The main objective of this evaluation was to determine whether the simplified Softmax computation could provide reliable classification results while reducing computational complexity and hardware requirements. During the simulation process, several input vectors were applied to the system to observe how the architecture processes the data and produces output probabilities. The simulation waveforms indicate that the proposed system successfully performs the required operations, including input comparison, maximum value detection, and probability estimation. The output signals generated by the system follow the expected pattern, confirming that the architecture functions correctly under different test conditions. One of the key aspects analyzed in this work is the computational efficiency of the proposed approach. In conventional Softmax implementations, the calculation of exponential values and normalization through division requires a significant number of arithmetic operations.

These operations increase the computational load and processing time. In contrast, the proposed approximate Softmax architecture replaces these complex calculations with simplified logic-based operations. As a result, the number of required arithmetic computations is greatly reduced. Another important factor considered in the analysis is hardware resource utilization. Since the proposed design eliminates the need for expensive exponential units and large division circuits, the overall hardware complexity is reduced. This leads to a smaller circuit area when implemented on digital hardware platforms such as FPGA or ASIC devices. Reduced hardware complexity also contributes to improved system efficiency and easier integration with neural network accelerators. The power efficiency of the proposed architecture is also an important outcome of this work. By minimizing complex arithmetic operations, the system consumes less energy during computation. This improvement makes the architecture particularly suitable for embedded systems and edge computing devices where power availability is limited. Energy- efficient neural network components are essential for applications such as mobile devices, IoT systems, and real-time intelligent systems.

In addition to efficiency improvements, the accuracy of the output results was also examined. Although the proposed system uses approximation techniques instead of exact exponential calculations, the final classification decision remains consistent with the expected behavior of the Softmax function. The system is able to correctly identify the most probable output class from the given input values. This demonstrates that the approximation method does not significantly affect the overall performance of the neural network inference process. The simulation results also highlight the speed advantage of the approximate approach. Since the architecture relies on simpler arithmetic and logical operations, the computation time is reduced compared to the traditional Softmax implementation. Faster

computation enables quicker inference, which is important for real-time applications such as object detection, speech recognition, and autonomous systems. Overall, the experimental evaluation confirms that the proposed approximate Softmax architecture provides an effective balance between computational efficiency and functional performance. The architecture reduces hardware complexity, power consumption, and processing time while still maintaining reliable classification behavior. These characteristics make the proposed system a promising solution for implementing deep neural networks in energy-constrained environments.

OUTPUT RESULT



VII) CONCLUSION

The main objective of this research was to reduce the computational complexity and energy consumption associated with the traditional Softmax computation. Conventional Softmax implementations require complex mathematical operations such as exponentiation and division, which increase processing time, hardware resource usage, and power consumption. These challenges become more significant when deep neural networks are deployed in embedded systems, mobile devices, and edge computing platforms where energy efficiency is an important requirement. To address these limitations, an approximate Softmax architecture was proposed. The proposed method simplifies the computation process by replacing complex exponential operations with approximation techniques and simplified logical operations. By reducing the number of arithmetic calculations required during the inference stage, the system achieves improved computational efficiency while maintaining reliable classification performance. The design and functional behavior of the proposed architecture were analyzed through simulation. The simulation results demonstrate that the system operates correctly and is able to identify the most probable output class from the given inputs. The results also show that the approximate Softmax logic significantly reduces the computational load compared to the conventional Softmax implementation. This reduction in complexity leads to faster processing and lower energy consumption.

Another important advantage of the proposed architecture is the reduction in hardware resource utilization. Since the design avoids complex exponential and division units, the overall hardware structure becomes simpler and more suitable for implementation on digital platforms such as FPGA or ASIC. This makes the architecture highly suitable for integration into modern neural network accelerators and low-power artificial intelligence systems.

VIII) ACKNOWLEDGMENT

The authors, Mr. Athul krishna R, Mr. Deepak S J, and Mr. Fayas Mohamed A would like to express their sincere gratitude to the institution for providing the necessary support and resources to carry out this research work successfully. The guidance, encouragement, and technical assistance offered by the Department of Electronics and Communication Engineering played a vital role in the completion of this project.

The authors also acknowledge the valuable laboratory facilities, infrastructure support, and academic environment provided by the institution, which enabled effective design, implementation, and testing of the proposed Approximate Softmax Architecture for Energy- Efficient Deep Neural Networks.

IX) REFERENCES

1. T. Chen, Z. Du, N. Sun, J. Wang, C. Wu, Y. Chen and O. Temam, "DianNao: A Small- Footprint High-Throughput Accelerator for Ubiquitous Machine-Learning," *ACM SIGARCH Computer Architecture News*, vol. 42, no. 1, pp. 269–284, 2014.
2. Y. LeCun, Y. Bengio and G. Hinton, "Deep Learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
3. M. Rastegari, V. Ordonez, J. Redmon and A. Farhadi, "XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks," *European Conference on Computer Vision*, vol. 9908, no. 1, pp. 525–542, 2016.
4. S. Mittal, "A Survey of Techniques for Improving Energy Efficiency in Deep Neural Networks," *International Journal of Computer Vision and Image Processing*, vol. 6, no. 4, pp. 1–21, 2016.
5. V. Sze, Y. H. Chen, T. J. Yang and J. S. Emer, "Efficient Processing of Deep Neural Networks: A Tutorial and Survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295–2329, 2017.
6. N. P. Jouppi et al., "In-Datacenter Performance Analysis of a Tensor Processing Unit," *IEEE Micro*, vol. 38, no. 2, pp. 10–19,2018.
7. V. Shatravin, D. Shashev and S. Shidlovskiy, "Implementation of the Softmax Activation for Reconfigurable Neural Network Hardware Accelerators," *Applied Sciences*, vol. 13, no. 23, pp. 12784–12795, 2023.
8. R. Kim, D. Lee, J. Kim, J. Park and S. E. Lee, "Hardware Accelerator for Approximation- Based Softmax and Layer Normalization in Transformers," *Electronics*, vol. 14, no. 12, pp. 2337–2347, 2025.
9. S. Hirayae, K. Yoshioka, Y. Tanaka and H. Tamuko, "Hardware-Oriented and Precisely Approximated Online Softmax for Deep Learning Models," *IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, vol. 2025, no. 1, pp. 1–6, 2025.