K-Nearest Neighbour Algorithm’s Implementation on Verilog and FPGA

Muhammad Hisham Bin Nauman, Farzad Khan

Computer Engineering, NUST CEME

FPGA Hackathon

K-Nearest Neighbour Algorithm’s Implementation on Verilog and FPGA

This project presents a hardware-accelerated implementation of the **K-Nearest Neighbours (KNN)** classification algorithm using **Verilog HDL** on a **Xilinx NEXYS 3 FPGA**. The design supports **K = 5** with a training dataset of **128 data points**, each characterised by a fixed number of features. The system efficiently classifies input vectors by computing **Euclidean distances**, maintaining a dynamic list of the top five nearest neighbours, and determining the output class through **majority voting**.

To address performance constraints, the architecture incorporates a **finite state machine (FSM)** optimised through pipelining, which significantly reduces the execution time from **385 to 145 clock cycles**.

The design was validated through extensive simulation using **ModelSim** and successfully deployed on the **NEXYS 3 FPGA**. The implementation highlights the power of custom hardware for machine learning algorithms, offering low-latency classification suitable for real-time embedded applications.

## Introduction

The K-Nearest Neighbours (KNN) algorithm is a widely used non-parametric method in machine learning for classification and regression tasks. However, its computational intensity, especially during the distance computation and sorting phases, makes it less suitable for real-time applications when implemented in software. FPGAs offer a parallel and deterministic hardware platform that can address these performance challenges. This project aims to design and implement a hardware-accelerated KNN classifier using Verilog HDL on a NEXYS 3 FPGA.

## Background and Motivation

The need for real-time machine learning inference in embedded systems has driven interest in hardware acceleration. Traditional implementations of KNN are software-based and slow, particularly for large datasets. Hardware implementation on an FPGA provides benefits like low latency, parallelism, and reconfigurability. The project was developed as part of an FPGA hackathon and serves as a benchmark for applying algorithmic optimisations and hardware design principles.

## Methodology

### Dataset Format and Assumptions

Each data point in the training set consists of a fixed number of features (e.g., 4 features), stored in internal FPGA memory. The input vector is provided via FPGA switches and buttons.

### Design Architecture

The hardware design includes the following modules:

#### Distance Calculator

Computes Euclidean distance using subtractors, squares, and adders.

#### K-Nearest Selector:

Maintains a sorted list of the 5 smallest distances.

#### Majority Voter:

Outputs the class with the highest frequency among the nearest neighbours.

#### FSM Controller:

Manages data flow through states: RESET, LOAD\_COMPUTE, FINAL\_STORE, and CALC\_MAJ.

### FSM Optimisation

Initial simulation showed 385 clock cycles per classification. By pipelining the READ and COMPUTE states and overlapping the UPDATE and CLASSIFY stages, we reduced this to 145 clock cycles.

## Implementation

The design was written in Verilog and simulated in ModelSim. Synthesizable modules were tested on the NEXYS 3 board using Xilinx ISE.

## Results

* Initial Clock Cycles: 385
* Optimised Clock Cycles: 145
* Dataset: 128 training points, K = 5
* FPGA Board: NEXYS 3
* Toolchain: Xilinx ISE 14.5, ModelSim

Simulation and on-board tests verified functional correctness and performance improvement. Inputs were correctly interpreted, and the system responded with correct classifications.

## Discussion

Implementing KNN on an FPGA demonstrated substantial speed improvements over software-based approaches. The optimisation of FSM and pipelining was crucial in reducing latency.

## Conclusion

This project highlights the viability of implementing machine learning algorithms like KNN on FPGA platforms. The successful deployment of a pipelined, KNN classifier showcases the potential of FPGAs in real-time AI applications, particularly where power efficiency and low latency are critical.

## Future Work

Future improvements could include:

* Support for floating-point data using DSP blocks
* Dynamic training set updates via UART or memory-mapped interface
* Integration with other communication protocols like SPI or I2C

**References**

Xilinx Inc. (2012). *NEXYS 3 FPGA Board User Guide*. Retrieved from

<https://digilent.com>

Patterson, D. A., & Hennessy, J. L. (2017). *Computer organization and design RISC-V*

*edition: The hardware/software interface* (5th ed.). Morgan Kaufmann Publishers.

ISBN: 978-0-12-407726-3