**Project Report**

**Real-Time Stream Data Processing using Hyperdimensional Computing and HLS**

Saeid Jamili, Marco Angioli

**Supervisor:** Professor Antonio Mastrandrea

**Introduction**

As Artificial Intelligence (AI) and Internet of Things (IoT) become increasingly prevalent, there is a pressing need for systems that can handle and process vast amounts of data efficiently in real-time. Despite the substantial processing power of cloud computing, it is frequently constrained by bandwidth and latency, particularly when dealing with critical tasks such as automated car driving or health care devices. Therefore, edge computing, which allows for data processing closer to the source, is rapidly gaining popularity. Our project combines the potency of Hyperdimensional Computing (HDC) with High-Level Synthesis (HLS) to form an effective system for real-time stream data processing at the edge.

Hyperdimensional Computing Hyperdimensional Computing, a novel computing approach, leverages high-dimensional vectors, known as hypervectors, for information processing. Drawing inspiration from the neuronal activity patterns observed in the human brain, HDC provides robustness and parallelism. The use of hypervectors allows for efficient representation and processing of complex data, making HDC particularly suited to real-time tasks.

**High-Level Synthesis (HLS) and Its Advantages**

High-Level Synthesis is a transformative technology that automates the process of translating high-level programming language descriptions into hardware description language. This process offers numerous advantages in terms of time and resource efficiency. HLS plays a pivotal role in our project, facilitating the seamless translation of high-dimensional computations into FPGA implementations, thereby significantly reducing complexity and development time associated with traditional hardware design processes. Additionally, HLS enables automatic generation of hardware accelerators during compile time, providing bespoke, application-specific optimizations that further enhance the performance and efficiency of the system.

**Hypervectors and Hardware Friendliness**

Hypervectors, owing to their intrinsic properties, are remarkably compatible with hardware designs, especially FPGAs. The attributes of hypervectors – high-dimensionality, inherent integral arithmetic, and the capability for bitwise operations – harmonize perfectly with the functional strengths of hardware design. This harmonization facilitates efficient utilization of hardware resources, thus making hypervectors an excellent choice for FPGA-based systems.

**System Architecture and Performance**

The architecture of our system, as illustrated in Figure 1, begins with a dataset and training code for the hypervectors. We've chosen a hypervector dimension of 1024 and a simple trivial data type for balance between computational demands and classification performance. Upon training, parameters are extracted for use in the inference part of our system, which is implemented on an FPGA using HLS.

A picture containing text, screenshot, diagram, design

Description automatically generated

Figure 1 Architecture of the System Designed for Classification on FPGA

Our system processes streaming input data features in real-time, generating output once all data features have been processed. The extracted data includes base vectors, level vectors, and class high dimensional vectors. Each incoming feature is quantized and encoded using the base and level vectors. The resultant high-dimensional vector is then compared with all class vectors using a similarity function, facilitating real-time classification of the input data.

**Results**

In a practical evaluation using the ISOLET dataset, our system was tested on over 1500 inputs. The system reached an inference speed of 150 classifications per second, operating with a 50MHz clock, thus showcasing its real-time processing abilities. Figure 2 demonstrates the HDC IP, generated by HLS, implemented in simple hardware, and connected to the PS for testing. Figure 3 depicts the simulation performed using a C++ testbench, and Figure 4 displays the co-simulation for waveform examination. The resource usage of the system is summarized in Table 1, and the results regarding speed and accuracy are presented in Table 2.

A picture containing text, diagram, plan, parallel

Description automatically generated

Figure 2 Designed hardware for connecting HDC IP to PS

A screenshot of a computer

Description automatically generated

Figure 3 Simulation with C++ testbench

A screenshot of a computer

Description automatically generated

Figure 4 Co-Simulation with C++ testbench

Table 2 Resource usage

|  |  |
| --- | --- |
| Resource | Usage |
| DSP | 7 |
| LUT | 4076 |
| FF | 3523 |
| BRAM | 48 |

Table 3 speed and accuracy

|  |  |
| --- | --- |
| Parameter | ISOLET Dataset |
| Features | 617 |
| Rows | 7797 |
| Accuracy | 0.71 |
| Speed (clock 50MHz) | 150 classifications/sec |

**Novelty**

Our project introduces an innovative blend of Hyperdimensional Computing (HDC) and High-Level Synthesis (HLS) for the efficient processing of streaming data. HDC brings to the table its ability to manage large volumes of data with remarkable speed and reliability. Concurrently, HLS simplifies the task of hardware design needed for our operations. This effective combination doesn't only equip us to handle streaming data in real-time but also paves the way for potential advancements in edge network devices, such as IoT devices.

**Conclusion and next steps**

This project demonstrates the potent synergy of Hyperdimensional Computing and High-Level Synthesis in tackling real-time stream data processing tasks. Our tests affirm the robustness, high performance, and real-world readiness of our system. The application of HLS further validates its promise as a paradigm for future hardware design, encouraging innovation and efficiency.

This endeavor underscores the potential of HLS and HDC to meet the demands of IoT devices and edge computing. Looking forward, we aim to enhance the system further to increase its efficiency. Specifically, our focus will be on optimizing resource usage, reducing associated memory requirements, and enhancing the system's speed, thereby ensuring our system continues to evolve and improve.