Performance Analysis of SIMD Vectorization for Neural Networks and Clustering Algorithms

Jaka Škerjanc, Erik Pahor

I. Introduction

This report analyzes SIMD vectorization performance for two machine learning algorithms: Multilayer Perceptron (MLP) for MNIST digit recognition and K-means clustering. Both implementations use AVX2 instructions and compare single vs. double precision floating-point performance.

II. Implementation Approach

AVX2 provides 256-bit wide registers that can process 8 single-precision or 4 double-precision values simultaneously. Our implementation uses key instructions including `_mm256_loadu_ps/pd`, `_mm256_set1_ps/pd`, `_mm256_add_ps/pd`, and `_mm256_mul_ps/pd`.

For the MLP, we vectorized the forward propagation function, focusing on input-to-hidden layer calculations and the sigmoid activation function. For K-means, we accelerated the Euclidean distance calculations between points and centroids. Both implementations handle edge cases where vector width doesn't align with data size.

III. Performance Results

A. MLP Vectorization

Tests with different hidden layer sizes (128-1024 neurons) showed consistent speedups:

Table I
MLP Performance Results

Hidden Layer Size	Double Precision	Float Precision
128	1.40x	1.71x
256	1.44x	1.86x
512	1.22x	1.78x
1024	2.38x	1.96x

Float precision consistently outperformed double precision, with speedups reaching 1.96x for float and 2.38x for double. Performance improved with larger layer sizes, particularly at 1024 neurons.

B. K-means Vectorization

Tests with different cluster counts (4, 8, 16) showed mixed results:

Table II
K-means Performance Results

Precision	Clusters	Speedup
double	4	0.96x
double	8	0.97x
double	16	0.96x
float	4	1.12x
float	8	1.15x
float	16	1.14x

Float precision showed consistent speedups (1.12x-1.15x), while double precision unexpectedly resulted in slight slowdowns (0.96x-0.97x). The number of clusters had minimal impact on relative performance.

IV. Analysis

The performance gains from vectorization can be attributed to:

- Parallel processing of multiple data elements
- Reduced loop overhead and memory access optimization
- Efficient use of CPU's SIMD execution units

The theoretical maximum speedup (8x for float, 4x for double) wasn't achieved due to:

- Memory access patterns and cache behavior limitations
- Overhead from non-vectorized code sections
- Data dependencies limiting parallelism

For K-means, double precision vectorization failed to deliver speedups likely due to:

- Memory access overhead outweighing computational benefits
- Less efficient cache utilization with larger data types
- Lower parallelism (4 doubles vs. 8 floats per vector)

V. Conclusion

SIMD vectorization effectiveness varies by algorithm and precision:

- MLP showed significant speedups in both precisions (1.22x-2.38x)
- K-means benefited only in float precision (1.12x-1.15x)
- Float precision generally outperformed double precision due to higher parallelism
- Larger problem sizes typically showed better vectorization benefits

These results demonstrate that SIMD vectorization is most effective when algorithms feature intensive floating-point calculations with optimizable memory access patterns, especially when single-precision accuracy is sufficient. Future work could explore more sophisticated vectorization techniques, such as loop unrolling and data layout transformations.