Вісник Тернопільського національного технічного університету
https://doi.org/10.33108/visnyk_tntu
Scientific Journal of the Ternopil National Technical University
2024, № 2 (114) https://doi.org/10.33108/visnyk_tntu2024.02
ISSN 2522-4433. Web: ***visnyk.tntu.edu.ua***

UDC 004.421:004.934.1’1

ACCELERATING DYNAMIC TIME WARPING FOR SPEECH RECOGNITION WITH SSE

Yurii Vash; Mariana Rol; Mykola Chyzhmar

Uzhhorod National University, Uzhhorod, Ukraine

Summary. *This study presents a significant enhancement to the Dynamic Time Warping (DTW) algorithm for real-time applications like speech recognition. Through integration of SIMD (Single Instruction Multiple Data) instructions to distance function, the research demonstrates how SSE accelerates DTW, markedly reducing computation time. The paper not only explores the theoretical aspects of DTW and this optimization but also provides empirical evidence of its effectiveness. Diverse dataset of 18 voice command classes was assembled, recorded in controlled settings to ensure audio quality. The audio signal of each speech sample was segmented into frames for detailed analysis of temporal dynamics. DTW search was performed on features set based on Mel Frequency Cepstral Coefficients (MFCC) and Linear Predictive Coding (LPC), combined with delta features. A comprehensive set of 27 features was extracted from each frame to capture critical speech characteristics. The core of the study involved applying traditional DTW as a baseline for performance comparison with the SSE-optimized DTW. The evaluation, focusing on computational time, included measurements like minimum, maximum, average, and total computation times for both standard and SSE-optimized implementations. Experimental results, conducted on datasets ranging from 5 to 60 WAV files per class, revealed that the SSE-optimized DTW significantly outperformed the standard implementation across all dataset sizes. Particularly noteworthy was the consistent speed of the SSE-optimized Manhattan and Euclidean distance functions, which is crucial for real-time applications. The SSE-optimized DTW maintained a low average time, demonstrating remarkable stability and efficiency, especially with larger datasets. The study illustrates the potential of SSE optimizations in speech recognition, emphasizing the SSE-optimized DTW's capability to efficiently process large datasets.*

Key words: *Dynamic Time Warping, Speech Recognition, Euclidean Distance, Manhattan Distance*.

https://doi.org/10.33108/visnyk_tntu2024.02.030 *Received 31.01.2024*

Problem Statement. Speech recognition systems are pivotal in various applications, necessitating real-time processing and low-latency responses. Among the various algorithms and techniques employed for this purpose, Dynamic Time Warping (DTW) has established itself as a robust method for aligning and comparing temporal sequences. DTW achieves a high accuracy in speech recognition in quiet environments that is not only comparable but often exceeds that of current machine learning techniques [1].

DTW is a powerful algorithm renowned for its ability to accurately align temporal sequences despite variations in length, speed, or timing. It used in various applications and areas. For instance, in finance, DTW helps analysts uncover hidden patterns in market trends through time-series data analysis [2]. In healthcare, DTW demonstrates its value by addressing the challenge of modeling patient data when availability is limited [3]. In the area of speech recognition, DTW shows unmatched ability to compare spoken words to reference models facilitates accurate recognition even when speech patterns do not align linearly. This capability is indispensable in voice-activated systems where diverse speaking speeds and pronunciations are the norm.

However, the computationally intensive nature of traditional DTW presents significant challenges, especially when real-time processing is required. This study introduces an optimized approach to DTW, utilizing Streaming SIMD Extensions (SSE) to substantially improve the computational efficiency of the algorithm. SSE, a form of Single Instruction

 Corresponding author: Yurii Vash; e-mail: yurii.vash@uzhnu.edu.ua

Multiple Data (SIMD) operations, enables simultaneous processing of multiple data points, thereby accelerating the core calculations within the DTW algorithm. By integrating SSE, this research aims to reduce the computational load of DTW, enhancing the algorithm's applicability in real-time speech recognition systems.

Analysis of the known results of the research. DTW relies on a distance o measure the similarity between two points in the sequences being compared. The most common metric is Euclidean distance, which provides a straightforward geometrical distance between two points [4]. However, DTW is not limited to this and can employ other distances like Manhattan, Chebyshev, or Mahalanobis distances, each offering different advantages depending on the specific characteristics of the data. Both the Euclidean and Manhattan distances, are particular instances of the Minkowski distance [5]. These distance metrics were selected to assess the performance of the SSE-optimized DTW algorithm.

The Euclidean distance is the most commonly used distance measure for time series analysis due to its adherence to metric properties, including non-negativity, symmetry, and the triangle inequality, which can help speed up search operations [5]. Euclidean distance is defined as:

$$d_{Euclidean}(x, y) = \sqrt{\sum_{k=1}^{n} (x_k - y_k)^2}$$

where x and y are two points in n-dimensional space, and x_k and y_k are the k-th elements of these points.

The Manhattan distance, differs from the Euclidean distance in that it calculates the sum of the absolute differences along each dimension, effectively moving horizontally and vertically, rather than diagonally. This makes it particularly suitable for grid-based systems where diagonal movement is not permissible [6]. Manhattan distance is defined as:

$$d_{Manhattan}(x, y) = \sum_{k=1}^{n} |x_k - y_k|$$

where x and y are two points in n-dimensional space, and x_k and y_k are the k-th elements of these points.

The essence of DTW is to construct a cost matrix D where $D(i, j)$ reflects the distance between points x_i from sequence X and y_j from sequence Y. The DTW algorithm then seeks the path through this matrix that minimizes the total cumulative distance, which can be represented mathematically as:

$$D(i, j) = d(x_i, y_j) + \min\{D(i-1, j-1), D(i-1, j), D(i, j-1)\}$$

The path corresponds to the alignment between the sequences, and the value of $D(N, M)$ gives the DTW distance between the sequences.

To enhance the performance of DTW, SSE can be utilized, particularly for the computation of distances. SSE instructions process multiple data points simultaneously, offering significant improvements in computational efficiency. The SIMD architecture is a cornerstone in the design of modern processors, enabling multiple data points to be processed in parallel under a single instruction, thereby optimizing performance through efficient data parallelism [7].

The SSE-optimized Euclidean distance between two vectors can be computed using the following formula, adapted to operate on four floating points at a time with SSE:

$$d_{SSE\_Euclidean}(a,b) = \sqrt{\sum_{i=1}^{n/4}(va_i - vb_i)^2}$$

Here, va_i and vb_i are the 4-length segments of vectors a and b loaded into SSE registers. The implementation involves parallel subtraction, squaring, and summing operations, executed through SSE intrinsic functions.

The SSE-optimized Manhattan distance between two vectors can be computed with the following formula, adapted for parallel processing on four floating points at a time with SSE:

$$d_{SSE\_Manhattan}(x,y) = \sum_{i=1}^{n/4}|va_i - vb_i|$$

In this formulation, va_i and vb_i represent the 4-length segments of vectors a and b that have been loaded into SSE registers. The computation is performed using parallel subtraction to find the differences, a bitwise operation to obtain the absolute values, and subsequent summation of these absolute values, all facilitated by SSE intrinsic functions.

Incorporating this SSE-based computation into the DTW algorithm results in the following optimized formula for computing the cost matrix. For simplicity, we will use the generic distance function d_{SSE}. First, let's define the recurrence relation for the cost matrix D for DTW:

$$D_{SSE}(i,j) = d_{SSE}\left(sample_i, reference_j\right) + \min(D1, D2, D3),$$

where

$$D1 = D_{SSE}(i-1, j-1), \text{ (the cost of the diagonal move)},$$
$$D2 = D_{SSE}(i-1) + penalty, \text{ (the cost of the horizontal move)},$$
$$D3 = D_{SSE}(i, j-1) + penalty, \text{ (the cost of the vertical move)}$$

The horizontal move $D2$ and the vertical move $D3$ represent a misalignment where one element from one sequence is aligned with multiple elements from the other sequence. To discourage this and to reflect the additional cost of such misalignments, a penalty is added. This penalty is a predefined value that increases the cost of the path if it includes many such horizontal or vertical moves, which corresponds to a less optimal alignment between the two sequences. The penalty in this DTW implementation serves to guide the path towards the best alignment by adding a cost to less desirable steps (horizontal and vertical) that do not directly match elements from the two sequences being compared.

This implementation effectively accelerates the distance calculation process, a computationally intensive part of the DTW algorithm, thereby enhancing the performance of the entire speech recognition process. The D_{SSE} function calculates the SSE-optimized distance between two feature vectors $sample_i$ and $reference_j$. The penalty is the additional cost for horizontal or vertical moves in the DTW path, which is typically used to maintain a certain path structure.

DTW cost matrix at position (i, j) is the sum of the SSE-optimized distance for the current position and the minimum of the three possible previous steps in the path, which can be either the previous diagonal, horizontal, or vertical positions, each with its associated cost.

We implemented SSE-optimized version of DTW in C++. The C++ standard library does not directly expose SSE operations; however, these can be accessed through specific compiler intrinsics [8] provided by headers such as <xmmintrin.h> for SSE instructions. The intrinsic types and functions allow direct manipulation of data in 128-bit XMM registers, enabling the parallel processing of floats. Among these primitives is __m128, a data type integral to SSE operations, which represents a vector of four single-precision floating-point values.

The Euclidean distance calculation is a prime example of an operation that benefits significantly from SSE optimization. In our implementation (Listing 1), we use __m128 to load vectors of four floats from the sequences being compared, employing _mm_loadu_ps. We then perform element-wise subtraction with _mm_sub_ps, square the result with _mm_mul_ps, and accumulate the squares with _mm_add_ps. The horizontal addition operation _mm_hadd_ps is a key step, summing values across the SIMD register pairs, and is applied twice to aggregate all four values into one. Finally, the square root of the sum yields the Euclidean distance, providing a measure of similarity between feature vectors.

Listing 1

Euclidean Distance Calculation with SSE Optimization

```
float EuclidianDistanceSSE(float* a, float* b, int size) {
   __m128 sum = _mm_setzero_ps();
   for (int i = 0; i < size; i += 4) {
      __m128 va = _mm_loadu_ps(a + i);
      __m128 vb = _mm_loadu_ps(b + i);
      __m128 diff = _mm_sub_ps(va, vb);
      __m128 sq_diff = _mm_mul_ps(diff, diff);
      sum = _mm_add_ps(sum, sq_diff);
   }

   sum = _mm_hadd_ps(sum, sum);
   sum = _mm_hadd_ps(sum, sum);

   // Store the result back to the float array
   float dist[4];
   _mm_store_ps(dist, sum);

   return sqrtf(dist[0]);
}
```

Manhattan distance calculations can also be optimized using SSE (Listing 2). The process involves loading vectors, computing the absolute value of differences using a bitwise ANDNOT operation with a sign bit mask (_mm_andnot_ps), and summing these absolute values. The horizontal addition is again utilized to sum the vector elements. The result is a more efficient computation of the Manhattan distance, which serves as another metric for DTW path costs.

Listing 2

Manhattan Distance Calculation with SSE Optimization

```
float ManhattanDistanceSSE(float* a, float* b, int size) {
    __m128 sum = _mm_setzero_ps();
    __m128 sign_mask = _mm_set1_ps(-0.f); // -0.f = 1 << 31

    for (int i = 0; i < size; i += 4) {
        __m128 va = _mm_loadu_ps(&a[i]);
        __m128 vb = _mm_loadu_ps(&b[i]);
        __m128 diff = _mm_sub_ps(va, vb);
        __m128 abs_diff = _mm_andnot_ps(sign_mask, diff);

        sum = _mm_add_ps(sum, abs_diff);
    }

    sum = _mm_hadd_ps(sum, sum);
    sum = _mm_hadd_ps(sum, sum);

    // Store the result back to the float array
    float dist[4];
    _mm_store_ps(dist, sum);

    return dist[0];
}
```

The provided code snippets (Listing 1 and 2) demonstrate the practical application of SSE in computing distances. Each line of code is meticulously crafted to ensure data parallelism. The Euclidean distance code loads unaligned float vectors, computes the squared difference, and accumulates the sum using SIMD operations. Similarly, the Manhattan distance code employs SSE to compute absolute differences efficiently.

Methodology. The study commenced with the collection of a comprehensive dataset comprising speech samples from 18 distinct classes, with the number of WAV files per class varying from 5 to 60. Each class corresponded to a specific command or phrase commonly used in voice-activated systems. The evaluation was conducted on a set of 10 preloaded WAV files to assess the efficacy of the DTW in speech recognition tasks. To ensure uniformity in audio quality, all recordings were carried out in a controlled setting, resulting in WAV files standardized at a sample rate of 44,100 Hz and a bit depth of 16 bits.

For each speech sample, the first step was to segment the continuous audio signal into frames, typically spanning 20–70 milliseconds. This segmentation allowed for the analysis of the signal's temporal dynamics, as speech is inherently non-stationary over long durations.

From each frame, a set of 27 features was extracted, encapsulating both spectral and temporal characteristics. This set included 13 Mel Frequency Cepstral Coefficients (MFCC), which provide a representation of the power spectrum of sound, and 13 Linear Predictive Coding (LPC) coefficients along with different delta coefficients, which model the vocal tract shaping the speech signal and dynamic of signal. These features were selected for their proven effectiveness in capturing the essential characteristics of speech relevant to recognition tasks.

The core of the methodology was the application of the DTW algorithm to align and compare the feature sequences extracted from the speech samples. Traditional DTW was first applied to establish a baseline for performance evaluation. This involved computing the cost matrix by calculating distances between all possible pairs of frames from the sample and reference sequences and finding the minimum cumulative distance path.

The optimized DTW algorithm was applied to the same dataset, and its performance was measured in terms of computational time. The measurements included the minimum, maximum, average, and total computation times for both the SSE-optimized and traditional algorithm implementations.

Experimental Results. The Euclidean distance results revealed a consistent pattern (Fig. 1): the SSE-optimized DTW showed a significant reduction in computation time across all dataset sizes compared to the regular DTW. Notably, as the number of files per class increased, the average time for the SSE-optimized DTW remained relatively stable, with minor fluctuations, averaging between 63 to 68 microseconds. In contrast, the regular DTW average time was substantially higher, ranging from 209 to 218 microseconds, showcasing a more pronounced increase with larger datasets.

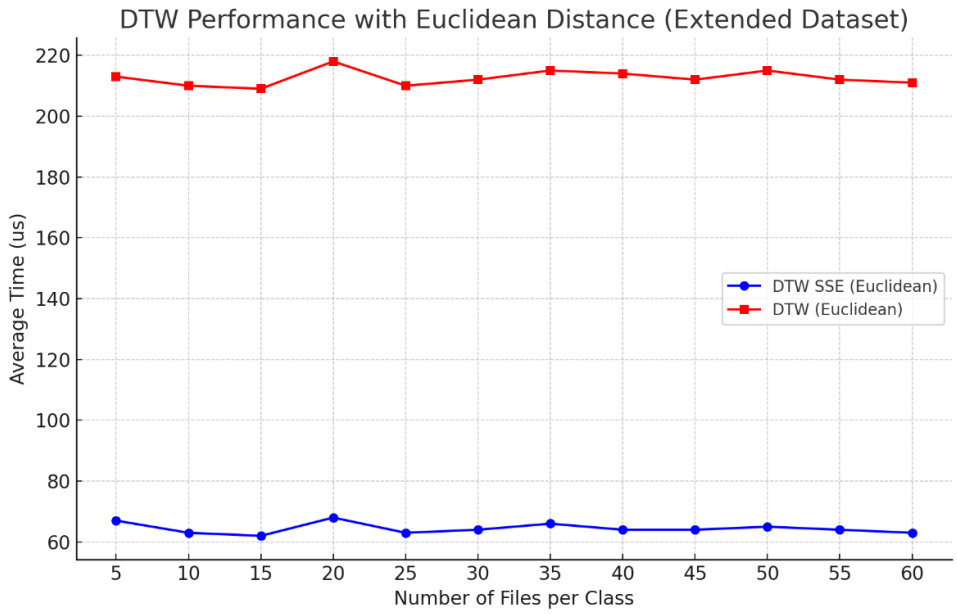

Figure 1. Performance comparison of DTW algorithms using Euclidean distance

For the smallest dataset (5 files per class), the SSE-optimized DTW achieved a minimum processing time of 9 microseconds, a maximum of 772 microseconds, and an average of 67 microseconds over 900 operations. The regular DTW, on the other hand, started at a minimum time of 30 microseconds, reaching a maximum of 1543 microseconds, with an average of 213 microseconds.

As the dataset scaled to 60 files per class, the SSE-optimized DTW impressively maintained a low average time of 63 microseconds, whereas the regular DTW increased slightly to an average of 211 microseconds. This demonstrates the scalability and efficiency of the SSE optimization, particularly in handling larger datasets where computational efficiency becomes increasingly crucial.

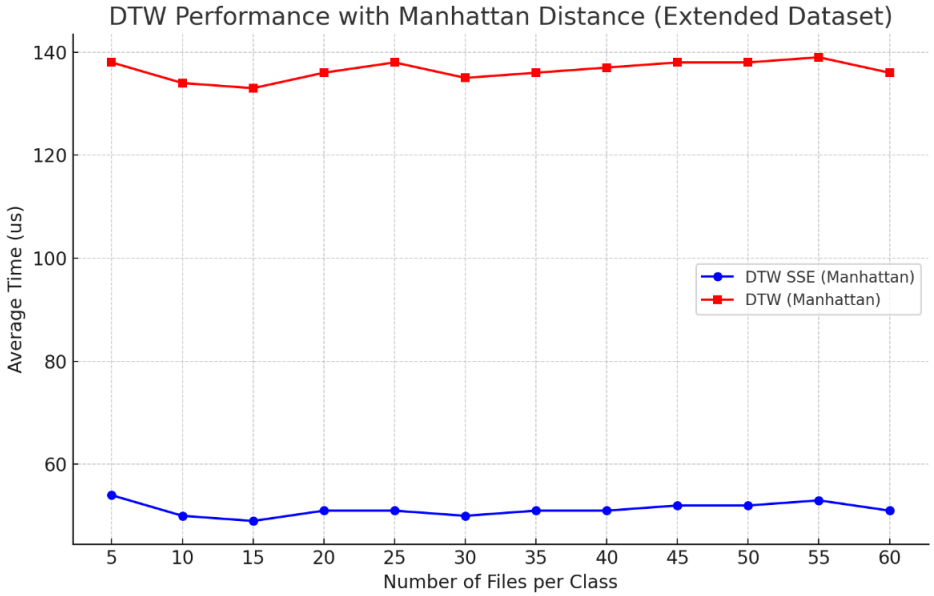

Figure 2. Performance comparison of DTW algorithms using Manhattan distance

The Manhattan distance results (Fig. 2) further corroborated the advantages of the SSE optimization. The SSE-optimized DTW outperformed the regular DTW consistently, maintaining an average computation time that was over four times faster. Starting with an average time of 54 microseconds for the smallest dataset, the SSE-optimized DTW showed remarkable stability, with the average time hovering around 50 to 53 microseconds for the largest dataset of 60 files per class. The regular DTW started at an average time of 138 microseconds and exhibited a slight upward trend, ending at an average of 136 microseconds for the largest dataset.

These results highlight the SSE-optimized DTW's capability to process speech recognition tasks more efficiently, with the Manhattan distance metric results slightly outperforming those of the Euclidean distance in terms of average computation time.

Conclusions. The experimental investigation of the SSE-optimized Dynamic Time Warping algorithm, as detailed in this article, provides a comprehensive understanding of its performance in speech recognition tasks. The results demonstrate the SSE optimization's significant impact on processing efficiency. Across a range of dataset sizes, the SSE-optimized DTW consistently outperformed the traditional DTW algorithm, without compromising the accuracy of the speech recognition.

The consistent speed advantage of the SSE-optimized DTW – evident in both Euclidean and Manhattan distance measurements – highlights the potential of SIMD optimizations in real-time speech processing applications. Notably, the average computation times for SSE-optimized DTW remained stable as the dataset size increased, a testament to the scalability of the optimization. For instance, even when the number of files per class reached 60, the SSE-optimized DTW maintained an average computation time of 63 microseconds for Euclidean distance, illustrating only a minor increase from the 62 microseconds average time observed with 15 files per class. Furthermore, the performance gap between SSE-optimized DTW and the standard implementation grew with the increasing number of files, underscoring the enhanced efficiency of the SSE approach in handling larger datasets.

The findings from this study establish SSE-optimized DTW as a highly effective approach for speech recognition, providing significant computational advantages that are essential for the deployment of responsive voice-controlled systems.

References

1. Jiang, S. and Chen, Z., (2023). Application of dynamic time warping optimization algorithm in speech recognition of machine translation. Heliyon, 9 (11), p. e21625. https://doi.org/10.1016/j.heliyon.2023.e21625
2. D'Urso, Pierpaolo & De Giovanni, Livia & Massari, Riccardo. (2021). Trimmed fuzzy clustering of financial time series based on dynamic time warping. Annals of Operations Research. 299. https://doi.org/10.1007/s10479-019-03284-1
3. Puri, Chetanya & Kooijman, Gerben & Vanrumste, Bart & Luca, Stijn. (2022). Forecasting Time Series in Healthcare With Gaussian Processes and Dynamic Time Warping Based Subset Selection. IEEE journal of biomedical and health informatics. PP. https://doi.org/10.1109/JBHI.2022.3214343
4. Baturinets A. (2022). Distance measures-based information technology for identifying similar data series. Scientific Journal of TNTU (Tern.), vol. 105, no. 1, pp. 128–140. https://doi.org/10.33108/visnyk_tntu2022.01.128
5. Cassisi, Carmelo & Montalto, Placido & Aliotta, Marco & Cannata, Andrea & Pulvirenti, Alfredo. (2012). Similarity Measures and Dimensionality Reduction Techniques for Time Series Data Mining. https://doi.org/10.5772/49941
6. Garg, Nidhi & Bisht, Amandeep. (2016). Comparative Analysis of DTW based Outlier Segregation Algorithms for Wrist Pulse Analysis. Indian Journal of Science and Technology. 9. https://doi.org/10.17485/ijst/2015/v8i1/101746
7. Xie, Chunhu & Wu, Huachun & Zhou, Jian. (2023). Vectorization Programming Based on HR DSP Using SIMD. Electronics. 12. 2922. https://doi.org/10.3390/electronics12132922
8. Intel Corporation, "Intel Intrinsics Guide". Available at: https://www.intel.com/content/www/us/en/docs/intrinsics-guide/index.html.

Список використаних джерел

1. Джианг С. та Чен З. (2023). Застосування алгоритму оптимізації динамічного часового вирівнювання у розпізнаванні мови машинного перекладу. Heliyon. 9 (11). С. e21625. https://doi.org/10.1016/j.heliyon.2023.e21625
2. D'Urso, Pierpaolo & De Giovanni, Livia & Massari, Riccardo. (2021). Trimmed fuzzy clustering of financial time series based on dynamic time warping. Annals of Operations Research. 299. https://doi.org/10.1007/s10479-019-03284-1
3. Пурі Ч., Коойман Г., Ванрумсте Б., Лука С. (2022). Прогнозування часових рядів у медицині з використанням гауссових процесів та відбору підмножин на основі динамічного часового варпінгу. Журнал IEEE з біомедичної та здоров'я інформатики. Том PP. https://doi.org/10.1109/JBHI.2022.3214343
4. Батурінець А. (2022). Інформаційна технологія визначення схожих рядів даних із використанням мір відстаней. Вісник ТНТУ. Том. 105. № 1. С. 128–140. https://doi.org/10.33108/visnyk_tntu2022.01.128
5. Кассісі К., Монтальто П., Аліотта М., Канната А., Пулвіренті А. (2012). Методи вимірювання подібності та техніки зменшення розмірності для майнінгу часових рядів. https://doi.org/10.5772/49941
6. Гарг Н., Бішт А. (2016). Порівняльний аналіз алгоритмів виявлення аномалій на основі DTW для аналізу пульсу на зап'ясті. Індійський журнал науки та технологій. Том 9. https://doi.org/10.17485/ijst/2015/v8i1/101746
7. Сі, Ч., Ву, Х., Чжоу, Ж. (2023). Векторизаційне програмування на базі HR DSP з використанням SIMD. Електроніка. Том 12. С. 2922. https://doi.org/10.3390/electronics12132922
8. Інтел Корпорація. Посібник «Інтел Інтрінсікс». URL: https://www.intel.com/content/www/us/en/docs/intrinsics-guide/index.html.

УДК 004.421:004.934.1'1

ПРИСКОРЕННЯ АЛГОРИТМУ ДИНАМІЧНОЇ ТРАНСФОРМАЦІЇ ЧАСОВОЇ ШКАЛИ ДЛЯ РОЗПІЗНАВАННЯ МОВЛЕННЯ З SSE

Юрій Ваш; Мар'яна Роль; Микола Чижмар

Ужгородський національний університет, Ужгород, Україна

Резюме. *Представлено значне вдосконалення алгоритму динамічної трансформації часової шкали (DTW) для програм реального часу, таких, як розпізнавання мови. Завдяки інтеграції інструкцій*

SIMD (Single Instruction Multiple Data) у функцію дистанції, дослідження демонструє, як SSE прискорює DTW, помітно скорочуючи час обчислення. Досліджено не лише теоретичні аспекти DTW та цієї оптимізації, але й надано емпіричні докази її ефективності. Зібрано різноманітний набір даних із 18 класів голосових команд, записаних у контрольованих умовах для забезпечення якості звуку. Аудіосигнал кожного зразка мовлення був сегментований на кадри для детального аналізу часової динаміки. Пошук DTW проводився на наборі функцій на основі мелчастотно кепстральних коефіцієнтів (MFCC) і лінійного прогнозованого кодування (LPC) у поєднанні з дельта-функціями. З кожного кадру було виділено повний набір із 27 ознак, щоб зафіксувати важливі характеристики мови. Основою дослідження було застосування традиційного DTW як бази для порівняння продуктивності з оптимізованим для SSE DTW. Оцінювання, зосереджено на обчислювальному часі, включало такі вимірювання, як мінімальний, максимальний, середній і загальний час обчислень як для стандартних реалізацій, так і для оптимізованих для SSE. Експериментальні результати, проведені на наборах даних від 5 до 60 файлів WAV на клас, показали, що оптимізований для SSE DTW значно перевершує стандартну реалізацію для всіх розмірів наборів даних. Особливо варто відзначити постійну швидкість оптимізованих для SSE функцій Манхеттена та Евклідової відстані, що має вирішальне значення для програм реального часу. Оптимізований для SSE DTW показує низький середній час, демонструючи чудову стабільність і ефективність, особливо з великими наборами даних. Дослідження ілюструє потенціал оптимізації SSE у розпізнаванні мовлення, підкреслюючи здатність оптимізованого для SSE DTW ефективно опрацьовувати великі набори даних.

Ключові слова: *алгоритм динамічної трансформації часової шкали, розпізнавання мовлення, Евклідова дистанція, Манхеттенська дистанція.*

https://doi.org/10.33108/visnyk_tntu2024.02.030 *Отримано 31.01.2024*