cuTimeWarp: Accelerating Soft Dynamic Time Warping on GPU

Alex Kyllo Afrooz Rahmati

March 8, 2021

Abstract

This report explores techniques for optimizing the computation of Soft Dynamic Time Warping, a differentiable sequence dissimilarity measure, on graphics processing units (GPU), for the purpose of enabling high-performance machine learning on time series datasets.

Introduction

Time series machine learning is a research area with countless useful applications such as recognizing sounds and gestures. Clustering or classifying large time series datasets is challenging partly because of the need to define a measure of dissimilarity between any two given time series. Furthermore, practical applications require finding common structure in time series despite different speeds or phases; for example, a word means the same whether spoken quickly or slowly. Another requirement for machine learning tasks is that the dissimilarity measure must be differentiable so that its gradient can be used as to minimize it as a loss function to find the best fit model. Finally, the measure must be efficient to calculate, because it will be calculated repeatedly many times during model fitting. To this end we will explore GPU acceleration of Soft Dynamic Time Warping (Soft-DTW) [1], a differentiable sequence dissimilarity measure, to enable high performance time series machine learning.

Background

TODO: add an intuitive explanation of what time series data is

There are multiple valid ways to compute a measure of dissimliarity between two time series; the simplest is Euclidean distance, which is the square root of the sum of squared pairwise differences between two time series x and y, each of length n (equation 1).

$$d(x, y) = \sqrt{\sum_{t=1}^{n} (x_t - y_t)^2} \tag{1}$$

The drawback of Euclidean distance in time series applications is that two structurally similar time series will produce a large distance if they are different speeds or out of phase (TODO: Add a figure to illustrate this). Dynamic Time Warping (DTW) was devised in the 1960s as an alternative time series dissimilarity measure to address this shortcoming. (TODO: find a historical citation)

The basic algorithm for DTW is to use Bellman's recursion, a dynamic programming technique, to find the lowest-cost path diagonally across a pairwise distance matrix. The computation cost for this approach is quadratic ($O(mn)$) for time series vectors of length m and n [1]. The formula for the DTW between time series x and y is given by equation 2. (TODO: explain the algorithm in more detail and provide a visualization)

$$DTW(x,y) = min_{\pi} \sqrt{\sum_{(i,j)\in\pi} d(x_i, y_j)^2} \tag{2}$$

Where $d(x_i, y_j)^2$ is the cost function, typically pairwise squared Euclidean distance. The loss function for DTW is not differentiable due to the min operation within the formula; a small change in the input time series may result in zero change in the path cost. However, we can create a differentiable version called Soft-DTW by replacing the min function with a soft-min function (equation 3) [1].

$$\text{soft-min}_{\gamma}(a_1, ..., a_n) = -\gamma log \sum_{i} e^{-a_i/\gamma} \tag{3}$$

Hence, Soft-DTW is parameterized by the smoothing constant gamma, which becomes a tunable hyperparameter in machine learning model training applications.

A common technique in machine learning with Soft-DTW is the computation of barycenters, which are centroids within the space of a set of time series. The differentiability of Soft-DTW allows for barycenter finding via gradient descent, and then new observations can be classified by finding the nearest barycenter. Sequence prediction and generation of is also possible using recurrent neural networks with Soft-DTW as a loss function [1].

(TODO: Explain process of barycenter finding with gradient descent on softdtw loss)

Prior to computing the Soft-DTW dissimilarity between any two time series, each time series should be *z-normalized*, that is, scaled so that its mean is equal to 0 and its standard deviation is equal to 1, to remove the problem of "wandering baselines" or "drift" in the measurements, as illustrated in [2] with an ECG classifier that yields incorrect results on un-normalized data due to drift that "may be caused by patient movement, dirty lead wires/electrodes, loose electrodes, and so on," and which "does not have any medical significance." Z-normalization also tends to make the iterative process of minimizing the cost function through gradient descent or Newtonian methods more efficient because its hyperplane is not disproportionately stretched in any one dimension, so the step size in any direction is the same relative to the scale of that dimension. (TODO: explain this better, find a citation)

Related Work

One important optimization we can apply is the use of Sakoe-Chiba bands, which prune the warping path search space, allowing path finding in subquadratic time. While this technique produces an approximation of the optimal path, in practice it has been shown to improve task performance by preventing pathological alignments where a very small portion of one time series maps onto a large portion of another [3].

Utilizing indexing to construct lower bounds on warping distance is an optimization technique for speeding up nearest neighbor search via early removal of poor candidates [3]. Shen and Chi (2021) proposes an optimization of nearest neighbor search of multivariate time series, leveraging the triangle inequality and quantization-based point clustering to restrict the search [4].

Xiao et al (2013) introduced a prefix computation technique for transforming the diagonal data dependence to improve parallel computation of the cost matrix on GPU [5]. Zhu et al (2018) demonstrates a method of optimizing memory access by taking advantage of the diagonal data dependency to rearrange the matrix so that elements on the same diagonal are stored contiguously [6]. A prior implementation of Soft-DTW on CUDA using PyTorch and Numba is capable of 100x performance improvement over the original Soft-DTW Cython code, but is limited to sequence lengths of 1024 (CUDA max block size) and leaves many opportunities for further CUDA optimizations such as the use of shared memory [7]. A 2015 paper describes a tiling approach called *PeerWave*, which utilizes direct synchronization between neighboring streaming multiprocessors (SMs)

to handle the inter-tile data dependency without atomic operations, locks, or other global barriers, leading to improved load balance and scaling properties [8].

In our project we will focus on this area of opportunity, optimizing matrix structure and memory access patterns to maximize parallelism and minimize memory latency in the computation of the warping path matrix.

Methods

To evaluate various performance optimizations on the Soft-DTW computation, we implemented a C++ and CUDA library called cuTimeWarp, which includes functions for computing the SoftDTW on CPU and GPU.

Given a set of many multivariate time series of the same length and number of variables, we can compute the Soft-DTW distance between every time series and every other time series in the set by computing, in parallel for each pair, a pairwise squared Euclidian matrix, then applying the Soft-DTW calculation on the distance matrix.

Test Data

For performance testing we selected datasets from the UCR Time Series Archive [9] and the UEA Multivariate Time Series Classification Archive [10].

Results

Discussion

Future Work

Our library provides CUDA kernels and C++ wrappers for computing pairwise Soft-DTW dissimilarity measures as well their gradients in parallel. Potential future work includes integrating this library with an optimization library that can iteratively minimize Soft-DTW cost to find barycenters among a set of time series, to assemble a nearest centroid classification system. Another area of potential is writing Python bindings to expose the Soft-DTW loss and gradient functions to deep learning frameworks such as TensorFlow or PyTorch, to enable the use of Soft-DTW loss as a training objective for recurrent neural networks. This will facilitate tasks such as classifying, predicting and generating sounds, gestures, and sensor data under the dynamic time warping geometry.

References

[1] M. Cuturi and M. Blondel, "Soft-DTW: A differentiable loss function for time-series," *arXiv:1703.01541 [stat]*, Feb. 20, 2018. arXiv: 1703.01541. [Online]. Available: http://arxiv.org/abs/1703.01541 (visited on 01/16/2021).

[2] T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, and E. Keogh, "Addressing big data time series: Mining trillions of time series subsequences under dynamic time warping," *ACM Transactions on Knowledge Discovery from Data*, vol. 7, no. 3, 10:1–10:31, Sep. 1, 2013, ISSN: 1556-4681. DOI: 10.1145/2500489. [Online]. Available: http://doi.org/10.1145/2500489 (visited on 03/07/2021).

[3] E. Keogh, "Exact indexing of dynamic time warping," in *Proceedings of the 28th international conference on Very Large Data Bases*, ser. VLDB '02, Hong Kong, China: VLDB Endowment, Aug. 20, 2002, pp. 406–417. (visited on 02/14/2021).

[4] D. Shen and M. Chi, "TC-DTW: Accelerating multivariate dynamic time warping through triangle inequality and point clustering," *arXiv:2101.07731 [cs]*, Jan. 19, 2021. arXiv: 2101.07731. [Online]. Available: http://arxiv.org/abs/2101.07731 (visited on 02/14/2021).

[5] L. Xiao, Y. Zheng, W. Tang, G. Yao, and L. Ruan, "Parallelizing dynamic time warping algorithm using prefix computations on GPU," in *2013 IEEE 10th International Conference on High Performance Computing and Communications 2013 IEEE International Conference on Embedded and Ubiquitous Computing*, Nov. 2013, pp. 294–299. DOI: 10.1109/HPCC.and.EUC.2013.50.

[6] H. Zhu, Z. Gu, H. Zhao, K. Chen, C. Li, and L. He, "Developing a pattern discovery method in time series data and its GPU acceleration," *Big Data Mining and Analytics*, vol. 1, no. 4, pp. 266–283, Dec. 2018, Conference Name: Big Data Mining and Analytics, ISSN: 2096-0654. DOI: 10.26599/BDMA.2018.9020021.

[7] M. Maghoumi, *Maghoumi/pytorch-softdtw-cuda*, original-date: 2020-05-02T23:28:24Z, Jan. 21, 2021. [Online]. Available: https://github.com/Maghoumi/pytorch-softdtw-cuda (visited on 01/23/2021).

[8] M. E. Belviranli, P. Deng, L. N. Bhuyan, R. Gupta, and Q. Zhu, "PeerWave: Exploiting wavefront parallelism on GPUs with peer-SM synchronization," in *Proceedings of the 29th ACM on International Conference on Supercomputing*, Newport Beach California USA: ACM, Jun. 8, 2015, pp. 25–35, ISBN: 978-1-4503-3559-1. DOI: 10.1145/2751205.2751243. [Online]. Available: https://dl.acm.org/doi/10.1145/2751205.2751243 (visited on 03/03/2021).

[9] H. A. Dau, A. Bagnall, K. Kamgar, C.-C. M. Yeh, Y. Zhu, S. Gharghabi, C. A. Ratanamahatana, and E. Keogh, "The UCR time series archive," *arXiv:1810.07758 [cs, stat]*, Sep. 8, 2019. arXiv: 1810.07758. [Online]. Available: http://arxiv.org/abs/1810.07758 (visited on 03/08/2021).

[10] A. Bagnall, H. A. Dau, J. Lines, M. Flynn, J. Large, A. Bostrom, P. Southam, and E. Keogh, "The UEA multivariate time series classification archive, 2018," *arXiv:1811.00075 [cs, stat]*, Oct. 31, 2018. arXiv: 1811.00075. [Online]. Available: http://arxiv.org/abs/1811.00075 (visited on 03/08/2021).