# Machine Learning for Launch Assessment: The Similarity-Based Launch Classification Tool (SLCT)

**Michal J. Dichter**
*Applied Technology Associates, a BlueHalo Company*

## ABSTRACT

The transformation of real-time data into actionable information is a fundamental requirement for the fast and accurate threat assessment of both orbital and suborbital launches. The Similarity-Based Launch Classification Tool (SLCT) uses a novel approach based on machine learning to classify two target variables, rocket type and target regime, from a discrete stream of state-vector observations. At the heart of the SLCT is a subsequence search algorithm that uses dynamic time warping (DTW) to measure the distance between a variable-length query sequence and fixed-length reference sequence. Armed with DTW as a time series comparison method, Monte Carlo cross-validation is used to optimize the accuracy of a $k$-nearest-neighbors ($k$-NN) classifier over a range of observation time intervals. Several run-time optimizations and approximate methods are proposed to accelerate the subsequence search.

## 1. INTRODUCTION

When an unknown launch event is detected, it is necessary to quickly and accurately assess the nature of the launch. In a typical scenario, a remote-sensing system, such as a ground- or space-based radar, detects the launch event and sets off a sequence of observations [1]. Regardless of the means of detection and tracking, one of the goals of launch assessment is to classify the launch (e.g. the rocket type and target regime) as accurately as possible, using as few observations as possible. Filter-based approaches to launch parameter estimation, and thus to rocket-type classification, are sensitive to the shortness of the observation time interval [2] and nonlinear dynamics of the rocket over its boost phase [3]. At the same time, physics-based approaches to target-regime classification, although based on sound principles, are too slow for real-time applications due to the length of the required prediction time interval [4].

Recently, machine learning algorithms, which are able to extract complicated, nonlinear relationships from large volumes of data, have been shown to produce accurate classification results at a fraction of the computational cost. Singh et al. used a real-time neural network and simulated kinematic telemetry to classify five types of ballistic missiles, each capable of executing a nominal, lofted, or depressed trajectory [5]. A deep-learning neural network was developed by Carpenter et al. to classify several types of short-range ballistic missiles [6]. The approach was extended by Eckert et al. to longer-range ballistic missiles [7]. Ritz et al. used an ensemble of neural networks to explore the dependence of missile geometry on kinematics [8].

Despite these advances, a common drawback of the above approaches is the requirement for the query sequence to have a fixed length. Once the data stream exceeds that fixed length, the set of observations must be truncated or downsampled to form an applicable query sequence. Truncation requires the operator or computer program to remove what may be valuable data from the set of observations. Models that employ this approach often assume or expect that more recent observations have greater predictive power than less recent observations, which is not always the case. On the other hand, downsampling the set of observations runs the risk of not adequately capturing the relevant dynamics.

The purpose of this work is to explore a proof of concept for the application of time series comparison methods to real-time kinematic telemetry. To that end, we propose the Similarity-Based Launch Classification Tool, or SLCT, which uses dynamic time warping (DTW) and $k$-nearest-neighbors ($k$-NN) to classify the rocket type and target regime of an unknown launch from a discrete stream of state-vector observations. We expect the elastic property of DTW, which allows for the sequences under comparison to evolve at different and time-dependent rates, to be a useful tool for launch assessment. For example, phase and frequency shifts to the kinematic telemetry caused by the performance of burns at different times and rates are detected and accounted for by DTW. This agnosticism to case-dependent parameters makes DTW an attractive candidate for pattern-matching tasks on small data sets.

The rest of this paper is structured as follows. In Section 2, we describe the data set that was used to develop and test our system. In Section 3, we present subsequence search DTW, the time series comparison method used by the SLCT. An example launch scenario is presented in Section 4. In Section 5, we choose a classification strategy and cross-validate our model. In Section 6, we reflect on the advantages and disadvantages of our approach, and offer some thoughts on how to reduce the processing requirements and enhance the accuracy of our model.

## 2. DATA SET

Our data set is composed of kinematic telemetry extracted from the public launch webcasts of three aerospace manufacturers. For each webcast, a multivariate time series of altitude and Earth-relative velocity magnitude is recorded. The sequences are captured at a rate of 1 Hz and span from liftoff to 600 seconds.

The rocket type and target regime of the payload (i.e. the target variables) are labeled for each launch. The rocket types are sorted into three classes by aerospace manufacturer: $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$. Each class encompasses a spectrum of models and booster configurations for a particular rocket family. The target regimes are also sorted into three classes: suborbital (SUB), low-Earth orbit (LEO), and trans-LEO (XLEO). A breakdown of the launch database (LD) by target variable and class is shown in Table 1.

Table 1: Data breakdown by target variable and class.

| | **Rocket type** | | | **Target regime** | | |
|---|---|---|---|---|---|---|
| | $\mathcal{A}$ | $\mathcal{B}$ | $\mathcal{C}$ | SUB | LEO | XLEO |
| **Launches** | 8 | 14 | 100 | 9 | 80 | 33 |

The LD is pre-processed by standardizing the altitude and velocity values by the catalog-wide means and standard deviations of the respective variables. This ensures that the time series comparison is not dominated by the offset or scale of any particular variable. The same global statistics are used to standardize the query sequence before it is processed by the SLCT.

## 3. DYNAMIC TIME WARPING

The SLCT uses a subsequence search algorithm to measure the DTW distance between a query sequence and a reference sequence. A full account of the method may be found in Chapters 3 and 7 of Reference [9]. However, the key concepts are presented here for completeness. To start, consider two multivariate time series, $X$ and $Y$, of lengths $N$ and $M$, respectively.

$$\begin{aligned} X &= \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\} \\ Y &= \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_M\} \end{aligned} \tag{1}$$

The components of $X$ and $Y$ are assumed to be two-element vectors of altitude and velocity. If we assume that $X$ represents the query sequence and $Y$ represents the reference sequence, then the time interval spanned by $X$ must be less than the time interval spanned by $Y$. A univariate example is shown in the left-hand panel of Figure 1.

Next, we define a cost function $c(\cdot)$ that returns the squared Euclidean distance between any two elements of $X$ and $Y$.

$$c(n,m) = \|\mathbf{x}_n - \mathbf{y}_m\|^2 \tag{2}$$

Evaluating the cost function for each element of $X$ and $Y$ produces the $N \times M$ cost matrix $C$, which is shown for the univariate example in the right-hand panel of Figure 1. Darker cells represent lower-cost matches, while lighter cells represent higher-cost matches.

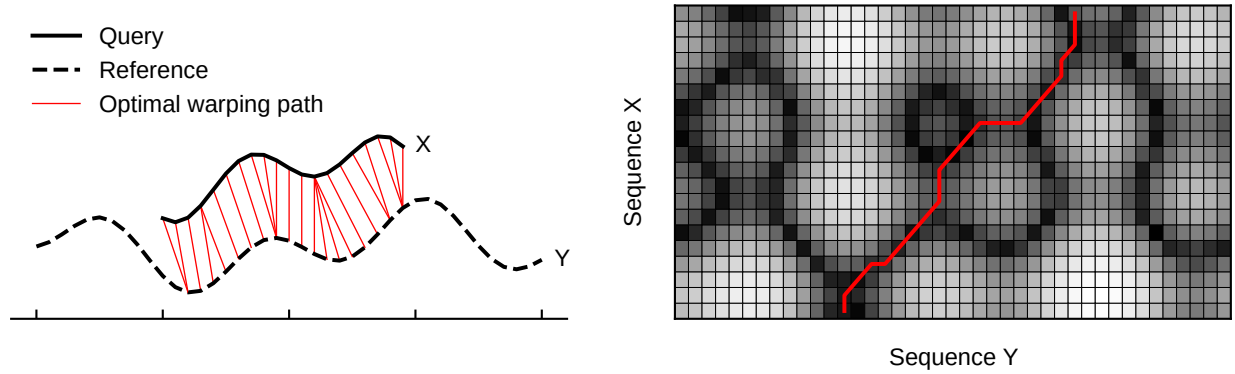


Fig. 1: (Left) Query sequence and reference sequence connected by optimal warping path. (Right) Cost matrix with optimal warping path superposed.

To compute an alignment between the two sequences, we define a *warping path* as any path through the cost matrix that obeys the following set of conditions.

- **Boundary condition**: The path must start at the first row of $C$ and end at the last row of $C$.
- **Step-size condition**: The matrix must be traversed using only moves from the set $\{(1,0),(0,1),(1,1)\}$.

The boundary condition ensures that the complete query sequence is matched to the reference sequence. The step-size condition ensures that no elements are excluded from the alignment and that the temporal order of each sequence is respected.

The cost of a warping path is defined as the sum of the cells through which it passes. The DTW distance is then defined as the cost of the optimal, or lowest-cost, warping path, divided by its length.

$$\mathrm{DTW}(X,Y) = \frac{1}{P}\sum_{p=1}^{P} c(n_p, m_p) \tag{3}$$

Informally, the optimal warping path runs along a valley of low cost, as shown by the red curve in the right-hand panel of Figure 1. Because each cell of the cost matrix represents a correspondence between an element of $X$ and an element of $Y$, the optimal warping path may also be thought of as a set of segments connecting the two sequences, as shown in the left-hand panel of Figure 1. This perspective emphasizes how DTW stretches and compresses the sequences to form an optimal alignment.

## 4. EXAMPLE LAUNCH SCENARIO

In this section, we apply subsequence search DTW to analyze a launch whose rocket type and target regime are assumed to be unknown. In order to respect the temporal order of launches in the LD, we select one of the more recent $\mathcal{B}$-class LEO launches and remove it from the LD. To simulate a late detection, a 1-Hz stream of state-vector observations is assumed to start 140 seconds after liftoff and terminate after 30 seconds. The state vectors are mapped as they arrive to the two-dimensional feature space of altitude and velocity. The converted state vectors are then appended to the query sequence.

For each new observation, we perform an exact subsequence search, in which the query sequence is compared to each launch in the leave-one-out LD. The DTW distances of the seven best matches from each class for both target variables are recorded and converted to similarity values between zero and one.

$$\mathrm{SIM}(X,Y) = e^{-\mathrm{DTW}(X,Y)} \tag{4}$$

Kernel density estimation is used to approximate the class-wise distributions of the resultant values. Each distribution is then normalized by its maximum value for visual ease. The result of this process is shown in Figure 2.

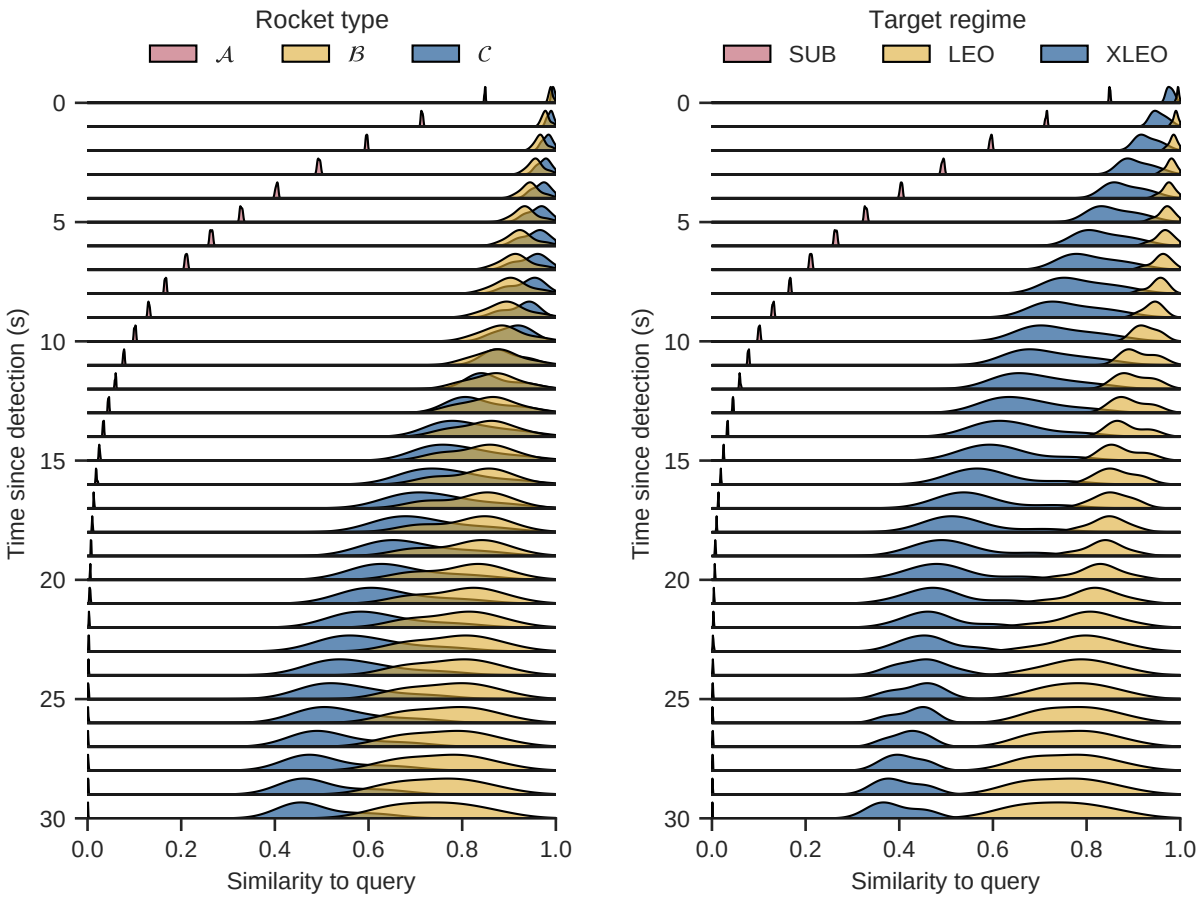


Fig. 2: Similarity of query sequence ($\mathcal{B}$-class LEO) to best-matched reference sequences from each class.

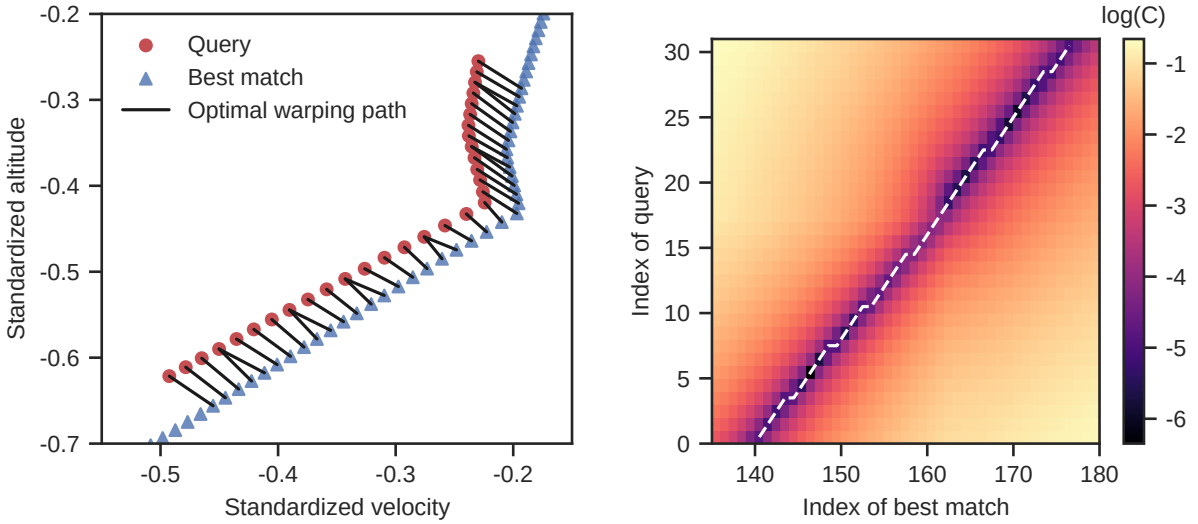


Fig. 3: (Left) Query and best-matched reference sequence in feature space connected by optimal warping path. (Right) Cost matrix with optimal warping path superposed.

At the start of the data stream, the distributions for the $\mathcal{B}$- and $\mathcal{C}$-class rockets, as well as for the LEO and XLEO classes, have a considerable amount of overlap. The overlap reflects the fact that the query sequence matches both classes equally well, and suggests that a single observation is not sufficient to discriminate between them. As the launch progresses, and the length of the query sequence grows, the overlapped distributions start to separate. At the end of the data stream, the strongest matches to the query sequence are $\mathcal{B}$-class LEO launches. The convergence of the subsequence search to the true rocket type and target regime demonstrates the effectiveness of DTW as a tool for processing kinematic telemetry. This sort of plot is useful because it captures how the similarity values evolve as a function of the observation time interval. Over the course of the launch, the query sequence traces out a progressively more distinctive trajectory in feature space, which discriminates it from many of the reference sequences and exposes weaker matches.

It is also useful to look at the query sequence and best-matched reference sequence in feature space. The left-hand panel of Figure 3 shows the two sequences connected by the optimal warping path. The fact that some elements of the query sequence are mapped to multiple elements of the reference sequence means that the two sequences are frequency-shifted with respect to each other, with the query sequence evolving at a relatively faster rate. This is also apparent from the fact that the optimal warping path deviates from the diagonal when superposed on the cost matrix, as shown in the right-hand panel of Figure 3. This example demonstrates why DTW is so effective on smaller data sets. Even though the best-matched reference sequence evolves at a relatively slower rate, the nonlinear nature of the warping path allows for common shapelets, such as the sharp turn that both sequences make in feature space, to be detected and matched.

## 5. MODEL SELECTION AND CROSS-VALIDATION

Armed with DTW as a tool to compute distances between sequences, we now require a classification strategy. One approach would be to treat the vector of distances between the query sequence and LD as a feature to be processed by a machine learning model, such as a neural network or random forest. Another approach would be to enforce the positive definiteness of the LD-to-LD distance matrix and apply a kernel method, such as a support-vector machine. However, coupled with DTW as a distance measure, the simple strategy of 1-NN (i.e. best-match-takes-all) often outperforms more complex approaches, particularly on smaller data sets [10]. Therefore, so as not to unduly complicate our model, we choose $k$-NN as our baseline classification strategy.

The accuracy of the classifier is optimized over two hyperparameters. The first hyperparameter is the number of nearest neighbors used to make the classification decision, which we select from the set $k = \{1,3,5,7\}$. The second hyperparameter is the weighting strategy, which controls the relative strength of each nearest neighbor. We test three popular approaches:

1. **Naive**: Each of the $k$-NN contributes equally to the classification decision.
2. **Distance-based**: Each of the $k$-NN is weighted by its distance to the query sequence.
3. **Frequency-based**: Each of the $k$-NN is weighted by the reciprocal of its class size.

For the distance-based weighting strategy, we use an approach in which the nearest of the $k$ matches is given a weight of one, and the most distant of the $k$ matches is given a weight of zero [11]. As such, the distance-based weighting strategy is only relevant for $k$ greater than three. For the frequency-based weighting strategy, we define aggregate classes by unique target-variable combinations. So, for example, $\mathcal{C}$-class LEO launches and $\mathcal{C}$-class XLEO launches are treated as separate classes. This approach has been proposed as a way to offset the adverse effects of skewed class-label distributions.

For each set of hyperparameters, the following cross-validation procedure is performed. First, a quasi-random launch is selected and removed from the LD. (More/less populous classes are under-/over-sampled.) Next, a launch segment, whose length is randomly sampled from the set $T = \{0,10,20,30\}$ seconds, is extracted from the first 30–300 seconds of the launch. The launch segment is then processed by the SLCT using the leave-one-out LD. The predicted and actual classes of the query sequence are recorded, and the above process is repeated 100,000 times. At the end of each Monte Carlo run, the error rate is calculated, and the cross-validation procedure is repeated for a new set of hyperparameters. The result of the grid search is shown in Table 2. Of the nine hyperparameter combinations, 3-NN with frequency-based weighting provides the best overall classification performance.

Table 2: Dependence of error rate on model hyperparameters.

| | **Weighting strategy** | | | | | |
|---|---|---|---|---|---|---|
| | **Rocket classification** | | | **Regime classification** | | |
| $k$ | Naive | Distance | Frequency | Naive | Distance | Frequency |
| 1 | 0.139 | - | - | 0.130 | - | - |
| 3 | 0.157 | - | 0.122 | 0.131 | - | 0.120 |
| 5 | 0.198 | 0.136 | 0.140 | 0.140 | 0.125 | 0.121 |
| 7 | 0.229 | 0.147 | 0.142 | 0.152 | 0.121 | 0.127 |

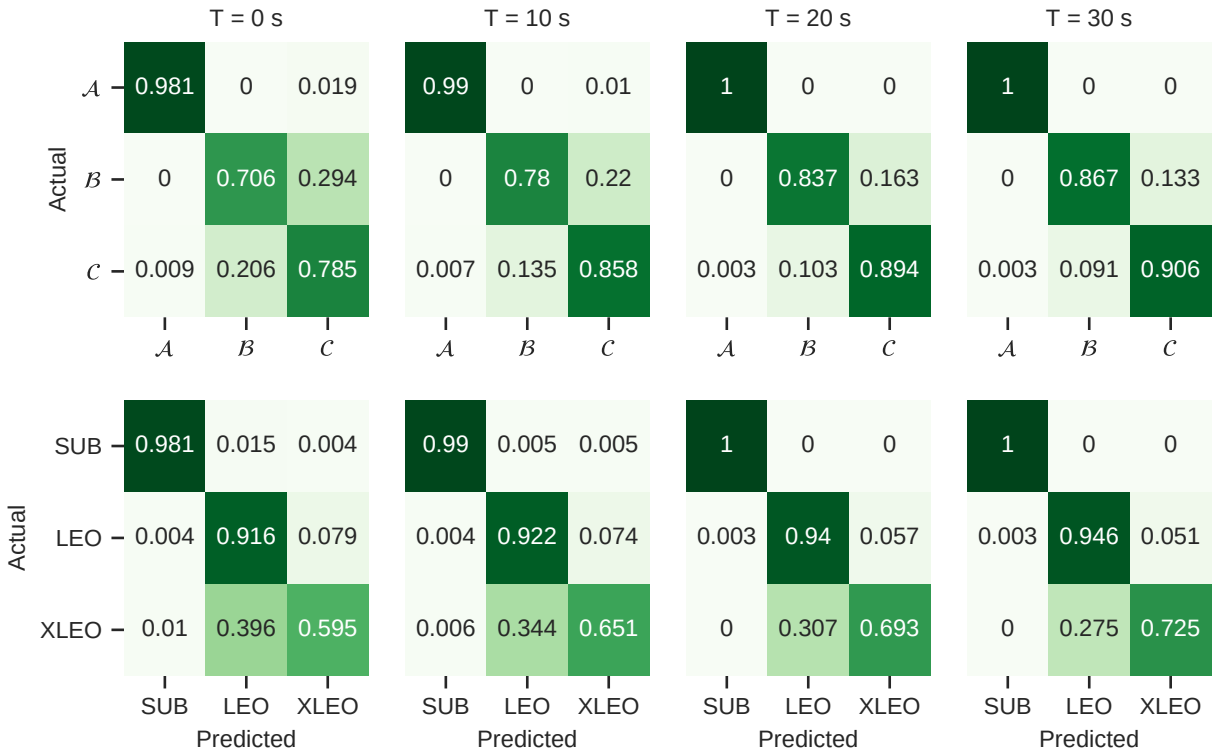


Fig. 4: Dependence of confusion matrix on observation time interval.

The performance of the best model is further deconstructed in Figure 4, where a confusion matrix is plotted for each of the considered observation time intervals. Diagonal elements represent the fraction of samples for which the predicted class is equal to the actual class, while off-diagonal elements represent the fraction of mislabeled samples. Larger values along the diagonal correspond to more accurate classifications. The two left-most plots show the model performance after a single observation, while the two right-most plots show the model performance after 30 seconds of observations.

In general, longer observation time intervals lead to more accurate classifications. In terms of the target-regime variable, suborbital launches are almost always labeled correctly. On the other hand, LEO and XLEO launches are sometimes confused for each other, particularly when the observation time interval is shorter. This is partly due to semantic differences between the class labels themselves: Orbital launches (LEO and XLEO) are conceptually closer to each other than they are to suborbital launches. The more fundamental problem, however, is the sparseness of our data set. Launches that follow less-popular paths through feature space do not have nearby reference sequences in the LD. Without nearby reference sequences, the model resorts to relatively distant nearest neighbors to make the classification decision, and erroneous results become more probable.

## 6. FUTURE WORK

To further optimize the performance of the SLCT, we plan to use simulated kinematic telemetry to sample the performance range of each rocket more densely. Increasing the size of the LD forces us to consider ways to accelerate the subsequence search. One of the most effective approaches is to abandon the reference sequence under test as soon as the best-so-far DTW distance is exceeded by a dynamically-computed lower bound. Rakthanmanon et al. used a cascade of lower bounds, as well as some other novel optimizations, to perform an exact subsequence search of one million reference sequences, for a query sequence of length 128, in a few tenths of a second [12]. This result suggests that an optimized subsequence search may be able to meet the speed requirements for launch assessment, given that number of time series required to adequately capture the performance range of a rocket is on the order of thousands [6]–[8].

Another approach, perhaps more scalable, is to settle for an approximate result. So far, we have only considered exact subsequence searches, in which the global-best match is always returned. In fact, there are a wealth of approximate nearest-neighbor algorithms, which are able to greatly reduce the processing requirements at the cost of a relatively small loss of accuracy. These algorithms are used by groups such as Spotify (ANNOY) and Facebook (FAISS) to suggest results in real time. One of the more recent breakthroughs comes from Malkov and Yashunin, whose graph-based search is logarithmic in the size of the data set and consistently outperforms other approximate methods in both speed and accuracy [13]. Of the two approaches – optimizing the exact search or using an approximate method – we expect that the latter has the most promise for real-time applications.

## 7. CONCLUSION

We have demonstrated an approach that combines DTW and $k$-NN to classify an unknown launch from a variable-length, multivariate time series of kinematic telemetry. The accuracy of the classifier was optimized using Monte Carlo cross-validation. The performance was shown to depend on both the length of the observation time interval and the completeness of the data set. Future work will focus on the use of approximate search methods and the development of a synthetic data set to accelerate the speed and enhance the accuracy of the SLCT.

## ACKNOWLEDGMENTS

This work was supported by the U.S. Air Force Research Laboratory contract FA9453-20-C-0021. The author gratefully acknowledges helpful feedback from Jeremy Wojcik and William Manning.

## REFERENCES

[1] M. L. Stone and G. P. Banner, "Radars for the Detection and Tracking of Ballistic Missiles, Satellites, and Planets," *Lincoln Laboratory Journal*, vol. 12, no. 2, pp. 217–244, 2000.

[2] S. Park, J. Jeong, C.-K. Ryoo, and K. Choi, "Detection and Classification of a Ballistic Missile in Ascent Phase," in *Proceedings of the 11th International Conference on Control, Automation, and Systems*, 2011, pp. 1235–1238.

[3] J. R. V. Zandt, "Boost Phase Tracking with an Unscented Filter," in *Signal and Data Processing of Small Targets 2002*, vol. 4728, 2002, pp. 263–274.

[4] M.-J. Tsai and F. A. Rogal, "Angle-Only Tracking and Prediction of Boost Vehicle Position," in *Signal and Data Processing of Small Targets 1991*, vol. 1481, 1991, pp. 281–291.

[5] U. K. Singh, V. Padmanabhan, and A. Agarwal, "Dynamic Classification of Ballistic Missiles Using Neural Networks and Hidden Markov Models," *Applied Soft Computing*, vol. 19, pp. 280–289, 2014.

[6] M. Carpenter, R. Hartfield, L. Zhou, and N. Speakman, "Statistical Learning for Munition Trajectory Prediction," in *AIAA Scitech 2019 Forum*. 2019, pp. 1–11.

[7] J. Eckert, M. Carpenter, R. Hartfield, and N. Cervantes, "Classification of Intermediate Range Missiles During Launch," in *AIAA Scitech 2020 Forum*. 2020, pp. 1–14.

[8] S. G. Ritz, R. J. Hartfield, J. A. Dahlen, J. E. Burkhalter, and W. S. Woltosz, "Near Real-Time Characterization of Unknown Missiles In Flight Using Computational Intelligence," in *2015 IEEE Aerospace Conference*, 2015, pp. 1–11.

[9] M. Müller, *Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications*, 1st. Springer Publishing Company, 2015.

[10] X. Xi, E. Keogh, C. Shelton, L. Wei, and C. A. Ratanamahatana, "Fast Time Series Classification Using Numerosity Reduction," in *Proceedings of the 23rd International Conference on Machine Learning*, 2006, pp. 1033–1040.

[11] S. A. Dudani, "The Distance-Weighted $k$-Nearest-Neighbor Rule," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-6, no. 4, pp. 325–327, 1976.

[12] T. Rakthanmanon, B. Campana, A. Mueen, *et al.*, "Searching and Mining Trillions of Time Series Subsequences under Dynamic Time Warping," in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2012, pp. 262–270.

[13] Y. A. Malkov and D. A. Yashunin, "Efficient and Robust Approximate Nearest Neighbor Search Using Hierarchical Navigable Small World Graphs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 4, pp. 824–836, 2020.