Time Series Classification under More Realistic Assumptions

Bing Hu
Outline

• Motivation
• Proposed Framework
  - Concepts
  - Algorithms
• Experimental Evaluation
• Conclusion & Future Work
Much of the progress in time series classification from streams is almost *Certainly Optimistic*

Because they have implicitly or explicitly made *Unrealistic Assumptions*
Assumption (1)

perfectly aligned atomic patterns can be obtained

Individual and complete gait cycles for biometric classification

walking  running  ascending-stairs
Assumption (1)

*perfectly aligned atomic patterns* can be obtained

However, the task of extracting individual gait cycles is *not trivial*!
Assumption (2)

The patterns are all equal length

However,

Heart beat can have different lengths

two heart beat of different lengths
Assumption (2)

The patterns are all equal length

Gun/Point problem is probably the most studied time series classification problem, having appeared in at least one hundred works.

UNREALISTIC!
All forty-five time series datasets contain only equal-length data.

Assumption (2)

The patterns are all equal length.

Contriving of time series datasets seems to be the norm.....
Assumption (3)

Every item that to be classified belongs to *exactly* one of the well-defined classes
Assumption (3)

Every item that is to be classified belongs to exactly one of the well-defined classes.
Assumption (3)

Every item that to be classified belongs to *exactly* one of the well-defined classes

A person cannot perform *walking* or *running* all the time...

The classification framework must be willing to say *I DO NOT KNOW*
Most of the literature implicitly or explicitly assumes one or more of the following:
Unrealistic Assumptions

- Copious amounts of *perfectly aligned atomic patterns* can be obtained.
- The patterns are *all equal length*.
- Every item that we attempt to classify *belongs to exactly one of the well-defined classes*.
Outline

• Motivation
• **Proposed Framework**
  - Concepts
  - Algorithms
• Experimental Evaluation
• Conclusion & Future Work
We demonstrate a time series classification framework that does not make *any* of these assumptions.
Our Proposal

- **Leverages weakly-labeled data**
  removes assumption (1) (2)
- **Utilizes a data dictionary**
  removes assumption (1) (2)
- **Exploits rejection threshold**
  removes assumption (3)

Assumptions:
(1) *perfectly aligned atomic patterns*
(2) *patterns are all of equal lengths*
(3) *every item to classify belongs to exactly one of the well-defined classes*

Unrealistic Assumptions:
- Copious amounts of *perfectly aligned atomic patterns* can be obtained
- The patterns are *all equal length*
- Every item that we attempt to classify *belongs to exactly one of our well-defined classes*
Weakly-Labeled data

such as “This ten-minute trace of ECG data consists mostly of arrhythmias, and that three-minute trace seems mostly free of them”

removing assumption (1)
Weakly-Labeled data

- Extraneous/irrelevant sections
- Redundancies

weakly-labeled data from Bob

Extraneous data
Weakly-Labeled data

How to mitigate the problem of weakly-labeled data?

• Extraneous/irrelevant sections
• Redundancies
Data Dictionary

- A (potentially very small) “smart” subset of the training data.
- It spans the concept space.

We want to perform ECG classification between Bob and other person’s heartbeat.
Concept space

Anything beyond the threshold, it is in other class

In the above figure, the concept space is one “*” and one “+”
Our algorithm does not know the patterns in advance.
We learn those patterns.
Unrealistic Assumptions

- Copious amounts of *perfectly aligned atomic patterns* can be obtained
- The patterns are *all equal length*
- Every item that we attempt to classify *belongs to exactly one of our well-defined classes*
Data Dictionary

The patterns to be classified can be of different lengths

- leisurely-amble
- normal-paced-walk
- brisk-walk
Unrealistic Assumptions

- Copious amounts of *perfectly aligned atomic patterns* can be obtained
- The patterns are *all equal length*
- Every item that we attempt to classify *belongs to exactly one of our well-defined classes*
Rejection Threshold

A byproduct of the data dictionary

```
if NN_Dist of query > threshold
    query is in the other class
```

<table>
<thead>
<tr>
<th>data dictionary</th>
<th>threshold</th>
<th>queries</th>
</tr>
</thead>
<tbody>
<tr>
<td>running</td>
<td>7.6</td>
<td>NN_dist &lt; 7.6</td>
</tr>
<tr>
<td>walking</td>
<td>6.4</td>
<td>NN_dist &gt; 6.4</td>
</tr>
<tr>
<td>ascending stairs</td>
<td>7.3</td>
<td>NN_dist &gt; 7.3</td>
</tr>
</tbody>
</table>

A person cannot perform running, walking, ascending-stairs all the time. There must exist other classes.
Desirable Properties of Data Dictionaries

• the classification error rate using D should be \textit{no worse} than \textit{(can be better)} using all the training data

\textit{Why ?}
Desirable Properties of Data Dictionaries

This is because the data dictionaries contain less spurious/misleading data.

weakly-labeled data

data dictionary
Desirable Properties of Data Dictionaries

- D can be a very small percentage of the training data
  - faster running time
  - resource limited device

for one hour of ECG data

Space: **3600Kbits**

Data dictionary

**Space:** 20 Kbits
Desirable Properties of Data Dictionaries

the number of subsequences within each class in $\mathcal{D}$ can be different.

walking

vacuum cleaning
Desirable Properties of Data Dictionaries

the number of subsequences within each class in $\mathbf{D}$ can be different

✓ For example, if the number of $S$ in $\mathbf{D}$ is larger than $\text{PVC}$, we can conclude that the variance of $S$ is larger than $\text{PVC}$
An Additional Insight on Data Redundancy

Data dictionary A

- class bears
- class bulls

Data dictionary B

- class bears
- class bulls

*leisurely-amble*
*normal-paced-walk*
*brisk-walk*

Our Solution: Uniform Scaling
Using the *Euclidean* distance, the misalignment would cause a large error. However, the problem can be solved by using the *Uniform Scaling* distance.

The *Uniform Scaling* distance is a simple generalization of the *Euclidean* distance.
An Additional Insight on Data Redundancy

Uniform Scaling

✓ to further reduce the size of data dictionary

✓ to achieve lower error rate

Imagine the training data does contain some examples of gaits at speeds from 6.1 to 6.5\,km/h, unseen data contains 6.7\,km/h
Outline

• Motivation
• **Proposed Framework**
  - Concepts
  - Algorithms
• Experimental Evaluation
• Conclusion and Future Work
Classification using a Data Dictionary

Before showing how to build the data dictionary, I want to show how to use it first.
We use the classic one nearest neighbor algorithm.
Classification using a Data Dictionary

We use the classic one nearest neighbor algorithm

data dictionary

<table>
<thead>
<tr>
<th>Activity</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>running</td>
<td><img src="signal_running.png" alt="Signal" /></td>
</tr>
<tr>
<td>walking</td>
<td><img src="signal_walking.png" alt="Signal" /></td>
</tr>
<tr>
<td>ascending stairs</td>
<td><img src="signal_stairs.png" alt="Signal" /></td>
</tr>
</tbody>
</table>

threshold

<table>
<thead>
<tr>
<th>Class</th>
<th>Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>running</td>
<td>7.6</td>
</tr>
<tr>
<td>walking</td>
<td>6.4</td>
</tr>
<tr>
<td>stairs</td>
<td>7.3</td>
</tr>
</tbody>
</table>

query: ?

Rejection Threshold

- A byproduct of the data dictionary
- If $\text{NN}\_\text{Dist of q} > t$
- $q$ is in the other class

A person cannot perform running, walking, ascending stairs all the time... There must exist other classes...
Building the Data Dictionary

Intuition

We show a toy dataset in the discrete domain to show the intuition. Our goal remains large real-valued time series data.

A *weakly-labeled* training dataset that contains two classes C1 and C2:

C1 = \{ dpacekfjklwalkflwalkklpacedalyutekwalksfj\}
C2 = \{ jhjhleapashljumpokdjkllleaphfleapfjjumpacgd\}
Building the Data Dictionary

Intuition

a training dataset that contains two classes C1 and C2:

C1 = {pacekfjk|walk|walk|klpacedalyutekwalksfj}
C2 = {jhjh|leapashl|jumpokdjk|leapfleapfjjumpacgd}

• weakly-labeled
• the colored text is for introspection only
Building the Data Dictionary

Intuition

C1 = \{ dpacekfjklwalkflwalkklpacedalyutekwalksfj\}
C2 = \{ jhjhleapashljumpokdjklleaphfleapfjjumpacgd\}

data dictionary

threshold

C1: \{ pace, walk \}
C2: \{ leap ; jump \}

r = 1
Building the Data Dictionary

Intuition

data dictionary

\[ C_1: \{ \text{pace, walk} \} \]
\[ C_2: \{ \text{leap, jump} \} \]

threshold

\[ r = 1 \]

Query:

\[ \text{ieap} \quad \text{NN}_\text{dist} = 1 \quad C_2 \]
\[ \text{kklp} \quad \text{NN}_\text{dist} = 3 \quad \text{other} \]
Building the Data Dictionary

Intuition

What is the result if we do not have data dictionary?

\[C_1 = \{ \text{dpacekfjklwalkflwalkkklpacedalyutekwalksfj}\}\]
\[C_2 = \{ \text{jjjleapashljumpokdjkllleaphfleapfjjumpacgd}\}\]

\[kklp \quad \text{dist} = 0 \quad \text{C1} \times\]
Building the Data Dictionary

Intuition

Consider a streaming data that needs to be classified:
.. ttgpacedgrteweerjumpwalkflqrafertwqhafhfahtfbseew..

How we build the data dictionary?

Collecting statistics about which substrings are often used for correct prediction
To use a ranking function to score every subsequence in C.

These “scores” rate the subsequences by their expected utility for classification of future unseen data.

We use these scores to guide a greedy search algorithm, which iteratively selects the best subsequence and places it in D.
Building the Data Dictionary

Algorithm

How do we know this utility?

*We estimate the utility by cross validation*

Three steps below
Building the Data Dictionary

**Step 1.** The algorithm *scores* the subsequences in C.

Procedure:

1. randomly extracted a large number of queries
2. cross-validation
3. rank every point in C using the SimpleRank function\[a\]

\[
rank(x) = \sum_j \begin{cases} 
1, & \text{if } \text{class}(x) = \text{class}(x_j) \\
-2 / (\text{num}_\text{of}_\text{class} - 1), & \text{if } \text{class}(x) \neq \text{class}(x_j) \\
0, & \text{other}
\end{cases}
\]

\[a\]K. Ueno, X. Xi, E. Keogh and D.J. Lee, Anytime Classification Using the Nearest Neighbor Algorithm with Applications to Stream Mining, ICDM, 2006
Building the Data Dictionary

SimpleRank function

<table>
<thead>
<tr>
<th></th>
<th>$S_1$</th>
<th>$S_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>classification accuracy</strong></td>
<td>70%</td>
<td>70%</td>
</tr>
</tbody>
</table>

- However, suppose that $S_1$ is also very close to many objects with different class labels (enemies).
- If $S_2$ keeps a larger distance from its enemy class objects, $S_2$ is a much better choice for inclusion in $D$.

[a]K.Ueno, X. Xi, E. Keogh and D.J.Lee, Anytime Classification Using the Nearest Neighbor Algorithm with Applications to Stream Mining, ICDM, 2006
Building the Data Dictionary

SimpleRank function \([a]\)

\[
rank(x) = \sum_j \left\{ \begin{array}{ll}
1, & \text{if } \text{class}(x) = \text{class}(x_j) \\
-2 / (\text{num}_\text{of}_\text{class} - 1), & \text{if } \text{class}(x) \neq \text{class}(x_j) \\
0, & \text{other}
\end{array} \right.
\]

- The intuition behind this algorithm is to give every instance a rank according to its \textit{contribution} to the classification.

- Score function \textit{rewards} the subsequence that return \textit{correct} classification and \textit{penalize} those return \textit{incorrect} classification.

\[\text{[a]}\]K. Ueno, X. Xi, E. Keogh and D. J. Lee, Anytime Classification Using the Nearest Neighbor Algorithm with Applications to Stream Mining, ICDM, 2006
Building the Data Dictionary

The iteration procedure:

**Step 1.** The algorithm *scores* the subsequences in C.

**Step 2.** The *highest* scoring subsequence is *extracted* and placed in D.

**Step 3.** We identify all the queries that are incorrectly classified by the current D. These incorrectly classified items are passed back to **Step 1** to re-score the subsequences in C.
Step 1. The algorithm *scores* the subsequences in C.

For simplicity, we use one query to illustrate how to score C.
We use one query to illustrate the ranking procedure

**Step 1**

**query** \( q \)

weakly-labeled data

<table>
<thead>
<tr>
<th>class 1</th>
<th>class 2</th>
<th>class 3</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="class 1 data" /></td>
<td><img src="image2.png" alt="class 2 data" /></td>
<td><img src="image3.png" alt="class 3 data" /></td>
</tr>
</tbody>
</table>

Perform one nearest neighbor classification

Two cases:

- when \( q \) is correctly classified
- when \( q \) is incorrectly classified
1. This query q is correctly classified as class 1
   NN_friend_dist = 10.4

2. found out the nearest neighbor distance in enemy (class 2 and class 3) is
   NN_enemy_dist = 13

3. For any subsequence that has nearest neighbor distance in friend class that is less than
   NN_enemy_dist, we give it a positive score.
   They are called nearest neighbor friends or likely true positives
query q

Two cases:

1. If $\text{NN\_friend\_dist} < \text{NN\_enemy\_dist}$
   - find nearest neighbor friends or likely true positives in the friend class

2. If $\text{NN\_friend\_dist} > \text{NN\_enemy\_dist}$
   - find nearest neighbor enemies or likely false positives in the enemy class
query q

1. This query q is wrongly classified as class 3
   \[ \text{NN}_\text{enemy}_\text{dist} = 13 \]

2. found out the nearest neighbor distance in friends (class 1)
   \[ \text{NN}_\text{friend}_\text{dist} = 16 \]
1. This query $q$ is wrongly classified as class 3
   $\text{NN}_\text{enemy}_\text{dist} = 13$

2. Found out the nearest neighbor distance in friend (class1)
   $\text{NN}_\text{friend}_\text{dist} = 16$

3. For any subsequence that has nearest neighbor distance in enemy class that is less than
   $\text{NN}_\text{friend}_\text{dist}$, we give it a negative score.
   They are called $\text{nearest neighbor enemies}$ or $\text{likely false positives}$

$\text{likely true positives}$
query q

Two cases:

If $NN_{friend\_dist} < NN_{enemy\_dist}$
find nearest neighbor friends or likely true positives in the friend class

If $NN_{friend\_dist} > NN_{enemy\_dist}$
find nearest neighbor enemies or likely false positives in the enemy class

$$rank(S) = \sum_k \begin{cases} 
1, & \text{likely true positives} \\
-2/(num\_of\_class - 1), & \text{likely false positives} \\
0, & \text{other}
\end{cases}$$
Building the Data Dictionary

Step 2

The *highest* scoring subsequence is *extracted* and placed in $D$. 

the point that has the highest score

the extracted subsequence
Building the Data Dictionary

Step 3

(1). Perform classification for all the queries using $D$. (2). The incorrectly classified items are passed back to Step 1 to re-score the subsequences in C.
Building the Data Dictionary

When to stop the iteration?

- The accuracy of classification using just the data dictionary cannot be improved any more.

- The size of the data dictionary.
Building the Data Dictionary

Learning the threshold distance

After the data dictionary is built, we learn a threshold to reject future queries, which do not belong to any of the learned classes.
Building the Data Dictionary

Learning the threshold distance

1. Record a **histogram** of the nearest neighbor distances of testing queries that are *correctly* classified using $D$
2. Record a **histogram** of the nearest neighbor distances of the queries in *other* classes
Uniform Scaling Technique

We replace the *Euclidean* distance with *Uniform Scaling* distance in the above data dictionary building and threshold learning process.
Outline

• Motivation
• Proposed Framework
  - Concepts
  - Algorithms
• Experimental Evaluation
• Conclusion and Future Work
Experimental Evaluation

An Example Application in Physiology

Eight hours of data sampled at 110Hz was collected from wearable sensors on eight subjects’ wrist, chest and shoes.

The activities includes:
normal-walking, walking-very-slow, running, ascending-stairs, descending-stairs, cycling, etc.
Experimental Evaluation

An Example Application in Physiology

Euclidean distance

Uniform Scaling distance

Using all the training data, the testing error rate is 0.22

Test error: randomly built $D$

Train error: Uniform Scaling

Test error: Uniform Scaling

Train error: Uniform Scaling

Error Rate

Percent of the training data used by the data dictionary
Experimental Evaluation

An Example Application in Physiology

Two examples of the rejected queries

Both queries contain significant amount of noise
Experimental Evaluation

An Example Application in Physiology

Rival Method

• We compare with the widely-used approach, which extracts signal features from the sliding windows. For fairness to this method, we used their suggested window size.

• We tested all the following classifiers: K-nearest neighbors, SVM, Naïve Bayes, Boosted decision trees, C4.5 decision tree
## Experimental Evaluation

An Example Application in Physiology

<table>
<thead>
<tr>
<th></th>
<th>Rival approach</th>
<th>Strawman</th>
<th>Our approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>error rate</td>
<td>0.364</td>
<td>0.221</td>
<td>0.152</td>
</tr>
<tr>
<td>amount of data used for classification</td>
<td>100%</td>
<td>100%</td>
<td>8.3%</td>
</tr>
<tr>
<td>assumptions</td>
<td>(1),(2),(3)</td>
<td>(1),(2),(3)</td>
<td>no assumption</td>
</tr>
<tr>
<td>running time</td>
<td>13 hours</td>
<td>28 hours</td>
<td>2.2 hours</td>
</tr>
<tr>
<td>rejected data</td>
<td>0</td>
<td>0</td>
<td>9.5%</td>
</tr>
</tbody>
</table>
The dataset includes ECG recordings from **fifteen subjects** with severe congestive heart failure. The individual recordings are each about **20 hours** in duration, samples at 250Hz.
Experimental Evaluation

An Example Application in Cardiology

Euclidean distance

Uniform Scaling distance

Using all the training data, the testing error rate is 0.102.
## Experimental Evaluation

An Example Application in Cardiology

<table>
<thead>
<tr>
<th></th>
<th>Rival approach</th>
<th>Strawman</th>
<th>Our approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>error rate</td>
<td>0.267</td>
<td>0.102</td>
<td>0.076</td>
</tr>
<tr>
<td>amount of data used for classification</td>
<td>100%</td>
<td>100%</td>
<td>2.1%</td>
</tr>
<tr>
<td>assumptions</td>
<td>(1),(2),(3)</td>
<td>(1),(2),(3)</td>
<td>no assumption</td>
</tr>
<tr>
<td>running time</td>
<td>78 hours</td>
<td>180 hours</td>
<td>3.6 hours</td>
</tr>
<tr>
<td>rejected data</td>
<td>0</td>
<td>0</td>
<td>4.8%</td>
</tr>
</tbody>
</table>
Experimental Evaluation

An Example Application in Daily Activities

The MIT benchmark dataset that contains 20 subjects performing approximately 30 hours of daily activities.

such as: running, stretching, scrubbing, vacuuming, riding-escalator, brushing-teeth, walking, bicycling, etc. The data was sampled at 70 Hz.
Experimental Evaluation

An Example Application in Daily Activities

Error rate Percent of data dictionary to all the training data

- 2.0%
- 3.0%
- 4.0%
- 5.0%
- 0.0%
- 1.0%

Using all the training data, the testing error rate is 0.237

Test error: randomly built D

Train error

Euclidean distance

Test error: uniform scaling
Train error: uniform scaling

Uniform Scaling distance

Euclidean train error for reference
Experimental Evaluation
An Example Application in Daily Activities

<table>
<thead>
<tr>
<th></th>
<th>Rival approach</th>
<th>Strawman</th>
<th>Our approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>error rate</td>
<td>0.314</td>
<td>0.237</td>
<td>0.152</td>
</tr>
<tr>
<td>amount of data used for</td>
<td>100%</td>
<td>100%</td>
<td>3.8%</td>
</tr>
<tr>
<td>classification</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>assumptions</td>
<td>(1),(2),(3)</td>
<td>(1),(2),(3)</td>
<td>no assumption</td>
</tr>
<tr>
<td>running time</td>
<td>52 hours</td>
<td>123 hours</td>
<td>4.8 hours</td>
</tr>
<tr>
<td>rejected</td>
<td>0</td>
<td>0</td>
<td>6.3%</td>
</tr>
</tbody>
</table>
Outline

• Motivation

• Proposed Framework
  - Concepts
  - Algorithms

• Experimental Evaluation

• Conclusion and Future Work
Conclusion

• Much of the progress in time series classification from streams in the last decade is almost *Certainty Optimistic*

•Removing those unrealistic assumptions, we achieve much *higher accuracy in a fraction* of time
Conclusion

• Our approach requires only very weakly-labeled data, such as “in this ten minutes of data, we see mostly normal heartbeats.....”, removing assumption (1)

• Using this data we automatically build a “data dictionary”, which contains only the minimal subset of the original data to span the concept space. This mitigates assumption (2)

• As a byproduct of building this data dictionary, we learn a rejection threshold, which allows us to remove assumption (3)
Thank you for your attention!

If you have any questions, please email bhu002@ucr.edu