Please note this draft reflects NAB v1.0.

The Numenta Anomaly Benchmark (NAB)

The purpose of this whitepaper is to provide details relevant to those looking to better understand NAB under the hood. We recommend taking a look at our conference publication [1] for an overall discussion of NAB and results.

Please use the following reference for this whitepaper:

> A. Lavin and S. Ahmad. (2015) *The Numenta Anomaly Benchmark* [White paper]. Redwood City, CA: Numenta, Inc. Available: https://github.com/numenta/NAB/wiki.

The general email for NAB related questions is nab@numenta.org.

This document was updated on May 11, 2017 to remove some obsolete data fomating requirements.

Table of Contents

1. Introduction

The Numenta Anomaly Detection Benchmark (NAB) attempts to provide a controlled and repeatable environment and tools to test and measure different anomaly detection algorithms on streaming data. NAB aims to provide a standard framework for which we can compare and evaluate different algorithms for detecting anomalies in streaming data. NAB consists of an open source repository available under Gnu AGPL v3, containing labeled benchmark data, code, file format descriptions, examples, and supporting documentation.

One important aspect of NAB is a scoring methodology that is designed for streaming applications. The NAB scoring system quantifies the degree to which the detector under test (DUT) will be valuable in practical applications. NAB compares the DUT detections against the ground truth labels, which is a combination of the individual labeling files, using a programmatic scoring function In addition we introduce the concept of “application profiles”, which enables us to vary the relative cost of the scoring metrics – true positives, false positives, false negatives. These components are discussed in detail below.

A second important aspect of NAB is that the benchmark dataset includes both artificial and real world data. The data files in the corpus include server metrics, machine sensor readings, social media metrics, ad click data, traffic volume, and more. Real-world data will be continuously added to the dataset, including applications such as financial data, security settings, positional data, etc.

Given a NAB version number *major.minor*, changes that affect the resulting scores will increase the *major* number; e.g. adding data and labels to the corpus, or changes to the scoring algorithm. The *minor* number increases with changes under the hood of NAB that do not affect the results.

1.1 *Reporting Results*

We encourage publishing results based on NAB. Please cite the ICMLA paper if you report your own NAB results in a paper or presentation. Please also send an email (ideally with PDF attached) to nab@numenta.org as we would appreciate hearing about it.

If an NAB user wants to have the results of their DUT’s NAB runs posted in the NAB repository along with the other algorithms, the user should send an email to nab@numenta.org, attaching their final score results file, their name and/or affiliation, a link to the custom DUT source code, and any other NAB inputs provided. A link or citation to any publication would be appreciated and will be included on the scoreboard.

The submission file should be a CSV formatted as specified in Appendix C. We should be able to run your detector and produce the same results files. All results will be reviewed before posting. It is possible to have results posted without including links to the NAB inputs, but this fact will be noted on the leaderboard.

2. NAB Details

2.1 *Algorithms included in NAB*

At the release of v1.0, four different detector algorithms are included in the NAB repository. The user can experiment with these detectors and replicate the results, or create new detectors to run through NAB – please refer to the NAB entry points instructions and diagram in the wiki. More discussion on the detectors can be found in the NAB paper [1].

- *HTM-based detector*: The Numenta detector, based on Hierarchical Temporal Memory (HTM), is included in the NAB code repository. The Numenta detector doesn't need training sets, automatically builds models for any number of metrics, and performs continuous learning on streaming datasets. Running the algorithm requires our open source repo NuPIC. For more details on anomaly detection with NuPIC please refer to the NuPIC wiki and Numenta's "Science of Anomaly Detection" whitepaper.

- *Etsy Skyline detector*: Skyline is an open source, real-time anomaly detection system, built to enable passive monitoring of hundreds of thousands of metrics, without the need to configure a model/thresholds for each one. It is designed for use wherever there is a large quantity of high-resolution timeseries data which need constant monitoring. Refer to the etsy/skyline Github for more information.

- *Null detector*: assigning a constant anomaly score to each data record, this detector defines a baseline from which we normalize the algorithms' scores.

- *Random detector*: provided as a trivial detector for comparison.

Additional anomaly detection algorithms have been tested by NAB but are not included in the repo. We've run Twitter's AnomalyDetection algorithms (with a detailed example in the wiki of how to score the R algorithm on NAB) and have posted the results to the NAB scoreboard. Reasons an algorithm would not be included in NAB: the algorithm is not available open source, or the algorithm runs too slow such that it would fail in real-world applications with streaming data.

2.2 *Benchmark dataset overview*

The NAB corpus contains the streaming data that detectors use as input during a benchmark test run. The benchmark dataset consists of a number of individual data files; each data file represents a sequence of data points over time that is interesting and potentially challenging for an anomaly detector. At the v1.0 release the benchmark dataset contains 58 individual data files. Five of these data files contain no anomalies, and instead represent normal patterns of data over time; anomaly detectors should not find anomalies in these files. The remaining data files each contain one or more anomalies. The majority of the data files are taken from real world situations, and there are also a few files of simulated data, to create simple anomalies in clear conditions. The dataset is located in the repo at nab/data/, including a readme detailing the individual data files. Details on data file formatting can be found in Appendix C.

2.3 *Ground Truth Anomaly Labels*

In order to score a DUT's anomaly detections, NAB needs a reference, or **ground truth** to score against. The v1.0 corpus includes seven real-world data files for which we know the anomaly causes; i.e. the truth labels are provided for us. The ground truth anomalies for the remainder of the benchmark dataset have been calculated following three steps:

1. A number of human labelers (listed in Appendix D) have labeled the benchmark dataset. Each individual labeler read through the published labeling guidelines, and then looked through the individual data files in the benchmark dataset, with the aid of the NAB data visualization tools – see the readme for more information. They recorded the timestamps at which anomalies appeared to start. The collection of these labels for all the data files makeup the **raw labels files**, one for each labeler and data batch.
2. Each raw label file is a dictionary of key-value pairs, where the dataset name identifies a list of labeled anomaly timestamps. Across these files, the timestamps are compared to identify the ground truth anomalies, yielding one ground truth label file. To determine the true anomalies, a custom "bucket-merge" algorithm (pseudocode is given in Appendix B) is used:
    a. For each given data file (i.e. the key), the start timestamps from all the raw labels are collected into **buckets** with other start timestamps within a certain range. That is, timestamps within a narrow range of one another are bucketed together because they (very likely) identify the same anomaly. Differences in anomaly start times amongst labelers can be attributed to human error. The size of the time range is set as 1/10 the size of an anomaly window (defined in step 3 below), both preceding and following the given timestamp. The rationale is straightforward: following the underlying assumption anomalies are rare, if multiple anomaly labels fall within

the same window, they identify the same sequence of anomalous data.

  b. If a given bucket contains more timestamps (raw labels) than the agreement threshold, it is classified as a true anomaly. The standard threshold is 0.50, so *a bucket must contain timestamps from 50% or more of the labelers in order to become a true anomaly label*.
  c. Each true anomaly bucket is then **merged** into a single timestamp based on frequency. That is, the timestamp which appears most often within a bucket is chosen to represent the entire bucket. Ties are determined by which timestamp occurs first. The resulting timestamps are the ground truth labels marking the start of the true anomalies.

3. The combined labels are then converted into **anomaly windows**, indicated by a beginning and ending timestamp. The motivation behind anomaly windows is twofold. First, anomalous data often occurs over time, rather than at a single point, and thus defining anomaly windows improves the NAB scoring efficacy. Second, anomaly windows allow the DUT to not be penalized during scoring if the DUT anomaly detections are slightly before or after the ground truth. We want these windows to be large enough to allow early detection of anomalies, but not so large as to artificially boost the score of inaccurate detectors. That is, the windows give the benefit of the doubt to the detector, but not so much that false positives would count as true positives. The anomaly windows are calculated using the following algorithm:
    a. The total amount of window length in a single data file was chosen to be 10% of the data file length. For instance, if a data file contains 4000 data points, then the total amount of relaxation *shared by all anomaly windows* in the data file is 0.1*4000, or 400 data points. The 10% parameter was validated by qualitatively inspecting the

dataset, and experimenting with a range of values showed no significant affect on the resulting scores, likely due to the scaled sigmoid scoring function (described later).

b. The total window length for a data file is then divided by the total number of ground truth anomalies in the data file, and the resulting number of data points makeup each anomaly window, 1/2 before the start of the window, and 1/2 at the end of the window. For instance, if the data file above has four ground truth anomalies, then each anomaly window is 400/4, or 100 data points – i.e. 50 data points both before and after the truth anomaly data point.
c. If two ground truth anomalies are in close proximity such that they overlap, the anomalies are combined into one large window. The rationale being if multiple anomalies fall within the same window length, we assume they identify the same anomalous data.

This label-combining phase, which results in the creation of the NAB ground truth labels, doesn't happen during every NAB test run; the programs in this phase were run once before the release of NAB. The code that created the ground truth labels, the individually labeled files, and all the data files are available in the NAB repository. The output of this method is the nab/labels/combined_windows.json. This file is used by the NAB optimization and scoring phases as the ground truth.

A subset of the benchmark dataset contains anomalies with known causes and are labeled accordingly, skipping to step 3 of the above process. These files are in the data/realKnownCause/ directory.

Following the labeling scheme we visually inspect the data and ground truth windows. This assists in validating the labeling process and parameters. From the inspections we found the data file "realAWSCloudwatch/iio_us-east-1_i-a2eb1cd9_NetworkIn.csv" needed manual adjustment to accommodate an early anomaly. That is, we manually set the first ground truth window of this file such that it would not overlap the probationary period. The may change in future NAB versions.

For new hand labelers to contribute labels to NAB, please email us at nab@numenta.org. You must be approved and adhere to the NAB Labeling Instructions.

3. NAB Test Phases

The NAB test process consists of three phases:

1. **Detector phase**: The DUT processes the data files, and records the **raw anomaly score** for each data point in the data files. The raw anomaly score is a floating-point number between 0 and 1.
2. **Threshold optimization phase**: The DUT's raw anomaly scores are optimized to find the highest scoring threshold that causes a data point to be tagged as an anomaly.
3. **Scoring phase**: The DUT's detections are scored with respect to the ground truth windows. Final scores are recorded for each data file.

A user can enter the NAB test process at the beginning of any one of the three phases, as long as the user provides the correct input data format. Please see the NAB entry points details and diagram. Following is a more detailed description of each phase.

3.1 *Detection phase*

During the detector phase, the DUT processes each data file in the benchmark dataset, taking the data points in sequence order. A **probationary period** is calculated for the beginning of each data file, where no scoring is done in this time window. The period is set at 750 data instances, but for data files shorter than 5000 instances we use 15% of the file length. The minimum and maximum value of each data file is available to the detector before processing begins; in most real-world applications the minimum and maximum values are known. No other information about the data file is given, and no look-ahead is allowed. For each point in the data file, the DUT must calculate a **raw anomaly score**, which is a floating point number between 0 and 1; the closer the number is to 1, the more likely this point is to be an anomaly. These raw anomaly scores are stored, each with its appropriate timestamp, in a specific output file for each data file, in the results folder. The NAB repository includes instructions on how to do a test run with the detectors included in the repository.

- *Detection phase inputs:* The inputs into this phase are the detector (DUT) and the benchmark dataset. The user can feed an optional, custom detector by creating a subclass of the base.py class in the nab/detectors directory.
- *Detection phase outputs:* The DUT's Raw Anomaly Scores are written into files named *detectorName_dataFileName.*csv; these files are located in the nab/results/ directory. Details on the format of these CSV files can be found in Appendix C.

3.2 *Threshold optimization phase*

The outputs of the detection phase are files (one for each data file) that have floating point raw anomaly scores (between 0 and 1) for every data point in the benchmark dataset. The next step in the NAB process is to threshold these raw anomaly scores into discrete values indicating the presence or absence of an anomaly. We constrain the detectors to use a single detection threshold for the entire dataset. For convenience, NAB includes a simple hill-climbing routine for finding the optimal threshold value given the scoring rules. Then the chosen threshold – i.e. that which resulted in the optimal score – is applied to the raw anomaly scores, resulting in detection labels for each timestamp, where a binary 1 represents the presence of an anomaly, and a 0 represents the absence of an anomaly. These detection labels are written into the same output files from Phase 1. Note that the labels column in the results file are not detection labels. Refer to Appendix D for result file format details

- *Threshold optimization phase inputs:* The inputs into this phase are the DUT's raw anomaly scores, the ground truth files, and the application profile (detailed below). Note that a user can enter the NAB test run at the beginning of the threshold optimization phase (Phase 2) by supplying the files *detectorName_dataFileName.*csv, that are populated with raw anomaly scores. File formats are described in detail in Appendix C.

- *Threshold optimization phase outputs:* The highest scoring thresholds for each detector and each scoring profile are written in a dictionary in *config/thresholds.json* file. Note that the scores corresponding to optimum thresholds in *thresholds.json* are without normalization. Details of scoring phase and normalization phase are discussed in section 3.3 and 3.4.

3.3 *Scoring phase*

In the scoring phase, NAB assigns a final score to the DUT's labels in the results file; each data file is scored separately. The scoring phase uses a *scoring function* and three different *scoring weights* to calculate the DUT's final score. This is discussed in further detail in the "Scoring" section below.

- *Scoring phase inputs:* The inputs into this phase are the DUT's anomaly detection files for each data file (after the threshold optimization phase is complete), the ground truth file, and the application profile. Note that a user can enter the NAB test run at the beginning of the scoring phase

(Phase 3) by supplying the results files and an entry in the thresholds JSON. File formats are described in detail in Appendices C and D.

- *Scoring phase outputs:* The scores are stored in a file called *detectorname_profile_*scores.csv, in the NAB results directory. These are raw NAB scores without normalization.

3.4 *Normalization phase*

In the normalization phase, NAB scales the raw scores w/ a normalization scheme – detailed in the NAB paper [1]. The purpose is to convey NAB scores on a more intuitive scale, where the maximum is 100 and a zero baseline is defined by the "null" detector. Note this phase can only be run following the scoring phase.

- *Normalization phase inputs*: The inputs are the raw NAB scores from the previous phase (scoring the DUT), the ground truth labels JSON (from which we calculate the perfect score), raw NAB scoring files from the null detector.

- *Normalization phase outputs*: The NAB scores for reporting, written to nab/results/final_results.json.

4. NAB Scoring

4.1 *Scoring function*

The NAB scoring algorithm uses a scaled sigmoid scoring function to quantify how good or bad a detection is, relative to a given anomaly window. Please refer to the NAB paper [1] for detailed discussion and examples.

4.2 *Scoring weights*

The **scoring weights**, stored in the application profile (detailed below) file config/profiles.json, are used in the scoring function to customize the values of correct and incorrect anomaly detections. The values for each of the scoring weights in the default application profile is 1.0. The user can vary the scoring weights by using different application profiles, which assign values to the binary classification metrics:

- *True positive (TP)*: correctly identified
- *True negative (TN)*: correctly rejected
- *False positive (FP)*: incorrectly identified
- *False negative (FN)*: incorrectly rejected

TP weight: the theoretical maximum number of points given for each anomaly detected. The value of the TP weight in the default application profile is 1.0. If a user wants to emphasize the importance of correct anomaly detection during the benchmark run, then the TP weight metric can be increased. This will increase the amount that the score is incremented every time an anomaly is detected correctly. Accordingly, if the user wants to deemphasize the importance of correct anomaly detection during the benchmark run, then lowering the value of the TP weight will decrease the amount added to the score every time an anomaly is detected correctly.

FP weight: the theoretical maximum number of points taken away for each FP label. The value of the FP weight in the default application profile is 1.0. If a user doesn’t care as much about false positives being detected, then decreasing this value will decrement the score by a lesser amount every time an anomaly is incorrectly identified. Accordingly, if the user wants to emphasize a low FP rate, then increasing the value of FP will decrement the score by a greater amount every time an anomaly is incorrectly identified.

FN weight: the fixed amount of points taken away whenever an anomaly is not detected at all. The score is reduced by this amount once for each anomaly which is not detected at all. The value of the FN weight in the default application profile is 1.0. If a user wants to make it more important that the detector never miss a real anomaly, then increasing the FN weight will decrement the score by a greater amount when an anomaly is missed. If the user cares less about true anomalies being missed, then decreasing the value of the FN weight will decrement the score by a lesser amount when an anomaly is missed.

More details on the derivation of the scoring function and its relation to standard algorithms scoring metrics are discussed in Appendix A.

With the anomaly windows defined to be 10% of a given data file (as discussed above), true positives are limited to 10% of the data. Yet false positives can occur in 90% of the data. *That is, of all the timesteps in each data file, only 10% are eligible to be true positives (within a window), while 90% can be false positives (outside a window)*. Thus we scale the score contribution of false positives such that they have the same cumulative weight as true positives – i.e. the score contributions from false positives are divided by 9. This is reflected in the application profiles below.

4.3 *Application profiles*

As described above, the three scoring weights – TP, FP, FN – can be varied by the user for a particular NAB test run; these scoring weights are stored in the **application profiles** file config/profiles.json. Along with a default application profile (in which all the scoring weights are set to 1.0), several different example application profiles are provided in the NAB repository. Note the weights shown below incorporate the FP scaling factor described in the previous section.

Standard application profile: This application profile attributes equal weight across the scoring metrics. In the NAB profiles.json file this profile is named "standard". The scoring weights in this profile are set as follows:

TP = 1
FP = 1
FN = .11

Alternate application profile #1: This application rewards a detector that has a very low FP rate; they would rather trade off missing a few true anomalies rather than getting multiple false positives. In the NAB profiles.json file this profile is named "reward low fp rate". The scoring weights in this profile are set as follows:

TP = 1 [give full credit for properly detected anomalies]
FP = 0.22 [decrement the score more for any false positives]
FN = .5 [decrement the score less for any missed anomalies]

Alternate application profile #2: This application rewards a detector that doesn't miss any true anomalies; they would rather trade off a few false positives than miss any true anomalies. In the NAB profiles.json file this profile is named "reward low fn rate". The scoring weights in this profile are set as follows:

TP = 1 [give full credit for properly detected anomalies]
FP = .055 [decrement the score less for any false positives]
FN = 2 [decrement the score more for any missed anomalies]

5. NAB Results

Please see the scoreboard for the current NAB results.

5.1 Interpreting Results

The final reported NAB scores are normalized such that the maximum is 100, and a “null” detector defines the baseline of 0. Details to the normalization procedure can be found in the NAB paper [1]. A couple important points to keep in mind when interpreting the results:

- The baseline score of 0 is determined by the null detector, which predicts a constant anomaly score across all data points. Note however this is not the minimum; negative scores are possible.
- The resulting NAB scores are relative to each other, within a common NAB version number; do not compare scores from version to version.

Negative contributions to score are from FPs and FNs, while TPs increase the score. Thus a *negative raw score implies the false detections by a given DUT outweighed the true detections*. Yet a DUT with a poor NAB score can still be useful, that is, as long as it beats the “random” detector. This detector simply assigns a random value between 0 and 1 for each data point in the benchmark dataset, which results in poor NAB scores.

Although one may only have interest in a single scoring profile, it is important to consider the results for several different profiles. For instance, consider a detection algorithm that makes very few anomaly detections – i.e. only flagging the most apparent anomalies. This DUT would score relatively well on the NAB "Standard" and "Reward low FP" profiles because it limits FPs while still getting a few TPs, yet this is not necessarily a quality anomaly detection algorithm. This DUT would score poorly in the NAB "Reward low FN" profile, which weights false negatives as more important, because it fails to detect many anomalies.

5.2 Visualizing Results

In the scripts/ directory we include a data visualizer (useful in labeling the files for anomalies) and a script to generate plots with plotly (this requires a (free) API key).

5.3 Score leaderboard

The wiki contains the NAB Scoreboard. Please note the scores should not be compared between NAB versions because revisions may affect the score output. Specifically, the addition of new data files to the benchmark dataset will result in different scores; additional data will prompt an update to the version number.

6. References

[1] A. Lavin and S. Ahmad, "Evaluating Real-time Anomaly Detection Algorithms – the Numenta Anomaly Benchmark," in *14th International Conference on Machine Learning and Applications (IEEE ICMLA'15)*, 2015.

7. Glossary

- Application profile: A set of variables that can be set by the user for any particular NAB test run. These are weights to give values to the relative importance of TP, FP, and FN detections. The standard scoring profile is used for runs where the user doesn't set these variables.
- Benchmark dataset/corpus: Collection of 58 time series data files (as of v1.0), the fixed dataset used to test the anomaly detection algorithms.
- Combined labeling file: ground truth labels, as a combination of the raw labeling files per the "bucket-merge" labeling algorithm. The labels show the presence of an anomaly in the benchmark dataset that the DUT will be scored against.
- Data file: Any one of a series of time sequence data in a specific format, chosen to be part of the benchmark dataset.
- Data visualizer: visualizer for the data files in the benchmark dataset, typically used by labelers. The plotting script is better for viewing the data and results.
- Detections phase: First phase of a NAB test run, where the DUT identifies anomalies in the benchmark dataset.
- Detector under test (DUT): Anomaly detector that is being tested by the benchmark.
- Ground truth (true) anomaly windows: windows of data in the benchmark dataset that are determined to be anomalous by the labelers.
- Labeler: Person who labels the data files in the benchmark dataset by hand, and provides a labels file.
- Normalization phase: Final phase of a NAB test run, where the raw scores from the scoring phase are normalized. The reported NAB scores are normalized.
- Raw anomaly score: Floating point value between 0 and 1 that is the output of the DUT for each data point in the benchmark dataset. A score closer to 1 indicates that the DUT has scored this more likely to be an anomaly.
- Raw labeling file: Human generated labels for anomalies in the benchmark test file, used to compute the ground truth labels in the combined labeling file.

- Raw NAB score: for a given detector, the sum of the raw scores across the data files. The reported NAB score is a normalized version of this score.
- Results file: File used by NAB to write the results of a benchmark run. NAB stores anomaly scores, detections and scores into this file at various points in the benchmark test run. This file can also be used as input if the user wants to enter the benchmark test mid-run.
- Scoring phase: Third phase of a NAB test run, where the DUT detections are scored and results files are written out.
- Threshold optimization phase: Second phase of a NAB test run, where the anomaly threshold is optimized to yield an optimal NAB score for the DUT.
- Window: a length of time for which a detected anomaly is a true positive. The window size is standardized for each data file, and the windows are located such that the true anomaly timestamp is at the center.

8. Appendices

A. Scoring algorithm
B. Label combining algorithm
C. Data format
D. Results format
E. Human labelers

Appendix A: Scoring algorithms

Sigmoid

The sigmoid, also called the sigmoidal curve or logistic function, is $S(t) = \frac{1}{1+e^{-t}}$.

It's commonly used as a differentiable step function to incorporate non-linearity into a system. We use the sigmoid in NAB scoring because it is continuous and real-valued between 0 and 1 (exclusive).

The scaled sigmoid in NAB scoring is defined to reward TPs within the true anomaly window as follows:

- A relative position of -3.0 is the far left edge of the anomaly window and corresponds to a 2*sigmoid(15) - 1.0 = 0.999999. This is the earliest TP possible for a given window; an earlier detection is an FP.
- A relative position of -0.5 reflects a slightly later detection and corresponds to a score of 2*sigmoid(0.5*5) - 1.0 = 0.84828.
- A relative position of 0.0 is the right edge of the window and corresponds to a score of 2*sigmoid(0) - 1 = 0.0. Any detection beyond this point are scored as a FP.
- Relative positions > 0 correspond to FPs increasingly far away from the right edge of the window. A relative position of 1.0 is past the back edge of the window and corresponds to a score of 2*sigmoid(-5) - 1.0 = -0.98661

Additional notes on scoring metrics

An anomaly detector is a binary classifier, where each step in time is labeled anomalous or not. Metrics to evaluate such classifiers arise from the resulting counts of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN).

		Ground Truth	
		TRUE	FALSE
Classifier Result	POSITIVE	TP	FP
	NEGATIVE	FN	TN

Appendix B: Label combining algorithm

Below is pseudocode for the “bucket-merge” method of combining raw user labels into the ground truth anomaly labels. The code can be found in nab/labeler.py.

```
for each datafile in the dataset
    labels = a list of the users' raw anomaly labels
    buffer = half of the estimated anomaly window size for this dataset
    for all timestamps in t
        if they're within the buffer of each other, then place in a bucket
    raw anomalies = the set of buckets
    for each bucket,
        if the size of the bucket exceeds the agreement threshold, then this bucket identifies a true anomaly
            merge the bucket to one timestamp (the most frequent in the bucket, w/ a tie decided by the earliest timestamp of the most frequent)
    set windows with true anomaly timestamps at center
```

Appendix C: Data format

The data files in the benchmark dataset all follow the format below:

- CSV format
- One header row
- Fields for "timestamp" and "value"
    - o "timestamp"
        - ▪ the time of the end of the metric collection window
        - ▪ e.g 2014-04-01 12:00:00.000000 is values collected

        - between 2014-04-01 11:55:00.000000 and 2014-04-01 12:00:00.000000
    - "value"
        - the collected metric, e.g. CPU utilization percent
        - either floats or integers (converted to floats internally)
- Records must be in chronological order
- Records must be continuous such that there are no missing time steps
- For NAB v0.8 the minimum file size is 1000 data instances; this will increase in future versions when the corpus is expanded

Appendix D: Results format

The detector under test will output a results file for each data file in the corpus , following the format below:

- CSV format
- One header row
- The header row MUST have the following fields
    - "timestamp”
        - in format “YYYY-MM-DD HH:MM:SS.s”, e.g. 2014-04-01 00:00:00.000000
    - "value": same format as input data files
    - “anomaly_score”: the DUT’s calculated anomaly likelihood values
        - float [0,1]
    - "label": ground truth for the class of a record, values are 0, 0.5, or 1
        - 0: this record is known to be non-anomalous
        - 0.5: this record's class is ambiguous
        - 1: this record is known to be anomalous
- Header rows may have other fields as well; these are ignored by the NAB

scorer. For instance, Numenta detectors result files have an additional field of “raw_score” which represents the score returned by HTM used to calculate the “anomaly_score”.

- Records must be in chronological order
- Records must be continuous such that there are no missing time steps

For each detector and profile, the overall scores for all files are written in *DetectorName_profileName.csv.* The files follow the format below:

- CSV with one header row, a row for each individual results data file, and totals row.
- The header row has the following fields:
    - “Score”: for a given results file, this is the sum of the scores for each data point; i.e. the sum of the S(t)_profileName column in that results file.
    - “TP”: number of true positive detections—data points in the results file with anomaly_score greater than the detector’s threshold, and label equal to 1.
    - “TN”: number of true negative data points—anomaly_score less than the detector’s threshold, and label equal to 0.
    - “FP”: number of false positive detections—anomaly_score greater than the detector’s threshold, and label equal to 0.
    - “FN”: number of false negative data points—anomaly_score less than the detector’s threshold, and label equal to 1.
    - “Total_count”: total number of data points in that results file.
- The “Totals” row sums the columns’ entries.

The final normalized scores for all NAB are written in *final_results*.json. These are the scores shown in NAB scoreboard.

Appendix D: Human labelers

Please note this draft reflects NAB v1.0.

Below is a list of the human labelers and their emails.

- Subutai Ahmad – sahmad@numenta.com
- Celeste Baranski – cbaranski@numenta.com
- Alexander Lavin – alavin@numenta.com