Preprint of book chapter:
CAOS-R: Character-based barcoding

This is a preprint of the following chapter: Tjard Bergmann, CAOS-R: Character-based barcoding, published in DNA Barcoding - Methods and Protocols, edited by Robert DeSalle, 2024, publisher Humana Press reproduced with permission of Springer Science. The final authenticated version is available online at: http://dx.doi.org/10.1007/978-1-0716-3581-0.

CAOS-R: Character-based barcoding

Author: Tjard Bergmann, *University of Veterinary Medicine Hannover, Hannover, Germany*

Contact: tjard.bergmann@tiho-hannover.de, *Institute of Ecology and Evolution*

Contents

CAOS-R: Character-based barcoding - Preprint-Version

1 Abstract

CAOS-Barcoding is a culmination of traditional taxonomy and modern DNA barcoding. CAOS identifies taxa by diagnostic characters as is done in traditional taxonomy and produces an identification matrix for taxon discrimination similar to DNA barcoding distance matrices. Here, I describe how to setup the CAOS-Barcoder and CAOS-Classifier software, which input data is needed and how to interpret the output data. With the CAOS-Barcoder single marker or concatenated data can be processed into diagnostic barcodes for taxon discrimination. The CAOS-Classifier can use the diagnostic barcodes for specimen identification.

2 Introduction

2.1 A history of CAOS

CAOS (Character Attribute Organization System) is a barcoding procedure which is highly inspired by the delimitation of phylogenetic species as described in Davis and Nixon in 1992. Davis, Nixon [1] described that phylogenetic species can be delimited by a procedure (population aggregation analysis) that involves a search for shared features within a population which are absent/different in other populations and therefore can be used as means of identification and differentiation. CAOS barcoding was developed on this principle and first described in a collaboration between the University of Columbia and the American Museum of Natural History in New York in 2002 [2, 3]. The CAOS algorithm in short compares attribute data (DNA, expression patterns, morphological data or other; Fig.1a-d) between two or more declared groups and searches for patterns that are unique for each group.

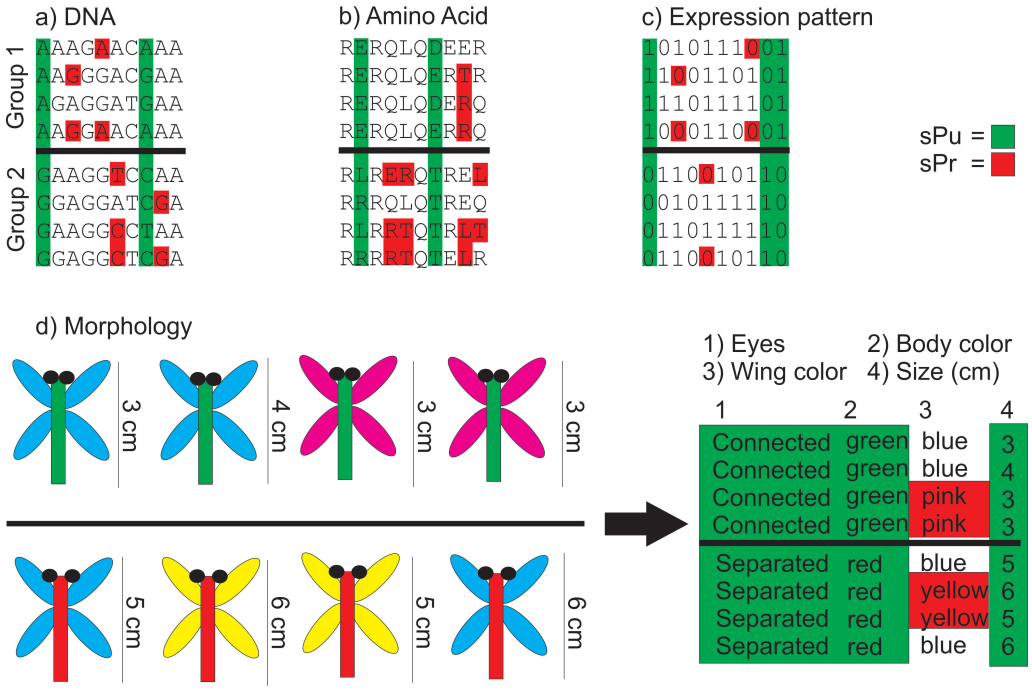

Figure 1: Different examples of character-based diagnostics for two taxa groups. Simple pure diagnostic characters (sPu) are labeled green and simple private diagnostic characters are labeled red. (a) DNA-based diagnostics. (b) Amino acid-based diagnostics. (c) Expression pattern-based diagnostics. (d) Morphology-based diagnostics. Here, four different morphological attributes are compared.

Identified character attributes can belong to one of two categories: 1) Simple pure characters (sPu), which are diagnostics shared by all members of a group at one specific location. 2) Simple private characters (sPr), which are diagnostics unique within a group but not shared by all members.

In the first publication using the CAOS algorithm Sarkar et al. [2] showed that CAOS can be used for highly accurate and robust cancer class prediction using microarray expression state patterns of genes (0 = decreased expression; 1 = increased expression) as character attributes. In the second publication, Sarkar et al. [3], demonstrated how phylogenetically characteristic amino acid states in homeoboxes can be identified by CAOS and used for high accuracy classification of homeobox groups.

Only six years later, in 2008, CAOS was integrated into a user-friendlier and DNA centered perl script called p-gnome. It was supplemented by a specimen classifier called p-elf [4].

A performance test on dragonflies and damselflies showed that with p-gnome character-based DNA barcoding was possible allowing discrimination of genera, species and populations in Odonata [5]. In 2011 as part of a collaboration between the American Museum of Natural History in New York, the University of Vermont and the University of Veterinary Medicine Hannover p-gnome was upgraded into the CAOS Workbench website platform. Here, p-gnome was split into the CAOS-Analyzer creating the barcodes and the CAOS-Barcoder transforming the barcodes into human readable code. The p-elf classifier was replaced by a more sufficient algorithm called CAOS-Classifier. The platform was tested successfully on turtles by Reid et al. [6]. As part of the publication the distance-based barcoding was compared to the character-based approach showing that complementing distance-based barcoding with character-based methods for identifying diagnostic sets of nucleotides provided better resolution in several cases where distance-based methods failed to distinguish species. Further tests of character-based barcoding potential were performed on the species-rich odonates in Bergmann et al. [7] and on a highly cryptic ant family in Paknia et al. [8]. Both publications showed higher specimen identification and taxon differentiation success based on diagnostic characters compared to the distance-based approach. When analyzing the unique avifauna of New Zealand (180 species) using the COI barcode region it was shown that of the 25 groups that were problematic to distinguish using neighbour-joining trees and/or distance methods, species within 13 groups could be correctly identified using diagnostic characters in CAOS [9]. While CAOS has been rarely used in recent years due to disconnected CAOS source websites (p-gnome, CAOS-Workbench), a new R coded CAOS-R has been developed in 2022 and is available on the more sustainable github platform (https://github.com/M0rph3u2x/CAOS-R).

3 Materials

3.1 CAOS-Barcoder

To perform character-based barcoding with CAOS, download the software from github (https://github.com/M0rph3u2x/-CAOS-R). The latest CAOS version is written in R (CAOS-R) and it is necessary to install R (version 4.2.2) on the computer (https://cran.r-project.org). Before running CAOS-Barcoder/Classifier, just once, run the script 'install_packages.R', this will install all necessary functions that are not available within the R core installation (see Note 1).

Always run CAOS-Barcoder/Classifier in the Editor RStudio (version 2021.09.0+351 or higher). All CAOS-Barcoder/-Classifier input data must be stored in the 'input_barcoder' or 'input_classifier' folder (same location as the CAOS-R scripts). Multiple projects can be processed procedurally if placed in the 'input_barcoder' folder. A CAOS-Barcoder project dataset consists of four components: 1) FASTA file, 2) Tree file, 3) Marker file and 4) Marker Patterns file. All files must share the same project specific label as first part of the name (e.g. if the project name is "data1" the components must be labelled: data1.fas, data1.tre, data1_Markers.txt, data1_Marker_Patterns.txt).

3.1.1 FASTA

The FASTA file format is very common in phylogenetics. It is a simplistic text format with a data descriptor and a core data fragment. The data descriptor (name of the specimen, sample id, etc...) is listed in the first row started by a ">" symbol (e.g. >cow_id455, see Note 2). In the next row or rows, the core data is listed (e.g. for a genetic marker: AAAACGCGCTCCTT, see Note 3). The next dataset is recognized when a ">" symbol is used as first symbol in a line to describe the identity of the next dataset (Fig. 2 A1 left).

With CAOS-R any kind of data can be used as long as the data is formatted according to the ruleset: If, for example, morphological data is used that is coded by multiple characters, each string must be separated by a "$" symbol (Fig. 2 A1 right, see Note 4).

It is possible to process concatenated data. To use a combination of multiple markers each marker data must be separated by a "&" symbol in the FASTA file. It is important that each marker subset has the same number of characters for each specimen within the marker dataset. If that is not the case missing data must be replaced with a token symbol (e.g. "?" or "-"; Fig.2A1 bottom).

3.1.2 Phylogenetic Tree

CAOS-R is dependent on a hierarchical organized guide. The guide will tell CAOS-R in which order diagnostic data of different taxa subgroups are compared against each other. A phylogenetic tree format is the ideal guide as it is easily built and understood for any form of data. As there are many popular tree formats available, please adhere to the newick tree architecture (Fig.2A2 top, see Note 5 & 6). The hierarchical structure provided by the tree file (*.tre) will be applied to all marker datasets. Trees with polytomies are supported by CAOS-R.

A) CAOS-Barcoder Input

1) FASTA

```
>cow_id455
AAAACGCGCTCCTT
AAAACGCGCTCCTT
>cow_id123
ATACCGCGCTCCTA
AAAACGCGCTCCTT
```

```
>Dragonfly_Ap12
yellow$green$5cm$eyes_connected
>Damselfly_Ap12
red$green$3cm$eyes_separated
```

```
>cow_id123
ATACCGCGCTCC--&AAAACGCG?TCCTT
```

2) Tree

```
Correct:
(((Mcaerulatus1_Panama,Mcaerulatus2_Panama),
Mcaerulatus3_Panama),Mcaerulatus4_Panama);
```

```
Wrong:
(((1,2),3),4);
1 Mcaerulatus1_Panama
2 Mcaerulatus2_Panama
3 Mcaerulatus3_Panama
4 Mcaerulatus4_Panama
```

3) Marker

```
16S&ND1&Color
```

4) Marker Patterns

```
ACGT?-&A$C$G$T$?$-&green$blue$yellow$red$brown$black
```

B) CAOS-Barcoder Output

1) CA_Table

Species Names	**68**	**112**	**130**	**143**	**151**
Mcaerulatus1_Panama	C	A	A	G	C
Mcaerulatus10_Panama	C	G	A	A	C
Mcaerulatus12_CostaRicaNord	C	A	A	G	T
Mcaerulatus13_CostaRicaNord	C	A	A	G	T
Mcaerulatus24_Mexico	T	A	C	T	A
Mcaerulatus25_Mexico	T	A	A	T	A

2) CA_Overview

Diagnostic Positions	Branch Positions	All Characters	Unique Characters	Diagnostic Characters	Simple Pure C.	Simple Private C.
68	1|2|3	C |C|T	C |C|T	- |- |T	- |- |T	- |- |-
112	1|2|3	A&G|A|A	G |- |-	G|- |-	- |- |-	G|- |-
130	1|2|3	A |A|A&C	- |- |C	- |- |C	- |- |-	- |- |C
143	1|2|3	G&A|G|T	G&A|G|T	A |- |T	- |- |T	A |- |-
151	1|2|3	C |T|A	C |T|A	C |T|A	C|T|A	- |- |-

Figure 2: Examples of the necessary input files and the resulting output files of the CAOS-Barcoder. (a) The CAOS-Barcoder needs four different input files to identify diagnostic positions in a given dataset. A FASTA file (A1), a phylogenetic tree file (A2), a marker file (A3) listing the names of the used markers, and a marker patterns file (A4) listing all legit character types that can be observed for a marker. (b) The CAOS-Barcoder creates multiple outputs. The most important ones are the CA_Table (CA character attribute) and the CA_Overview. The CA_Table (B1) lists all specimens within a node and their characters at diagnostic positions.

3.1.3 Markers

As CAOS-R can process more than one marker per reference dataset it is important to let CAOS-R know the names of each provided marker. This is done via the "Markers.txt" file. In the file, list all used markers in the same order as used in the concatenated datasets and separate the marker names with an "&" symbol (Fig.2A3, see Note 6).

3.1.4 Marker Patterns

For each marker the user can define which set of characters or character strings are allowed as diagnostic units. The user defined diagnostics are listed within the "Marker_Patterns.txt" file (Fig.2A4). If all diagnostics for a marker consists of single characters they can be listed next to each other (e.g. DNA: ACGT). If diagnostics are character strings each diagnostic string must be separated by a "$" symbol (e.g. color: red$green$blue). The diagnostics for one marker cannot be a combination of single and multi characters except if separated by a "$" symbol (e.g.: A$blue$G). Diagnostic units that are not listed in the "Marker_Patterns.txt" file are ignored by CAOS-R when identifying diagnostic attributes (see Note 6).

3.2 CAOS-Classifier

Once, a reference dataset has been barcoded with the CAOS-Barcoder, the barcodes can be used to identify query data with unknown origin using the software "CAOS_Classifier.R". The Classifier needs four components to process the data. 1) A query FASTA file containing all specimen sample data that needs to be identified. 2) The reference barcode database (created by the CAOS_Barcoder). 3) A marker priority file (listing the priority given to each marker). 4) A reference FASTA file containing the aligned reference data (the same FASTA as used in the CAOS-Barcoder). All input data must be placed in a folder named "input_classifier" (see Note 7).

3.2.1 Query FASTA

The query FASTA file contains the information of specimen that need to be identified. The structure must be set up in the same order as used in the reference FASTA file. It is important that each marker data in the file has been aligned to the reference data and contains the same number of characters as used in the reference or has data gaps masked by replacement symbols such as "-" or "?". Within the folder "input_classifier" store the query FASTA file inside the subfolder "query".

3.2.2 Reference barcode database

The reference barcode database is the output from the CAOS_Barcoder and should be placed within the folder "input_classifier" inside the subfolder "reference". The diagnostic barcode data inside the database (specifically the CA_Overview_Node tables and the CA_Taxa table) is used to classify the query specimen.

3.2.3 Marker priority file

The marker priority file allows the user to rank the markers based on their diagnostic value. If some markers are known to be more effective in distinguishing taxa than other used markers, they can be ranked (e.g. 16S>ND1). Ranking the markers has a big impact on the classification process as lower ranked markers will get a penalty when comparing diagnostic matches between the used markers (for more details about the ranking system read method section 3.2 CAOS-Classifier). When all markers are ranked the same, each marker name in the file must be separated by a "&" or "=" symbol. So, for example, "16S=ND1>Color" means that "16S" and "ND1" are equally weighted while the marker "Color" will get a penalty. In contrast, "16S>ND1>Color" means that "ND1" will get a minor and "Color" a higher penalty (see Note 8). Store the file in "input_classifier"->"reference"->project folder (see Note 7).

3.2.4 Reference Fasta file

The reference FASTA file is the same file as used in the CAOS-Barcoder. The FASTA file is necessary for aligning the query and best hit reference after matching the query by the diagnostic barcodes. Store the file in "input_classifier"->"reference"->project folder (see Note 7).

4 Methods

4.1 CAOS-Barcoder

The CAOS-Barcoder will sort and compare all reference data based on the guiding tree. The software will start at the root of the tree and compare each branch of the node against each other in search for simple pure or private diagnostics.

It will save identified diagnostics in two table formats “CA_Table” and “CA_Overview”. Each node in the tree will get a numbered id label. An overview picture called “Tree_Preview.png” will be created in the output folder to facilitate locating node specific barcodes (Fig.3).

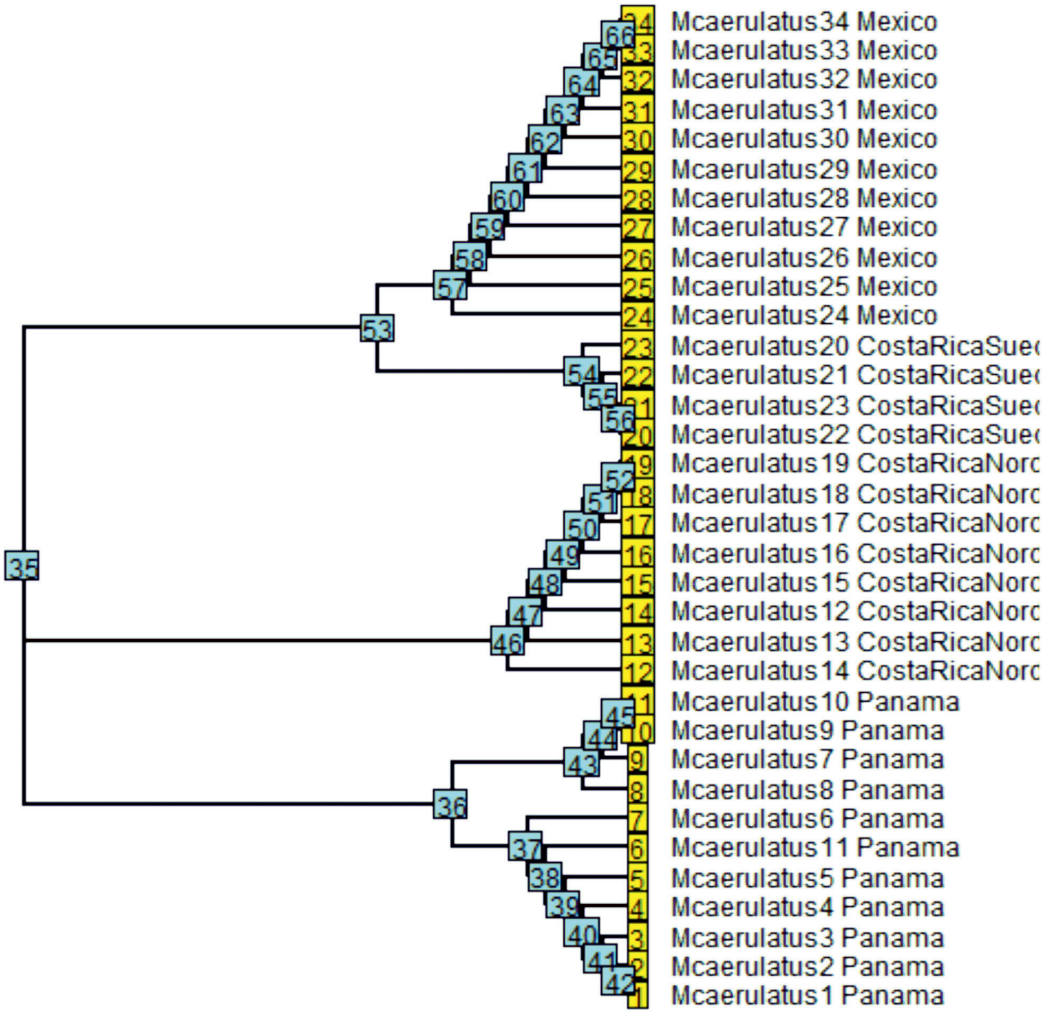

Figure 3: The CAOS-Barcoder creates an overview tree with numbered nodes. This tree will help users correlate the CA_Tables to each node in the tree as both use the same node ids.

In the CA_Table all diagnostic positions will be listed in the top row followed by all specimen and specimen specific data for the diagnostic positions. Specimens from different branches are separated by empty rows (Fig.2B1).

In the CA_Overview diagnostics of different quality are listed. Here, the first column lists the diagnostic position. The second column lists the branch id for each diagnostic in the node (e.g., 1 = branch 1). The following columns show diagnostics of different diagnostic value (each branch specific set of diagnostic data is separated by “|”. The column “All characters” lists all characters observed in each branch at this position. The column “Unique characters” only shows characters that are not observed in all branches but can be shared by multiple branches and are therefore not necessarily branch specific. The column “Diagnostic characters” only highlights characters that are unique to each branch. The column “SimplePure Characters” only shows diagnostics that are unique to one branch and shared by all taxa in the branch, while in the column “SimplePrivate Characters” unique diagnostics are listed that are not shared by all members of the branch (Fig.2B2).

In another table called “CA_Taxa_All” the node positions with barcode content, the branch ids and taxa names of each branch are listed. This file is important for the CAOS_Classifier to connect the diagnostics to the reference taxa names.

Each of the described tables is created for every one of the applied markers. All markers get their own output subfolder and are stored in the project labelled main folder within “output”.

4.2 CAOS-Classifier

The CAOS-Classifier reads in all CA_Overview_Node tables of the loaded reference database and sorts them by their node id. In the next step it compares the reference diagnostics of each marker and branch against the query dataset. Matching characters with simple pure diagnostics will score two points, a match with simple private diagnostics one point and a match with unique characters (if no sPu and sPr were detected) will award half a point. After screening all diagnostic positions, the hit score is compared between all markers and branches. The marker and branch with the highest score will be followed to the next node. Here, the branch diagnostics for all markers will be screened for the best match with the query. This process of comparing, scoring and following the highest hit score will be repeated until no further resolution is possible by diagnostics and taxa identification. At this point, using the "CA_Taxa" table, all remaining reference taxa will be listed as best hits. Several output files are generated during the classification process. A "best_hits" table will list the processed query names and best hit identifications (Fig.4a).

CAOS-Classifier Output

a) Best Hits

Query	Hits
Q1_Mcaerulatus_Panama	Mcaerulatus1_Panama,Mcaerulatus2_Panama

b) Hit Score

Marker	Node	Raw_Score	Priority_Score
16S	35	14.5|6.5|4	14|6|4
Nd1	35	14.5|6.5|4	14|6|4
Color	35	6.0| 0|0	3|0|0

c) Score Overview

Marker	Node	CA_Position	sPu_CAs	sPr_CAs	Unique_CA	Query_CA	Score
16S	35	68	- |- |T	- |- |-	C|C|T	C	0.5|0.5|0
16S	35	112	- |- |-	G|- |-	G|- |-	A	0.5|0.5|0
16S	35	130	- |- |-	- |- |C	- |- |C	A	0.5|0.5|0
16S	35	143	G|A|T	- |- |-	G|A|T	G	2.5|0.5|0

Figure 4: A description of the most informative CAOS-Classifier outputs. (a) The best hits table will show the closest matches between reference data and a query file. (b) The hit score will provide an overview of how well each marker supported the identification of a query file at each node in the tree (the scores for each branch are separated by a "|" symbol). (c) The score overview provides a detailed history of the scores each marker, node, and diagnostic position accumulated during the classification process.

For each query a query name labeled output folder will be created containing hit score tables, score overview tables and alignment files. The hit score tables will list the scores of matching diagnostics for each diagnostic node (Fig.4b). The raw score column will show the unbiased diagnostic score, while the priority score will list scores based on the marker priority setting. Here, the priority setting is 16S=ND1>Color, which the program translates into the rankings 1,1,2. The priority score is created by dividing the raw score by the ranking number.

The "score_overview" table provides a complete track of scores given for each marker, node and diagnostic position. Therefore, this table helps users to understand how the closest match between query and reference was identified (Fig.4c).

In this example the query shares a unique CA at position 68 with the first and second branch, scoring them 0.5 points. No match is identified for CA position 112 and 130 so the score stays at 0.5. At position 143 a sPu diagnostic is shared between the first branch and the query so 2 points are added to the first branch score.

For each identified reference best hit an alignment text file is created listing full sequence alignments between the query and the detected best hit.

5 Notes

1. CAOS-R was build to be run within the RStudio Editor (https://posit.co) which offers a great overview of the loaded code, data and ongoing processes.
2. The id labels will be edited by CAOS-R if they contain empty spaces (e.g. “>Sample 1 ND1 Germany” turns into “>Sample_1_ND1_Germany”). This procedure prevents errors when processing the data.
3. The core data can be presented in a single row or multiple rows, for example, truncated to 50 symbols per line.
4. An R script for automated conversion of morphological data from XLSX to FASTA format and other tools to facilitate working with CAOS-R are available on github (https://github.com/M0rph3u2x/CAOS-R).
5. The names in the described tree must be identical to the names used within the FASTA file. The program is case sensitive so “DATA1”, “Data1” and “data1” will be interpreted as three different taxonomic units. Names within the tree file should not be replaced with numbers (Fig. 2 A2 bottom).
6. It is important to press “Enter” at the end of the last entry line or else a formatting error might occur when the file is loaded into CAOS.
7. Within the “input_classifier” folder store the query FASTA file inside the subfolder “query”. Copy the CAOS-Barcoder project output inside the subfolder “reference”. Store the reference FASTA file alignment and the Marker_-Priority.txt file inside the project folder. If still unsure how to place the input files correctly, investigate the provided input data example on https://github.com/M0rph3u2x/CAOS-R.
8. It is not allowed to use the “<” symbol. Therefore, the definition “16S<ND1<Color” will cause an error. Re-ordering the markers in the “Marker_Priority.txt” file (e.g. Color>ND1>16S) will also cause an error in the current version of the CAOS-Classifier. Instead, if user assigned priority is required connect the marker names by “:” (16S:ND1:Color) and assign priority values in a second row (e.g. “5:1:3”; 1=highest priority).

6 References

1. Davis JI, Nixon KC. Populations, Genetic Variation, and the Delimitation of Phylogenetic Species. Systematic Biology. 1992;41(4):421-35. doi: 10.1093/sysbio/41.4.421.
2. Sarkar IN, Planet PJ, Bael TE, Stanley SE, Siddall M, DeSalle R, et al. Characteristic attributes in cancer microarrays. Journal of Biomedical Informatics. 2002;35(2):111-22. doi: 10.1016/S1532-0464(02)00504-X.
3. Sarkar IN, Thornton JW, Planet PJ, Figurski DH, Schierwater B, DeSalle R. An automated phylogenetic key for classifying homeoboxes. Molecular Phylogenetics and Evolution. 2002;24(3):388-99. doi: 10.1016/S1055-7903(02)00259-2.
4. Sarkar IN, Planet PJ, DeSalle R. caos software for use in character-based DNA barcoding. Molecular Ecology Resources. 2008;8(6):1256-9. doi: 10.1111/j.1755-0998.2008.02235.x.
5. Rach J, DeSalle R, Sarkar IN, Schierwater B, Hadrys H. Character-based DNA barcoding allows discrimination of genera, species and populations in Odonata. Proceedings of the Royal Society B: Biological Sciences. 2008;275(1632):237-47. doi: 10.1098/rspb.2007.1290.
6. Reid BN, Le M, McCord WP, Iverson JB, Georges A, Bergmann T, et al. Comparing and combining distance-based and character-based approaches for barcoding turtles. Mol Ecol Resour. 2011;11(6):956-67. doi: 10.1111/j.1755-0998.2011.03032.x.
7. Bergmann T, Rach J, Damm S, DeSalle R, Schierwater B, Hadrys H. The potential of distance-based thresholds and character-based DNA barcoding for defining problematic taxonomic entities by CO1 and ND1. Molecular Ecology Resources. 2013;13(6):1069-81. doi: 10.1111/1755-0998.12125.
8. Paknia O, Bergmann T, Hadrys H. Some ‘ant’swers: Application of a layered barcode approach to problems in ant taxonomy. Molecular Ecology Resources. 2015;15(6):1262-74. doi: 10.1111/1755-0998.12395.
9. Tizard J, Patel S, Waugh J, Tavares E, Bergmann T, Gill B, et al. DNA barcoding a unique avifauna: an important tool for evolution, systematics and conservation. BMC Evolutionary Biology. 2019;19(1):52. doi: 10.1186/s12862-019-1346-y.