2017-06-01

The OGCleaner: Detecting False-Positive Sequence Homology

Masaki Stanley Fujimoto
Brigham Young University

The OGCleaner: Detecting False-Positive Sequence Homology

Masaki Stanley Fujimoto

A thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Master of Science

Mark J. Clement, Chair
Quinn Snell
Christophe Giraud-Carrier

Department of Computer Science

Brigham Young University

ABSTRACT

The OGCleaner: Detecting False-Positive Sequence Homology

Masaki Stanley Fujimoto
Department of Computer Science, BYU
Master of Science

Within bioinformatics, phylogenetics is the study of the evolutionary relationships between different species and organisms. The genetic revolution has caused an explosion in the amount of raw genomic information that is available to scientists for study. While there has been an explosion in available data, analysis methods have lagged behind.

A key task in phylogenetics is identifying homology clusters. Current methods rely on using heuristics based on pairwise sequence comparison to identify homology clusters. We propose the Orthology Group Cleaner (the OGCleaner) as a method to evaluate cluster level verification of putative homology clusters in order to create higher quality phylogenetic tree reconstruction.

Keywords: machine learning, orthology clusters, phylogenetics

ACKNOWLEDGMENTS

I would like to thank the love and support provided by so many. My wife, Madison. My parents, Andy and Yoshie. My siblings. My committee members: Dr. Mark Clement, Dr. Quinn Snell and Dr. Christophe Giraud-Carrier. My great friend and mentor Paul Bodily. I would also like to thank the help that I received from friends and colleagues: Anton Suvorov, Cole Lyman, Nozomu Okuda, Nick Jensen, Michael Watkins, Shayla Draper, Christopher Tensemeyer, Aaron Dennis, Dr. Rogan Carr, Lawrence Ripsher, Jared Price, Dr. Seth Bybee, and Dr. Perry Ridge.

Without all of you, this would not have been possible, thank you.

Contents

1 Background

In this work, we present the Orthology Group Cleaner (the OGCleaner) as a method for filtering false-positive homology clusters used during phylogenetic tree reconstruction published in BMC Bioinformatics [12] and released as an applicatipn in Bioinformatics [13].

One of the most fundamental questions of modern comparative evolutionary phylogenomics is to identify common (homologous) genes that originated through complex biological mechanisms such as speciation, multiple gene losses/gains, horizontal gene transfers, deep coalescence, etc. [23]. When homologous sequences are identified, they are usually grouped and aligned together to form clusters. Homologous DNA (and those translated to amino acids) sequences can be further subdivided into two major classes: orthologs and paralogs. Orthologs are defined as homologous genes in different species that arose due to speciation events, whereas paralogs have evolved from gene duplications. Moreover, orthologous genes are more likely to exhibit a similar tempo and mode of evolution, thus preserving overall sequence composition and physiological function. Paralogs, instead, tend to follow different evolutionary trajectories leading to subfunctionalization, neofunctionalization or both [14]. Nevertheless this phenomenon, called the ortholog conjecture, is still debatable [8] and requires additional validation since it has been shown that even between closely related species some orthologs can diverge such that they eventually loose common functionality.

The accurate detection of sequence homology and subsequent binning into aforementioned classes is essential for robust reconstruction of evolutionary histories in the form of phylogenetic trees [7]. To date, numerous computational algorithms and statistical methods have been developed to perform orthology/paralogy assignments for genic sequences (for review see [25]). Methodologically these approaches employ heuristic-based or evidence (phylogenetic tree)-based identification strategies, which produces varying frequencies of false-positive or negative results. The majority of heuristic algorithms rely on the principle of Reciprocal Best Hit (RBH, [25]) where BLAST [3] hit scores (e-values) approximate evolutionary similarity between two biological sequences. Further algorithmic augmentations of those heuristics, for instance Markov graph clustering (unsupervised

learning) [28], enables the definition of orthologous/paralogous clusters from multiple pairwise comparisons. Despite their relatively low computational complexity, these algorithms have been shown to overestimate the number of putative homologies (i.e., higher rates of false-positive detection compared to evidence-based methods [5]).

In this current era of next-generation sequence data researchers have gained access to tremendous amounts of "omic" data, including for non-model organisms. Phylogenetic information, including species trees, is very limited, unreliable and/or completely unavailable for some poorly studied taxa, thus evidence-based methods are not directly applicable to infer homology. Ebersberger et al. [9] first attempted to circumvent this problem by using a novel hybrid approach (HaMStR) for extraction of homologous sequences from EST/RNA-seq data using a profile Hidden Markov Model (pHMM) [10] based on a similarity search coupled with subsequent RBH derived from re-BLASTing against a reference proteome. The innovative feature of their approach is in the utilization of pHMM as an additional evidence for homology. This architecture incorporates characteristics of multiple sequence alignments (MSA) for user pre-defined core orthologs. Then, a HMM search is performed with each individual pHMM using matching criterion applied to find putative orthologs in the proteome of interest. This method, however, has limitations and weaknesses, such as:

- Proteome training sets composed of phylogeneticaly "meaningful" taxa for construction of core ortholog clusters may not be available
- Identification of informative core ortholog clusters may be somewhat cumbersome due to incomplete and/or low coverage sequencing
- The pHMMs may not contain any relevant compositional or phylogenetic properties about biological sequences that constitute MSA
- Inability to explicitly identify paralogy limits the use of HaMStR for some evolutionary applications

Hence, homologous clusters inferred from various multiple sequences require further vali-

dation to improve confidence in orthology/paralogy classification. Here, we propose a unique approach to identify false-positive homologies detected by heuristic methods, for example HaMStR or InParanoid [36], called the Orthology Group Cleaner (the OGCleaner). Our machine learning method uses phylogenetically-guided inferred homologies to identify non-homologous (false-positive) clusters of sequences. This improves the accuracy of heuristic searches, like those that rely on BLAST.

2 Methods

The OGCleaner is a machine learning approach to removing low quality, putative homology clusters that have been identified by other methods in order to improve phylogenetic tree reconstruction. To do this, we first construct a ground-truth dataset which consists of known homology and non-homology clusters. We then select attributes for use in training machine learning algorithms. Finally, we apply it to two real datasets. One real data provided by a phylogenetic benchmarking suite. The other, a novel set of species where the phylogenetic tree is not well-established.

2.1 Construction of ground-truth training sets

The ground-truth data set is built around data gathered from OrthoDB, one of the most comprehensive collections of putative orthologous relationships predicted from proteomes across a vast taxonomic range [38]. This data is particularly useful for construction of training sets since OrthoDB clusters were detected using a phylogeny-informed approach collated with available functional annotations. Hence, training sets constructed from OrthoDB clusters have the inherent benefit of both an evolutionary and physiological assessment resulting in more precise filtering for false-positive homology.

The key to our method was the development of labeled training sets that were used to train supervised machine learning classifiers. Previously, homology clusters were known and annotated in OrthoDB. There were, however, no annotated clusters that represented non-homology clusters from random alignments. Thus, we created and annotated our own set of non-homology clusters

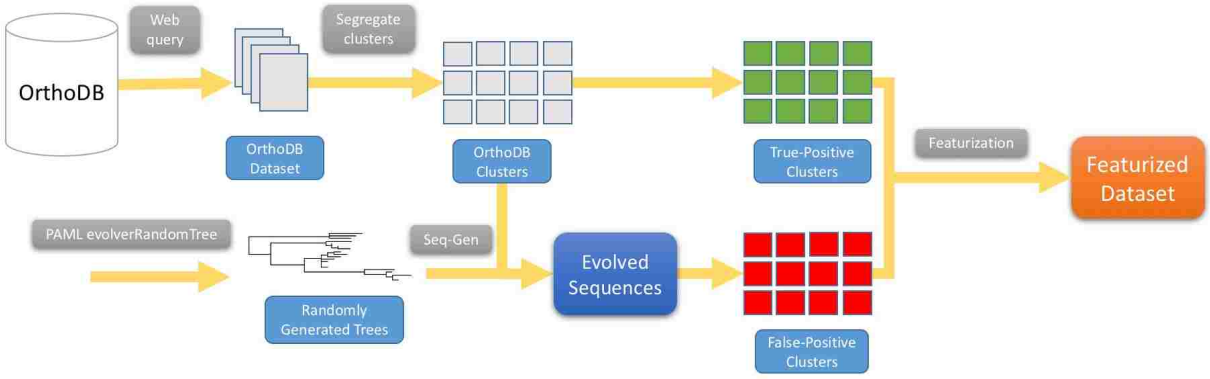

Figure 1: Cluster generation workflow. This is the process of generating homology and non-homology clusters for training the machine learning algorithms.

through a generative process. We created these clusters in two different manners: randomly aligned sequences and evolving sequences from the homology clusters. Cluster generation can be seen in Figure 1.

We extracted 5,332 homology (H) clusters from the predefined OrthoDB profile called "single copy in >70% of species" across the entire arthropod phylogeny in the database, and then aligned them. Non-homology (NH) clusters were generated by:

1. Using the alignment of randomly drawn sequences from the totality of the protein sequences with cluster size sampled from Poisson (λ), where λ=44.3056 was estimated as the average cluster size of Hs
2. Evolving the sequences taken from H clusters

This process of evolving sequences was accomplished by using PAML [40] to generate random binary trees for each sequence within a cluster. The discretized number of terminal branches for each random tree was sampled from a normal distribution with mean 50 and a standard deviation of 15. Within each of the clusters, individual sequences were evolved using their respective randomly generated tree using Seq-Gen [35]. We used WAG + I [39] as the substitution model for the amino acid sequences during the evolving process specifying the number of invariable sites (i) at 0%, 25% and 50%. Then, to form NH clusters, a single evolved sequence from the terminal

branches was selected randomly from each tree.

It is unknown how closely the synthetic NH clusters resemble non-homology clusters that are encountered in actual analyses. By following the previously mentioned process, we believe that the simulated NH clusters are realistic clusters in which the evolved sequences are diverged enough to be considered as non-homologous to each other. During run-time, we see that the OGCleaner does filter homology clusters that result in better phylogenetic trees (see Tables 8, 9 and 10). During benchmarking, the OGCleaner identifies clusters as non-homology and results in better metrics after their removal. This suggests that putative homology clusters found during analysis on real data resemble the synthetic non-homology clusters that the OGCleaner was trained on.

We attempt to assess how realistic the false-positive clusters are by using T-SNE to visualize the H and NH clusters in two dimensions (see Figure 2). As can be seen in the figure, the generated instances overlap many of the true-positive clusters from OrthoDB. This suggests to us that the generated false-positive clusters have features that are similar to the original clusters and are therefore realistic to some degree.

From the H and NH clusters, two different sets of training, validation and testing partitions were formed. The first set (EQUAL) had an equal number of homology, randomly aligned, 0% invariable-site evolved, 25% invariable-site evolved and 50% invariable-site evolved clusters within the combination of training, validation and testing data sets. The second set (PROP) consisted of 50% of the training set as homology clusters while the remaining half of the training set was composed of equal parts randomly aligned, 0% invariable-site evolved, 25% invariable-site evolved and 50% invariable-site evolved clusters. The combined data sets were then partitioned into training, validation and testing. Initial testing revealed EQUAL and PROP to perform about the same. Thus, we retained only the PROP data set for testing and use PROP in our released implementation. This was done by randomly sampling from the pool of clusters and assigning 80% of the clusters (8,800) to training, 10% (1,100) to validation and the last 10% (1,100) to testing.

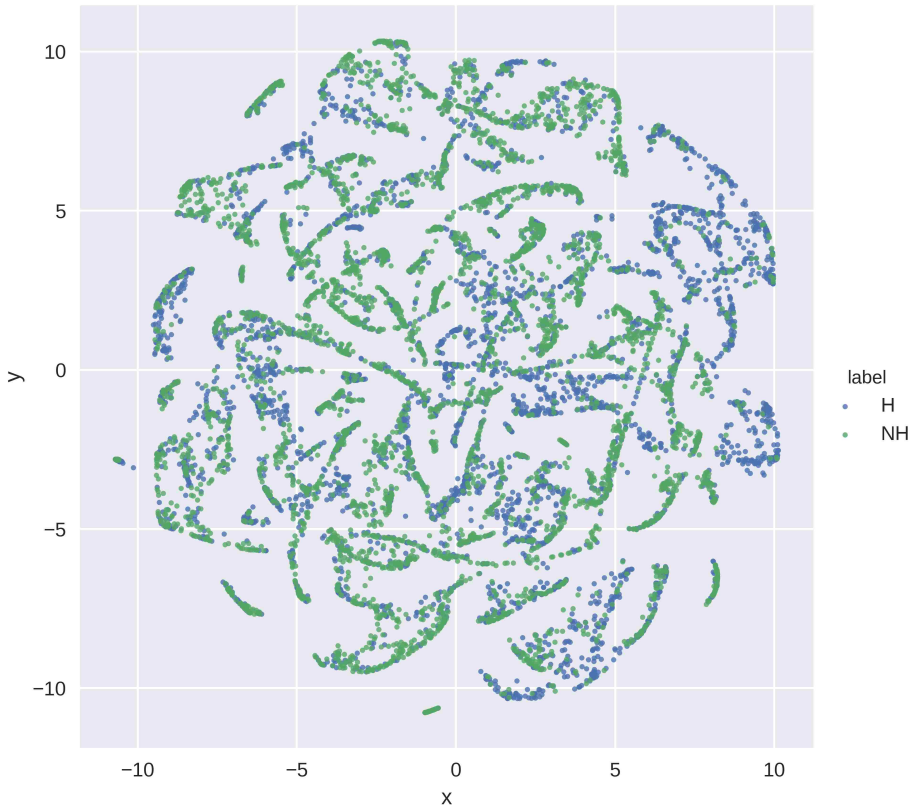

Figure 2: T-SNE manifold learning dimensionality reduction applied to the true-positive OrthoDB clusters (blue, H) and generated false-positive clusters (green, NH). Instances from both classes overlap and are not easily separable suggesting that the generated clusters are similar to the true-positive clusters.

2.2 Attribute selection

Ten different attribute features were selected (Table 1) and calculated for individual MSA of putative homology clusters and for training Hs and NHs as well. To identify randomly aligned positions in MSAs, we utilized ALISCORE [31], software based on the principle of parametric Monte Carlo resampling within a sliding window. This approach is more objective and exhibits less conservative behavior contrasted to commonly used non-parametric approaches implemented in GBLOCKS [4, 27]. We expected the number of randomly aligned positions for false-positive homologies to be higher than for true homologs. Additionally, several other simple metrics (the number of sequences forming MSAs, alignment length, total number of gaps, total number of amino acid residues and range defined as the difference between longest and smallest sequences within MSAs) were also derived. Overall, incorporation of these attributes into a training set was used to increase the robustness of the performance of the machine learning algorithm. We also obtained amino acid composition for each sequence from each cluster and binned it into four classes according to physicochemical properties of amino acids (charged, uncharged, hydrophobic and special cases), then compositional dispersion was calculated using an unbiased variance estimator corrected for sequence length. Here we assumed that amino acid composition between closely related sequences would be preserved by analogous weak genome-wide evolutionary constraints [24, 37] and thus have diminished variance.

For detection of false-positive homology we utilized different supervised machine learning algorithms in order to learn from the labeled data instances. Supervised machine learning algorithms take in labeled instances of a particular event as input. From these labeled instances, the algorithm can then learn from the features associated with the instance to perform classification on other, unlabeled instances. A number of different algorithms were used in order to find a model that performed well. Initially, we used the Waikato Environment for Knowledge Analysis (WEKA) software [18] for training different supervised machine learning classifiers and for evaluating the test data sets. Eventually, we moved to using scikit-learn for our own software package [33]. A set

Feature	Description
Aliscore	The number of positions identified by Aliscore as randomly aligned
Length	The length of the alignment
# of Sequences	The number of sequences in the alignment
# of Gaps	Number of base positions marked with a gap
# of Amino Acids	Number of amino acids in the alignment
Range	Longest non-aligned sequence length minus shortest non-aligned sequence length
Amino Acid Charged	Standard deviation for the proportions of amino acids in the charged class for each sequence
Amino Acid Uncharged	Standard deviation for the proportions of amino acids in the uncharged class for each sequence
Amino Acid Special	Standard deviation for the proportions of amino acids in the non-charged and non-hydrophobic class for each sequence
Amino Acid Hydrophobic	Standard deviation for the proportions of amino acids in the hydrophobic class for each sequence

Table 1: All Features that were used in order to train the machine learning algorithm. Each of these features was calculated for each of the clusters Machine learning.

of models was trained and compared using the arthropod data set (see Section 4.1 for additional information).

A number of different machine learning algorithms were evaluated. These algorithms included: neural networks, support vector machines (SVMs), random forest, Naive Bayes, logistic regression, and two meta-classifiers. A total of seven models were trained for the arthropod data set. A meta-classifier uses a combination of machine learning algorithms in tandem to perform classification. The two different meta-classifiers utilized stacking with a neural network as the meta-classifying algorithm. Stacking takes the output classifications for all other machine learning algorithms as input and then feeds them into another machine learning algorithm. The learning algorithm that is stacked on the others is then trained and learns which machine learning algorithms it should give more credence when performing classification. One of the meta-classifiers incorporated all the previously mentioned learning algorithms (neural network, SVM, random forest, Naive Bayes, and logistic regression). The other meta-classifier used all the previously mentioned learning algorithms except for logistic regression. All parameters for each machine learning algorithm are summarized in Table 2.

Algorithm	Parameters
Neural Network	weka.classifiers.functions.MultilayerPerceptron -L 0.1 -M 0.05 -N 3000 -V 0 -S 0 -E 40 -H a
Support Vector Machine (SVM)	weka.classifiers.functions.SMO -C 1.0 -L 0.001 -P 1.0E-12 -N 0 -V 1 -W 1 -K ”weka.classifiers.functions.supportVector.PolyKernel -C
Random Forest	weka.classifiers.trees.RandomForest -I 10 -K 0 -S 1
Naive Bayes	weka.classifiers.bayes.NaiveBayes
Logistic Regression	weka.classifiers.functions.Logistic -R 1.0E-8 -M 1
Meta-Classifier w/o Logistic Regression	weka.classifiers.meta.Stacking -X 10 -M ”weka.classifiers.functions.MultilayerPerceptron -L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H a” -S 1 -B ”weka.classifiers.trees.RandomForest -I 10 -K 0 -S 1” -B ”weka.classifiers.bayes.NaiveBayes ” -B ”weka.classifiers.functions.SMO -C 1.0 -L 0.001 -P 1.0E-12 -N 0 -V 1 -W 1 -K ”weka.classifiers.functions.supportVector.PolyKernel -C 250007 -E 1.0””
Meta-Classifier w/Logistic Regression	weka.classifiers.meta.Stacking -X 10 -M ”weka.classifiers.functions.MultilayerPerceptron -L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H a” -S 1 -B ”weka.classifiers.functions.Logistic -R 1.0E-8 -M 1” -B ”weka.classifiers.functions.MultilayerPerceptron -L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H a” -B ”weka.classifiers.trees.RandomForest -I 10 -K 0 -S 1” -B ”weka.classifiers.bayes.NaiveBayes ” -B ”weka.classifiers.functions.SMO -C 1.0 -L 0.001 -P 1.0E-12 -N 0 -V 1 -W 1 -K ”weka.classifiers.functions.supportVector.PolyKernel -C 250007 -E 1.0””

Table 2: The machine learning parameters used for each of the different algorithms in initial training and testing with WEKA.

By testing with WEKA, we determined preliminary performance of each of the different algorithms. For our own software package and further analysis, we decided to not include the meta-classifier w/logistic regression (retaining only the meta-classifier with all algorithms) as there appeared to be no added benefit of two meta-classifiers.

3 Real Data Set Construction

We tested this approach on two different real data sets: a benchmark data set with well-established phylogenetic trees and novel data set where the phylogenetic tree is not well-established. The novel data set was constructed in the following manner:

3.1 Library preparation and RNA-seq

For the experimental data set (OD_S) we used 18 Odonata (dragonflies and damselflies) and 2 Ephemeroptera (mayflies) species. Total RNA was extracted from the eye tissues of each taxon using NucleoSpin RNA II columns (Clontech) and reverse-transcribed into cDNA libraries using the Illumina TruSeq RNA v2 sample preparation kit that both generates and amplifies full-length cDNAs. Prepped Ephemeroptera mRNA libraries were sequenced on an Illumina HiSeq 2000 producing 101 bp paired-end reads by the Microarray and Genomic Analysis Core Facility at the Huntsman Cancer Institute at the University of Utah, Salt Lake City, UT, USA, while all Odonata preps were sequenced on a GAIIx producing 72 bp paired-end reads by the DNA sequencing center at Brigham Young University, Provo, UT, USA. The expected insert sizes were 150 bp and 280 bp respectively. Raw RNA-seq reads were deposited in the National Center for Biotechnology Information (NCBI), Sequence Read Archive (see Table 3 for SRA IDs).

3.2 Read trimming and de novo transcriptome assembly

The read libraries were trimmed using the Mott algorithm implemented in PoPoolation [22] with default parameters (minimum read length = 40, quality threshold = 20). For the assembly of the

Library	Reads Before Trimming	Reads After Trimming	N50	Max Contig Length	Min Contig Length	# Contigs	# Peptides (TransDecoder)	SRA ID (NCBI)
OD07_Cordulegaster_maculata	7207518	6819629	983	16508	201	28163	11877	SRR2164542
OD08_Anax_junius	6267456	5978119	807	9457	201	19987	8519	SRR2164543
OD10_Hetaerina_americana	6447244	6057989	1141	10815	201	34384	13373	SRR2164551
OD11_Ischnura_verticalis	6183018	5790475	879	7894	201	27001	10568	SRR2164552
OD12_Gomphus_spicatus	6498099	6273168	1502	11612	201	37936	10611	SRR2157378
OD13_Nehalennia_gracilis	5894197	5516510	1027	11774	201	33766	12694	SRR2157379
OD18_Chromagrion_conditum	7607629	7189745	1188	8671	201	30453	9256	SRR2157380
OD25_Stylurus_spiniceps	6840281	6597769	1249	23861	201	37436	13568	SRR2157381
OD28_Neurocordulia_yamaskanensis	6410925	6061801	1261	15978	201	34984	12905	SRR2157382
OD36_Argia_fumipennis_violacea	5955971	5600701	1076	11410	201	35049	12754	SRR2157383
OD42_Archilestes_grandis	8736454	8367974	1179	9315	201	34318	12285	SRR2164544
OD43_Hetaerina_americana_2	3585483	3411596	738	7343	201	24192	7318	SRR2164545
OD44_Enallagma_sp	5646110	5370720	940	6446	201	27135	8085	SRR2157367
OD45_Libellula_forensis	5591547	5383248	1125	13425	201	31962	11352	SRR2164546
OD46_Libellula_saturnata	5961628	5717528	1397	13986	201	35045	13326	SRR2164547
OD62_Ischnura_hastata	10080263	9551907	1777	11200	201	40154	13651	SRR2164548
OD64_Anax_junius_2	9657180	9195840	1133	21566	201	30833	13117	SRR2157371
OD_Ischnura_cervula	7105927	6702900	1156	15621	201	40741	14253	SRR2157372
R_E001_Baetis_sp	16352942	16113853	1786	10772	201	30517	16743	SRR2164549
R_E006_Epeorus_sp	13846765	13701079	1303	23453	201	45886	16782	SRR2164550

Table 3: Sequence Read Archive statistics and IDs.

transcriptome contigs we used Trinity [16], currently the most accurate de novo assembler for RNA-seq data [41], under the default parameters.

3.3 Downstream transcriptome processing

In order to identify putative protein sequences within the Trinity assemblies we used TransDecoder (`http://transdecoder.github.io`), the utility integrated into the comprehensive Trinotate pipeline (`http://trinotate.github.io`) that is specifically developed for automatic functional annotation of transcriptomes [17]. TransDecoder identifies the longest open reading frames (ORFs) within each assembled DNA contig, the subset of the longest ORFs is then used to empirically estimate parameters for a Markov model based on hexamer distribution. The reference null distribution that represents non-coding sequences is constructed by randomizing the composition of these longest contigs. During the next decision step, each longest determined ORF and its 5 other alternative reading frames are tested using the trained Markov model. If the log-likelihood coding/noncoding ratio is positive and is the highest, this putative ORF with the correct reading frame is retained in the protein collection (proteome). For more details about the RNA-seq libraries, assemblies and predicted proteomes see Table 3.

Drosophila Species	NCBI ID	# of bases (in Gb)	Platform
D. ananassae	SRR166825	13.6	Illumina HiSeq 200 PE
D. biarmipes	SRR346718	6.4	Illumina HiSeq 200 PE
D. ficusphila	SRR346748, SRR346751	12.1	Illumina HiSeq 200 PE
D. mauritiana	SRR1560444	7.7	Illumina HiSeq 200 PE
D. melanogaster	SRR1197414	9.6	Illumina HiSeq 200 PE
D. miranda	SRR899848	13	Illumina HiSeq 200 PE
D. mojavensis	SRR166833	11.1	Illumina HiSeq 200 PE
D. pseudoobscura	SRR166829	15.1	Illumina HiSeq 200 PE
D. simulans	SRR166816	17.2	Illumina HiSeq 200 PE
D. virilis	SRR166837	15.1	Illumina HiSeq 200 PE
D. yakuba	SRR166821	13	Illumina HiSeq 200 PE

Table 4: Drosophila data sets.

3.4 Construction of Drosophila data set

Ten high quality Drosophila raw RNA-seq data sets (DROSO) were obtained from NCBI (Table 4). First we trimmed the reads using PoPoolation [22] and subsampled the read libraries to the size of the smallest (Drosophila biarmipes). Then, two additional data sets corresponding to 50% and 10% of the scaled libraries were constructed by randomly drawing reads from the original full-sized libraries. Finally, de novo transcriptome assembly and protein prediction were conducted as outlined above for these three data sets. These data sets were used to test whether homology clusters derived from low-coverage RNA-seq libraries contain more false-positives.

3.5 Gene homology inference

To predict probable homology relationships between proteomes we used the heuristic predictor InParanoid/MultiParanoid based on the RBH concept [1, 36]. Among various heuristic-based methods for sequence homology detection, OrthoMCL [28] and InParanoid [36] have been shown to exhibit comparable high specificity and sensitivity scores estimated by Latent Class Analysis [5], so in the present study we exploited InParanoid/MultiParanoid v. 4.1 for the purpose of simplicity in computational implementation. InParanoid initially performs bidirectional BLAST hits (BBHs) between two proteomes to detect BBHs in the pairwise manner. For this step, we set default parameters with the BLOSUM62 protein substitution matrix and bit score cutoff of 40 for all-against-all BLAST search. Next, MultiParanoid forms multi-species groups using the notion

of a single-linkage. Due to inefficient MultiParanoid clustering algorithm, we had to perform a transitive closure to compile homology clusters for all species together. Transitive closure is an operation performed on a set of related values. Formally, a set S is transitive if the following condition is true: for all values A, B, and C in S, if A is related to B and B is related to C, then A is related to C. Transitive closure takes a set (transitive or non-transitive) and creates all transitive relationships, if they do not already exist. When a set is already transitive, its transitive closure is identical to itself. In the case of the pairwise relationships produced by InParanoid, we constructed orthologous clusters using the notion of transitive closure, where gene identifiers were the values, and homology was the relationship.

For example, our OD_S data set consisted of N=20 proteomes, we performed $N(N-1)/2 = 190$ pairwise InParanoid queries. A simple transitive closure yielded 13,998 homology clusters for OD_S. The DROSO data set yielded 20,676, 18,584 and 17,067 homology clusters for 100%, 50% and 10% respectively. Then putative homologous genes were aligned to form individual MSA homology clusters for the subsequent analyses using MAFFT v. 6.864b [21] with the "-auto" flag that enabled detection of the best alignment strategy between accuracy- and speed-oriented methods.

Additionally, we utilized HaMStR v. 13.2.3 [9] under default parameters to delineate putative orthologous sequences in the OD_S proteome sets. 5,332 core 1-to-1ortholog clusters of 5 arthropod species (Ixodes scapularis, Daphnia pulex, Rhodnius prolixus, Apis mellifera and Heliconius melpomene) for training pHMM were retrieved from the latest version of OrthoDB [38]. We used Rhodnius prolixus (triatomid bug) as the reference core proteome because this is the closest phylogenetically related species and publically available proteome to the Ephemeroptera/Odonata lineage [30]. As previously described, each core ortholog cluster was aligned to create MSA using MAFFT and converted into HMM profile using HMMER v. 3.0 [11]. BBHs against the reference proteome were derived using reciprocal BLAST.

4 Implementation

The OGCleaner is implemented in python2 and utilizes machine learning algorithms to classify putative homology clusters of amino acid sequences as homology or non-homology clusters. It can be downloaded, along with example data sets, from `https://github.com/byucsl/ogcleaner`. To train the models, positive and negative examples that represent true-positive and false-positive homology clusters are required. Our overall workflow for generating training data is depicted in Figure 1. See Figure 3 for additional implementation details and example workflow. True-positive homology clusters are gathered from OrthoDB, a hierarchical catalog of orthologous sequences ([26]). Alternatively, a user can provide their own orthology groups for training.

Algorithm 1 False-positive cluster generation

1: **procedure** GENERATEFPCLUSTERS
2: Initialize C by duplicating *true-positive clusters*
3: **for each** $c \in C$ **do**
4: **for each** $seq \in c$ **do**
5: *rtree* $\leftarrow$ random phylogenetic tree with n leaf nodes
6: *evolvedSeqs* $\leftarrow$ *n* evolved sequences based on *rtree* and *seq*
7: *evolvedSeq* $\leftarrow$ randomly selected sequence from *evolvedSeqs*
8: *seq* $\leftarrow$ *evolvedSeq*

False-positive clusters are generated by following Algorithm 1. PAML's ([40]) evolver-RandomTree module is used to generate random phylogenetic trees (Alg. 1 Op. 5). A randomly generated tree with *n* leaf nodes and a single sequence from a cluster are provided to Seq-Gen ([35]) which generates *n* evolved sequences (Alg. 1 Op. 6). A single evolved sequence is randomly chosen from the *n* evolved sequences (Alg. 1 Op. 7) and takes the place of the original sequence in the cluster (Alg. 1 Op. 8). A newly generated cluster consisting of the evolved sequences is still, by definition, a homology cluster because its sequences were derived from the same ancestral sequences. In our released implementation, the training set is 50% H clusters and 50% NH clusters. The new cluster should, however, have diverged enough from the original ancestral sequences to show characteristics of a false-positive homology cluster.

Both the true-positive and false-positive cluster data sets are then featurized for model

		Predicted	
		H	NH
Actual	H	860	13
	NH	42	920

Table 5: Confusion matrix of test instances (1835 total instances) for the neural network model.

	Precision	Recall	F1-Score	Support
H	0.95	0.99	0.97	873
NH	0.99	0.95	0.97	962
avg/total	0.97	0.97	0.97	1835

Table 6: Per-class performance of the test-set measured with precision, recall and F1-score with support (number of instances) for each class.

training. Clusters are aligned using MAFFT ([20]). Cluster features are then extracted using our own feature extraction scripts and Aliscore ([27], [31]). A full list of the used features is found in Table 1. Pandas dataframes are used for data handling ([29]). Using scikit-learn, various models are provided to the user for cluster classification ([33]). A meta-classifier that implements stacking created by the authors is also provided to the user. The default model for users is a neural network which has shown to provide superior results compared to other models (see Figures 4 and 5 for performance graphs). A confusion matrix showing a breakdown of the neural network's classification accuracy can be found in Table 5 as well as the precision, recall, and F1-score in Table 6.

4.1 Training Data Set

The training data set was used as input to the machine learning model for parameter selection. For the arthropod data set, 80% of the data were used for training, while 10% of the data was reserved for validation and the last 10% for testing. Machine learning algorithms were utilized to learn from the combination of the H and NH clusters in the data set to differentiate the two. A trained model could then be used to classify unlabeled instances as homologous and non-homologous. There were a total of 8,800 instances in the OrthoDB arthropod data set that were used as a training set,

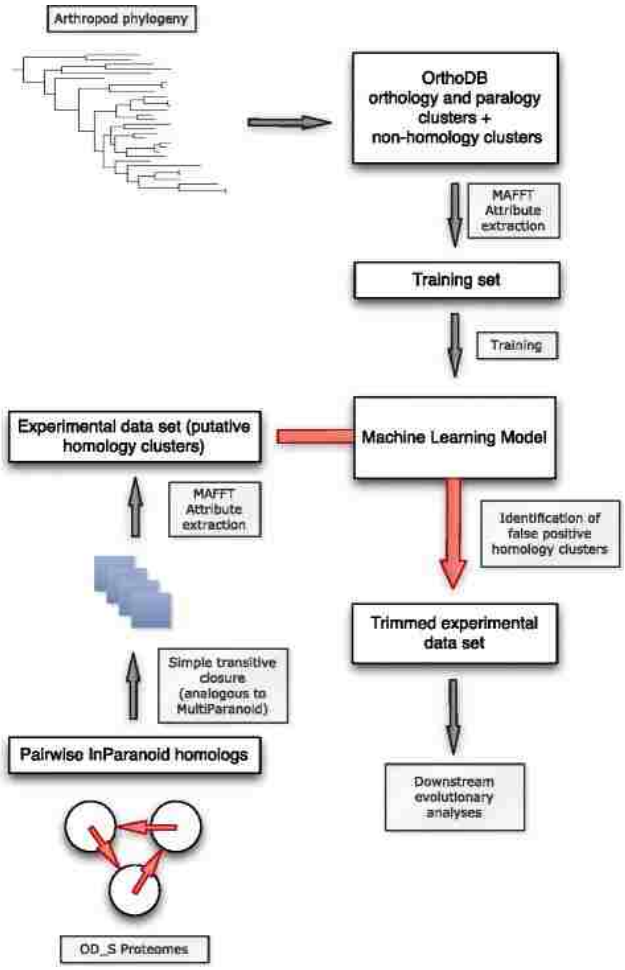

Figure 3: A diagram of the overall workflow of the OGCleaner. This figure shows the different steps that were used in developing our machine learning model. Arthropod phylogeny was generated in previous studies and deposited in OrthoDB. These sequences were then gathered from OrthoDB and used as our orthology and paralogy clusters. They were combined with generated non-homology clusters. The combination represents our training data set used to train the machine learning algorithms. The experimental data were assembled with proteins inferred from the assemblies. InParanoid was then used to identify putative homologs. Once putative homologs were identified they were input into the trained machine learning algorithms for classification and subsequent cluster trimming.

4,378 H and 4,422 NH clusters.

4.2 Validation Data Set

The validation data sets were used after the model had been trained on the training data set. By using the trained model on the validation set, the efficacy of the model could be seen. 10% of the arthropod data set formed the arthropod validation set. The models trained using the arthropod training set were validated only with the arthropod instances. If the model did not perform adequately on the validation set, different parameters for the machine learning algorithms were modified in an attempt to improve the performance of the models. The re-trained models would then revalidate on their same, respective validation sets. The process was repeated until adequate performance of the learning algorithm was reached. The OrthoDB arthropod validation set consisted of 1,100 instances, 566 H and 534 NH clusters.

4.3 Testing Data Set

All general steps of our pipeline are summarized in Figure 3 using the example of OD_S processing. Testing data sets were used only after all the models were finished being trained and validated. This is to ensure an honest measure of the predictive capacity of the models because the testing data were never used in order to evaluate how our model was built and to modify the models. The last 10% of the arthropod data set was used as the arthropod test set. The arthropod test set from the OrthoDB contained 1,100 instances for both the PROP and Equal data sets. The PROP data set had 555 H and 545 NH clusters. The EQUAL data set had 207 H and 893 NH clusters.

4.4 Real Data Set

We tested our filtering process by applying the arthropod classifiers trained on the ground-truth data set to the DROSO and OD_S data sets. Unlike the testing sets mentioned in the previous section, the ground-truth for these data sets was unknown. We examined the number of clusters filtered and conducted a manual inspection of a subset of the filtered clusters to verify the removal of only

false-positive homology clusters. Because there are, to the authors' knowledge, no other post-processing methods for cluster filtering that exist our approach is novel. The filtering processes that do exist are heuristic-based approaches, such as an e-value cutoff, that are built-in modules of the clustering software. Therefore, for comparison, we only examined the number of clusters filtered from the output of InParanoid and HaMStR.

4.5 Miscellaneous Parameters

Seq-Gen Parameters

When creating false-positive homology clusters, Seq-Gen is used to evolve the sequences so that they are no longer homology clusters but are more related than clusters of random sequences. Seq-Gen is set to evolve sequences using WAG +I. Invariable sites vary amongst clusters between 0, 25 and 50% so that there is a variety of sequence conservation amongst clusters during evolving.

5 Results and Discussion

Model validation shows how a variety of learning algorithms perform using a bootstrap analysis. Figures 4 and 5 show the performance of the different models using the training/validation and testing data sets. Comparing the different machine learning algorithms, the multi-layer perceptron (MLP) is the model that performs the best. We varied the size of the training set (from 1% to 100% of training instances). On the train/validation data sets, random forest performs the best (~100% accuracy) but does not perform well on the test data set (~90% accuracy) suggesting that it is overfitting to the data. The models behave differently when given varied amounts of data to train on. All models except for Naive Bayes increased in accuracy as the training data grew. Naive Bayes held constant with ~70% accuracy despite the amount of data provided. Additionally we tested which features were the most meaningful for classification using the MLP model (Figures 8 and 9). We found that the attribute that had the best predictive power to be Aliscore. We believe that

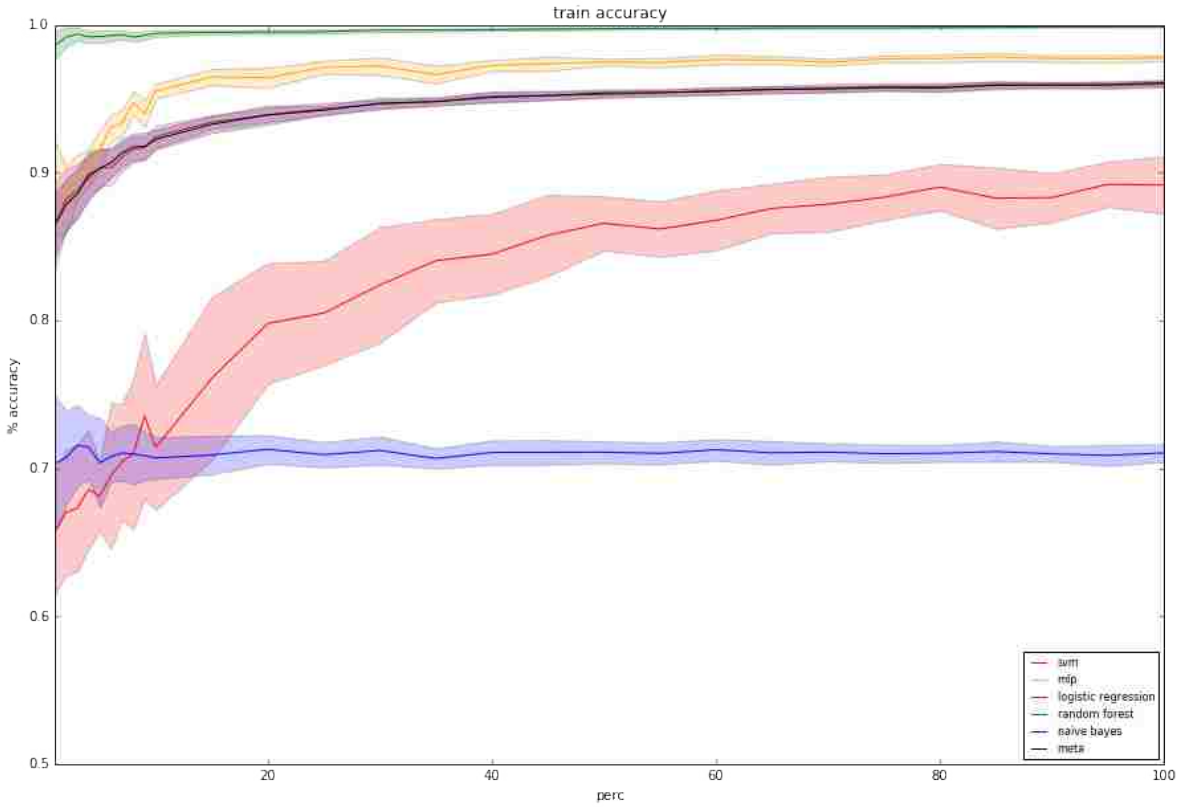

Figure 4: Training data set accuracy.

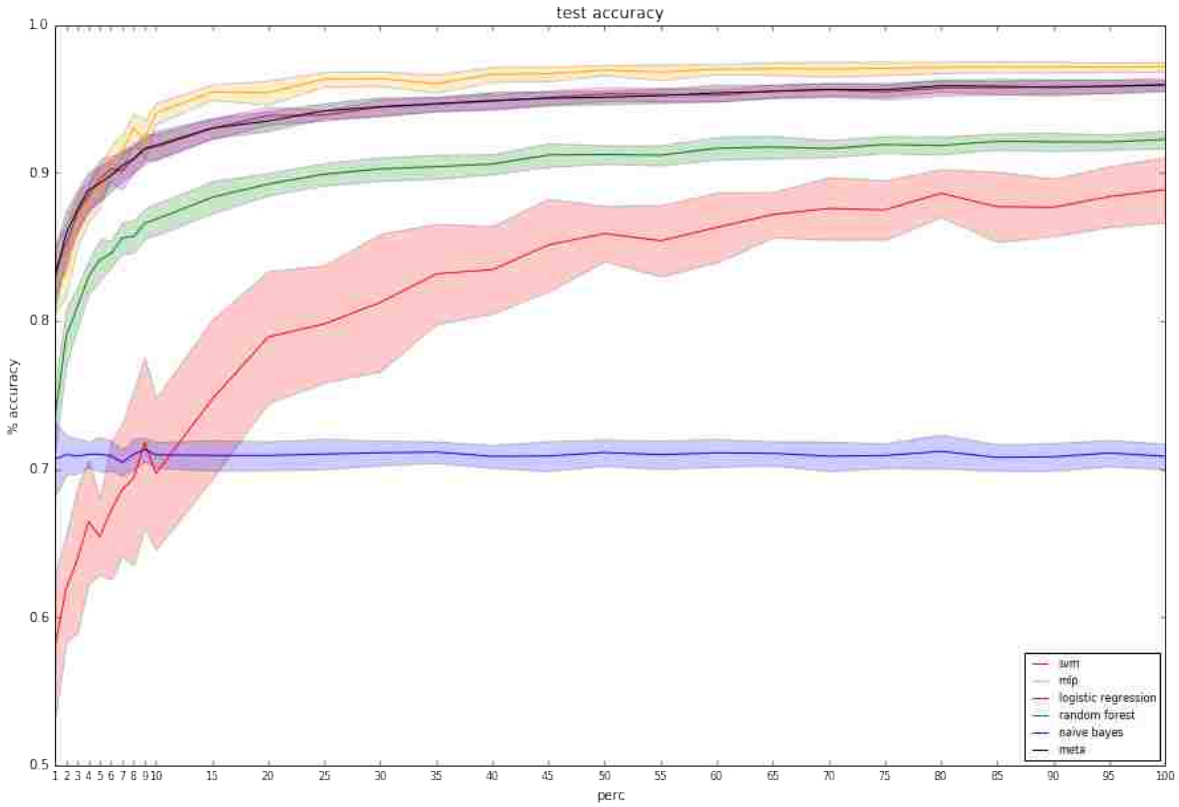

Figure 5: Testing data set accuracy

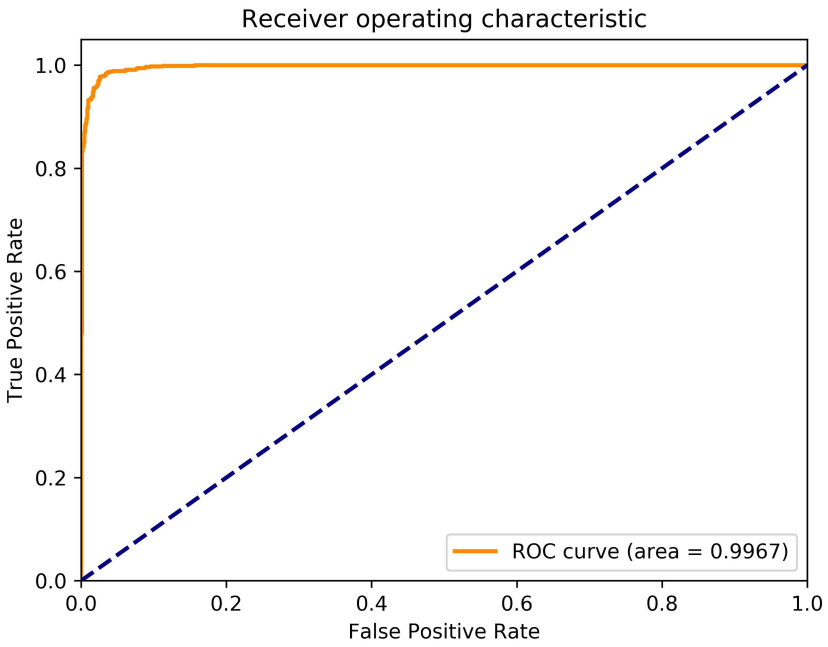

Figure 6: Area under the Receiver Operating Characteristic curve (AUROC) for the neural network model. This shows the trade-off between sensitivity (true-positive rate) and specificity (false-positive rate) by varying classification confidence thresholds. An area under the curve (AUC) of nearly 1 shows that there is little trade-off between sensitivity and specificity.

	Kept	Removed
InParanoid	10500	3497
HaMStR	1231	896

Table 7: Summary of InParanoid and HaMStR cluster filtering. The number of clusters that were kept and removed for the OD_S clusters from InParanoid and HaMStR.

this may be the case because clusters with large amounts of randomly aligned positions probably reflect an NH cluster. Creating a heuristic for this attribute may be possible but not straight forward as suggested by random forest's inability to leverage this feature to outperform the MLP.

Lower coverage data sets are often used when performing transcriptomic and evolutionary analyses especially on non-model organisms. For instance, in a recent paper [32] the authors inferred a phylogeny of many insect species using relatively small RNA-seq library sizes averaging at ~3Gb compared to Drosophila data sets. We expected the number of false-positive clusters to increase with the decreasing sequencing depth. In order to examine this, three DROSO data sets were tested for the presence of false-positives.Indeed, we found that the number of false-positive homology clusters increased in the subsampled DROSO data sets (15.7%, 17.8% and 29.9% for 100%, 50% and 10% DROSO data sets respectively). These subsampled data sets allowed us to see the results that are common when homology clustering is performed on small libraries. Applying the filtering process to the InParanoid and HaMStR OD_S clusters resulted in many removed clusters (Table 7), implying that heuristic-based methods have increased rates of false-positives. The removal of many clusters showed the overall poor quality of many of the putative homology clusters (for comparison between homology and false-positive homology clusters see Figure 7). This was expected due to the low quality transcriptome assembly that was caused by sequencing depth in addition to biological factors such as interspecific differential expression. The filtering process preserved higher quality clusters and finished almost instantly resulting in huge time savings when compared to manually curating the clusters. Overall our method can be applied to filter homology clusters derived from closely related (e.g. Drosophila species) as well as highly diverged taxa (e.g. Odonata species). We also note that the trimming procedure behaves more

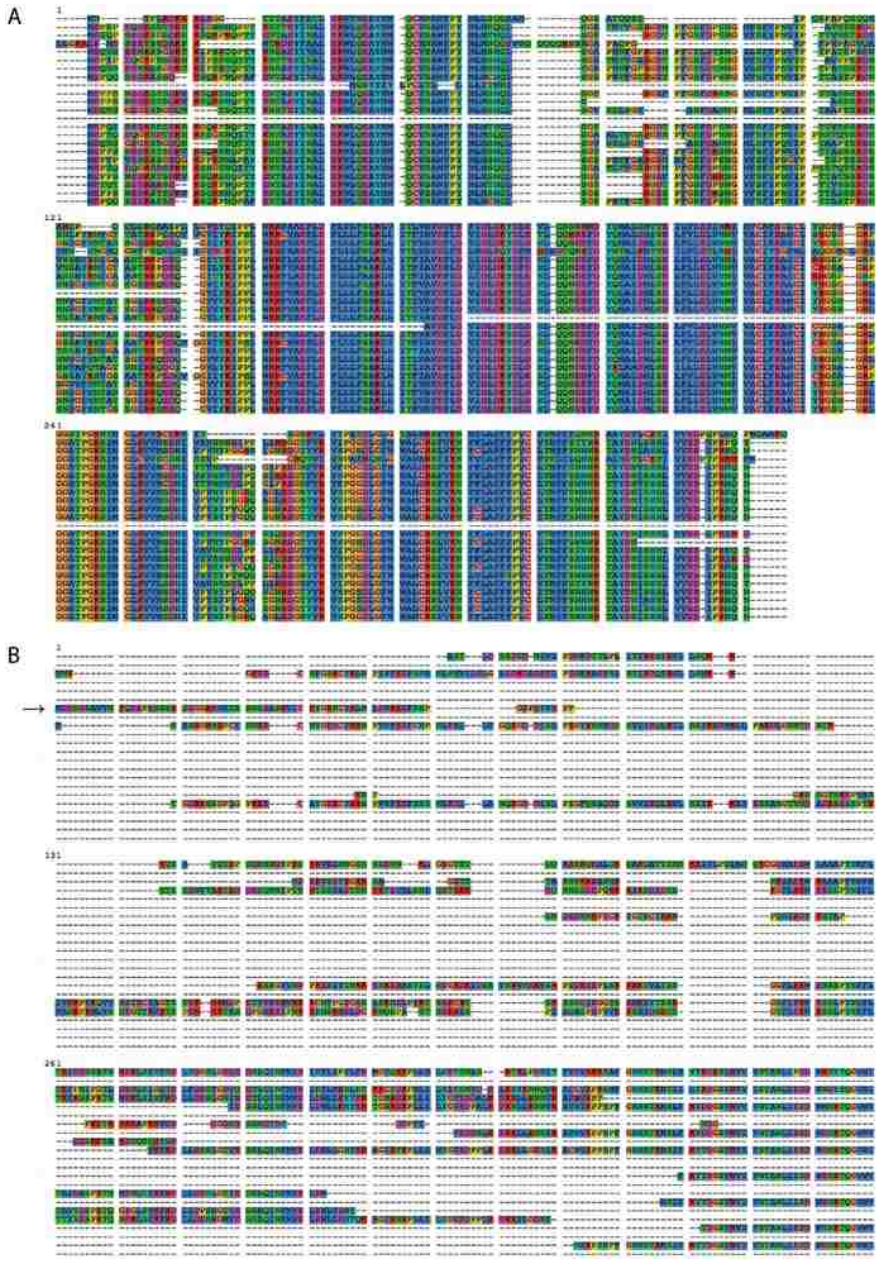

Figure 7: Examples of a high quality homology (a) and false-positive homology (b) All sequences within the homology cluster (a) belong to one protein family (FAM81A1-like protein). The sequence in the false-positive homology cluster indicated by the arrow represents Aprataxin and PNK-like factor whereas other sequences represent tyrosyl-DNA phosphodiesterase.

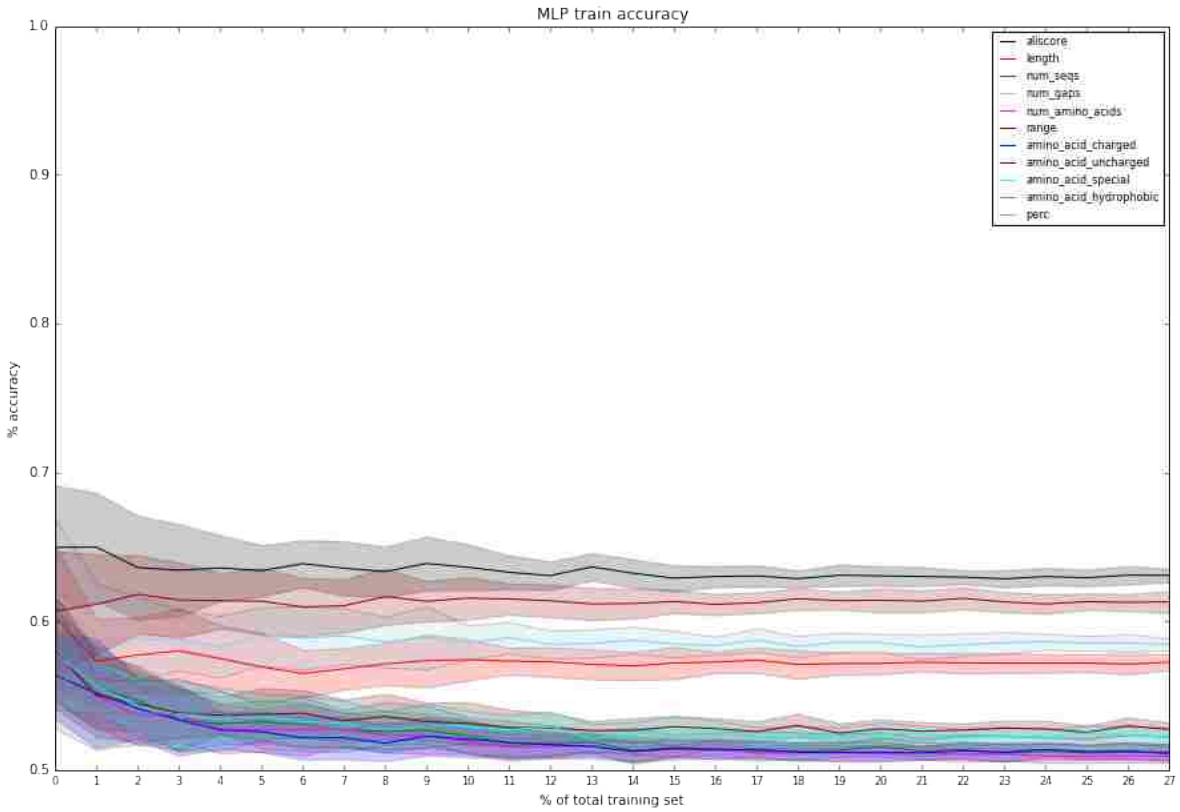

Figure 8: Training data set per attribute performance using MLP.

conservatively with increasingly diverged sequences.

Feature validation shows how each individual features classifying power. Results can be seen in Figures 8 and 9 The same bootstrap analysis scheme as model validation is used except that each feature is only tested using a neural network. All features had testing set accuracies between 50% (as good as random guessing) and ~65% accuracy. These results suggest that it is a combination of the different features that allow the trained models to predict with high accuracy and that it is not a singular feature providing the majority of classification power to the trained models.

Performance was measured by comparing to OrthoMCL [28] using The Orthology Benchmark [2] for evaluating the accuracy of orthology inference. Benchmarking was done using the Quest for Orthologs (QoF) Reference Proteomes v5 (2011-04) data set [15]. This data set is

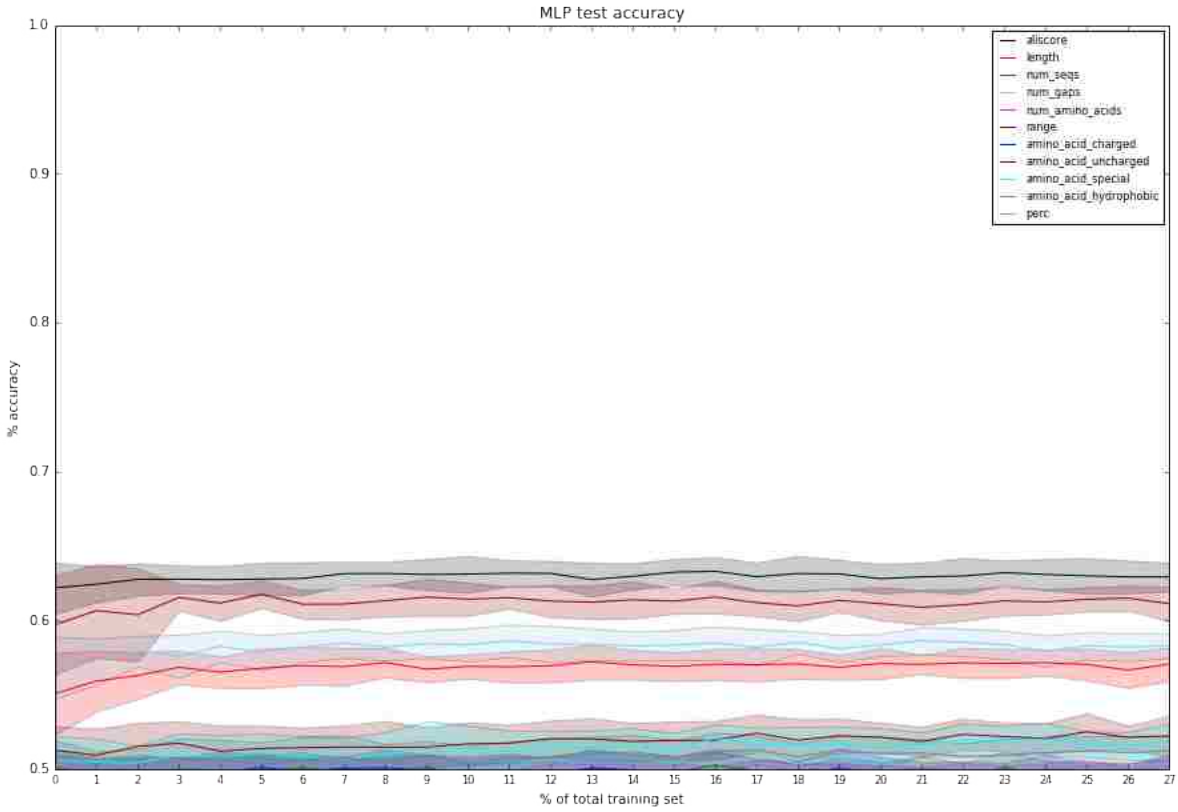

Figure 9: Testing data set per attribute performance using MLP.

generated from UniProtKB [6] and consists of 66 manually compiled proteomes that are a subset of the UniProt reference proteomes. OrthoMCL was used to generate a base set of orthology clusters. The OGCleaner was then applied to the clusters and provided overall better performance than the original OrthoMCL clusters. Comparison using The Orthology Benchmark (`http://orthology.benchmarkservice.org/`) are shown in tables 8, 9, and 10. We believe that improvements will be even greater when applied to non-model organisms that have lower-quality proteome assemblies.

Trained models can also be evaluated on held-out test data with known labels. This can be used to verify model performance especially if using your own orthology groups for model training. Visualization of performance is provided by matplotlib [19] and IPython [34].

		OrthoMCL		OrthoMCL + The OGCleaner		Best Method		
		pos. predictive value rate	true-positive rate	pos. predictive value rate	true-positive rate	pos. predictive value rate	true-positive rate	method name
SwissTree	APP	0.7722±0.130807	0.7093±0.135725	0.7722±0.130807	0.7093±0.135725	0.9718±0.054428	0.8023±0.119034	PANTHER 8.0 (all)
	ASTER	0.9892±0.0210743	0.5562±0.0759241	0.9892±0.0210743	0.5562±0.0759241	0.9898±0.019797	0.5927±0.0750839	RBH / BBH
	BAMBI	0.9756±0.0472181	0.7477±0.116392	0.9756±0.0472181	0.7477±0.116392	0.9783±0.0421432	0.8411±0.0979582	RBH / BBH
	BAR	0.9906±0.0183171	0.6375±0.0732421	0.9906±0.0183171	0.6375±0.0732421	0.9921±0.0154937	0.7553±0.0655001	phylomeDB
	CASP	0.5±0.692965	0.5±0.692965	0.5±0.692965	0.5±0.692965	-	-	-
	CITE	0.92±0.150397	0.3538±0.164395	0.92±0.150397	0.3538±0.164395	0.9592±0.0783502	0.7231±0.153846	PANTHER 8.0 (all)
	Clusterin	0.8462±0.277375	0.8462±0.277375	0.8462±0.277375	0.8462±0.277375	-	-	-
	GH14	0.5±0.692965	0.5±0.692965	0.5±0.692965	0.5±0.692965	-	-	-
	HOX	0.877±0.0824079	0.3835±0.0806901	0.8739±0.084336	0.3728±0.0802418	0.9381±0.0628574	0.3799±0.0805455	RBH / BBH
	MAPT	0.7778±0.384124	0.7778±0.384124	0.7778±0.384124	0.7778±0.384124	-	-	-
	NOX	0.6547±0.0480272	0.8634±0.0398372	0.6547±0.0480272	0.8634±0.0398372	0.9964±0.00708859	0.9632±0.0218327	metaPhOrs incomplete
	POP	0.9897±0.0199997	0.9369±0.0469591	0.9897±0.0199997	0.9369±0.0469591	0.9903±0.0189365	0.9903±0.0189365	PANTHER 8.0 (all)
	PSEN	0.9282±0.049488	0.9898±0.0198976	0.9282±0.049488	0.9898±0.0198976	0.9897±0.0199997	0.9847±0.0243064	Ensembl Compara (e81)
	RPS	0.9565±0.0833443	0.9565±0.0833443	0.9565±0.0833443	0.9565±0.0833443	-	-	-
	SERC	0.8504±0.0347562	0.6347±0.0405388	0.8504±0.0347562	0.6347±0.0405388	0.9978±0.00426551	0.845±0.0304671	metaPhOrs incomplete
	SUMF	0.95±0.0955186	0.95±0.0955186	0.95±0.0955186	0.95±0.0955186	-	-	-
	TRFE	0.5455±0.24026	0.8182±0.227931	0.5455±0.24026	0.8182±0.227931	0.9091±0.16989	0.9091±0.16989	EggNOG
	VATB	0.9331±0.0302875	0.8761±0.038692	0.9331±0.0302875	0.8761±0.038692	0.9959±0.00808248	0.8654±0.0400905	EggNOG
TreeFamA	TreeFamA	0.7891±0.00444051	0.6933±0.00470478	0.8067±0.0045604	0.6296±0.00492701	0.9484±0.00264818	0.6894±0.00472112	metaPhOrs incomplete

Table 8: Agreement with Reference Gene Phylogenies. Higher values are better for 'pos. predictive value rate' and 'true-positive rate'. The '-' character represents when OrthoMCL + The OGCleaner provided the best performance under the 'Best Method' columns. Improvements are marked in green and weaker results in red. OrthoMCL was used to generate a base set of orthology clusters. The OGCleaner was then applied to the clusters using the provided pre-trained model filtering out low-quality clusters.

		OrthoMCL			OrthoMCL + The OGCleaner			Best Method		
		# completed tree samplings (of 50k trials)	avg RF distance (genetree,speciestree)	avg fraction incorrect trees	# completed tree samplings (of 50k trials)	avg RF distance (genetree, speciestree)	avg fraction incorrect trees	# completed tree samplings (of 50k trials)	avg RF distance (genetree, speciestree)	avg fraction incorrect trees
Species Tree Discordance	Eukaryota	10032	0.3097±0.0076	0.446±0.010	9608	0.2959±0.0076	0.431±0.010	13702 metaPh0rs	0.0498±0.0036 PANTHER 8.0 (LDO only)	0.1017±0.0085 OMA Groups (RefSet5)
	Fungi	2570	0.296±0.013	0.470±0.019	2478	0.286±0.013	0.464±0.020	2572 PANTHER 8.0 (all)	0.194±0.025 OMA Groups (RefSet5)	0.340±0.040 OMA Groups (RefSet5)
Generalized Species Tree Discordance	LUCA	5787	0.340±0.016	0.370±0.017	5408	0.322±0.016	0.357±0.017	-	0.167±0.019 OMA Groups (RefSet5)	0.174±0.020 OMA Groups (RefSet5)
	Eukaryota	6611	0.320±0.011	0.607±0.017	6278	0.311±0.011	0.600±0.018	6724 PANTHER 8.0 (all)	0.176±0.017 OMA Groups (RefSet5)	0.402±0.035 OMA Groups (RefSet5)
	Vertebrata	25192	0.502±0.013	0.842±0.013	23959	0.477±0.013	0.831±0.013	-	0.1649±0.0062 InParanoidCore	0.623±0.017 InParanoidCore
	Fungi	19058	0.125±0.012	0.125±0.012	18359	0.127±0.012	0.127±0.012	18627 PANTHER 8.0 (all)	0.0403±0.0070 OMA Groups (RefSet5)	0.0403±0.0070 OMA Groups (RefSet5)

Table 9: Generalized Species Tree Discordance Benchmark. Lower values are better for 'avg RF distance(genetree , speciestree)' and 'avg fraction incorrect trees'.

	OrthoMCL		OrthoMCL + The OGCleaner		Best Method	
	avg Schlicker	# ortholog relations	avg Schlicker	# ortholog relations	avg Schlicker	# ortholog relations
Gene Ontology Conservation	0.4895±0.0012	135009	0.4942±0.0013	120232	0.5202±0.0023 OMA Groups (RefSet5)	174752 PANTHER 8.0 (LDO only)
Enzyme Classification Conservation	0.9392±0.0010	121268	0.94868±0.00091	113995	0.98693±0.00072 OMA Groups (RefSet5)	120204 PANTHER 8.0 (all)

Table 10: Species Tree Discordance Benchmark. Higher value is better for 'avg Schlicker' and '# ortholog relations' is better.

6 Conclusion

We have demonstrated a machine learning method released as the Orthology Group Cleaner (the OGCleaner) that can be used to differentiate homology and non-homology clusters based on characteristics of known good and bad clusters. These results can be seen in our trained models' ability to achieve high classification accuracy on the test data sets as well as by examining the number of clusters that were removed from the experimental OD_S data set. We developed a training set of known good and bad clusters that was previously unavailable and made supervised machine learning impossible. Using a feature set that we developed, we tested various machine learning algorithms and found that when trained on our training data sets that the multi-layer perceptron (neural network) consistently outperformed all other models.

Applications of our method were also seen as we applied them to other data sets. Our method was especially useful when applied to the OD_S data set, by filtering out many clusters as false-positive homology. We showed that our method is effective in settings where non-model organisms are being studied and the transcriptome assembly quality is low primarily due to low coverage sequencing or partial RNA degradation.

This paper has demonstrated the usefulness of machine learning in finding homology clusters by quickly removing low quality clusters without using any additional heuristics. The clusters that are retained can then be used later in higher quality phylogeny reconstruction and/or other analyses of gene evolution. In the future, we aim to explore machine learning approaches to clustering sequences more deeply to produce more refined and reliable homology clusters.

References

[1] Andrey Alexeyenko, Ivica Tamas, Gang Liu, and Erik LL Sonnhammer. Automatic clustering of orthologs and inparalogs shared by multiple proteomes. *Bioinformatics*, 22(14):e9–e15, 2006.

[2] Adrian M Altenhoff, Brigitte Boeckmann, Salvador Capella-Gutierrez, Daniel A Dalquen, Todd DeLuca, Kristoffer Forslund, Jaime Huerta-Cepas, Benjamin Linard, Cécile Pereira, Leszek P Pryszcz, et al. Standardized benchmarking in the quest for orthologs. *Nature Methods*, 13(5):425–430, 2016.

[3] Stephen F Altschul, Warren Gish, Webb Miller, Eugene W Myers, and David J Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403–410, 1990.

[4] J Castresana. Selection of conserved blocks from multiple alignments for their use in phylogenetic analysis. *Molecular Biology and Evolution*, 17(4):540–552, 2000.

[5] Feng Chen, Aaron J Mackey, Jeroen K Vermunt, and David S Roos. Assessing performance of orthology detection strategies applied to eukaryotic genomes. *PloS ONE*, 2(4):e383, 2007.

[6] UniProt Consortium et al. Reorganizing the protein space at the universal protein resource (uniprot). *Nucleic Acids Research*, 40(D1):D71, 2011.

[7] Frédéric Delsuc, Henner Brinkmann, and Hervé Philippe. Phylogenomics and the reconstruction of the tree of life. *Nature Reviews Genetics*, 6(5):361–375, 2005.

[8] Christophe Dessimoz, Toni Gabaldón, David S Roos, Erik LL Sonnhammer, Javier Herrero, Quest for Orthologs Consortium, et al. Toward community standards in the quest for orthologs. *Bioinformatics*, 28(6):900–904, 2012.

[9] Ingo Ebersberger, Sascha Strauss, and Arndt von Haeseler. Hamstr: profile hidden markov model based search for orthologs in ests. *BMC Evolutionary Biology*, 9(1):1, 2009.

[10] Sean R. Eddy. Profile hidden markov models. *Bioinformatics*, 14(9):755–763, 1998.

[11] Sean R. Eddy. Accelerated profile hmm searches. *PLoS Computational Biology*, 7(10): e1002195, 2011.

[12] M Stanley Fujimoto, Anton Suvorov, Nicholas O Jensen, Mark J Clement, and Seth M Bybee. Detecting false positive sequence homology: a machine learning approach. *BMC Bioinformatics*, 17(1):101, 2016.

[13] M Stanley Fujimoto, Anton Suvorov, Nicholas O Jensen, Mark J Clement, Quinn Snell, and Seth M Bybee. The ogcleaner: filtering false-positive homology clusters. *Bioinformatics*, 33 (1):125–127, 2017.

[14] Toni Gabaldón and Eugene V Koonin. Functional and evolutionary implications of gene orthology. *Nature Reviews Genetics*, 14(5):360–366, 2013.

[15] Toni Gabaldón, Christophe Dessimoz, Julie Huxley-Jones, Albert J Vilella, Erik LL Sonnhammer, and Suzanna Lewis. Joining forces in the quest for orthologs. *Genome Biology*, 10(9):1, 2009.

[16] Manfred G Grabherr, Brian J Haas, Moran Yassour, Joshua Z Levin, Dawn A Thompson, Ido Amit, Xian Adiconis, Lin Fan, Raktima Raychowdhury, Qiandong Zeng, et al. Full-length transcriptome assembly from rna-seq data without a reference genome. *Nature Biotechnology*, 29(7):644–652, 2011.

[17] Brian J Haas, Alexie Papanicolaou, Moran Yassour, Manfred Grabherr, Philip D Blood, Joshua Bowden, Matthew Brian Couger, David Eccles, Bo Li, Matthias Lieber, et al. De novo transcript sequence reconstruction from rna-seq using the trinity platform for reference generation and analysis. *Nature Protocols*, 8(8):1494–1512, 2013.

[18] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The weka data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18, 2009.

[19] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[20] Kazutaka Katoh and Daron M Standley. Mafft multiple sequence alignment software version 7: improvements in performance and usability. *Molecular Biology and Evolution*, 30(4): 772–780, 2013.

[21] Kazutaka Katoh, Kazuharu Misawa, Kei-ichi Kuma, and Takashi Miyata. Mafft: a novel method for rapid multiple sequence alignment based on fast fourier transform. *Nucleic Acids Research*, 30(14):3059–3066, 2002.

[22] Robert Kofler, Pablo Orozco-terWengel, Nicola De Maio, Ram Vinay Pandey, Viola Nolte, Andreas Futschik, Carolin Kosiol, and Christian Schlötterer. Popoolation: a toolbox for population genetic analysis of next generation sequencing data from pooled individuals. *PloS ONE*, 6(1):e15925, 2011.

[23] Eugene V Koonin. Orthologs, paralogs, and evolutionary genomics 1. *Annual Review of Genetics*, 39:309–338, 2005.

[24] David P Kreil and Christos A Ouzounis. Identification of thermophilic species by the amino acid compositions deduced from their genomes. *Nucleic Acids Research*, 29(7):1608–1615, 2001.

[25] David M Kristensen, Yuri I Wolf, Arcady R Mushegian, and Eugene V Koonin. Computational methods for gene orthology inference. *Briefings in Bioinformatics*, 12(5):379–391, 2011.

[26] Evgenia V Kriventseva, Fredrik Tegenfeldt, Tom J Petty, Robert M Waterhouse, Felipe A Simão, Igor A Pozdnyakov, Panagiotis Ioannidis, and Evgeny M Zdobnov. Orthodb v8: update of the hierarchical catalog of orthologs and the underlying free software. *Nucleic Acids Research*, 43(D1):D250, 2015.

[27] Patrick Kück, Karen Meusemann, Johannes Dambach, Birthe Thormann, Björn M von Reumont, Johann W Wägele, and Bernhard Misof. Parametric and non-parametric masking of randomness in sequence alignments can be improved and leads to better resolved trees. *Frontiers in Zoology*, 7(1):1, 2010.

[28] Li Li, Christian J Stoeckert, and David S Roos. Orthomcl: identification of ortholog groups for eukaryotic genomes. *Genome Research*, 13(9):2178–2189, 2003.

[29] Wes McKinney. Data structures for statistical computing in python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 51 – 56, 2010.

[30] Karen Meusemann, Björn M von Reumont, Sabrina Simon, Falko Roeding, Sascha Strauss, Patrick Kück, Ingo Ebersberger, Manfred Walzl, Günther Pass, Sebastian Breuers, et al. A phylogenomic approach to resolve the arthropod tree of life. *Molecular Biology and Evolution*, 27(11):2451–2464, 2010.

[31] Bernhard Misof and Katharina Misof. A monte carlo approach successfully identifies randomness in multiple sequence alignments: a more objective means of data exclusion. *Systematic Biology*, 58(1):21–34, 2009.

[32] Bernhard Misof, Shanlin Liu, Karen Meusemann, Ralph S Peters, Alexander Donath, Christoph Mayer, Paul B Frandsen, Jessica Ware, Tomáš Flouri, Rolf G Beutel, et al. Phylogenomics resolves the timing and pattern of insect evolution. *Science*, 346(6210):763–767, 2014.

[33] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[34] Fernando Pérez and Brian E. Granger. IPython: a system for interactive scientific computing. *Computing in Science and Engineering*, 9(3):21–29, May 2007.

[35] Andrew Rambaut and Nicholas C Grass. Seq-gen: an application for the monte carlo simulation of dna sequence evolution along phylogenetic trees. *Computer Applications in the Biosciences: CABIOS*, 13(3):235–238, 1997.

[36] Maido Remm, Christian EV Storm, and Erik LL Sonnhammer. Automatic clustering of orthologs and in-paralogs from pairwise species comparisons. *Journal of Molecular Biology*, 314(5):1041–1052, 2001.

[37] Guang-Zhong Wang and Martin J Lercher. Amino acid composition in endothermic vertebrates is biased in the same direction as in thermophilic prokaryotes. *BMC Evolutionary Biology*, 10(1):1, 2010.

[38] Robert M Waterhouse, Fredrik Tegenfeldt, Jia Li, Evgeny M Zdobnov, and Evgenia V Kriventseva. Orthodb: a hierarchical catalog of animal, fungal and bacterial orthologs. *Nucleic Acids Research*, 41(D1):D358–D365, 2013.

[39] Simon Whelan and Nick Goldman. A general empirical model of protein evolution derived from multiple protein families using a maximum-likelihood approach. *Molecular Biology and Evolution*, 18(5):691–699, 2001.

[40] Ziheng Yang. Paml 4: phylogenetic analysis by maximum likelihood. *Molecular Biology and Evolution*, 24(8):1586–1591, 2007.

[41] Qiong-Yi Zhao, Yi Wang, Yi-Meng Kong, Da Luo, Xuan Li, and Pei Hao. Optimizing de novo transcriptome assembly from short-read rna-seq data: a comparative study. *BMC Bioinformatics*, 12(Suppl 14):S2, 2011.