A Hierarchical Approach to Taxonomic Classification of Uncatalogued Species

410.734: Practical Introduction to Metagenomics Final Project

Rayna Hamilton[1]

[1]Advanced Academic Programs, Johns Hopkins University

Submitted December 19, 2022

ABSTRACT

Most metagenomic sequence classifiers are intended to produce species- or subspecies-level assignments to organisms whose genomes are catalogued in taxonomic databases. While this approach is integral to determining the breakdown of metagenomic communities, it may prevent assignment of reads from species not present in databases, even when such organisms belong to well-described higher taxonomic levels. Here, a hierarchical arrangement of support vector machines is investigated for its ability to classify prokaryotic genome fragments from species not seen during training into the taxonomic levels of Domain, Phylum and Genus. We demonstrate that this architecture enables detection of many sequences which might otherwise be left unclassified and can dynamically determine at which level a sequence can be confidently classified.

INTRODUCTION

Metagenomic analysis involves profiling the entire microbial breakdown of environmental sample, often using next-generation DNA or RNA sequencing data [1]. Taxonomic binning, defined as the assignment of sequence reads to species or higher, less specific levels of the taxonomic hierarchy, depends upon accurate classification tools trained on databases of known microbial species [2]. The classifier Kaiju performs assignment by translating each query sequence in all 6 possible reading frames, then scanning a protein database for exact k-mer matches, where a k-mer is defined as a sequence with a pre-specified length. This classifier may assign reads to a lowest common ancestor when multiple matches score equally [3]. The classifier Kraken instead generates a database which maps each possible nucleotide k-mer to its lowest common ancestor (all genomes which contain that k-mer). Query sequences are then assigned to the best-scoring taxonomic node, where score is determined by the number of k-mers shared [4]. While such methods exhibit excellent precision and recall on organisms present in their training database, their reliance on long, exact k-mer matches may prevent detection of species not seen during training, as evidenced by Kraken's 33% genus-level sensitivity on such sequences [4],[5]. Consequently, many of these reads, which may originate from novel species not currently catalogued in databases, may be left unclassified. While it is not possible for a classifier to assign a correct species label to a species which it has not seen before, it may be possible to assign such sequences to higher taxonomic levels if a sufficient range of organisms from the associated taxonomic group was seen during training.

One transformation of sequence data which may enable sufficient generalization within large taxonomic groups is production of tetranucleotide frequency vectors from microbial genome fragments. It was demonstrated in 2010 that such vectors facilitate accurate separation of organisms from different genera by support vector machines (SVMs) [6]. SVM classification depends upon construction of a separating hyperplane, which is selected by maximizing the distance between the hyperplane and the training data points. To prevent overfitting ("memorizing" the input data, including its noise and errors), SVMs are generally implemented with a soft margin that allows some training points to be misclassified. Additionally, datasets which cannot initially be separated by a linear hyperplane may be transformed using a kernel function, which adds additional dimensions to the dataset that facilitate separation [7].

Depending on how closely a sequence from some unknown organism resembles data seen during training, a classifier may only have sufficient evidence to produce a more general classification, such as a Domain or Phylum assignment. One model architecture which enables dynamic decision-making based on classifier confidence is an arrangement of classifiers into a hierarchy identical to the associated taxonomic hierarchy. In this architecture, a Domain-level classifier is run on all sequences, then it passes each sequence to the appropriate Phylum-level classifier if it is sufficiently confident in its prediction. This process is repeated at more specific taxonomic levels until each sequence is given a final classification or is declared unclassified. The viral metagenome classifier CHEER has applied this approach to neural networks in order to achieve excellent Order-, Family- and Genus-level classification accuracy of viral sequences not seen during training [5]. We will determine whether such an approach can be applied to support vector machines trained on tetranucleotide frequency vectors from prokaryotic genome fragments.

METHODS

The 774 prokaryotic genomes previously used for SVM classification of tetranucleotide vectors were downloaded using the NCBI datasets command-line tool [6],[8]. To ensure that classes used in training had sufficient data to capture their diversity, phyla containing less than 10 member organisms and genera containing less than 3 member organisms were removed. Phyla which contained only one member genus after this point were ignored. For demonstration purposes, 2 Archaea and 2 Bacteria phyla with 2-3 member genera and less than 20 member species (Crenarchaeota,

Classifier Name	Train Accuracy	Test Accuracy	Train F1	Test F1	Unseen Accuracy	Unseen F1
All to Domain	1	0.973927	1	0.973828	0.896208	0.896198
Bacteria to Phylum	1	0.974352	1	0.96922	0.972364	0.951795
Archaea to Phylum	1	0.977252	1	0.973952	0.960492	0.956236
Spirochaetes to Genus	1	0.987359	1	0.981701	0.929529	0.932329
Euryarchaeota to Genus	1	0.978701	1	0.975057	0.947398	0.921669
Tenericutes to Genus	1	0.957688	1	0.922924	0.602061	0.526643
Crenarchaeota to Genus	0.993612	0.992231	0.993611	0.99223	0.991974	0.990194

Table 1. Classifier accuracy and F1 scores for sequences seen during training, sequences not seen during training, and sequences from organisms not seen during training.

Euryarchaeota, Tenericutes, Spirochaetes) were selected for further analysis (Table S1). Training data was constructed by dividing each genome into 500-nucleotide fragments, then counting the occurrences of each tetranucleotide in each fragment. Tetranucleotide counts were converted into frequencies by dividing by the total number of tetranucleotides per fragment (497) [6].

Each classifier was trained only on fragments from its own member organisms. Support vector machines were constructed using the svm.SVC() function within scikit-learn [9]. The radial bias function kernel, which has been previously demonstrated as highly effective at separating this dataset, was used in all trials [6]. The optimal SVM hyperparameter values (gamma and C) were selected using the scikit-learn function GridSearchCV [10]. Each hyperparameter combination was assessed by four-fold cross-validation, or evaluation of four models trained on different 75% subsets of the dataset. Model success was evaluated using F1 score, a harmonic mean of precision and recall which is resistant to bias due to differing class size (Figure S1) [11].

After hyperparameter selection, models were trained on 90% of the training data from their member organisms. Probability models capable of predicting classifier confidence were also trained by specifying probability=True in the svm.SVC() function call [10]. Model accuracy and F1 score was evaluated on the remaining 10% of data points. A tree data structure was then constructed containing classifier nodes which pass data points to the appropriate child node if confident in their prediction, or otherwise return the parental classification. This hierarchical classifier was evaluated on a set of genome fragments from 25 organisms from the same genera as training organisms, but which had not been seen during training (Table S2). A reasonable confidence threshold (required for nodes to report a classification) was chosen by plotting classifier confidence for correctly and incorrectly classified sequences.

RESULTS

Unsurprisingly, all classifier nodes achieved over 99% accuracy and F1 score on datapoints seen during training (Table 1). The overall high testing accuracy and F1 scores (>=97% for all but the Tenericutes classifier) indicates that the models are also generally successful at classifying sequences which they have not seen before, if the sequences are from organisms seen during training. The overall lower "unseen" scores, representing sequences from organisms not seen during training, indicate that such sequences are more difficult to classify. The especially poor unseen performance of the Tenericutes classifier results from a strong tendency to classify nearly all unknown organism sequences as *Mycoplasma*, rather than *Ureaplasma* (Figure 1).

Plotting each model's classification confidence reveals distinct distributions for correctly- and incorrectly-classified unknown organism data points (Figure 2). For all but the Tenericutes classifier, prediction confidences are concentrated near 1.0 for correctly classified data points, but are spread evenly for incorrectly classified points. This result indicates that prediction confidence is a good metric for determining whether output classifications are

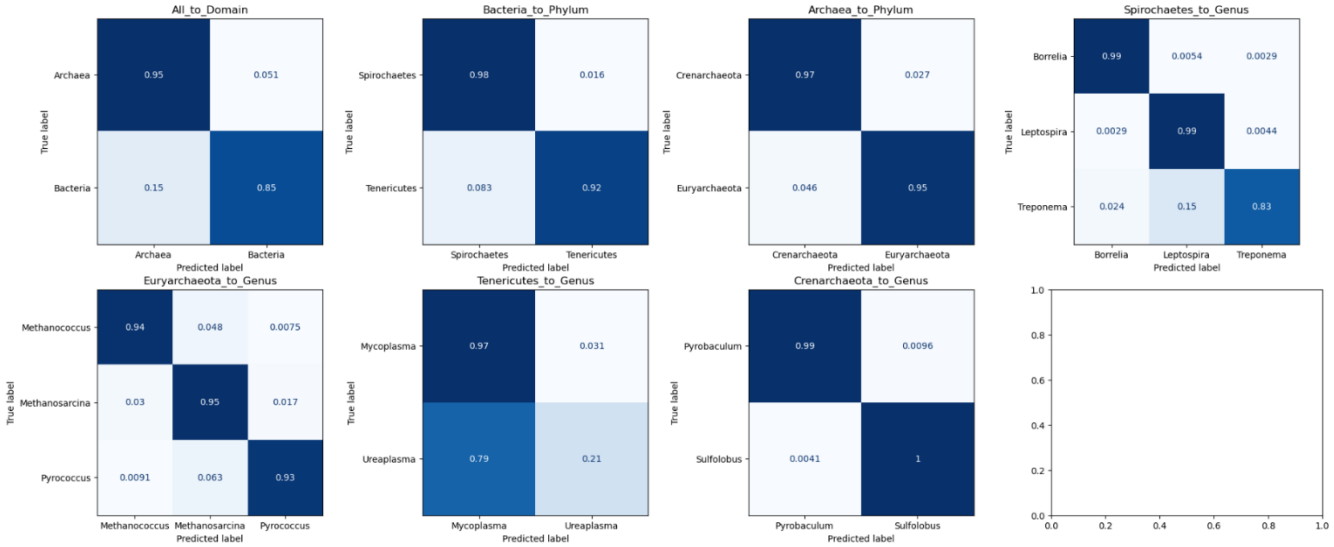

Figure 1. Confusion matrices depicting classifier performance on sequences from organisms not seen during training.

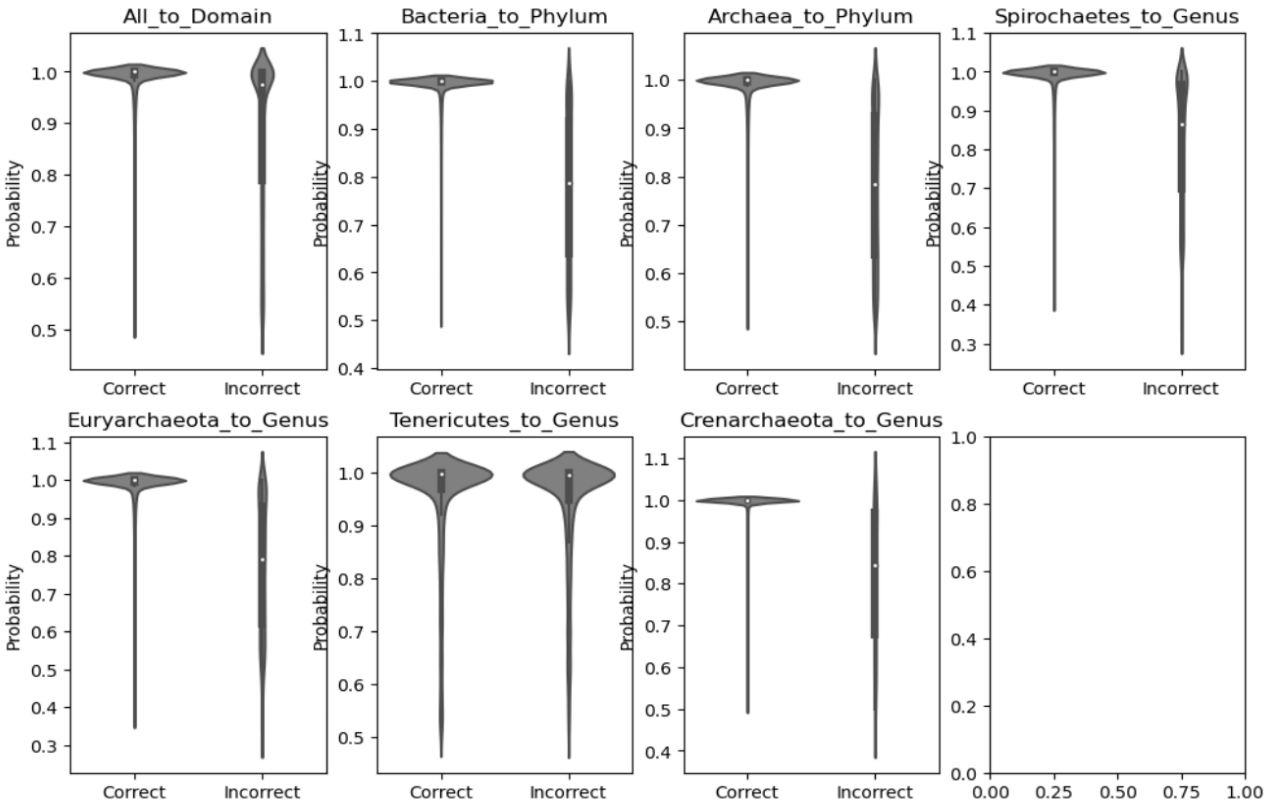

Figure 2. Classification confidence distributions for correctly- and incorrectly-classified sequences from unknown organisms.

correct. Consequently, many incorrect classifications can be avoided by leaving low-confidence datapoints unclassified. Based on these results, a stringent confidence threshold of 99% was chosen for the hierarchical model.

After confidence threshold selection, sequences from the 25 never-before-seen organisms were classified using the hierarchical model. With the stringent 99% confidence threshold, 25.7% these sequences did not yield sufficient confidence to be classified. These sequences likely exhibit tetranucleotide frequency vectors too dissimilar from those seen during training for a confident classification to be assigned. This number could be reduced through use of a lower confidence threshold, however doing so will generally increase the number of incorrect classifications. The hierarchical model was able to assign confident and correct genus-level classifications to 53.5% of sequences from organisms which it had not seen before, a significant improvement over the 33% genus-level sensitivity seen in Kraken [4]. It is, however, important to remember that this study was performed on a small subset of organisms using long genome fragments, rather than potentially error-containing short reads. The number of incorrect classifications increases from 4.6% at the domain level to 5.5% at the genus level. This result indicates that most misclassifications result from the domain-level classifier, an unsurprising result given its 89.6% unknown organism accuracy and F1 score. Use of a wider variety of organisms during training could likely improve this performance.

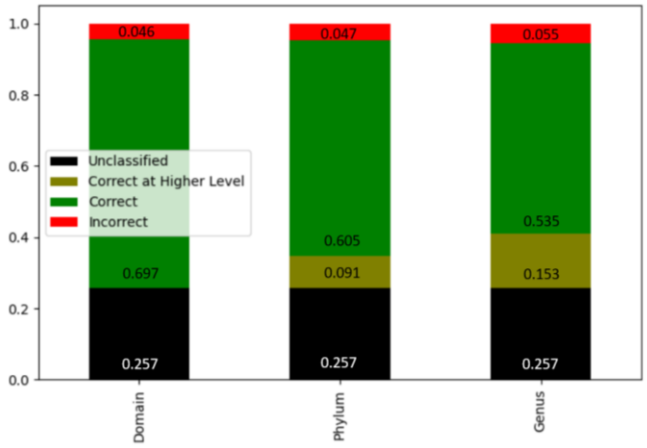

Figure 3. Hierarchical model classification breakdown for three taxonomic levels. For this display, “Correct at Higher Level” indicates that the model was not sufficiently confident to output a classification at the specified taxonomic level, but its higher classification was correct, for example classification of a Leptospira sequence as Bacteria or Spirochaetes.

DISCUSSION

Long k-mer based methods such as Kaiju and Kraken have previously demonstrated excellent classification performance on species and subspecies seen during training [3],[4]. However, assignment of higher taxonomic levels to species which are not currently catalogued in databases has been less thoroughly investigated in the literature [5]. This study demonstrates that application of support vector machines to prokaryotic metagenome fragments facilitates reasonable higher-level classification performance on organisms not seen during training. Use of the classification threshold also enables a user-configurable trade-off between the number of sequences classified and accuracy, a useful feature if putting this model into production. Performance of this hierarchical model was most limited by the Tenericutes genus-level

classifier, a model whose poor performance likely resulted from insufficient diversity in the training organism dataset. Specifically, two of the *Ureaplasma* genomes used present in the training data represented strains from the same species (*Ureaplasma parvum*). It is likely that the *Ureaplasma* organisms used in this training set were very similar to each other, relative to the *Mycoplasma*. Consequently, the model likely learned to classify all sequences not from that small region of the data space as *Mycoplasma*. This issue could likely be avoided with a more diverse organism training set.

If such a model was put into production, a wider variety of organisms would be used, which would likely capture more of the diversity present in each taxonomic group and thus enable more accurate assignment of member organisms. More specifically, a production-quality metagenomic classifier would be trained on a subset of available organisms, then tested on those not seen during training to determine unknown organism classification accuracy. The models could then be re-trained on all available organisms in the database. This training should also be performed on simulated next-generation sequencing datasets, rather than long genome fragments, to ensure that models are able to maintain accuracy when presented with shorter, potentially error-containing reads. However, use of larger training sets will further extend the time needed for the already time-consuming hyperparameter tuning step. Use of this specific model architecture may thus not be practical, especially when applied to the many taxonomic groups not included in this study. Additional model types, architectures and datatypes should thus be investigated to determine the most practical approach to taxonomic classification of currently unclassified species. One approach which could improve accuracy is use of multiple k-mer lengths during frequency calculation, then feature selection to determine the lengths best capable of discriminating between groups at a given taxonomic level. It may for example be determined that longer k-mers give more useful information at lower taxonomic levels and vice versa. If a suitable architecture and datatype is found, an appropriate pipeline would involve first running a species-level classifier on metagenomic reads to identify sequences associated with known species. A higher-level taxonomic classifier could then be run on the remaining unclassified reads. Alternatively, both methods could be run on the same input data and their results compared (in the case of disagreement, the more specific classification, usually from the species-level classifier, would be chosen). Further research into accurate and efficient unknown organism classification methods is needed to elucidate more of the unclassified diversity present in many metagenomic datasets.

CONCLUSIONS

Arrangement of multiple support vector machines into a hierarchical model results in reasonable accuracy and F1 scores when classifying genome fragments from previously-unseen organisms at higher taxonomic levels. One significant advantage of this architecture is the ability to decide on a per-read basis how specific a taxonomic assignment can be confidently produced. Additionally, the confidence threshold feature of this model enables modifying the trade-off between the number of sequences classified and classification accuracy. Application of such a model architecture to sequences from a wider variety of organisms may thus enable classification of sequences from uncatalogued organisms at higher taxonomic levels.

CODE AVAILABILITY

The Jupyter notebook pipeline is located in both the RHamilton_final_pipeline.ipynb file and the RHamilton_final_pipeline.pdf printout. All package versions and information on the Google Cloud instance used is located in documentation.txt.

DATA AVAILABILITY

Model classifications and correct unknown classifications are located in the unknown_predictions.csv and correct_unknown_classifications.csv files, respectively. All genomes used and associated accessions are specified in Table S1 and Table S2 of supplementary_information.pdf. The file table_S1.csv which the pipeline takes as input is directly from the supplementary information of a previous publication [6]. All final trained models are found within final_models.zip.

REFERENCES

1) Orvis, J. (2022). Taxonomic analysis, phylotypes and binning. Personal Collection of J. Orvis, Johns Hopkins University, Baltimore MD.
2) Ye, S. H., Siddle, K. J., Park, D. J., & Sabeti, P. C. (2019). Benchmarking Metagenomics Tools for Taxonomic Classification. *Cell*, *178*(4), 779–794. https://doi.org/10.1016/j.cell.2019.07.010
3) Menzel, P., Ng, K. L., & Krogh, A. (2016). Fast and sensitive taxonomic classification for metagenomics with Kaiju. *Nature Communications*, *7*(1). https://doi.org/10.1038/ncomms11257
4) Wood, D. E., & Salzberg, S. L. (2014). Kraken: ultrafast metagenomic sequence classification using exact alignments. *Genome biology*, *15*(3), R46. https://doi.org/10.1186/gb-2014-15-3-r46
5) Shang, J., & Sun, Y. (2021). CHEER: HierarCHical taxonomic classification for viral mEtagEnomic data via deep leaRning. *Methods (San Diego, Calif.)*, *189*, 95–103. https://doi.org/10.1016/j.ymeth.2020.05.018
6) Perry, S. C., & Beiko, R. G. (2010). Distinguishing microbial genome fragments based on their composition: evolutionary and comparative genomic perspectives. *Genome biology and evolution*, *2*, 117–131. https://doi.org/10.1093/gbe/evq004
7) Noble W. S. (2006). What is a support vector machine?. *Nature biotechnology*, *24*(12), 1565–1567. https://doi.org/10.1038/nbt1206-1565
8) National Library of Medicine. (n.d.). NCBI Datasets. National Center for Biotechnology Information. Retrieved November 20, 2022, from https://www.ncbi.nlm.nih.gov/datasets/
9) sklearn.svm.SVC. (n.d.). Scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
10) sklearn.model_selection.GridSearchCV. (n.d.). Scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
11) sklearn.metrics.f1_score. (n.d.). Scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html