Contents

1 Introduction

1.1 State of the art

Binning as a computational challenge

Recent progress in next-generation sequencing (NGS) technologies allowed affordable sequencing of metagenomes[1] [1] and thus, increased drastically the amount of available samples. In this work, we used data from Oxford Nanopore Technologies MinION sequencers, which offer in a compact format cost-efficient multi-kilobase reads sequencing possibilities, possibly on-field.In this context and be it for health, hydrology, food industry, assigning each read to its correct original taxon has become a key step in any environmental analysis. Binning is all about labelling a mix of reads with their correct assignments ; it is comparable to class assignment coupled with demultiplexing, but with a twist : in the worst case scenario, we don't know how many bins are inside the sample, nor which are the bins. Many tools have been developed to bin metagenomic samples, from unique k-mers frequencies [2], supported by local alignments [3], or based upon 16S [4]. All those approaches rely on training machine or deep learning models [5] on selected DNA features. The challenge is that reads are relatively small and do not target a specific region of the genome, and prediction may be hard for highly conserved regions. Our work is focusing on the k-mers approach, which offers the advantages of being scalable to whole genome analysis.

Overview of alignment-free approaches for read assignment

Research about composition-based taxonomy using short patterns and small words was studied early on [6], in particular codon usage [7], highlighting compositional biases between species : tenuous within a single genome but obvious between different ones. At that time, lack of data and processing power slowed down this research line. Tetranucleotide frequencies were proved to have significance regarding phylogeny [8]. Advances in indexing capacities allow to use longer words called k-mers for classification from genomic data and *state-of-the-art* algorithms have moved from composition analysis to longer k-mer presence/absence analysis [9]. For instance, Kraken [10], certainly the most popular tool for species identification from short reads, relies on long k-mers (k=31 by default). Using k-mers as identification signatures is one of the most used alignment-free approaches [11, 12, 13]. However, it raises two limitations : size of indexes may become gigantic to cover the number of available genome sequences, and error mitigation is low, as such tools relies on

[1] Collection of genomes, issued from a same environmental sample.

perfect matches to assign a taxa. We end this short review with a wiely used tool, PhyloPythia [14], relying both on partial alignment and k-mer frequencies. It is based on the selection of reads corresponding to specific genes, and then uses compositional approach to assign a taxa. However, it is designed for short reads and its reliability for erroneous long read sequences is quite impacted, as the supervising part included in the latest version relies on local alignment [15]. We ended up comparing ourselves to their results, as they also tested their software on the same size of data.

Phylogenies in bacteria are quite subject to changes, and well-known and used markers such as 16S are not so relevant [16]. Horizontal gene transfer is common in bacteria [17], a feature that is not accounted for by a classical purely tree-structured representation. As emerging genotypes are inferred by multiple relatives issued from the community diversity, a graph-like structure would be more adapted. Some alternatives, seemingly more respectful of evolutionary relations are currently being figured out [18], requiring our approach to accept changes in phylogeny to stay relevant.

1.2 Learning a classifier from genomic data

MinION sequencers need a basecalling[2] step to get the nucleotidic sequence. Guppy, a neural network based software responsible for this task receives frequent updates which increase its reliability, though 5 to 6% of errors represented as indels and substitutions remains across a whole genome [19]. In such conditions, matching a read on a genome may be tricky and composition-based approaches are more flexible on this particular kind of data. We will be focusing here on small words (k in-between 4 and 6) on a reduced alphabet $X = \{A, T, C, G\}$.

We aim to develop an alignment-free method, with a high resilience against sequencing errors, identifying the family of reads to feed this data to a strain identifier software named ORI[3], to lower its execution times by discarding non-relevant taxas.

1.3 A *state-of-the-art* classification algorithm : XGBoost

As this work elaborates on gigantic amounts of sparse and heterogeneous data, I was enjoined to implement XGBoost [22] as the core classifier. This scalable regression tree boosting algorithm is well-known for solving *state-of-the-art* supervised classification problems [23]. It learns binary

[2]Basecalling is the conversion from the continuous electric signal (fast5 files) issued from the pore of the sequencer to a discrete suite of nucleotide (fastq files). As the DNA flow rate through the pore is not constant, mistakes are easy to be made when converting analogous signal to a discrete series of bases.

[3]ORI (Oxford nanopore Reads Identification) [20] is a software developed at the INRIA of Rennes to identify strains from erroneous long reads ; based on a qgram analysis, it can yield results even on noisy data with a good confidence level [21].

regression trees. A binary regression tree is a structure that represents an iterable decision process [24]. At each step, a feature is tested and two branches are created, one for success and one for failure. The process is repeated until a leaf is reached, where a value of a predicted variable is estimated. To create reasonably-sized trees, a parameter named tree depth represents the maximum number of nodes that any path, from the root to a leaf, should not exceed. The word *regression* in regression trees stands for *regression function*. Rather than giving a raw probability to be of a specific class, each function returns a score that will be aggregated to create the output. For a dataset $D = \{(x_i, y_i)\}$ with n rows of m features x_i of values y_i, a tree-based learning model will use Z functions to predict output : $\hat{y_i} = \sum_{z=1}^{Z} f_z(x_i)$ for $f_z \in F$ where F is the regression trees space.

What makes XGBoost so powerful is its gradient boosting. Once a tree is done, meaning all paths leads to a leaf, a test set is processed through the tree. The data is weighted, miss-classified data get higher weights and well-classified data get lower weights. Another tree is grown in order to classify this data. Formally, it is about integrating multiple models in order to, from an ensemble of weak learners, emerge a strong learner [25] : the library uses a greedy selection algorithm [26], trying at each step to maximize the current classifier local performances in the global objective of increasing full classifier accuracy. To prevent the process of being endless, a number of boostings is set : it is the number of successive best function addition (meaning the ones which upgrades the best our model) to each node of our model. It can be visualized as the number of trees any prediction will go through, which does not implies that model will only be made of this number of trees. When XGBoost executes a model, it iterates through all kept trees of model and sums prediction of each tree of F : aggregation results in a score per class $\hat{y_i}$, which is returned as a vector. This vector can have different data types, according to the classification objective : it can be multiclass or binary depending on the number of classes we have in our training data, and can be a single class attribution or a probability tied to each class.

2 Materials and methods

This section introduces some names for genomes sets I will use for the remaining of the report :

- **Bernard's dataset** : 107 bacterial and archaeal genomes from [11] ; comprising 2 domains, 16 phyla, 24 groups, 44 orders and 76 families.
- **Supported dataset** : 129 bacterial and archaeal genomes issued from NCBI's public database *refseq*, where only a few families with *at least 3 representatives* were kept ; it comprises 2

domains, 8 phyla, 14 groups, 26 orders and 43 families.

- **Lactobacillales dataset** : 687 bacterial genomes from NCBI's public database *refseq*, where only the ones from the Lactobacillales' order were kept, splitted among 70 families.
- **Refseq dataset** : 16969 bacterial and archaeal genomes from NCBI's public database *refseq*, within 2 domains, 42 phyla, 104 groups, 247 orders and 3333 families. It is the whole bacteria and archaea *refseq*, with one strain per specie, as of may 2022.

2.1 Genomic data composition : studies across classification levels

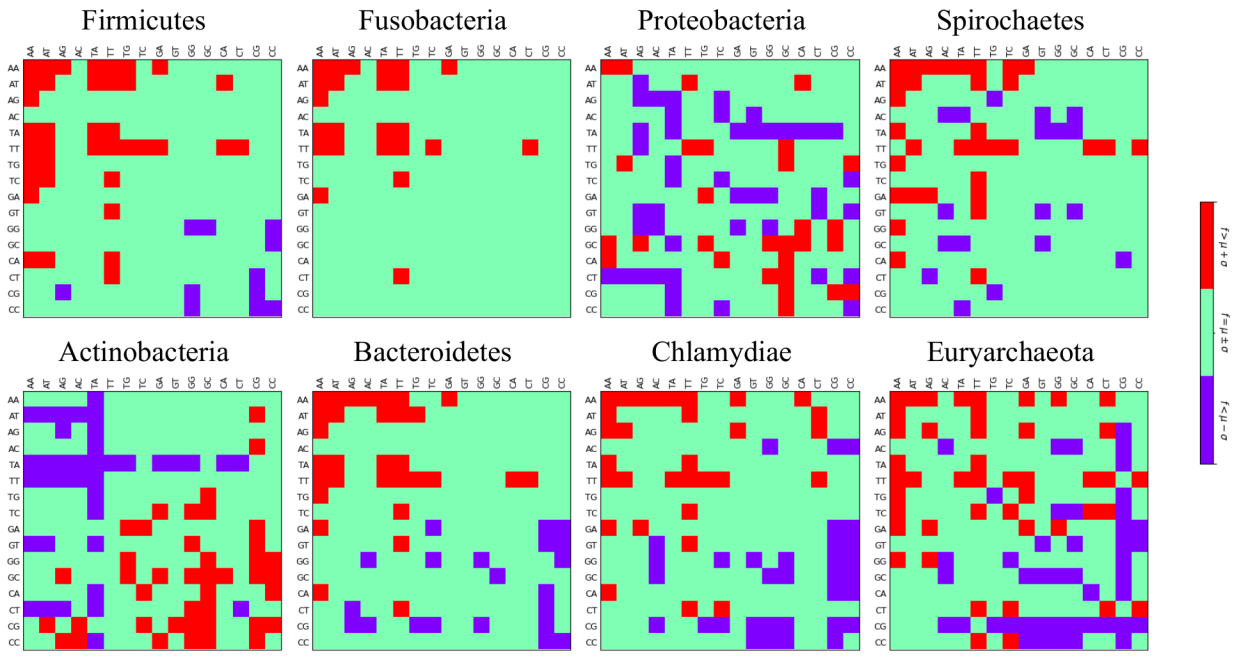

Figure 1. Distribution of k-mer frequencies across phlya from the supported database. Colors indicates if the mean frequency on all known complete genomes in a phylum is above, below or in-between global average plus or minus one standard deviation.

A crucial step in applying the XGBoost algorithm is to define the set of features. Frequencies of k-mers (Fig. 1), standard deviation from k-mer frequencies (Fig. 2) were computed for Bernard's and supported datasets at genome scale, each time resulting in distinct results per clade, across all test levels. Frequencies are not stable inside taxonomic units, giving us a notion of taxon signature. However, to learn how to divide lower ranks, we need to see if those signatures shows inner vatiations. All k-mers are not relevant : for instance, k-mers such as CCAC, TTAG, TCAG or ATGT shows little if no variability of frequencies across all observed genomes 2 and thus can't be used as high-level signatures (domain, phyla) but they might be relevant inside sub-populations where their variations could meet an acceptable threshold. For smaller fragments, as shown in Fig. 3,

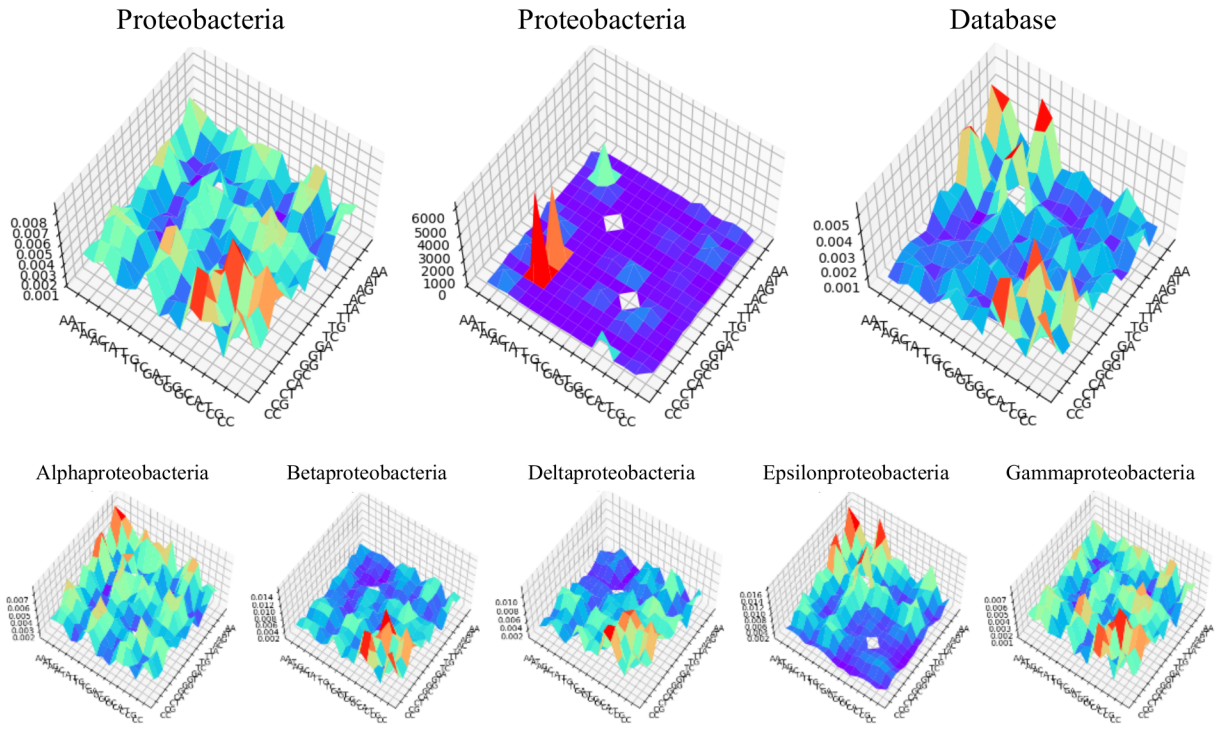

Figure 2. Top left : surface of σ_{4-mers} (standard deviation of 4-mers frequencies, estimated for each genome of the supported database on 100 randomly sampled fragments of 10.000 bp with seed=1111 and step=1) frequencies for *Proteobacteria* genomes of the supported database. Top middle : feature gain of the XGBoost model used for predictions of the groups of *Proteobacteria*. Top right : surface of σ_{4-mers} frequencies across all genomes of the supported database. Bottom : surfaces of σ_{4-mers} for groups of *Proteobacteria* in the supported database. Those five groups are assigned during prediction using the features displayed on the top middle.

some 4-mers frequencies are somewhat stable across reads issued from a same taxon though they are differently represented across all groups, meaning strong features could be associated to this taxon. Some others shows high variance, meaning that they're probably good candidates for a split at a lower taxonomical level. We settled on k-mer frequencies as our main set of features.

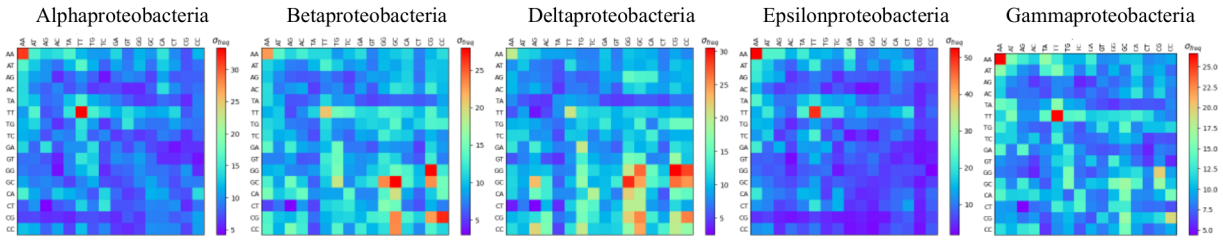

Figure 3. Matrices of mean of standard deviation of 4-mers counts. 4-mers frequencies were estimated for each genome of the supported database on 100 randomly sampled fragments of 10.000 bp with seed=1111 and step=1. The higher the value, the greater the frequency sparsity.

2.2 Machine learning of signatures with gradient-boosted regression trees

For all experiments described below, the same set of parameters was used to tune the learning algorithm. I used XGBoost version 1.6.1. Models were built in approximate-local mode[4]. For each, 10 boostings with the gbtree method were used and a tree maximal depth of 10. The classification objective was multi:softprob[5] to get raw matrices of probabilities. A step size shrinkage[6] of 0.3 was used with a minimal child weight[7] of 1 and all other parameters were let as default. Configuration files used for this report and the paper are available in the software repository. WISP makes a prediction for each taxonomic level, each one depending on the previous one. For each, the three steps described below apply : preprocessing, decision tree inference and postprocessing.

Preprocessing : from genomic to training data

The XGBoost library can use the LIBSVM [27] format, for which I developed a set of functions to read and write. Encoding is straightforward : each line starts with the dynamic database-dependant clade mapping (if any) ; for each feature, values are k-mers counts and keys are composed of A, T, C and G, which are respectively coded as 0, 3, 1 and 2 ; hence the feature 3312 is the 4-mer TTCG. Finally, if the species is known (reference dataset), its name is written as a commentary at the end of the line. I choose LIBSVM over simpler formats to allow on-disk operations from XGBoost, as this format bypass dataset size limitations. It will be mandatory to use the full NCBI's *refseq* as training set. For the *learning sets*, any reference genome is optimally divided in n fragments of size m, with the objective of maximizing coverage before depth, and features are k-mers filtered by a seed. Seeds are composed of 0 and 1, respectively discarding or keeping the nucleotide. In all our experiments, we used $m = 10000$. Databases were elaborated in two versions, $v1$ and $v2$. Those two versions used seeds 11111 for domain and phylum, and 1111 for group, order and family levels. In $v1$ databases, $n = 50$ fragments were sampled for domain and family, and $n = 100$ fragments for other levels. As of $v2$ databases, domain dataset is made of $n = 100$ fragments, and $n = 500$ fragments for other levels. As of the *unknown sequences*, they are splitted randomly in n' fragments of size m, ($m = 10000$ and $n' = 400$) and processed with the same parameters. All fragments are computed two times, for direct and reverse strand hypothesis. This way, we don't have to guess if

[4]Approximate mode splits the dataset at each node calculation, by quantile, and subsequently allows to use large datasets on disk, by only loading at each step subsets in the memory. It is mandatory for huge datasets as the ones we manipulate. Local re-defines splits at each node, which is better for deep trees and scenarios where data is lacking.

[5]Multi stands for multiclass attribution, and softprob for a vector-shaped return of probabilities tied to each class.

[6]Column subsampling ratio applied to grow each of the trees in order to escape overfitting.

[7]Minimal sum of weights to keep a child node in any tree, the higher the sum, the more conservative the tree.

our read is $3' \rightarrow 5'$ or $5' \rightarrow 3'$. I also filtered out k-mers corresponding to homopolymers, as basecalling errors of MinION reads are at 30% located in homopolymeric regions. Once the data is preprocessed, it is ready to enter the model.

On the usage of decision trees

Modelling and setting up the right learning parameters is the critical point of any supervised machine learning approach : a bad setting may cause the model to overfit learning data or to not learn any rule at all. XGBoost documentation came handy for those adjustments. In practice, basic tweaking took a lot of runs to get tangible results. Splitting rules are simple and self-explanatory, though their aggregation and relative importance are harder to capture in representations such as Fig. 4. Evaluation of any model requires a shared dataset to evaluate the impact of each individual parameter, select the ones giving the best and more robust results.

Figure 4. First tree of *Lactobacillales* order model computed upon genomes from Bernard's dataset (left) and a zoom on a single decision step (right). Exploring XGBoost models is possible thanks to graphical representation of splits, however as any model is made of hundreds of trees, other views such as feature gain can yield more relevant information about our data than the tree view.

For the tweaking step, a set of 20 *Lactobacillales* genomes was evaluated against Bernard's dataset. Objective was a tradeoff between high-speed indexing allowed by the small training and validation sets, while having real biological data at-hand to evaluate both performance and over-fitting of our method. At each run, one single parameter was changed. We investigated tree depths ranging from 6 to 12, number of boostings from 6 to 12, and k-mer size from 4 to 6, with and without spaced seed. I tuned parameters until having perfect recall on the 20 samples.

Postprocessing : discarding of spurious and ubiquitous data

After any prediction at a level l of the taxonomy, we obtain a matrix $M_{i,j}$ of predictions $\hat{y_i}$, crossing reads and taxa at level l : each read x is associated to a vector of predictions v_j^l for taxon j at level l. All reads are not meaningful : in the case x was sampled inside highly-shared and conserved sequence, x might be associated to many classes with roughly the same confidence level. To discard ubiquitous or ambiguous reads, we have implemented dedicated threshold functions.

We apply a softmax function[8] $\sigma(v)_j = \frac{e^{v_j}}{\sum_{k=1}^{K} e^{v_k}}$ on each vector of $M_{i,j}$ to force $\sum_{j=0}^{x} v_j^l = 1$.

[8]Function that normalizes a vector of K real numbers into a probability distribution of K discrete outputs.

A threshold function is then applied. For instance, if $max(v_x^l) - \mu(v_x^l) \geq \theta_x$, then prediction $\hat{y}_i$ is kept, else it is discarded. With low quantities of data, we might also encounter for an important part of reads tied on taxa with very similar scores and we have decided to retain several alternatives during classification. To explore secondary paths, we have introduced a threshold θ_{reads}, and the classifier outputs every class ψ satisfying $\frac{\sum_{i=0}^{n} v_\psi^l}{||i||} \geq \theta_{reads}$.

2.3 Validation : from simulated data to MinION lectures

Classifier validation : generation and binning of mock reads

Once XGBoost and its support functions were implemented, I had to validate the classifier. For this purpose, a mock dataset containing 20 randomly generated false genomes of 100.000 bp was used. Each sequence was split in 50 random fragments of 10.000 bp, which were used as training set. A certain level of errors (1-5% indels/SNP) was induced inside the generated sequences ; a test set containing 10 fragments of 10.000 bp for each sequence was created, and we tried to identify, for each fragment and each set of fragment, the 'species' it was belonging to.

Integrating biological data : considerations regarding phylogeny

Looking at k-mer frequencies and XGBoost models, I figured that higher taxonomical rank features were not the same as lower-rank ones. From this observation, we proposed to build predictions hierarchically, and use at a taxonomic level the dataset of genomes corresponding to the taxon predicted at the level just above. The taxonomic tree is built on-the-fly (example on Fig. 6), according to genomes present in the database ; we selected five levels (domain, phylum, group, order, family) for multiple reasons. First, they are the most universal to retrieve from NCBI's taxonomy, being the five most well-annotated levels ; second, internship objective was to determine sample's family ; and third, each of those levels represents a different set of classification cases and challenges, be it by the number of classes or sparsity of representatives.

Accuracy estimations : Bernard's and supported datasets

In order to validate in conditions as close as possible to real-life experiments, we elaborated a *baseline* and a *leave-one-out* scenario. It is about, respectively, letting the tested sample in the training set, and removing it from the training set. This way, we seek to emulate two applications : retrieving the taxon of an already known genome or guessing it for a nearby observed organism. In the first case, we aim to get a fully accurate classification ; in the second, we want to have the most specific taxonomic level possible. To simulate at best MinION reads for all species, errors

were simulated according to the known noise level of MinION reads basecalling [19] by randomly adding, deleting or substituting one nucleotide[9] with a fixed probability of error all along the read. This way, we validated we could scale to even larger datasets and were ready for real datas.

Real case scenario : Lactobacillales binning

In order to evaluate WISP performances in a real usecase, software was tested against a metagenomic sample sequenced by E. Roux, composed of 200.301 reads of Lactobacillales above 10.000 bp, spreaded in-between 11 distinct barcodes (Tab. 3). Two experiments were made : firstly, algorithm was set to bin without demultiplexing[10], and every read was assigned to a class. Secondly, a run was made on demultiplexed assemblies, to evaluate performance on real MinION data.

3 Results

3.1 Simulated data : from read to sample binning

Error-free data : maximum awaited accuracy

In this first case, binning of respectively 42.800 and 51.600 reads was realised for Bernard's and supported datasets. Reads were random samplings of 10.000 bp in reference genomes, and each read was predicted independently. It is the easiest prediction problem, as there's no sequencing error induced. Consensus is set according to $max \sum_{i=0}^{n} \hat{y}_i$. A read prediction is considered correct if it is attributed to its real class and a sample prediction is correct if the consensus of the prediction matrix is equal to the sample class. We obtained overall very good results (see Tab. 1), with relatively few data. For instance with the supported dataset, for respectively domain, phylum, group, order and family, mean size of the learning set is 6450, 6450, 1613, 921 and 248 sequences.

During this validation process, I built confusion matrices (Fig. 5) to display success rates for our classification algorithm for each taxonomic level. This allow to control accuracy, specificity and sensitivity for the current database. Those matrices displayed that distinction between phyla *Actinobacteria* and *Proteobacteria*, and families *Shigella* and *Escherichia* were ambiguous, observations we also made later on with leave-one-out scenarios.

Erroneous data : simulating MinION reads

Runs on mock erroneous reads sustained acceptable results, as displayed in Tab. 2. Errors in classification are focused in a few taxa (see Fig. 6) : some families got a recall of 1, while others felt

[9] Goal was to emulate indels and SNP's, which can both be present in Guppy's outputs.
[10] Goal was to emulate a lifelike scenario on a metagenomic sample.

Table 1. Accuracy (Acc), specificity (S_p) and sensitivity (S_n) of assignments of individual errorless reads for the supported database. The row *Global* highlights the cumulated loss through all levels.

	baseline			leave-one-out		
	Acc	S_p	S_n	Acc	S_p	S_n
Domain	99.70	99.91	99.91	99.33	99.69	99.69
Phylum	96.37	99.70	99.34	88.20	98.47	90.37
Group	99.29	99.99	99.98	92.99	99.51	89.95
Order	99.10	100	100	95.41	99.83	96.63
Family	99.15	100	100	90.92	99.80	92.09
Global	93.74	99.60	99.23	70.66	97.32	72.11

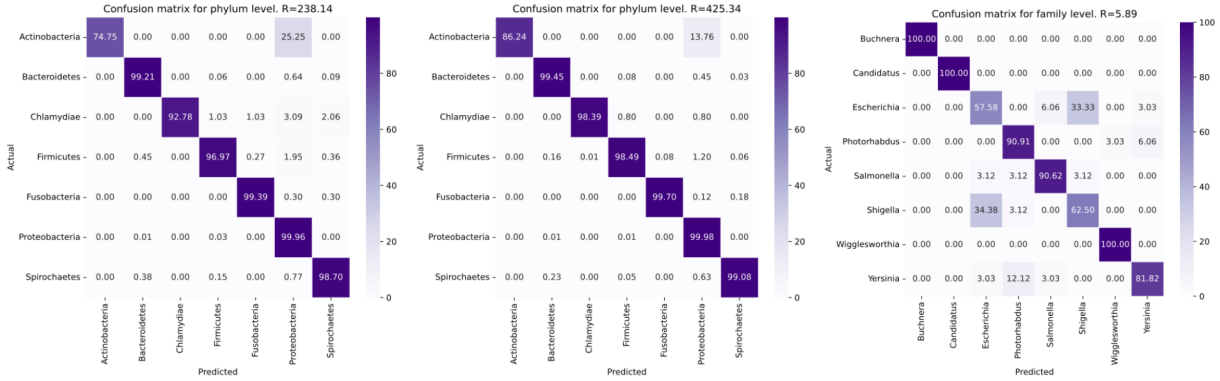

Figure 5. Read confusion matrices for phyla level, assuming domain is *Bacteria* (left and center) and family level, assuming order is *Enterobacterales* (right). Percentages are computed by line ; for instance, on the left figure, 96.97% of *Firmicutes* reads were correctly assigned. They were averaged over 10 cross-validation tests on our trained model. Left and right figures are issued from the supported database with 100 samplings in each genome, whereas center figures is issued from the same reference set, but with 500 samplings per genome. Ratio R is calculated by dividing the sum of correctly assigned reads by the sum of classification errors.

down completely. Those fails are not due to lack of data, as the dataset was for instance perfectly balanced at family level (3 representatives for baseline, 2 for leave-one-out) and sampled with the same *maximum coverage* method. Inspecting signatures gives us clues on the issue : both sparsity inside taxa (see Fig. 3) and proximity with others (see Fig. 1) hurts algorithm capacity to create models that accept some deviation from training set.

Comparisons of results for erroneous and errorless runs are displayed in Fig. 7. It shows recall rates in *leave-one-out* scenario on samples of 400 reads. To read Bernard's dataset results, one must keep in mind that all taxa that are absent from the learning set aren't represented.

3.2 Real dataset : binning of a MinION metagenomic sample

Runs on raw reads without demultiplexing showed satisfactory results with the Lactobacillales training set (Fig. 8), with solely 7% of reads assigned to families we did not have in our sample, in a split with 70 classes. As we could not get from demultiplexer a table matching reads and barcode but

Table 2. Accuracy (Acc), specificity (S_p) and sensitivity (S_n) of assignments of individual 6% destroyed reads for the supported database. The row *Global* highlights the cumulated loss through all levels.

	baseline			leave-one-out		
	Acc	S_p	S_n	Acc	S_p	S_n
Domain	99.85	99.93	98.87	99.19	99.60	99.60
Phylum	93.40	99.43	98.88	83.60	97.64	89.30
Group	98.71	99.98	99.89	91.40	99.42	93.24
Order	98.46	99.99	99.88	95.05	99.87	96.25
Family	98.27	99.99	99.74	88.80	99.74	88.33
Global	89.06	99.32	97.28	63.97	96.31	70.51

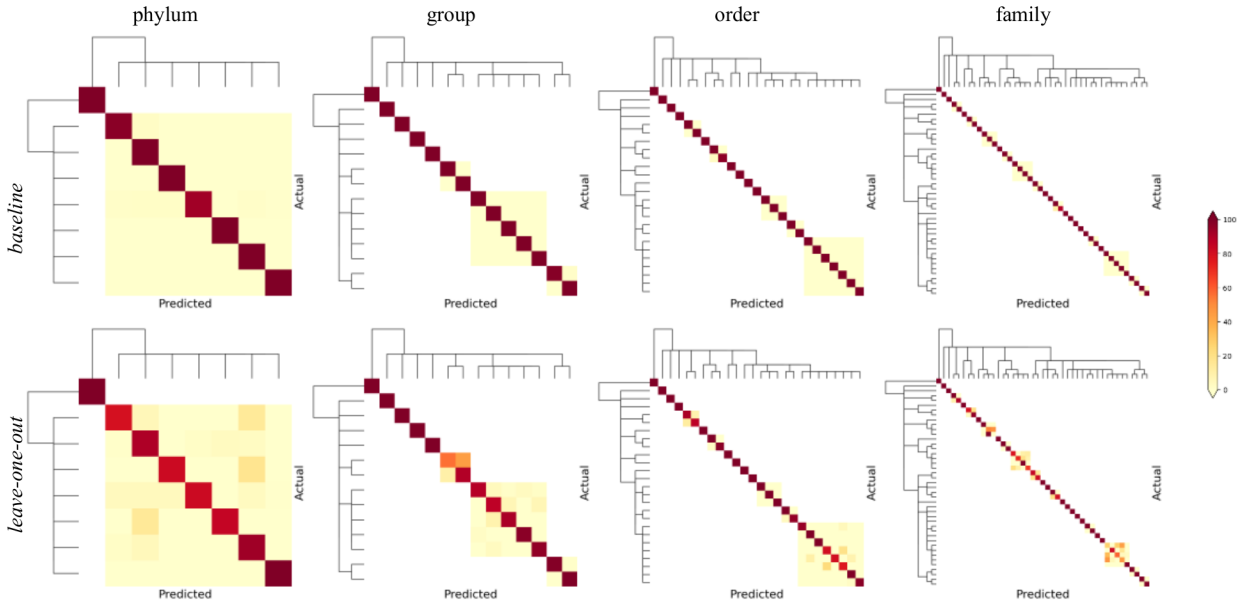

Figure 6. Individual read assignments at (from left to right) phylum, group, order and family for baseline (top) and leave-one-out (bottom) scenarios with the supported database. All reads include 6% SNP/indels errors. Dendrograms give the taxonomy used at the previous taxonomic level. White areas are the classification cases our algorithm will not encounter.

only assemblies, we could not output on this sample mesures such as accuracy[11], sensitivity[12] and specificity[13] as we did before. As of results at sample level, results over assemblies are displayed in Tab. 3. Like before, consensus helps to mitigate hard to assign reads, as they are splitted among all samples. As of CIRM-BIA658 and CIRM-BIA315, it is quite surprising with the supported database to see *Desulfurovibrio* as this family is part of the *Deltaproteobacteria* group.

[11] Accuracy (Acc) is a metric qualifying how many correct predictions were made out of the total set of prediction, by dividing those two numbers.

[12] Sensitivity (S_n) refers to the true positive rate over a set of predictions.

[13] Specificity (S_p) refer to the true negative rate over a set of predictions.

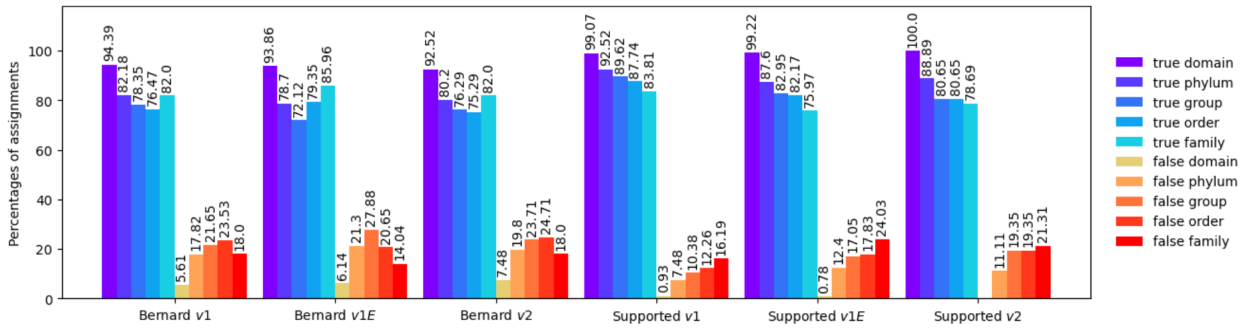

Figure 7. Global prediction results in *leave-one-out* scenario, for Bernard's and supported databases. For $v1$ style database (resp. 50, 100, 100, 100, 50 samplings in reference genomes for domain, phylum, group, order and family) results are shown with ($v1E$) and without ($v1$) errors. As of $v2$ style (resp. 100, 100, 500, 500, 500 samplings in reference genomes for domain, phylum, group, order and family) results are only for errorless reads, as we had worse results in the best case scenario than $v1$ style.

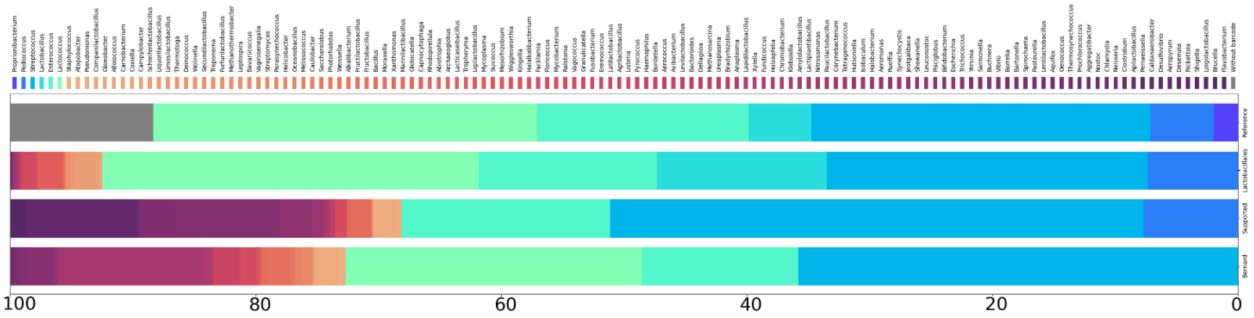

Figure 8. Percentages of read assignments using Bernard, supported and Lactobacillales train sets (bottom to top) on the Lactobacillales metagenomic sample. Reference (top) is outputted from demultiplexer, and gray are reads without attached barcode. Tints of blue are families that are in the metagenomic sample, tints of red are badly binned reads.

4 Discussion

4.1 Key determination factors

Learning database is, as illustrated previously, a key factor to any determination. Number of relatives in the database of the taxon to be predicted has a direct impact on prediction quality. If we sample too much, the learned model may overfit the training data ; as we observed in our tests between $v1$ and $v2$ type databases (Fig. 5), we lost prediction power at read level for leave-one-out scenarios by sampling more in our references genomes, while confusion matrices showed better results : this overfitting evidence is comforted by our global results, displaying less well-attributed species with any $v2$ database than $v1$ (Fig. 7). We also need to look out for a tradeoff between diversity and relatives. Supported database was sampled from a small part of the *refseq* and show on the MinION metagenomic sample results worse than the less weighted, smaller Bernard's database (Fig. 8) even if with our validation process supported datased showed far better results (Figs. 6, 7).

Table 3. Consensus results for assembly-based experiment. CIRM-BIA1 is not in the Lactobacillales order but in Actinomycetales ; and as Streptomyces and Mycobacterium are inside this order, it is not surprising to see such assignments.

Studied case			Presence in training set			Determined family		
Barcode	CIRM code	Specie name	Bernard	Supported	Lactobacillales	Bernard	Supported	Lactobacillales
01	BIA1	Proprionibacterium freudenreichii	no	no	no	Streptomyces	Mycobacterium	Schleiferilactobacillus
02	BIA1328	Enterococcus faecalis	yes	yes	yes	Enterococcus	Enterococcus	Enterococcus
03	BIA44	Lactococcus lactis	yes	no	yes	Lactococcus	Streptococcus	Lactococcus
04	BIA79	Lactococcus cremoris	relative	no	yes	Lactococcus	Streptococcus	Lactococcus
05	BIA1640	Pediococcus parvulus	no	relative	yes	Lactiplantibacillus	Pediococcus	Pediococcus
06	BIA76	Lactococcus garvieae	relative	no	yes	Streptococcus	Streptococcus	Lactococcus
07	BIA658	Lactobacillus delbrueckii	no	no	yes	Lactiplantibacillus	Desulfovibrio	Lactobacillus
08	BIA315	Lactobacillus delbrueckii	no	no	yes	Lactiplantibacillus	Desulfovibrio	Lactobacillus
09	BIA739	Enterococcus faecalis	yes	yes	yes	Enterococcus	Enterococcus	Enterococcus
10	BIA1638	Streptococcus equinus	relative	relative	yes	Steptococcus	Streptococcus	Streptococcus
11	BIA20	Streptococcus thermophilus	relative	yes	yes	Streptococcus	Streptococcus	Streptococcus

As I was limited in time, I made all experiments with 10.000 bp reads, though the median length of MinION reads is in-between 7000-8000 bp, and did not run all experiments with the *refseq* set. Tuning parameters for lower-sized reads would expand coverage, and thus, usecases.

4.2 Confronting and comforting current classification

Confusion between the *Shigella* and *Escherichia* families is a well-known classification issue [28] and they should be merged ; with Bernard's dataset, our algorithm is only assigning correctly from 40 to 60% of reads, which is a coherent result given that the higher level only contain those two classes. It also emphasize limitations of our model : the differences between those families are more of a subspecies/strains nature, and oour method is not suitable for such fine distinctions. Across all our experiments, phyla seemed to be the hardest taxonomic unit to capture, be it for reads or samples of a few hundred reads. A closer look at results showed most of errors are focused on *Fusobacteria* and *Firmicutes* ; 14.9% of *Fusobacteria* are badly assigned in results (Fig. 6), though this error rate was greatly mitigated from the 40.4% miss-classifications we had with Bernard's dataset, emphasizing the need to balance the learning set. Once again, it does not come as a surprise : those two phyla are close together genome-wide [29] supposedly because of horizontal gene transfer, while being two physiologically and ecologically distinct units.

4.3 Overview of WISP pros and cons : classification biases

Early ending of classification

As our methodology relies on defining the nearest neighbor of the sample species in a set of reference genomes, we can only assign a taxon if at least one reference genome exists for this taxon. One could have a good assignment at higer taxonomic level, but the problem is that our algorithm always tries to reach the family level. To lower this effect, we could imagine to build our

own taxonomic tree from k-mer frequencies ; such approaches has already successfully been used to generate new taxonomies [30]. We can calculate it in the form of a set of 2-dimensional proximity graphs. Nodes in a same graph would belong to the same taxonomic level. Then, we could then settle each read of the sample inside this space, and compute the average distance to each node of each level to output exploitable results. Explored classes would then not be the classes matching the threshold θ_{reads}, but the p nearest neighbors of each prediction $\hat{y}_i$. We also thought of incorporating ORI-like feature ; for reads binning, seek to minimize the total number of families that qualifies the maximum of reads. Another perspective is to create databases from different sets of genomes at each level and for each taxa, to query more precisely at each step while maintaining decent prediction speed ; as illustrated in Fig. 8, Lactobacillales set was precise at determining proportions ; and though others datasets had around 30% of reads miss-attributed to some other classes, consensus was converging to Lactobacillales.

Algorithm adaptability to taxonomy evolution

Currently, WISP retrieves the NCBI's taxonomy for each genome in the reference set during the building phase. When a read is assigned to a class, it serves to determine lower-rank possible taxa list. It implies that given any set of genomes, classification will be linked to this genome set and that if a change occurs inside classification, one would only need to retrieve the new taxonomy.

Bacteria domain does not only acquire genomes changes through reproduction, as horizontal transfer do complicates classification. The *tree of life* paradigm requires the use of fairly high thresholds in the prediction to anticipate and tolerate future horizontal transfers. Another perspective is to use more accurate taxonomies like the one from EMBL-EBI [31].

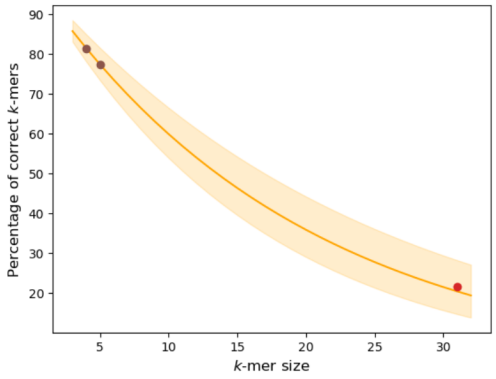

Figure 9. Percentage of expected correct k-mers after basecalling. The envelope displays impact of error rates in-between 4% and 6%, and the line represent a 5% error rate that the newer version of Guppy is supposed to have. Red dot represents the estimated error rates for each k-mer used by Kraken (using $k = 31$). Brown dots are the estimated error rates for each of the k-mers used by WISP.

On the use of small k-mers to dodge more errors

Smaller words helps skipping errors, as those are more distributed between features which individually have higher probability to be correct, than longer words requiring perfect matches (Fig. 9). XGBoost models uses only few features from the data we provide (Fig. 2) so we have additional chances to skip erroneous data. More-

over, as the models only consider if count of a k-mer is above or below a threshold, reducing amount of something which is already below threshold or increasing quantity of something which is already above won't matter for final prediction.

4.4 Complexity and scaling

Creating a training set and learn on it is the most time-consuming step. Those two steps in $v1$ style took respectively 1:30h, 2:30h, 2:00h and 3-8:00h for Bernard's, supported, Lactobacillales and refseq genomes sets. It is unclear how to introduce useful optimizations that XGBoost is not already considering (e.g. feature preselection). Optimisations are still to be made. We can hardly do better solely with python, as we currently use optimised types, generators and parallelism. Model complexity has a huge impact on prediction time. Respectively on Bernard's, supported and Lactobacillales datasets processing 200.301 MinION reads took 29:50h, 47:50h and 23:40h on 4 cores, peaking at 55 Gb of RAM used, that is, an average run time of respectively 0:54s, 0:86s and 0:43s per read. As our algorithm spawns one process per core, software has a linear scaling when increasing allocated resources as long as it is not limited by disk access speed.

4.5 Comparison to other algorithms

We wanted to put our results in perspective to other binning algorithms. Kraken [10] is commonly used for this task, and PhyloPythia [32] was tested on long reads. I also tried to install MetaMaps [3], and a VW implementation [9], but all software failed at some point to install. We ended by comparing ourselves solely on theorical PhyloPythia's results, which gives a rough baseline. On the same size of fragments, for known genome samples (baseline), we obtained an all-levels sensitivity/specificity range 99.4-100% (PhyloPhytia 97.7-98.5%) / 100% (PhyloPhytia 83.7-98.0%). For unknown genomes (leave-one-out), we obtained at all levels 90.0-99.9% sensitivity (PhyloPhytia 80.0-87.9%) and 98.5-99.8% specificity (PhyloPhytia 81.7-92.1%).

5 Conclusion

Through this work, I elaborated a reliable and documented software, tested as of its architecture. Results emphasized the strengths of compositional approaches on erroneous data. We showed that taxa-specific k-mers frequencies could emerge from experiments : cross-validation can identify hard-to-determine taxas. Though is it hard to extract splitting rules from XGBoost trees (see Fig. 4), feature gain is a highly valuable information for model analysis. One of the most interesting perspective is to scale this work to huge datasets, e.g., the whole NCBI's *refseq* database.