Distinguishing Gene Transfer Agents from their Phage homologs

Taylor Neely

Advisor: Olga Zhaxybayeva

Department of Computer Science, Dartmouth College

June 2, 2016

Abstract. Gene Transfer Agents (GTAs) are viral-like particles that facilitate horizontal gene transfer. Identification of GTAs in bacterial genomes by traditional search methods such as BLAST is problematic, as GTAs closely resemble phages, and homologs encoding for both GTAs and prophages will be picked up by a BLAST search. In order to identify new GTAs and distinguish them from their phage homologs, we developed a machine learning approach for characterizing and classifying GTA and phage genes. Our software, GTA_Hunter, is capable of distinguishing between known GTAs and phages with high accuracy. GTA_Hunter identified new true GTA genes and ruled out other GTA homologs from further investigation. Furthermore, our results suggest that orders *Rickettsiales* and *Rhodospirillales*, previously thought to not have GTAs, may contain some of the genes that encode for GTAs.

Keywords: gene transfer agent, machine learning, support vector machine, identification

1 Introduction

Microbial cooperation is evolutionary phenomenon in which organisms act in the benefit of each other, sometimes at the cost of their own well-being. Microbes have a host of social traits that allow them to interact with one another, including quorum sensing, biofilms, chemotactic signaling, and horizontal gene transfer [1]. These traits are vulnerable to cheating at the individual level, making their continued propagation within the population a complex problem. One way microorganisms can benefit one another is through the sharing of genes. By sharing environmentally favored genes, the fitness of the recipients increases [2]. Acquisition of new genes through horizontal gene transfer (HGT) can have significant evolutionary consequences [3]. Various mechanisms of HGT can allow for this gene exchange to occur. One specific mechanism for HGT is through Gene Transfer Agents (GTAs).

GTAs are virus-like particles produced by and encoded within some archaeal and bacterial genomes [4]. Unlike bacterial viruses (phages) GTAs are too small to encapsulate all of their encoding genes and instead package random pieces of the host DNA [5]. When a GTA-producing cell lyses and releases GTA particles, the

packaged DNA can be delivered to neighboring cells and potentially introduce novel traits into small fraction of the population. It is hypothesized that this facilitated method of HGT allows for and assists cooperative behavior among a given microbial population through the exchange of beneficial genes at the cost of the individual cell. The presence of GTA could affect the evolution of cooperation in microbial populations. To test this hypothesis, we first must be able to correctly identify GTA genes within genomes.

The first GTA was discovered in *Rhodobacter capsulatus*, a bacterium within the *α-proteobacteria* class, and is referred to as RcGTA [6]. RcGTAs are encoded by multiple genes in the genome, 17 of which are found together in a structural cluster. However, the majority of GTA genes have homologs in *bona fide* phages, and as a result, sequence similarity based search methods such as BLAST are not able to distinguish GTA genes from their prophage homologs [7] in bacterial genomes. Therefore, current computational tools are inept for correct GTA gene classification and as a result unable to confidently identify GTA genes from the structural cluster. While sequences similarity searches have proven inadequate in identifying true GTAs, there should still be ways in which GTAs differ from their prophage homologs, as GTAs are functionally different from phages and also appear to evolve at a much slower rate [4]. To address this, we have developed an alternate approach to correctly classify true GTA genes amidst the collection of homologs of both GTA and phage origins that were identified by previous methods. This approach can be considered as a sequence composition method in contrast to traditional sequence similarity methods such as BLAST [8].

For a sequence composition method to work, it is necessary to characterize each DNA or protein sequence beyond just its sequential order and capture a more comprehensive view of the gene as a whole. One way to do this is by breaking down each gene into a series of k-mers, strings of consecutive nucleic or amino acids in a given nucleic acid or protein sequence, and their respective frequencies. We use amino acid k-mers, as amino acids are more varied and evolutionarily constrained than nucleic acids. There is bias towards specific amino acids due to selection on protein function. It has been shown that k-mers work well for gene characterization in a number of similar biological scenarios [8,9,10]. Additional sequence composition methods include pseudo amino acid composition (PseAAC) [11] and physicochemical property-based composition [9].

In addition to the typical amino acid frequencies used in amino acid composition models, PseAAC captures sequence-order specific information by looking at pairs of adjacent or nearby amino acids and scoring them based on the difference of their physical properties – hydrophobicity, hydrophilicity, and side-chain mass. Physicochemical property-based composition takes a similar approach, expanding on the specific amino acid properties while losing the sequence-specific information. In this approach, amino acids are grouped by their physicochemical properties, such as residue charge, polarity, and ability to form bonds.

Because sequence composition approaches are able to numerically represent a protein sequence as a vector of respective values across any and all genes, these quantitative methods can then be used by mathematical

models such as machine learning algorithms. Machine learning classifiers are able to learn the distinguishing features of various families of genes and assign a family to unclassified genes. We can therefore use such an algorithm to distinguish true GTA genes from their phage homologs.

There are various machine learning algorithms that can be used for our application. Because we have a set of genes that we know belong to GTA and viruses , we can take a supervised-learning approach. Supervised learning is done by defining a function from labeled training data that best describes the classes involved – in our case, GTA and virus. There are a variety of supervised approaches, including K-nearest neighbor (KNN) [12], naïve Bayes [13], artificial neural network (ANN) [14], and support vector machine (SVM) [15]. Because we will be working with small, unbalanced datasets, our choice in classifiers is limited. For example, naïve Bayes requires a balanced dataset [16], while an ANN typically requires a large dataset [17]. Due to these constraining factors, we opted for an SVM classifier due to its good performance with small and unbalanced training sets [16].

GTA_Hunter includes tools for both feature generation and classification. Using training sets consisting of GTA homologs from GTA clusters in α-proteobacteria genomes and bona fide phages, we built, trained, and evaluated a weighted SVM and then applied it to classify uncertain GTA homologs as either GTA or virus. The SVM performed well for the majority of GTA genes, with accuracy scores ranging from 93.33% to 100%. Classification of GTA-like genes revealed potential GTAs previously missed by other methods. Here, we show that our sequence composition method and classifier can distinguish between GTA and phage genes.

2 GTA_Hunter

We implemented our software in Python 3.5.1. The code can be found on GitHub at `https://github.com/tneely/GTA_Hunter`. Figure 1 outlines our classification pipeline. More detailed descriptions of our feature generation, weighting, and SVM algorithms can be found in section 3, Methods.

Fig. 1. Processing pipeline. RcGTA homologs are identified using BLAST [7]. Known GTA and viral genes are clustered by their phylogenetic distance and assigned weight based on cluster size. Amino acid sequences are then converted into a set of features to be used in the SVM. The SVM is first trained on known GTA and viral genes. After training, unknown genes are then assigned a class by the SVM.

2.1 Weighting

Our access to bacterial genomes is limited to those sequenced and made available by other labs. This can result in a genome sequencing bias, where research interests drive the sequencing of many genomes specific to a certain taxonomic group. Due to this bias, our training data may contain an abundance of GTA homologs from phylogenetically close organisms. we want to reduce how much closely related gene sequences impact training the classifier. To do so, we use a pairwise distance matrix produced by RAxML (version 8.2.4) using the PROTGAMMAJTT model [18] after alignment by MUSCLE (version 3.8.31) [19] to cluster genes base on their phylogenetic distance to one another. GTA_Hunter takes these distance matrices as input in order to assign gene weights. Genes that cluster together are assigned a fraction of their original weight based on their cluster's size. The extent of the clustering can be controlled by distance threshold t.

2.2 Features

Feature generation is a necessary step for classification. GTA_Hunter takes as input a collection of genes represented as amino acid sequences in FASTA file format. The amino acid sequences of genes are transformed into a feature set using one or more of the following sequence composition methods:

- k-mer composition: amino acid sequences are broken up into "words" of non-negative length k and their frequencies counted for each gene.
- Pseudo amino acid composition: amino acid frequencies for the 20 native amino acids are counted for each gene. Chemical differences between adjacent amino acids are summed to create additional features. The number of features and level of adjacent or nearby amino acids compared is determined by the parameter λ with $0 \leq \lambda \leq L$, where L is the length of the protein sequence.
- Physicochemical composition: amino acids are classified into one or more of 19 physical and chemical properties. There properties are summed over all amino acids in the sequence of each gene.

2.3 SVM

GTA_Hunter utilizes a weighted soft-margin SVM as the classifier. The SVM is trained on known GTA and viral genes using the feature and weight inputs described above. The soft-margin can be tuned through the non-negative parameter c, which controls how important it is to satisfy the SVM margin constraint. The larger the c value, the less value we place on correctly classifying the genes during training. After training, the SVM is capable of classifying GTA homologs of unknown origins as either GTA or virus. The SVM currently supports three kernels capable of mapping the features to higher dimensional implicit feature spaces.

– Linear: a simple inner product between two feature sets.

– Polynomial: the dot product of two feature sets raised to a user defined power m.

– Gaussian: two feature sets are transformed according to the Gaussian function and user defined σ.

3 Methods

Fig. 2. The main RcGTA structural cluster. Genes are represented by arrows and are numbered from 1 to 15. Genes are scaled in respect to the length of the encoding sequence. Genes colored blue denote genes with viral homologs that were used in the SVM.

3.1 Training set preparation

To effectively train the SVM so that it can differentiate between GTA and viral genes, a training set of true GTA and viral gene homologs is required. Our training sets, shown in Table 1, for use in the SVM classifier consist of homologs of RcGTA structural cluster genes from 257 complete α-proteobacterial genomes and *bona fide* phage genomes from Viral RefSeq database (release 71) [20]. To identify "true RcGTA" genes, 17 genes from RcGTA structural cluster were used as queries in BLASTP (version 2.2.30+; E-value < 0.001; query and subject overlap by at least 60% of their length) [7] searches against the database of complete α-*proteobacterial* genomes (downloaded from the NCBI ftp site at `ftp://ftp.ncbi.nlm.nih.gov/genomes/Bacteria/`). Furthermore, in order to minimize the inclusion of prophage homologs, only ORFs that had other structural cluster homologs (> 8) within $\sim$ 10 kb were used to build Position Specific Scoring Matrices that were used to search the database again to find distant homologs with other structural homologs within $\sim$ 10 kb [21]. GTA homologs that appeared in a genome together with at least 8 other structural cluster homologs will be referred to as "large cluster" GTA genes, and those with less than 8 supporting homologs are "small cluster" GTA genes. The genes from large clusters were then classified as "true" RcGTA [Shakya et al., unpublished]. To identify viral homologs, RcGTA genes were used as queries in a BLASTP (E-value < 0.001; query and subject overlap by at least 60% of their length) search of a database consisting of the Viral RefSeq database (release 71) [20]. Due to few or no viral homologs for some of the genes in the structural cluster, we limited our training set to just 10 genes [Figure 2]. Table 1 denotes the gene functions and the number of GTA and viral homologs per gene.

Table 1. Training sets of known GTA and viral genes derived from RcGTA homologs found in α-proteobacteria and phage genomes, respectively.

RcGTA gene	Protein function	# of GTA genes	# of viral genes	Total # of genes
g2	Terminase	78	232	310
g3	Portal	78	243	321
g4	Prohead protease	71	181	252
g5	Major capsid protein	73	219	292
g6	Head-tail adapter complex	75	68	143
g8	Head-tail adapter complex	84	64	148
g12	Glycoside hydrolase	75	15	90
g13	Tail protein	70	41	111
g14	Cell wall peptidase	77	89	166
g15	Tail protein	68	39	107

While our GTA selection process chose likely GTAs due to the requirement that they must be part of large clusters, this does not guarantee that all homologs are true GTAs. To ensure our training sets only contain true GTAs and viruses, we manually curated the training set to remove genes that could be viral based on their phylogeny. Phylogenetic trees were constructed from sequences aligned by MUSCLE (version 3.8.31) [19] using FastTree (version 2.1, JTT+CAT substitution model) [22]. Genes that did not group together with genes of the same class (GTA or viral) were removed. In all, 52 GTA homologs were removed across all 10 RcGTA gene training sets.

3.2 Test set preparation

Our trained SVM classifier was used on GTA homologs from α-proteobacteria genomes that did not satisfy the conditions above to be included in the training set. The test sets were derived from the same search method used for identifying GTA in the training set. Genes that belonged to small clusters (less than eight structural cluster homologs) were included as part of the test set [Table 2].

The majority of bacterial genomes sequenced contain prophage sequences within them [23]. While GTAs are currently known to be in four taxonomic groups, we believe that their existence may be more widespread and that many of the assumed prophage sequences are in fact be GTA. To test this hypothesis, we looked at homologs of RcGTA genes within *α-proteobacteria* that were found in prophage regions identified jointly by PhiSpy [24], Phage_Finder [25], and PHAST [26] using BLASTP (version 2.2.30+; E-value < 0.001; query and subject overlap by at least 60% of their length) [21]. These were included in a separate test set [Table 2].

Table 2. Small cluster (< 8 genes) and putative prophage RcGTA homologs evaluated by the SVM classifier. Numbers denote number of GTA genes classified and (total number of evaluated genes).

RcGTA gene	Protein function	Small cluster	Prophage
g2	Terminase	13(16)	0(3)
g3	Portal	12(92)	2(26)
g4	Prohead protease	36(72)	11(20)
g5	Major capsid protein	11(76)	0(13)
g6	Head-tail adapter complex	9(11)	4(5)
g8	Head-tail adapter complex	3(4)	2(2)
g12	Glycoside hydrolase	18(63)	2(2)
g13	Tail protein	25(49)	3(5)
g14	Cell wall peptidase	17(26)	3(3)
g15	Tail protein	17(37)	2(8)

3.3 Feature generation

There are many possible ways to characterize a proteins sequence. To characterize the protein sequences for use in a machine learning environment, each gene is represented as an n-dimensional vector. When using k-mers for the feature set, each of the n unique k-mers in the training set are represented for each gene. k-mers are generated by scanning along each protein sequence with window size k. Once k-mer generation is completed, each sequence is then re-scanned and its k-mer frequencies transformed into a feature array.

Pseudo amino acid composition (PseAAC) is a method of characterizing proteins sequences without completely losing sequence-order information [11]. The first 20 features of PseAAC denote the frequencies of the 20 native amino acids that encode for proteins. Additional features are sums of rank-difference correlation factors, where we compare amino acids i and $i+\lambda$ over the average of three different properties: hydrophobicity [27], hydrophilicity [28], and side-chain mass [Figure 3].

We define the feature set $F = (f_i)$ for PseAAC as the following:

$$f_i = \begin{cases} \dfrac{r_i}{\sum_{i=1}^{20} r_i + w \sum_{k=1}^{\lambda} s_k}, & \text{if } 1 \leq i \leq 20, \\ \dfrac{w s_{i-20}}{\sum_{i=1}^{20} r_i + w \sum_{k=1}^{\lambda} s_k}, & \text{if } 21 \leq i \leq 20 + \lambda. \end{cases}$$

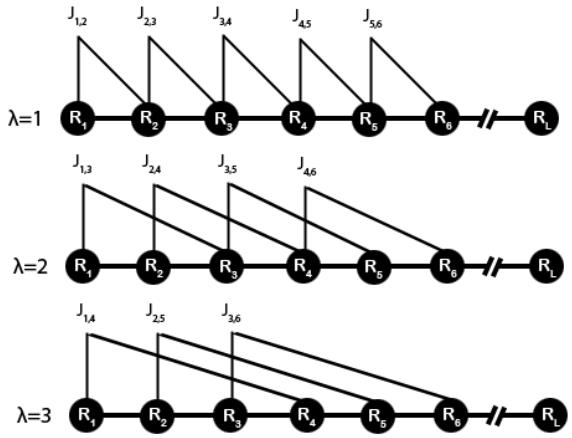

Fig. 3. Pseudo amino acid composition compares adjacent amino acids in a protein sequence to calculate s_k for each λ value.

where r_i is the frequency of the ith native amino acid, w is a weight constant, and

$$s_k = \frac{1}{L-k} \sum_{i=1}^{L-k} J_{i,i+k}$$

$$J_{i,j} = \frac{1}{3}[(H_1(R_j) - H_1(R_i))^2 + (H_2(R_j) - H_2(R_i))^2 + (M(R_j) - M(R_i))^2]$$

with $H_1(R_i)$, $H_2(R_i)$, and $M(R_i)$ denoting the hydrophobicity, hydrophilicity, and side-chain mass of amino acid R_i, respectively.These scores are all subjected to a standard conversion by normalization of the values in H_1, H_2, and M.

Physicochemical properties can characterize proteins in a non-sequence specific manner. Each amino acid is classified into one of 19 different chemical properties as shown in Table 3 [9]. For a given gene, we scan the sequence and sum the scores of the amino acids in a 19 dimensional feature vector and normalize by the length of the sequence.

3.4 Weighting of the training set

To prevent highly similar genes from closely related genomes from dominating the SVM minimization equation over fewer, more unique sequences during training, we introduced a weighting system into the soft-margin SVM. To assign weight to each gene, we first clustered them based on pairwise phylogenetic distances generated by RAxML using the PROTGAMMAJTT model (version 8.2.4) [18] after alignment by MUSCLE [19]. Farthest neighbor hierarchical clustering was used to group nearby genes until the minimum cluster distance exceeds a set threshold. The threshold can be adjusted to minimize the bias due to similar sequences in

Table 3. Physicochemical properties of amino acids.

Physicochemical property	Amino acids
Charged residue	D, R, E, K, H
Hydrophilic and neutral	N, Q, S, T, Y
Basic polar or positively charged	H, K, R
Acidic polar or negatively charged	D, E
Aliphatic	A, G, I, L, V
Aromatic	F, W, Y
Small	T, D, N
Tiny	G, A, S, P
Large	F, R, W, Y
Hydrophobic and aromatic	W, F
Hydrophobic and neutral	A, C, G, I, L, M, F, P, W, V
Amidic	N, Q
Cyclic	P
Hydroxylic	S, T
Sulfur-containing	C, M
H-bonding	C, W, N, Q, S, T, Y, K, R, H, D, E
Acidic and amidic	D, E, N, Q
Ionizable	D, E, H, C, Y, K, R
Disulfide bond	C

the training set. Weights are assigned based on how many genes cluster together. Algorithm 1 shows the pseudocode for our implementation.For example, for each gene in a cluster of size 10 we will assign it a weight of $\frac{1}{10}$.

3.5 Support vector machine implementation

SVMs have been around for decades [15] and have been successfully applied to address various biological questions [29,30,31,32]. An SVM suits our particular classification requirements, as it is capable of performing well even when given small training sets and have shown to be effective for similar biological applications [9]. During training, the SVM constructs a hyperplane to separate the two categories in the training set. Points that fall on one side of the hyperplane are assigned to one class, while those on the other side are assigned to the other class. The hyperplane is constructed such that it best separates the two classes by maximizing the distance to the nearest training-data point for both classes, creating a buffer or margin between the two classes before a given point's classification changes.

Algorithm 1 Assigning gene weights

Let $G = (g_1, ..., g_n)$ be an array of genes g_i for $1 \leq i \leq n$ and $g_i.w$ be denote the weight of gene g_i
Let $D = (d_{ij})$ be an n x n matrix, where d_{ij} is the pairwise distance between genes g_i and g_j
Let t be the distance threshold, where $t \geq 0$

procedure CLUSTER(G, D, t)
    Let C be an array of size n
    for $i = 1$ to n **do**
        $C[i] = [G[i]]$
    end for
    while $C.length > 1$ **do**
        Let $minDist = \infty$ be the minimum cluster distance
        Let $minCluster1 = minCluster2 = Nil$
        for $i = 1$ to $C.length$ **do**
            for $j = 1$ to $C.length$ **do**
                if $i \neq j$ **then**
                    Find the two most distant genes g_i and g_j in $C[i]$ and $C[j]$
                    if $D[g_i, g_j] < minDist$ **then**
                        $minDist = D[C[i], C[j]]$
                        $minCluster1 = C[i]$
                        $minCluster2 = C[j]$
                    end if
                end if
            end for
        end for
        if $minDist > t$ **then**
            return C
        else
            $C.MERGE(minCluster1, minCluster2)$
        end if
    end while
    return C
end procedure

procedure WEIGHT(C)
    for $i = 1$ to $C.length$ **do**
        for $j = 1$ to $C[i].length$ **do**
            $C[i][j].w = 1/C[i].length$
        end for
    end for
end procedure

We compute the soft-margin SVM by solving the expression

$$min\left(\frac{1}{2}||w||^2 + C\sum_{i=1}^{n}\xi_i\right)$$

$$\text{s.t. } y_i(\boldsymbol{w}\boldsymbol{x_i} + b) \geq 1 - \xi_i \text{ and } \xi_i \geq 0 \ \forall i$$

where $\boldsymbol{x_i}$ and y_i are the feature vector and label of any given gene g_i, respectively, and $\boldsymbol{w}$ and b define the hyperplane $f(\boldsymbol{x}) = \boldsymbol{w}\boldsymbol{x} + b$. C is a user defined cost associated with the slack variable ξ. We use ξ as a trade-off parameter (i.e. the cost for each misclassified example). We can represent this minimization problem in the Lagrangian dual form as

$$\max_{\alpha}\sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{i=j}^{n}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i\boldsymbol{x}_j$$

$$\text{s.t. } \sum_{i=1}^{n}\alpha_i y_i = 0 \text{ and } 0 \leq \alpha_i \leq C \ \forall i$$

We can solve the dual form equation above efficiently using a quadratic programming solver.

Traditional SVMs can only correctly classify between two linearly separable sets. In non-linear cases or when the training sets contain outliers, a soft-margin approach is required for correct classification. A soft-margin SVM allows for some genes to be misclassified during training such that the classifier can derive a more accurate hyperplane separating the two classes. The slack cost c defines how lenient the soft-margin SVM is in allowing for genes to be misclassified. With the introduction of the soft-margin, highly similar sequences can have an increased influence on the position of the hyperplane, as misclassifying two or more similar sequences can be less optimal than misclassifying one. Weighting can be used to reduce the impact of these highly similar sequences during training. Typically, C is represented as an n x n diagonal matrix with values c. For the weighted case, each $C[i,i] = \frac{c}{d_i}$, where d_i is the weight assigned to the ith gene. Gene weights act as modifiers to the soft-margin, allowing more or less room for slack in the margin depending on how many genes were clustered together. Thus, we chose to adopt a weighted soft-margin SVM to use when training it on GTA and viral gene k-mer sets. Algorithm 2 shows the pseudocode of our implementation.

Training was done for each gene in the RcGTA cluster that had sufficient GTA and viral homologs [Figure 2]. Our SVM classifier was written in Python (version 3.5.1) and is trained by solving the weighted soft-margin Lagrangian dual problem using the CVXOPT quadratic programming solver [33]. Although additional kernels were implemented, only the linear kernel was used during analysis, and all genes were confirmed to be linearly separable (data not shown).

Algorithm 2 SVM Implementation

Let $G = (g_1, ..., g_n)$ be an array of genes g_i for $1 \leq i \leq n$
Let $X = (g_i.x)$, $Y = (g_i.y)$, and $W = (g_i.w)$ be the feature sets, classes, and weights respectively for genes $g_i \in G$ and $g_i.y = 1$ if g_i is a GTA and $g_i.y = -1$ if it is a virus
Let *QUAD-PROG* be a quadratic programming solver
procedure SVM-TRAIN(G,c)
    Compute $lagrange\text{-}multipliers = QUAD\text{-}PROG(X, Y, W * c)$
    Let $alphas = \{a_i \in lagrange\text{-}multipliers : a_i > 1x10^{-5}\}$
    Let $support\text{-}vectors = \{g_i \in lagrange\text{-}multipliers : a_i > 1x10^{-5}\}$
    return *alphas*, *support-vectors*
end procedure

Let q be an unclassified gene, where $q.x$ is the feature set for q
procedure SVM-PREDICT(alphas, support-vectors, q)
    Let $score = 0$
    for $a_i \in alphas$ and $g_i \in support\text{-}vectors$ **do**
        $score = score + (a_i * g_i.y * (g_i.x * q.x))$
    end for
    if $score > 0$ **then**
        return "GTA"
    else
        return "virus"
    end if
end procedure

The SVM/weighting schema is influenced by two different parameters. First, c represents the soft-margin parameter of the SVM. Higher c values increase the "softness" of the margin, while lower c values bring the SVM closer to its original hard margin form. Second, t represents the pairwise-distance threshold at which point we no longer cluster the genes.

3.6 K-nearest neighbors

K-nearest neighbors (KNN) is a simplistic classification model. For any given gene, we assign its class based on the classifications of its neighbors. We assign the class to the gene based on the majority class shared by its K closest neighbors (Euclidean distance). We implemented KNN using the scikit-learn Python package [34].

3.7 Multinomial naïve Bayes

Naïve Bayes classifiers calculate and assign condiational probabilities to features based on their occurrence and which class they belong to. Genes are assigned a class based its maximum a posteriori probability between the two classes. Naïve Bayes models assume all features are independent. We implemented multinomial naïve Bayes using the scikit-learn Python package [34].

3.8 Model training, cross validation, and assessment

The training data was evaluated using a ten-fold cross-validation scheme, dividing the dataset into 10 different parts. Nine parts are combined to form the training set, while the tenth part is used as the test set and its SVM assigned classifications compared to the known classes. This is repeated ten times so that each of the ten parts can be used as the test set. This cross validation method was repeated ten times, and the cross validation scores over each repetition was averaged.

Cross validation results were evaluated by their accuracy scores. Accuracy refers to the number of correctly classified genes divided by the total number of genes. Accuracy scores were weighted such that GTA and viral classes have equal impact, regardless of respective sizes.

4 Results and discussion

We ran GTA_Hunter on the genes listed in Figure 2. Our first goal was to determine how well our model worked and under what features and parameters. Once this was established we moved on to determine whether or not some GTA homologs are true GTAs.

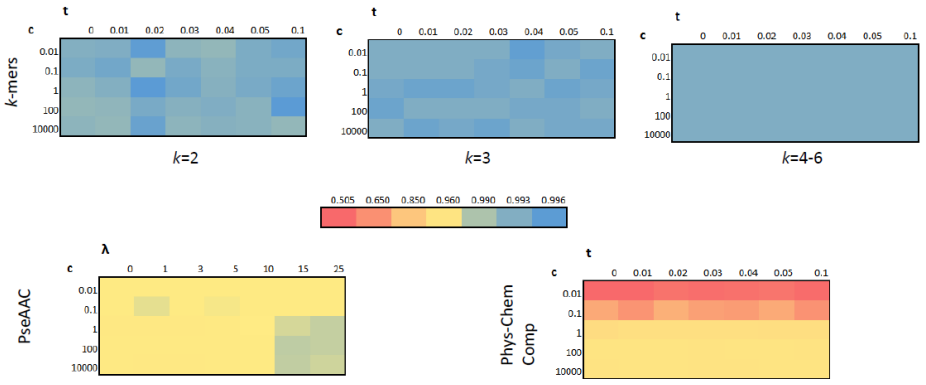

Fig. 4. Representative cross validation. We examined the performance of our SVM classified over a variety of soft-margin c and weight thresholds t for each feature type. Accuracy at each parameter is shown by as indicated by the heat map in the center. The representative is from gene 5.

4.1 SVM evaluation and parameter tuning

GTA gene homologs were assessed by training the SVM on known GTA-like genes (GTA and viral) using a variety features and SVM parameters. We analyzed performance over a variety of k-mer sizes, PseAAC λ values, and physicochemical composition features. k-mers features tended to outperform PseAAC and

physicochemical composition, with the latter performing the worst [Figure 4]. Optimal k-mer size varied between genes, with lower k values typically performing the best. Combining features together did not improve cross validation results. When combining k-mers and PseAAC, average performance went up, while best performance either did not change or performed worse (data not shown). Combining physicochemical composition with any of the other features tended to lower the accuracy scores than if that feature was used alone. Table 4 shows the highest accuracy scores attained for each gene and the conditions that it was run under. It is interesting to note that for several of the genes we had clustering thresholds of $t = 0$. In some cases our training sets are diverse enough that clustering only reduces the amount of available data and hurts SVM performance. The opposite is true for larger clustering thresholds such at $t = 0.05$ for $g4$.

4.2 Alternative classifiers: KNN, naïve Bayes, and random

We compared our SVM to alternative classifiers such as KNN and multinomial naïve Bayes, as well as when the class labels are randomly assigned to the training set. SVM tended to outperform other classifiers Table 4. However, KNN and naïve Bayes both performed well, occasionally at or better than the SVM score. In all, SVM appears to be the best approach. That all three classification models perform well suggests that our chosen features are capable of fully characterizing each gene. One concern with many classifiers is the over-fitting of the data. Excessively complex models are capable of modeling the random error or noise for each gene instead of the underlying trend. This is especially true when we have many features, as with the k-mers, but few samples. SVMs tend to be resistant to over-fitting, but this is dependent on how we tune the soft-margin parameter c. By randomizing class labels on the genes during training, we can see if our SVM model is over-fitting the data. If the model is over-fitting we would expect the accuracy in cross validation to deviate strongly from 0.5. However, we see in Table 4 that this is not the case.

4.3 GTA-like genes within small clusters

GTA homologs that had less than eight other structural cluster homologs were classified as either GTA or viral. These genes are referred to as belonging to "small clusters". Because genes in small clusters lack some of the RcGTA gene homologs in their structural cluster, genes within these clusters can be either true prophages or part of an RcGTA structural cluster that has lost some of its genes, hence decaying.

GTA homologs within small clusters were classified by an SVM trained using the parameters defined in Table 4 for each gene. Of the 446 genes that are part of the small clusters from 261 α-proteobacteria genomes across all 10 RcGTA gene homolog test sets being classified, 161 genes were classified as GTA. The classified genes were grouped by taxonomic order to reveal trends in gene classification [Figure 5].

Table 4. Best cross validation accuracies obtained using at indicated c and t parameters. Features used for each gene were either k-mers or a combination of k-mers and PseAAC (λ). Performance of alternative classification methods, naïve Bayes and K-nearest neighbors, are also shown.

Gene	Features	Model	Parameters	GTA	Viral	Accuracy
g2	$k = 3$	SVM	$c = 0.1$, $t = 0.01$	78	232	1
	$\lambda = 25$	SVM (random)	$c = 0.1$, $t = 0.01$	11.5	186.8	0.476
		Naïve Bayes	Multinomial	78	232	1
		KNN	$k = 3$	78	217.4	0.969
g3	$k = 3$	SVM	$c = 1$, $t = 0$	78	243	1
	$\lambda = 25$	SVM (random)	$c = 1$, $t = 0$	15.1	188.7	0.485
		Naïve Bayes	Multinomial	78	243	1
		KNN	$k = 3$	78	242.2	0.998
g4	$k = 2$	SVM	$c = 1$, $t = 0.05$	71	181	1
	$\lambda = 5$	SVM (random)	$c = 1$, $t = 0.05$	19.9	133.3	0.508
		Naïve Bayes	Multinomial	70.5	179.4	0.992
		KNN	$k = 5$	71	177.2	0.990
g5	$k = 2$	SVM	$c = 1$, $t = 0.02$	72.4	218.2	0.994
		SVM (random)	$c = 1$, $t = 0.02$	16.9	157.8	0.476
		Naïve Bayes	Multinomial	72	219	0.993
		KNN	$k = 3$	73	213.2	0.987
g6	$k = 2$	SVM	$c = 0.01$, $t = 0.01$	74	67.1	0.987
		SVM (random)	$c = 0.01$, $t = 0.01$	40.6	31.7	0.504
		Naïve Bayes	Multinomial	74.8	66.3	0.986
		KNN	$k = 8$	74	68	0.993
g8	$k = 2$	SVM	$c = 0.1$, $t = 0.01$	84	63	0.992
	$\lambda = 3$	SVM (random)	$c = 0.1$, $t = 0.01$	49.1	28	0.511
		Naïve Bayes	Multinomial	84	63	0.992
		KNN	$k = 8$	84	62	0.984
g12	$k = 4$	SVM	$c = 0.1$, $t = 0.04$	74.9	12.7	0.923
		SVM (random)	$c = 0.1$, $t = 0.04$	68.4	1	0.489
		Naïve Bayes	Multinomial	75	12.9	0.930
		KNN	$k = 3$	73.8	14.5	0.975
g13	$k = 2$	SVM	$c = 0.1$, $t = 0$	70	41	1
	$\lambda = 15$	SVM (random)	$c = 0.1$, $t = 0$	45.9	13.3	0.490
		Naïve Bayes	Multinomial	70	41	1
		KNN	$k = 2$	70	41	1
g14	$k = 3$	SVM	$c = 0.01$, $t = 0.03$	77	89	1
	$\lambda = 15$	SVM (random)	$c = 0.01$, $t = 0.03$	31.3	50.3	0.486
		Naïve Bayes	Multinomial	77	84.4	0.974
		KNN	$k = 3$	77	88.4	0.997
g15	$k = 4$	SVM	$c = 100$, $t = 0.04$	68	39	1
	$\lambda = 15$	SVM (random)	$c = 100$, $t = 0.04$	48.7	11	0.499
		Naïve Bayes	Multinomial	68	38.7	0.996
		KNN	$k = 2$	45.4	39	0.834

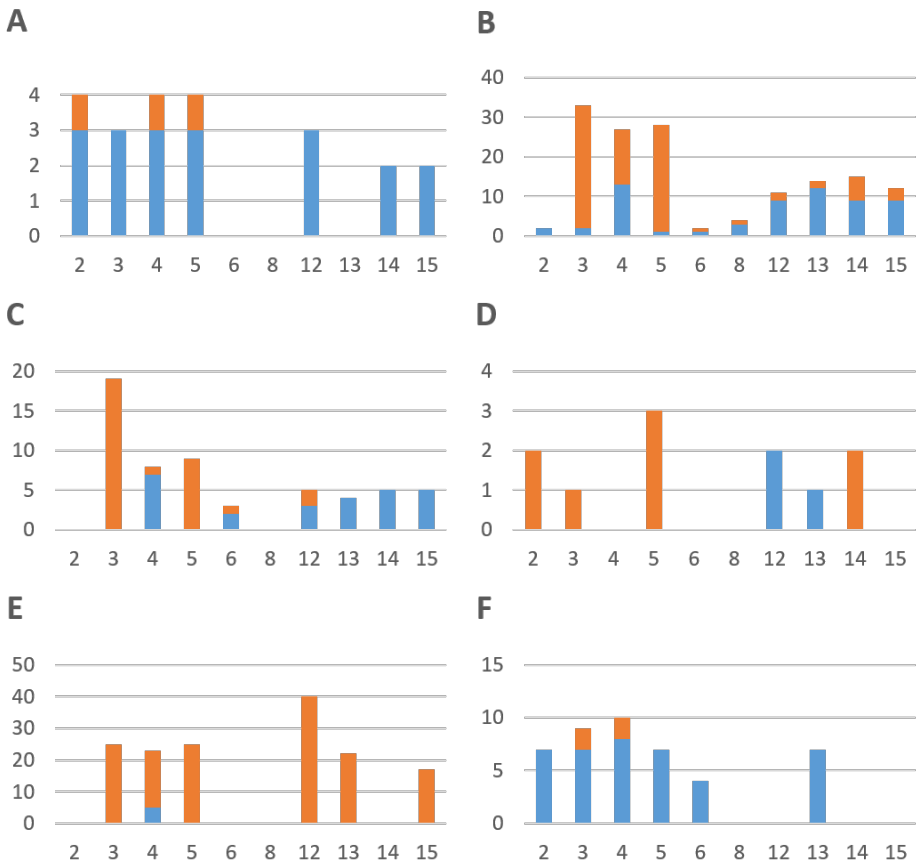

Fig. 5. Small cluster classification results grouped by order. The x-axis refers to the specific RcGTA gene, and the y-axis is the frequency of homologs classified for a given RcGTA gene. The test set was evaluated with the feature and parameters shown in Table 4. A. *Caulobacterales*, n=5, B. *Rhizobiales*, n=34, C. *Rhodobacterales*, n=11, D. *Rhodospirillaes*, n=6, E. *Ricketsiales*, n=42, F. *Sphingomonadales*, n=8.

We examined six taxonomic orders: *Caulobacterales*, *Rhizobiales*, *Rhodobacterales*, *Rhodospirillaes*, *Ricketsiales*, and *Sphingomonadales*. Of these six groups, *Caulobacterales*, *Rhizobiales*, *Rhodobacterales*, and *Sphingomonadales* contain known GTAs from large clusters in addition to the small clusters classified. Some of the small cluster genes from these orders are classified as GTA [Fig 5A-C,F]. Many of these GTA-classified small cluster genes are part of genomes that already contain large clusters (data not shown). This is especially true for *Rhodobacterales* and *Sphingomonadales*, where all genomes sequenced contained large clusters. Because all of these genomes already contain GTAs, we speculate that many of the short cluster genes within these orders that were classified as GTA by the SVM may be a result of duplication events. More importantly, genomes in orders *Rhodospirillaes* and *Ricketsiales* contain no large clusters and therefore no known "true" GTAs. However, we see for both *Rhodospirillaes* [Fig 5D] and *Ricketsiales* [Fig 5F] that a small number of the small cluster genes have been classified as GTA. This suggests that *Rhodospirillaes* and *Ricketsiales* used to have functional GTAs, these genes have since decayed. If this is true, then all orders previously mentioned

may have shared a common ancestor that expressed GTAs. Further investigation and analysis on the nature of the GTA-classified genes in *Rhodospirillaes* and *Ricketsiales* is needed to draw any definite conclusions.

One interesting trend to note is that for within the orders *Rhizobiales* and *Rhodobacterales*, the SVM preferentially classified genes encoding for head proteins as virus and genes encoding for tail proteins as GTA [Fig 5B-C]. In particular, one organism, *Methylobacterium nodulans* in the order *Rhizobiales*, has 4 small clusters of 5 genes in its genome. For each of the 4 small clusters in *M. nodulans*, gene $g3$ is classified as virus while genes $g12$, $g13$, $g14$, and $g15$ as GTA. On average, with the exception of gene $g15$, genes that encode for proteins that appear first within the RcGTA structural cluster have a longer sequence length of about 250 more amino acids than those that encode for proteins later on [Figure 2]. With shorter sequence lengths, there are fewer amino acids to characterize in a given gene. Therefore, we will see fewer unique k-mers among gene homologs with short sequence lengths. In contrast, we would expect to see more unique k-mers in longer sequences. Because differences in sequence length affect k-mer diversity, the size of the feature set, will also be affected. Richer feature sets allow for more accurate classification. Alternatively, late genes encode for tail structural proteins and their sequences less conserved, making classification more difficult. The inclusion of PseAAC features does not seem to be enough to offset this bias. Consequently, the SVM's ability to classify genes as GTA or virus is impacted the size of the feature sets used. Despite high accuracy for cross-validation for both early and late appearing genes in the structural cluster, the SVM will have more difficulty classifying genes with small feature sets, as smaller feature sets inherently hold less information.

4.4 GTA-like genes within putative prophage regions

In addition to looking at structural cluster size to determine whether or not RcGTA homologs are true GTAs, we can also use prophage identifying tools such as PhiSpy [24], Phage_Finder [25], and PHAST [26] to detect putative prophages. Because majority of bacterial genomes sequenced contain prophage sequences within them [23] and GTA genes are often identified as prophages, we speculated that some prophage sequences may actually be GTA. We can evaluate GTA homologs in prophages to determine if they are actually GTAs. The set of GTA homologs within prophage regions were classified using SVM models with the input parameters and features listed in Table 4. The GTA homologs genes found in prophage regions as identified jointly by PhiSpy, Phage_Finder, and PHAST are few in number, totaling 87 with the majority being from genes $g3$, $g4$, and $g5$ [Table 2]. This is partly due to the fact that we remove all large cluster genes from the results, leaving only small cluster genes which by definition lack enough genes to be considered GTA and by association, prophage. Few of the genes selected were located in the same prophage region, with the maximum number of GTA-like genes in a given prophage region being 3. The limited cluster size for the genes makes it difficult to reach any classification consensus for a given cluster.

Table 5. SVM classification results of GTA homologs in prophage regions. Genes which have been classified as GTA are denoted in blue, while those classified virus are orange. The genes are grouped by organism as indicated in the table.

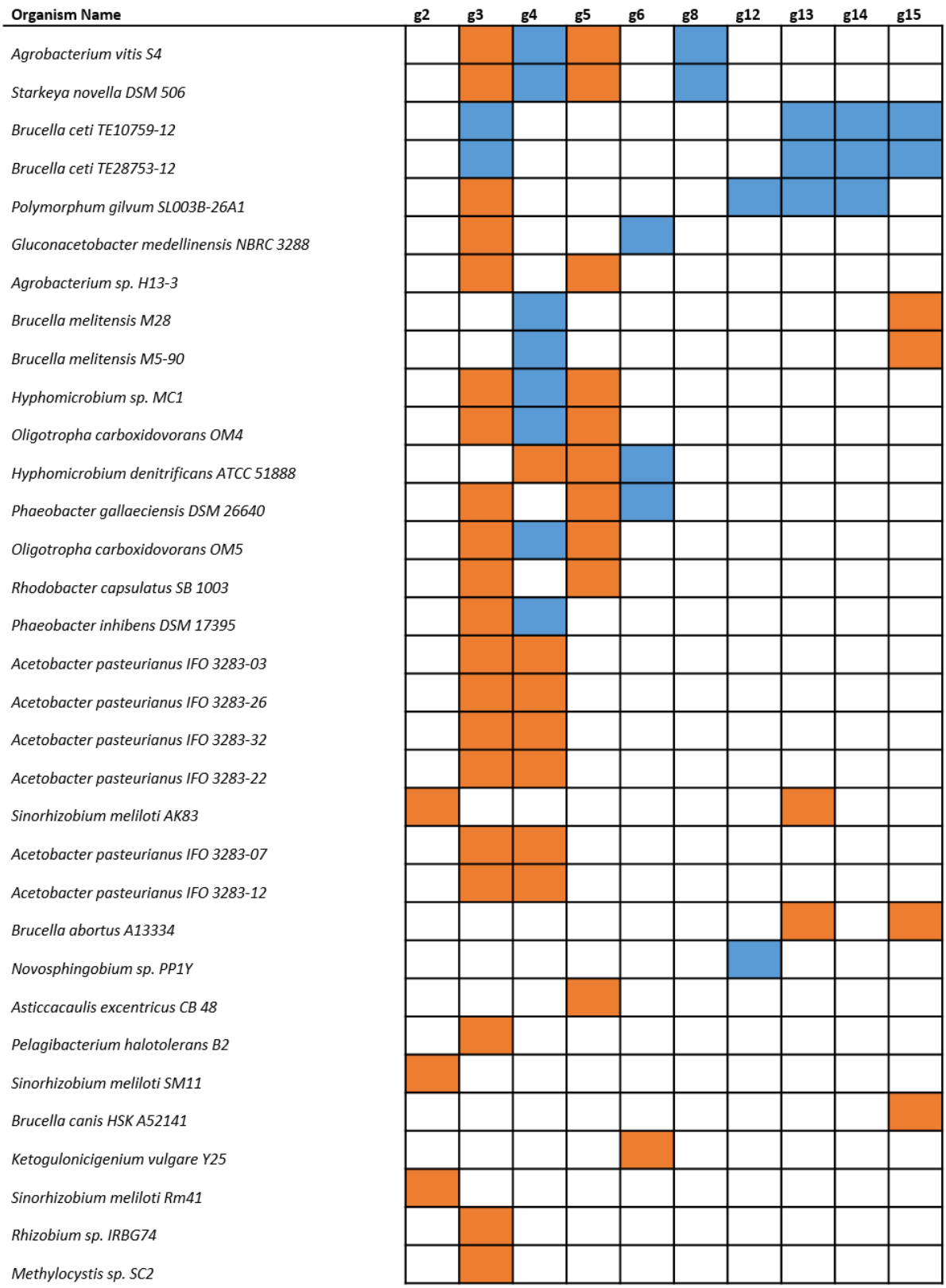

Table 5 groups the genes based on which organism they belong to. The SVM classified 68 of the 87 genes as viral, with only 19 being classified as GTA. Only three organisms contained at least 3 genes that were classified as GTA. Two that we examined in particular are *Brucella ceti* TE28753-12 and *Polymorphum gilvum* SL003B-26A1.

4.5 *Brucella ceti* TE28753-12 GTA-classified homologs from prophages are true GTAs

B. ceti is a bacterial species within *Rhizobiales*. Multiple species of *Brucella* have a complete GTA structural cluster. *B. ceti*, however does not have a structural cluster but houses a putative prophage region with 59 genes, 5 of which are homologs of GTA genes [26]. Our SVM classified 4 ($g3$, $g13$, $g14$, and $g15$) of those genes as GTA. In addition to the GTA-like genes tested by our SVM classifier, the prophage region also contains an RcGTA homolog for gene $g9$ between $g3$ and $g13$. Phylogenic trees of the GTA-like genes and their known GTA and viral homologs showed the SVM classified genes grouping closely with other GTAs. In each case, the GTA classified gene of *Brucella ceti* grouped together genomes from other *Brucella* [Figure 6]. The phylogenetic grouping, coupled with the SVM classification makes it highly likely that the genes are of GTA origins.

4.6 *Polymorphum gilvum* SL003B-26A1 GTA classified homologs from prophages may not be "true" GTAs

P. gilvum is a novel species isolated from saline soil contaminated with crude oil. It can use crude oil as a carbon and energy source and is adapted to saline soil at a temperature of 45°C [35]. Its genome has a known GTA structural cluster along with 9 putative prophage regions, one of which has 4 GTA homologs. 3 of the 4 genes that are homologs of $g12$, $g13$, and $g14$ were classified as GTA. Additionally, this region contains an RcGTA $g15$ homolog as well that was not included in the test set due to an insufficient number of viral homologs.

Because *P. gilvum* already contains a large GTA cluster, it is possible that the GTA-classified genes from the prophage region are simply duplicates of the genes from the large structural cluster present in the genome. To test this hypothesis, we reconstructed the phylogenetic history of these genes and their homologs from both GTA and phage. The phylogenetic history, however, revealed that the GTA-classified genes did not group with their large cluster homologs in *P. gilvum*. Additionally, the phylogeny shows that all genes did not group with any of their GTA homologs, but instead grouped with *bona fide* phages, suggesting that these genes are likely not the result of a duplication from the structural cluster [Figure 7]. Interestingly, while $g13$ and $g14$ grouped with phages at the local level, they group with known GTAs as a whole. This shows

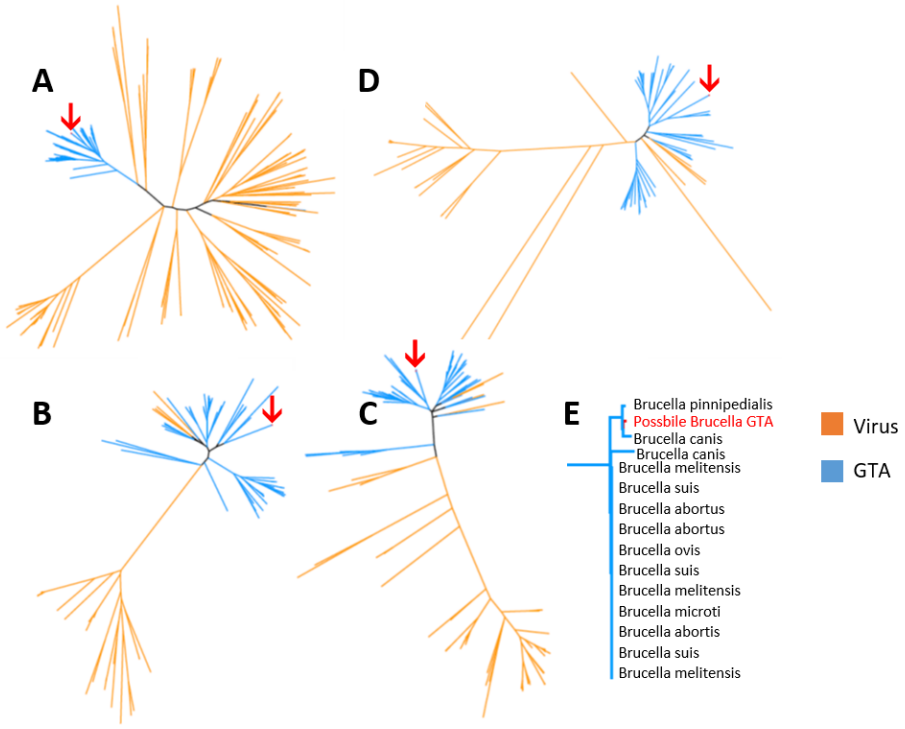

Fig. 6. Phylogenetic trees of *Brucella ceti* GTA genes. The trees were constructed using known GTA and viral genes from the training set. A. $g3$, B. $g13$, C. $g14$, and D. $g15$ from RcGTA homologs. Blue branches indicate GTA genes, and orange branches indicate viral genes. Red arrows denote the GTA-like genes from *Brucella ceti*. E. A typical branch containing *Brucella* GTA genes and a *Brucella ceti* GTA classified gene.

that our soft-margin SVM classifier is still working as intended. The differences in phylogenetic history of GTA genes in the prophage region and large cluster suggest that the GTA-classified homologs in prophages may have been picked up by phages at some point in time or acquired through horizontal gene transfer. It is important to note that while $g12$ and $g13$ did not group closely with their respective known large cluster GTA genes in *P. gilvum*, $g14$ still did. The relationship between these GTA-classified homologs in prophages and known GTAs is still not clear.

5 Conclusion and outlook

Our goal was to develop a software package that can be used to confidently identify GTA genes from across various genomes starting with just their sequence information. We have shown that GTA_Hunter can accurately classify GTA and phage genes, reducing the time and effort needed to sift through BLAST results for true GTAs. The high cross-validation accuracies for k-mer features regardless of classification model confirms that k-mer composition is an effective way to characterize amino acid sequences. Additionally, the high cross-validation accuracies for our SVM add confidence to our approach. GTA-classified homologs found

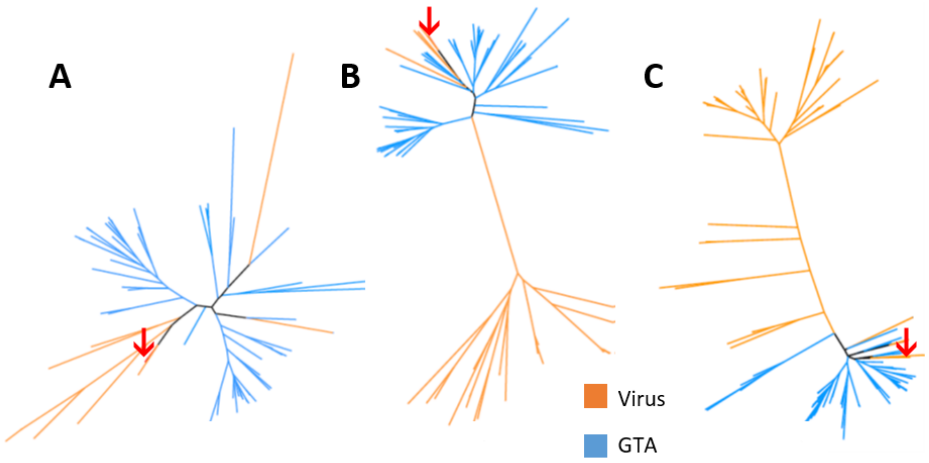

Fig. 7. Phylogenetic trees of *Polymorphum gilvum* GTA genes. The trees were constructed using known GTA and viral genes from the training set. A. *g*12, B. *g*13, and C. *g*14 from RcGTA homologs. Blue branches indicate GTA genes, and orange branches indicate viral genes. Red arrows denote the GTA-like genes from *Polymorphum gilvum*.

in putative prophages like those in *B. ceti* are highly likely to be of GTA origin. Using GTA_Hunter we have identified many genes of interest. Those in small clusters that have been classified as GTA and contain large clusters within their genomes may have arisen from duplication events. Furthermore, GTA-classified small cluster genes in orders such as *Rhodospirillaes* and *Ricketsiales* that lack any known large cluster GTAs are of particular interest. Further investigation into these genes may help us better understand the evolutionary history of GTAs. Additionally, GTA homologs that were classified as virus can be ruled out from further investigation, saving both time and effort. This will allow for more efficient research in the future.

6 Acknowledgments

I would like to thank Professor Olga Zhaxybayeva and all members of the Zhaxybayeva lab, especially Migun Shakya, for their expertise and guidance my project, as well as Dan Birnbaum for his initial work on the project. Additionally, I would like to thank the James O. Freedman Presidential Scholars Program for providing me the opportunity and funding necessary for my work. The funding for this project also came from the Simons Foundation Investigator award from the Simons Foundation to O.Z., and National Science foundation award DEB-1551674 to O.Z.

References

1. J. Smith, J. Van Dyken and P. Zee, "A generalization of Hamilton's rule for the evolution of microbial cooperation," *Science*, p. 1700–1703, 2010.
2. R. Hardin, "The Genetics of Cooperation," *Analyse & Kritik*, pp. 57-65, 2006.
3. O. Zhaxybayeva and W. F. Doolittle, "Lateral gene transfer," *Current Biology*, pp. R242-R246, 2011.
4. A. Lang, O. Zhaxybayeva and J. Beatty, "Gene Transfer Agents: virus-like elements of genetic exchange," *Nature Reviews Microbiology*, pp. 472-482, 2012.
5. T. B. Stanton, "Prophage-like gene transfer agents-novel mechanisms of gene exchange for *Methanococcus, Desulfovibrio, Brachyspira,* and *Rhodobacter* species.," *Anaerobe*, vol. 13, no. 2, pp. 43-49, 2007.
6. B. Marrs, "Genetic Recombination in *Rhodopseudomonas capsulata*," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 71, no. 3, pp. 971-973, 1974.
7. S. Altschul, W. Gish, W. Miller, E. Myers and D. Lipman, "Basic local alignment search tool," *Journal of Molecular Biology*, vol. 215, pp. 403-410, 1990.
8. H. Soueidan, L.-A. Schmitt, T. Candresse and M. Nikolski, "Finding and identifying the viral needle in the metagenomic haystack: trends and challenges," *Frontiers in Microbiology*, 07 January 2015.
9. R. Kaundal, S. S. Sahu, R. Verma and T. Weirick, "Identification and characterization of plastid-type proteins from sequence-attributed features using machine learning," *BMC Bioinformatics*, 2013.
10. G. Qing-Bin, W. Zheng-Zhi, Y. Chun and D. Yao-Hua, "Prediction of protein subcellular location using a combined feature sequence," *Federation of European Biochemical Societies*, pp. 3444-3448, 2005.
11. K.-C. Chou, "Prediction of protein cellular attributes using pseudo-amino acid composition," *PROTEINS: Structure, Function, and Genetics*, no. 43, pp. 246-255, 2001.
12. N. S. Altman, "An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression," *The American Statistician*, vol. 46, no. 3, pp. 175-185, 1992.
13. A. McCallum and K. Nigam, "A Comparison of Event Models for Naive Bayes Text Classification," *AAAI-98 workshop on learning for text categorization*, vol. 752, 1998.
14. K. Hornik, M. Stinchcombe and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 356-366, 1989.
15. C. Cortes and V. Vapnik, "Support vector networks," *Machine Learning*, pp. 273-293, 1995.
16. N. Japkowicz and S. S, "The class imbalance problem: a systematic study," *Intell Data Anal*, vol. 6, pp. 429-450, 2002.
17. G. M. Foody, M. B. McCulloch and W. B. Yates, "The effect of training set size and composition on artificial neural network classification," *International Journal of Remote Sensing*, vol. 16, no. 9, pp. 1707-1723, 1994.
18. A. Stamatakis, "RAxML Version 8: A tool for Phylogenetic Analysis and Post-Analysis of Large Phylogenies," *Bioinformatics*, 2014.
19. R. C. Edgar, "MUSCLE: multiple sequence alignment with high accuracy and high throughput," *Nucleic Acids Research*, vol. 32, no. 5, pp. 1792-1797, 2004.

20. T. Tatusova, S. Ciufo, B. Fedorov, K. O'Neill and I. Tolstoy, "RefSeq microbial genomes database: new representation and annotation strategy," *Nucleic Acids Res*, vol. 42, no. 1, pp. 553-559, 2014.

21. S. Altschul, T. Madden, S. A. J. Zhang, Z. Zhang, W. Miller and D. Lipman, "Gapped BLAST and PSI-BLAST: a new generation of protein database search programs," *Nucleic Acids Res.*, vol. 25, pp. 3389-3402, 1997.

22. M. Price, P. Dehal and A. Arkin, "FastTree 2 – Approximately Maximum-Likelihood Trees for Large Alignments," *PLOS One*, vol. 5, no. 3, p. e9490, 2010.

23. C. Canchaya, C. Proux, G. Fournous, A. Bruttin, and H. Brussow, "Prophage Genomics," *Microbiology and Molecular Biology Reviews*, vol 67, no. 2, p. 238-276, 2003.

24. S. Akhter, R. Aziz and R. Edwards, "PhiSpy: a novel algorithm for finding prophages in bacterial genomes that combines similarity- and composition-based strategies," *Nucleic Acids Research* 40, p. e126, 2012.

25. D. Fouts, "Phage_Finder: Automated identification and classification of prophage regions in complete bacterial genome sequences," *Nucleic Acids Research* 34, pp. 5839-5851, 2006.

26. Y. Zhou, Y. Liang, K. Lynch, J. Dennis and D. Wishart, "PHAST: A Fast Phage Search Tool," *Nucleic Acids Research* 39, pp. W347-W352, 2011.

27. C. Tanford, "Contribution of Hydrophobic Interactions to the Stability of the Globular Conformation of Proteins," *Journal of American Chemistry*, vol. 84, no. 22, pp. 4240-4247, 1962.

28. T. P. Hopp and K. R. Woods, "Prediction of protein antigenic determinants from amino acid sequences," *PNAS*, vol. 78, no. 6, pp. 3824-3828, 1981.

29. R. Kaundal, R. Saini and Z. PX, "Combining Machine Learning and Homology-Based Approaches to Accurately Predict Subcellular Localization in Arabidopsis," *Plant Physiology* 154, pp. 36-54, 2010.

30. S. Hua and Z. Sun, "Support vector machine approach for protein subcellular localization prediction," *Bioinformatics* 17, pp. 721-728, 2001.

31. R. Kaundal, A. Kapoor and G. Raghava, "Machine learning techniques in disease forcasting: a case study on rice blast prediction," *BMC Bioinformatics* 7, p. 485, 2006.

32. J. Ward, L. McGuffin, B. Buxton and D. Jones, "Secondary structure prediction with support vector machines," *Bioinformatics* 19, pp. 1650-1655, 2003.

33. M. Anderson, J. Dahl and L. Vandenberghe, "CVXOPT 1.1.8," 2016.

34. 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," J*ournal of Machine Learning Research*, vol. 12, pp. 2825-2830, 2011.

35. Y. Nie, Y.-Q. Tang, Y. Li, C.-Q. Chi, M. Cai and X.-L. Wu, "The Genome Sequence of *Polymorphum gilvum* SL003B-26A1T Reveals Its Genetic Basis for Crude Oil Degradation and Adaptation to the Saline Soil," *PLOS ONE*, 16 February 2012.