Semi-Supervised Learning applied to HIV-1 virus subtyping

Pablo Andrés Millán Arias
School of Computer Science
University of Waterloo
Waterloo, ON, N2L 3G1
pmillana@uwaterloo.ca

Abstract

The classification of the human immunodeficiency virus type one (HIV-1) can have several applications in clinical and epidemiological studies as well as in treatment response and in vaccine developments. Over the past three decades there has been a huge effort trying to find an efficient solution to this problem, with the most popular approaches relying on alignment-based methods that have proven to be very effective but at a high computational cost. In this project, an alignment-free method is studied and combined with a semi-supervised learning technique, obtaining 92% of accuracy using only 10% of the labeled samples of a particular database.

1 Introduction

The predominant type of the HIV-1 virus is the M group and it circulates as nine pure subtypes and 55 different circulant recombinant forms (CRFs).[1] Given a particular DNA sequence, the initial problem consist in predicting the correct subtype of the sequence. However, this initial view of the problem has changed because now we need to address the reality of a dynamic genetic evolutionary process, through which new unique recombinant forms (URFs) are constantly emerging.[2]

The major methods for virus subtyping can be classified into two broad categories: Alignment-based and alignment free methods [3]. In the first one, each input sequence is compared against a set of predefined subtype reference sequences, a sliding window is then used to find homologous sequences with the same subtype. Some of the most popular methods such as BLAST, REGA and SQUEAL can be assigned to this category [1]. On the other hand, the alignment-free methods rely on the sequence composition, that is, the number and distribution of nucleotide in the sequences (e.g COMET and CASTOR).

As machine learning is becoming more popular, one of the latest aligment-free approaches to this classification problem relies on the use of k-mer (substrings of length k) frequencies as a representation of the sequences. These k-mer frequencies can be used to create an embedding such that the feature vectors are constructed from the number of occurrences of all 4^k possible k-mers (given the nucleotide alphabet $\{A, C, G, T\}$). A supervised learning classifier such as a multilayer perceptron or a support vector machine (SVM) is then trained on the feature vectors. This methodology has provided state-of-the-art performances.

In practice, however, relatively little labeled data are available since it requires an expert to assign the correct subtype to a sequence with a certain degree of confidence. In this work we used the k-mer frequency embedding to explore the semi-supervised learning framework trying to give a better use of the unlabeled data. We describe experimental comparison between two semi-supervised learning approaches to the problem of virus subtyping. In particular, we explored two Graph-Based Methods and the Neighbourhood Kernel. The report is organized as follows: we begin with a description of the semi-supervised learning paradigm and its more general assumptions in section 2, this section summarizes chapers 2,7 and 11 of [4]. In section 3 we describe our data and show that it is a good fit for the semi-supervised learning requirement. The results of the experiments are presented in Section 4 and we discuss the results obtained from the experiments in the last section.

2 Prior-Work

2.1 The Semi Supervised Learning Approach

As it can be inferred from its name, semi-supervised learning (SLL) is in a sweet spot between unsupervised and unsupervised learning. The algorithms are provided with unlabeled samples alongside with some additional information that can be, but is not limited to the labels of some of the samples. For this particular case, the data set $X = (x_i)_{i \in [n]}$ can be divided into the labeled samples $X_l := (x_1, \ldots, x_l)$ and the unlabeled samples $X_u := (x_{l+1}, \ldots, x_{l+u})$ and only the labels $Y_l := (y_1, \ldots, y_l)$ are provided.

One may tent to believe that compared with a supervised algorithm that uses only labeled data, we would have a more accurate prediction by considering unlabeled examples. However, there is an important requisite: The distribution of the examples must be relevant for the classification problem, in other words, it is required that the knowledge that we plan to obtain about $p(x)$ with the addition of unlabeled samples, has to carry some additional information about $p(y \mid x)$. If this condition is not met, using unlabeled samples lead to a misguided inference.

In that sense, for semi-supervised learning to work, some certain assumptions will have to hold on:

- *The smoothness assumption:* If two points x_1, x_2 in a high-density region are close, then so should be the corresponding outputs y_1, y_2.
- *The cluster assumption:* If points are in the same cluster, they are likely to be on the same class. This is equivalent to say that the decision boundary should lie in a low-density region.
- *The manifold assumption:* Data can be embedded into a small dimensional manifold.

In practice, semi-supervised learning algorithms will try to implement these or some of these assumptions and the algorithms are classified accordingly in the literature. For the particular problem of HIV-1 subtyping I focused my attention in one Graph-based method that is built on the manifold assumption and one clustering method that tries to find a Low-Density separator using a cluster Kernel.

2.2 Label Propagation on a Similarity Graph

Data points can be naturally represented by an empirical graph $g(V, E)$ where nodes $V = \{1, \ldots, n\}$ represent the training samples and the edges E represent the similarity between them. This similarity can be represented by a weigh matrix $\mathbf{W}$ such that $\mathbf{W}_{ij} \neq 0$ iff x_i and x_j are neighbours. The weight matrix $\mathbf{W}$ can be, for instance, the k-nearest neighbor matrix: $\mathbf{W}_{ij} = 1$ iff x_i is among the k-nearest neighbors of x_j or vice versa. Another typical weight matrix is given by the expression: $\mathbf{W}_{ij} = e^{-\gamma \|x_i - x_j\|^2}$, that can be thought as a radial basis kernel function. Note that using a radial basis kernel function will produce a fully connected graph which is represented in memory by a dense matrix. On the other hand, the KNN weight matrix will be sparse and that can drastically reduce computation time.

Given the graph, the idea is that each node propagates its label to its neighbours according to the similarity matrix until the process converges. Algorithm 1 explains mathematically this intuition, the estimated labels on both labeled and unlabeled data are denoted by $\hat{Y} = \left(\hat{Y}_l, \hat{Y}_u\right)$. The way the labels are being propagated is simply by taking the weighted average of the labels over the neighbours. Note that only the labels of the unlabeled data are changing in each iteration.

Note that in the label propagation algorithm the final labels of the initially labeled data must match exactly. However, in practice this can be relaxed with the addition of a penalty and a regularization term. Another approach, similar label propagation algorithm was proposed in 2004 and at each

```
Result: Estimated Lables $\hat{Y} = \left(\hat{Y}_l, \hat{Y}_u\right)$
1 Compute weight matrix using the RBF or the Knn Kernels;
2 Compute the diagonal degree matrix $\mathbf{D}_{ii} \leftarrow \sum_j W_{ij}$ ;
3 Initialize $\hat{Y}^{(0)} \leftarrow (y_1, \ldots, y_l, 0, 0, \ldots, 0)$ ;
4 while not convergence to $\hat{Y}^{(\infty)}$ do
5     $\hat{Y}^{(t+1)} \leftarrow \mathbf{D}^{-1}\mathbf{W}\hat{Y}^{(t)}$;
6     $\hat{Y}_l^{(t+1)} \leftarrow Y_l$
```
Algorithm 1: Label propagation (Zhu and Ghahramani, 2002)

step,the node i receives a contribution from its neighbors j (weighted by the normalized weight of the edge (i, j)), and an additional small contribution given by its initial value. This algorithm exploits the properties of the Graph Laplacian and its information about the distribution of the data that is represented by the eigen values.

```
Result: Estimated Lables $\hat{Y} = \left(\hat{Y}_l, \hat{Y}_u\right)$
1 Compute weight matrix for $i \neq j$ ;
2 Compute the diagonal degree matrix $\mathbf{D}_{ii} \leftarrow \sum_j W_{ij}$ ;
3 Compute the normalized graph Laplacian $\mathcal{L} \leftarrow \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$ ;
4 Initialize $\hat{Y}^{(0)} \leftarrow (y_1, \ldots, y_l, 0, 0, \ldots, 0)$ ;
5 Choose $\alpha \in [0, 1)$
6 while not convergence to $\hat{Y}^{(\infty)}$ do
7     $\hat{Y}^{(t+1)} \leftarrow \alpha\mathcal{L}\hat{Y}^{(t)} + (1-\alpha)\hat{Y}^{(0)}$;
```
Algorithm 2: Label Spreading (Zhu et al., 2004)

The convergence rate of these algorithms depends on specific properties of the graph such as the eigenvalues of its Laplacian. In general, for the worst case the complexity is $O(kn^2)$, where k is the number of neighbors of a point in the graph. In the case of a dense weight matrix, the computational time is $O(n^3)$

2.3 The Neighborhood Kernel for Semi-Supervised Learning

This Kernel approach has a very clear intuition and unlike other Kernel approaches is very fast and general. It makes use of two complementary similarity measures: The base kernel representation which is useful for the classification purposes and a distance measure that describe how close are examples to each other. In that sense, the concept of Neighbourhood is introduced:

$$\mathrm{Nbd}(x) = \{x' \mid d(x, x') \leq thr\}$$

Finally, for a fixed original feature representation, x is represented by the average of the feature vectors for members of its neighborhood:

$$\Phi_{nbd}(x) = \frac{1}{|\mathrm{Nbd}(x)|} \sum_{x' \in \mathrm{Nbd}(x)} \Phi_{\text{orig}}(x')$$

In terms of the original kernel, the neighborhood kernel is the defined by:

$$k_{nbd}(x, y) = \frac{\sum_{x' \in \mathrm{Nbd}(x), y' \in \mathrm{Nbd}(y)} k_{orig}(x', y')}{|\mathrm{Nbd}(x)||\mathrm{Nbd}(y)|}$$

This simple Kernel fits with the cluster assumption given the fact that for a set of points in the feature space that form a cluster and a region R formed by the union of the convex hulls of the neighborhood point sets, then the averaged vector will always stay inside the convex hull of the vectors in the neighborhood.

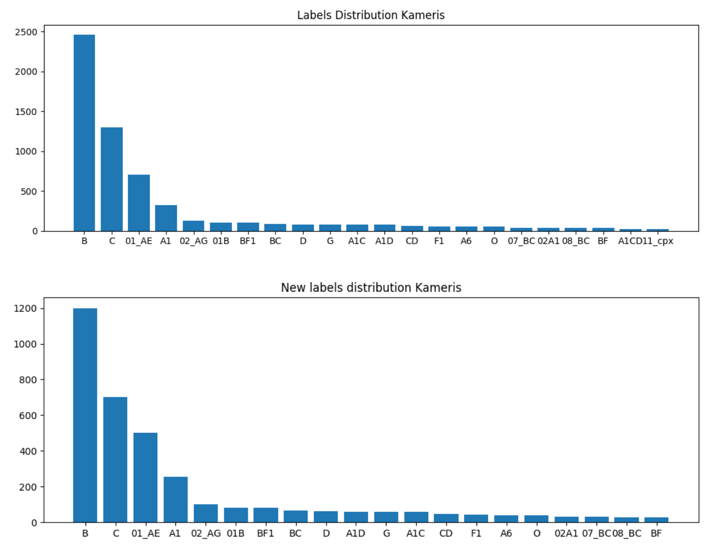

Figure 1: Label distribution of the data base. Original manually curated database (top). After sub sampling (bottom)

3 Experiments

3.1 Database Description

One of the most popular databases for HIV-1 virus subtyping and for genome in general is the NCBI-GenBank database. For our experiments we will use the same data set used in [1] that from now on will be referred as Kameris. This is a manually curated subset of the NCBI-GenBank with sequences up to late 2017.

As can be seen from the original data set labels distribution in figure 1, the data set was not balanced and that may lead to misguidance in the training process of our algorithms that are highly dependent in the distribution of the data. For our experiments, we dropped all the classes with less than twenty examples per class and performed sub-sampling of the data in the B,C and $01AE$ classes but trying to maintain the original distribution. We then divided the data in 75% train and 25% test, making sure that every sub type is represented in both the training and the testing set.

To determine if our data were suitable for the semi-supervised learning approach. We use Kameris' open source library for extracting the features of a given sequence, this feature vectors Φ_{orig} correspond to the normalized histogram of the occurrences of the 4^k different k-mer frequencies in a particular sequence. After extracting the features with $k = 5$ for convenience, we perform a simple inspection of the data using the first two and the first three components in the Principal Component Analysis (PCA). As can be seen from Figure 2 the distribution of our data for this particular embedding gives us useful information for the classification process in the sense that data with the same labels are connected by high-density paths.

3.2 Implementation Details

As mentioned, the new features correspond to a normalized histogram of the different k-mers. Then one have to be careful when selecting the similarity metrics. For the Graph based algorithms we didn't consider the Radial Basis Kernel both because its computational complexity and because it relies on the euclidean distance of every sample. For building the KNN matrix we accept the euclidean distance but under the assumption that for a small number of neighbours, since the manifold is Riemannian, it will be locally like an Euclidean space. For both Graph-Based Algorithms we used the open-source implementation in the sci-kit library.

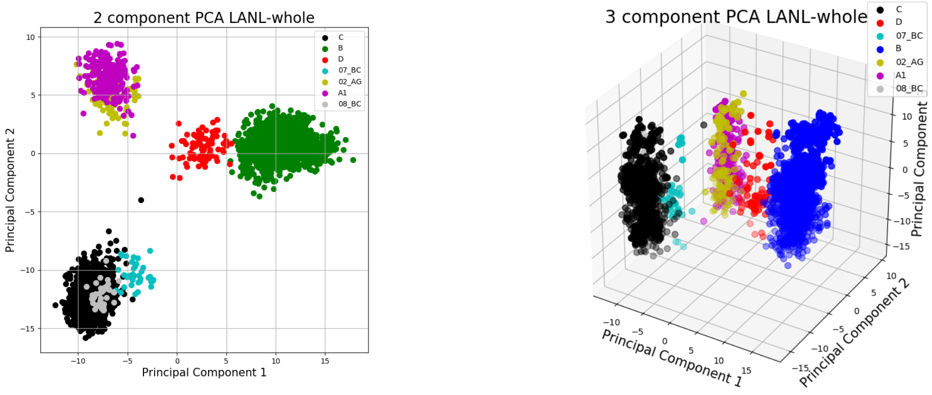

Figure 2: Principal component Analysis of the Whole-genome data set. Two principal component and 17% of the variance exhibited (left). Three principal components and 20% of the variance exhibited (righ).

In the case of the Neighborhood Kernel, there is evidence that using PSI-BLAST for the similarity measure will lead to good results. But since we would like to keep or method alignment free we explored with three different histogram distances:

- **Manhatan Distance**: Also known as the *cityblock* distance, rances from 0 to 2.

$$D_{MD}(P,Q) = \sum_{i=1}^{N} |p_i - q_i|$$

- **Bhattacharyya Distance**: Is defined between two probability functions from a divergence

$$BD(P,Q) = \sum_{i=1}^{N} \sqrt{p_i q_i}$$

  which ranges from 0 to 1, with 1 for P and Q being identical. The Bhattacharyya distance is defined from its divergence as $D_{BD}(P,Q) = -\ln BD(P,Q)$

- **Histogram Intersection**: Ranges from 0 to 1, with 1 for P and Q being identical

$$HI(P,Q) = \sum_{i=1}^{N} \min{(p_i, q_i)}$$

  to make it a distance we take $D_{HI}(P,Q) = 1 - HI(P,Q)$

For this implementation we built the function to compute the Neighbnourhood kernel and used it alongside with the sic-kit implementation of the Support Vector Machine algorithm. For both methodologies, we randomly dropped the labels of the 90% and the 50% of the training set.

3.3 Results

In this section we present the different results obtained, it is important to highlight that the state-of-the-art for this particular data set is **96.75%** of accuracy [1] before giving the results .

In Table 1 we summarize the results obtained for the Neighborhood Kernel using the different distance metrics and a different sizes of the labeled training set. For this models we tuned the threshold hyper-parameter using cross validation and the results are summarized in Figure 3 for each distance metric.

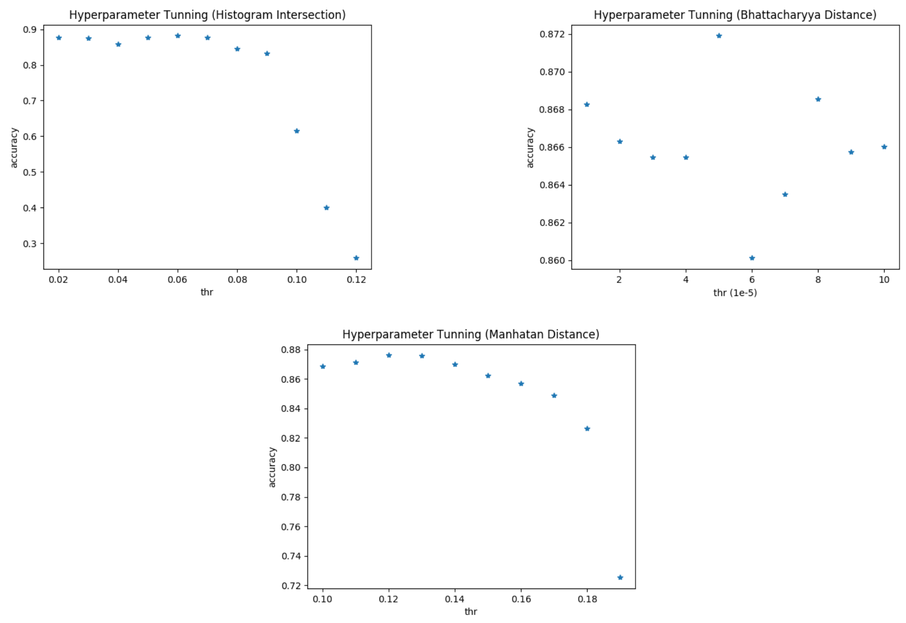

Figure 3: Cross-Validation training accuracy for each similarity metric in the Neighborhood Kernel. Manhattan Distance (Bottom). Bhattacharyya Distance (Upper Right) Histogram Intersection (Upper Left).

Table 1.

Model	% Labeled Data	Accuracy
Manhattan	50%	0.942
Manhattan	10%	0.921
Bhattacharryya	50%	0.913
Bhattacharryya	10%	0.90
Intersection	50%	0.924
Intersection	10%	0.897

Table 2 summarizes the results obtained for the Graph-Based methods using a different size of the labeled training set. For this models we did not tune the hyper-parameters using cross validation, but we use some heuristics to select both the number of neighbors and the value of the parameter α in the Label Spreading Technique. We selected 5 neighbours and $\alpha = 0.8$

Table 2.

Model	% Labeled Data	Accuracy
Label Propagation	50%	0.874
Label Propagation	10%	0.614
Label Spread	50%	0.913
Label Spread	10%	0.648

Finally we present the confusion matrices for the best model of each approach as well as the confusion matrix obtained using the supervised Learning approach.

4 Discussion

As expected, the use of k-mer frequencies is a good fit for the semi-supervised approach and each of the methods perform well, being the Neighborhood Kernel with the Manhattan distance the best model. This was expected because for the Neighborhood Kernel there is not a fixed number of neighbors for each sample, so the cardinality of the Neighborhood depends merely on the density of the region, following the clustering assumption and making the model more robust to noise. Regarding the selection of the distance metrics I would have thought that the best one would be the Bhattacharyya distance since it is specially designed for histograms and probability distributions.

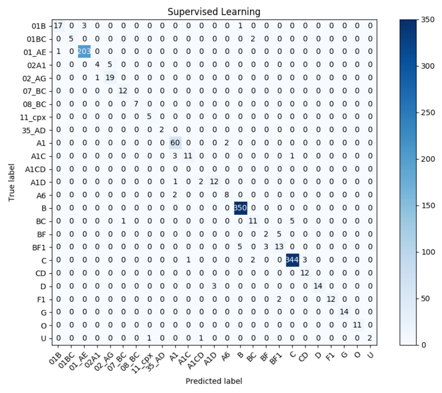
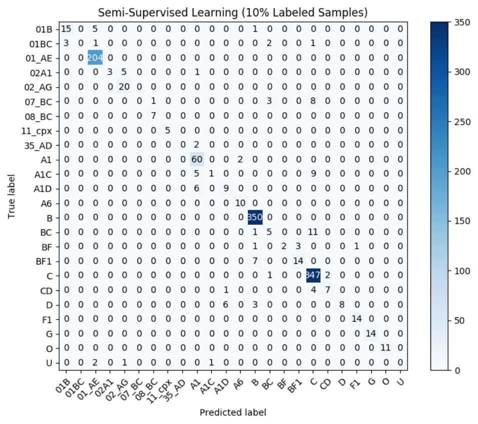
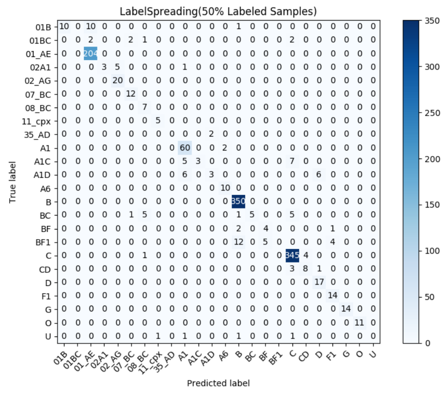

Figure 4: Confusion Matrices for the different approaches Label Spread (Bottom). Neighborhood Kernel (Upper Right) Supervised Learning (Upper Left).

However, for our experiments the value of the distance is too small and that makes difficult the selection of a good threshold. Finally, as expected, the Histogram Intersection performs slightly worse than the other methods as it is more sensitive to noise.

Regarding the Graph-Based methods, convergence was harder when fewer labeled samples were provided and the classification accuracy is also lower. This is not surprising since as it is mentioned in [4], the problem is equivalent of solving a linear system using the Jacobi iteration Matrix. Giving more unlabeled data, increases the size of the system making the convergence slower.

Finally, by looking at the confusion matrices one can evidence that the error rate is grater for classes with low representation because the dependency on the original distribution of the data.

As a result of this project we ended up with a model that is slightly worse than the state-of-the-art classifiers but it uses just 10% of the training samples. As a personal experience it was nice learning about semi-supervised algorithms since this topic was not covered during class. As a future work it could be interesting the algorithms with raw data including CRFs and URFs to test it's performance

References

[1] Stephen Solis-Reyes, Mariano Avino, Art Poon, and Lila Kari. An open-source k-mer based machine learning tool for fast and accurate subtyping of hiv-1 genomes. *PLOS ONE*, 10.1371, 2018.

[2] Taylor BS, Sobieszczyk ME, McCutchan FE, and Hammer SM. The challenge of hiv-1 subtype diversity. *The New England journal of medicine*, 358,15, 2008.

[3] Struck D, Lawyer G, Ternes AM, Schmit JC, and Bercoff DP. Comet: adaptive context-based modeling for ultrafast hiv-1 subtype identification. *Nucleic acids research*, a42,18, 2014.

[4] Olivier Chapelle, Yoshua Bengio, Bernhard Scholkopf, and Alexander Zien. *Semi-Supervised Learning*. The MIT Press, 2006.

[5] Rushdi Shams. Semi-supervised classification for natural language processing. *CoRR*, abs/1409.7612, 2014. URL http://arxiv.org/abs/1409.7612.

[6] Yusei Kobori and Satoshi Mizuta. Similarity estimation between dna sequencesbased on local pattern histograms ofbinary images. *CoRR*, abs/1808.00496, 2018. URL https://www.biorxiv.org/content/biorxiv/early/2015/03/05/016089.full.pdf.

[7] Olivier Chapelle, Jason Weston, and Bernhard Schölkopf. Cluster kernels for semi-supervised learning. In *NIPS*, 2002.