Phenotype Inference from Genotype in RNA Viruses

Chuang Wu

CMU-CB-14-101

July 2014

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Roni Rosenfeld, Chair
Jaime Carbonell
Elodie Ghedin (New York University)
Gilles Clermont (University of Pittsburgh

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Keywords: Machine Learning, Combinatorial Filtering, Active Learning, Disjunctive Normal Form Learning, Phenotype Inference

For my wife, son and the upcoming princess

Abstract

The phenotype inference from genotype in RNA viruses maps the viral genome/protein sequences to the molecular functions in order to understand the underlying molecular mechanisms that are responsible for the function changes. The inference is currently done through a laborious experimental process which is arguably inefficient, incomplete, and unreliable. The wealth of RNA virus sequence data in the presence of different phenotypes promotes the rise of computational approaches to aid the inference. Key residue identification and genotype-phenotype mapping function learning are two approaches to identify the critical positions out of hitchhikers and elucidate the relations among them.

The existing computational approaches in this area focus on prediction accuracy, yet a number of fundamental problems have not been considered: the scalability of the data, the capability to suggest informative biological experiments, and the interpretability of the inferences. A common scenario of inference done by biologists with mutagenesis experiments usually involves a small number of available sequences, which is very likely to be inadequate for the inference in most setups. Accordingly biologists desire models that are capable of inferring from such limited data, and algorithms that are capable of suggesting new experiments when more data is needed. Another important but always been neglected property of the models is the interpretability of the mapping, since most existing models behave as 'black boxes'.

To address these issues, in the thesis I design a supervised combinatorial filtering algorithm that systematically and efficiently infers the correct set of key residue positions from available labeled data. For cases where more data is needed to fully converge to an answer, I introduce an active learning algorithm to help choose the most informative experiment from a set of unlabeled candidate strains or mutagenesis experiments to minimize the expected total laboratory time or financial cost. I also propose Disjunctive Normal Form (DNF) as an appropriate assumption over the hypothesis space to learn interpretable genotype-phenotype functions.

The challenges of these approaches are the computational efficiency due to the combinatorial nature of our algorithms. The solution is to explore biological plausible assumptions to constrain the solution space and efficiently find the optimal solutions under the assumptions.

The algorithms were validated in two ways: 1) prediction quality in a cross-validation manner, and 2) consistency with the domain experts' conclusions. The algorithms also suggested new discoveries that have not been discussed yet. I applied these approaches to a variety of RNA virus datasets covering the majority of interesting RNA phenotypes, including drug resistance, Antigenicity shift, Antibody neutralization and so on to demonstrate the prediction power, and suggest new discoveries of Influenza drug resistance and Antigenicity. I also prove the extension of the approaches in the area of severe acute community disease.

Acknowledgments

I gratefully thank my advisor Dr. Roni Rosenfeld for his guide and support in my PH.D. research. I am fortunate to have worked with him during my Ph.D. training. Roni has helped me improve research skills, always inspired me with his insightful analysis of the results, and gave me advice in improving my academic writing.

A great debt is owed to Dr. Elodie Ghedin for our pleasant and fruitful collaboration. The research made great progress due to her valuable data and constructive thoughts. It has been a very unique and happy experience to work with her. Her knowledge and insights in Virology, especially in Influenza, taught me and inspired me in my research.

I am thankful to my dissertation committee members Dr. Jaime Carbonell and Dr. Gilles Clermont for their insightful questions, comments, and suggestions that helped me to improve my dissertation. Dr. Gilles Clermont also provided valuable data and domain knowledge to the project. Because I worked remotely in the project, Gilles has been very nice and helpful to guide the research and helped the writing. He has very high standard in the quality of the work, and kept revising the manuscript until it is satisfied. I really appreciate the opportunity to work with him.

I want to thank my wife and son for their unconditional love and support.

Contents

List of Figures

List of Tables

Chapter 1

The background and significance

1.1 RNA virus genotypes and phenotypes

RNA viruses (including retroviruses), such as HIV, Influenza, Dengue and West Nile, impose very significant disease burdens throughout the world, e.g., HIV infected patients are at a high risk of developing AIDS, which is now the fourth-leading cause of death worldwide [7]; Influenza A viruses are pathogens causing respiratory tract infections: in 2009 a new H1N1 virus emerged and caused a pandemic, infecting millions worldwide with 18,449 reported deaths worldwide [8]. Even during typical epidemic years, approximately 250,000 - 500,000 people worldwide die as a result of severe complications associated with influenza infections [9].

RNA viruses have a relatively small genome, this is probably because the lack of RNA error correction mechanisms that puts a limit on the size of RNA genomes [10]. RNA viruses are characterized as having short generation time, high replication rate and high mutation rate compared to DNA viruses, i.e., 10^4 higher rate than that of DNA viruses [10]. Variations are dominated by point mutations (rather than by cross over). Some viruses, e.g., influenza, have segmented genome reassortment [11].

Because of their very short generation time and low replication fidelity, RNA viruses exhibit extensive variability at the nucleic acid and protein level which results in fast adaptation rate,

and great ability to escape the immune system and antiviral drugs. For example, resistance has developed to all HIV drugs [12], and some drug resistance mutations are probably present before the start of therapy [13]. Influenza resistance to Neuraminidase Inhibitor is rare but the resistance mutations are emerging and the resistance is getting more prevalent [14]. Not all antibodies actually neutralize the virus, and not all neutralizing antibody neutralize all variants of the virus. A small number of mutations can render a viral antigen not neutralizable.

Understanding RNA virus phenotypes from genotype is in great need of computational tools to guide infection treatment, yet the prediction of phenotypes from genotype has proved challenging. A few important phenotypes and details that have been studied in the thesis are listed in appendix A.1.

1.2 Approaches to understand RNA virus Phenotypes from Genotypes

Two major approaches to understand RNA virus Phenotypes are:

- Key protein protein residues identification
- Genotype-phenotype function learning

Key protein residues identification takes a set of aligned protein sequences with phenotype labels as input, and identifies the position(s) that determine the phenotype changes, i.e., the mutations at these positions are a cause of the phenotype change.

Genotype-phenotype function learning also takes the alignment of protein or DNA/RNA sequences and their phenotype labels as input, and either learns a classification function $f : X \rightarrow \{+, -\}$ that maps the genotype to a binary function, or learns a regression function $f : X \rightarrow \{R\}$ that maps the genotype to real value.

These two approaches are important first steps in elucidating the mechanism responsible for that phenotype. They are also crucial steps towards inferring the phenotype from sequence

alone, which has broad uses in clinical decision making (e.g., antiviral drug choice based on drug resistance) and in public policy (e.g. vaccine formulation based on immunogenicity and cross-reactivity).

1.2.1 Current bench experiment approaches

In biological bench experiments, key identification is usually accomplished by a set of reverse-genetics laboratory experiments involving point directed mutagenesis and/or crossover between different strains, modifying a protein of interest or a section thereof. The procedure is laborious, time consuming and expensive. Generating a single new data point involves extraction of the appropriate nucleic acid sequence, mutagenesis, reconstitution of the slightly modified virus, and testing of the phenotype. Generating a single data point can thus take weeks, and may fail for a variety of reasons. The complete identification process can take many months. The positions to be mutated are selected by the experimenter, often based on human analysis of a single pair of viral strains which are maximally similar at the amino acid level yet exhibit opposite phenotypes. The experimenter then focuses on the residue positions where these two strains differ. Various combinations of the positions are introduced back and forth in an attempt to establish the necessary and sufficient conditions for the change in phenotype. Note in particular that no use is made of other available strains or isolates, even when (as is often the case) their sequence and/or phenotype are already known or can be readily derived (both sequencing and phenotypic assaying are often faster and cheaper than reverse-genetics experiments). This procedure is commonly employed by many experimental labs, on diverse phenotypes and viruses, as well as in non-viral contexts (e.g. [15], [16], [17], to name a few). From a computational perspective, though, the procedure as currently practiced is highly suboptimal. It is vulnerable to *inefficiency* (more experiments used than needed), *incompleteness* (settling on one explanation when other, equally simple ones exist), and the occasional *incorrect* inference (since inference is done informally).

1.2.2 Current computational approaches

Computational learning of genotype-phenotype mapping in RNA viruses explores machine learning techniques, such as support vector machine (SVM) regression [7], decision tree classification [18], statistical models [19][20], neural networks [21][22], recursive partitioning [23], linear stepwise regression [24], support vector regression [25], least-squares regression [25], and least angle regression [25]. These models learn from a training data set and then test their performance using a test data set. In terms of prediction accuracy, some classifiers outperform other classifiers for some datasets but lose for others. For HIV drug resistance data, SVM was shown to be superior to the other methods [26]. However, the performances of these methods were around 80% for HIV drug resistance datasets [25], suggesting that domain-specific knowledge has great potential to improve performance.

Existing work on statistical inference of genotype-phenotype relationship focuses on population genetics, using linkage analysis and association studies. Linkage analysis is not applicable to our case because crossover is not a significant force in the evolution of most RNA viruses. Similarly, association studies are not applicable here because they can only detect single-locus associations, or else require exceedingly large datasets: for a typical scenario where up to a few dozen labeled sequences are available and the phenotype depends on 2-4 key residues that interact in a complex fashion, there is not enough power in statistical tests to identify these residues. More specifically, tests like those described in [6][27][28] look for association between each individual residue position and the phenotype. But if the phenotype is determined by a complex interaction among, say, four residue positions, then there will be only moderate association between any one of these positions and the phenotype label, and this association may not be reliably detected with the limited number of labeled sequences that are usually available. This is a weakness shared by all methods that look for phenotypic association with individual residue positions (call these "*position-specific association methods*" (PSAM)). This deficiency in the real data was described in [1]. Although position-specific association methods can be expanded to look for

phenotypic associations with any pair or triplet of residues etc., the exponential growth in the number of covariates further reduces the power of the tests. An even more serious limitation of these methods is that they assume that the labeled data were independently sampled, a patently false assumption in most cases of interest.

Rule induction algorithms, such as simultaneous covering by decision tree algorithm [29], and ordered list of classification rules induction [30] can also mine if-then rules, but they only discover small number of rules for efficient prediction or classification purposes, and their rule forms are highly constrained by the tree structures. Sequence analyses using logic regression [31] and Monte Carlo Logic regression [32] adaptively identify weighted logic terms that are associated with phenotypes. These approaches do not explore the whole hypothesis space to identify all possible solutions, hence they are not guaranteed to extract the global optimal solution.

Many state-of-the-art machine learning approaches have been applied to RNA virus genotype-phenotype mapping, such as support vector machine (SVM) regression [7], decision tree classification [18], statistical models [19][20], neural networks [21][22], recursive partitioning [23], linear stepwise regression [24], support vector regression [25], least-squares regression [25], and least angle regression [25]. These models learn from a training data set and then test their performance using a test data set. The effort focuses on the prediction accuracy in the cross validation manner. These approaches lack the intention to learn biological meaningful and interpretable functions. Nonetheless, we will comprehensively compare our Disjunctive Normal Form (DNF) learning algorithms with these approaches regarding prediction qualities.

We are also not aware of any statistical or computational methods designed specifically to infer genotype-phenotype relationships in RNA viruses or other situations dominated by point mutations and small to moderate size datasets.

1.3 Goal and approaches

The goal is to appropriately exploit domain knowledge to design algorithms that can eliminate the vulnerabilities in key residue identification, improve prediction performance of genotype-phenotype mapping, and specifically solve three fundamental problems in these two approaches that have not been addressed in RNA viruses research: the scalability of the data, the capability to suggest new experiments, and the interpretability of the function.

- Scalability of the data. A common scenario of inference done by biologists with mutagenesis experiments usually involves a small number of sequences. These sequences are usually highly reliable since the mutations are Engineered. In this highly biased data, most statistics based approaches would fail in establishing significant solutions. On the other hand, high throughput sequencing generates huge amount of data with low reliability.
- Capability to suggest new experiments. This mostly applies to the study where the data are insufficient to infer the solutions. Domain experts usually design combinatorial bench experiments to narrow down the solutions. Without a computational algorithm to thoroughly search the candidate space, this procedure is usually suboptimal and takes more time and financial cost to finish.
- Interpretability of the mapping functions. This is an important but always neglected property of computational approaches. Most existing models behave as 'black boxes', and provide limited insights to the domain experts for guidance of experiment design and decision making.

We propose to design a combinatorial algorithm for key residue identification, and use Disjunctive Normal Form (DNF) to learn genotype-phenotype mapping. Combinatorial algorithms perform better on the small and highly biased datasets, because they neglect the distributions at each position throughout the data points and are capable of detecting any signal that potentially leads to phenotype changes. Combinatorial algorithms require fewer sequences to establish the

statistical power than statistical approaches. Nonetheless, the shortcomings shared by combinatorial approaches are that they are computationally inefficient and very sensitive to errors. The approaches are exploring biological plausible assumptions to constrain the solution space and guarantee finding the optimal solutions in the setup, and increasing the reliability of the data and designing soft versions of algorithms to allow appropriate amount of errors.

Specifically, for small to medium amounts of mutagenesis data, a combinatorial model is designed to identify the key residues consistent with the data by testing all plausible hypotheses. When the data is insufficient to learn the solution and more experiments are needed, the Active Learning approach is applied to suggest the most informative experiment to reduce the amount of further laboratory work and optimize the lab cost; Disjunctive Normal Form is proposed as an appropriate bias over the solution space to learn the genotype-phenotype mapping function.

The algorithms are first evaluated and validated on the simulated data and retrospective data where answers are known, and then proceeded to learn real biological datasets where no answers are known yet.

The structure of the thesis is the following:

- Demonstrate that solutions to genotype-phenotype mapping often come from a constrained space
    - Empirical evidence of domain knowledge
    - Spatial concentration of Key Residues (“Fingerprints”)
- Develop algorithms to exploit the constrained space(s)
    - Develop algorithms for Key Residue identification
        - Algorithm description
        - Demonstrate the usefulness of the approaches
            - Validate consistency and convergent efficiency
            - Retrospective validation of the algorithms

        - Application to unresolved problems
    - Develop Active Learning algorithms for these setups
        - Algorithm description
        - Demonstrate the usefulness of the approaches
            - Application of the algorithms to unresolved problems
    - Develop algorithms for mapping function learning
        - Algorithm description
        - Demonstrate the usefulness of the approaches
            - Validate consistency and convergence
            - Retrospective validation of the algorithms
            - Application of the algorithms to unresolved problems
- Applications to real unresolved problems
    - Influenza Antigenicity drift
    - Influenza drug resistance
    - Patient hospital mortality with severe community acquired pneumonia

Chapter 2

A Demonstration that solutions to Genotype-Phenotype mapping often come from a constrained space

2.1 Evidence of a small number of key residues in determining the phenotype changes

We develop algorithms that assume the number of key residues in the genotype-phenotype mapping is small. This assumption is important because it reduces the number of hypotheses from exponential (in the length of the protein) to polynomial, and constrains the form of the mapping functions. This assumption is biologically plausible as can be seen from the following examples representing diverse phenotypes:

- **Drug resistance:** Antiviral drug resistance of RNA viruses is a pervasive problem. Resistance develops to nearly all antiviral drugs soon after they are introduced. In Influenza, resistance to the M2 ion channel blockers amantadine and rimantadine is associated with two mutations in the M2 protein [33]. Resistance to the neuraminadase inhibitor Oseltamivir is

associated with mutations in the NA active site in positions 292, 294, 274, 119 [34], [35].

- **Immunogenicity:** In HIV-1, decreased immunogenicity was shown to be caused by three mutations in the gag protein [36].
- **Pathogenicity:** In Avian Influenza, dramatically increased pathogenicity was found to be associated with a small number of mutations in the polyprotein cleavage site [37].
- **Antigenicity:** In Influenza A, the investigation of the differences between vaccine strain (A/Panama/2007/99) and the circulating (A/Fujian/411/02-like) viruses showed that two mutations in the hemagglutinin protein are responsible for the antigenic drift [15].
- **Neutralization:** In HIV-1, neutralization escape was found to be associated with a small number of point mutations in gp120, gp41 ectodomain and in the cytoplasmic tail of gp41 [38], [39]. In Dengue type 2 virus, neutralization by the 3H5 mab was found to be abrogated by mutations in the envelope protein at positions 383, 384 or 385 [17].
- **Tropism** defines the cells and tissues of a host in which a particular virus can grow. In HIV-1, mutations at positions 425, 426 and 427 make the virus lose its ability to infect a monocyte cell [40].

2.2 Spatial concentration of Key Residues ("Fingerprints") in molecule bindings

RNA virus drug resistance, Antigen/Antibody binding, Infectivity, and so on, are determined by the structure binding between two protein chains. Genotype changes that happen in the binding active sites modify the structure of the functional proteins and potentially lead to phenotype changes. The binding is determined mostly by a binding region in the 3D space in which amino acids interact. Taking Antigen/Antibody binding for example, the contact area is smaller than $600\mathring{A}^2$, and the interaction involves a fraction of amino acids that reside outside facing the bind-

ing partner. The majority of key residues are concentrated in "neighborhoods" in 3D spaces. To better visualize the binding area, we plot the distances in 3D space against the position in protein sequence. The plots, named "contact maps" (Figure 2.1, 2.2) are generated for 250 Ag:Ab binding complexes retrieved from Protein Data Bank (PDB) in order to visualize the interacting regions of antigens, focusing on regions where the Ag and Ab are close to each other (e.g. $\leq 4\mathring{A}$). The "contact maps" show that the small area is composed of a couple of short regions along primary sequence that are the determinants of Ag:Ab binding. We call these piecemeal contiguous regions **"fingerprints"**. Hence, in our algorithm for identifying key residues in Ag:Ab binding, we will assume that these key residues are spatially clustered into a small number of "fingers".

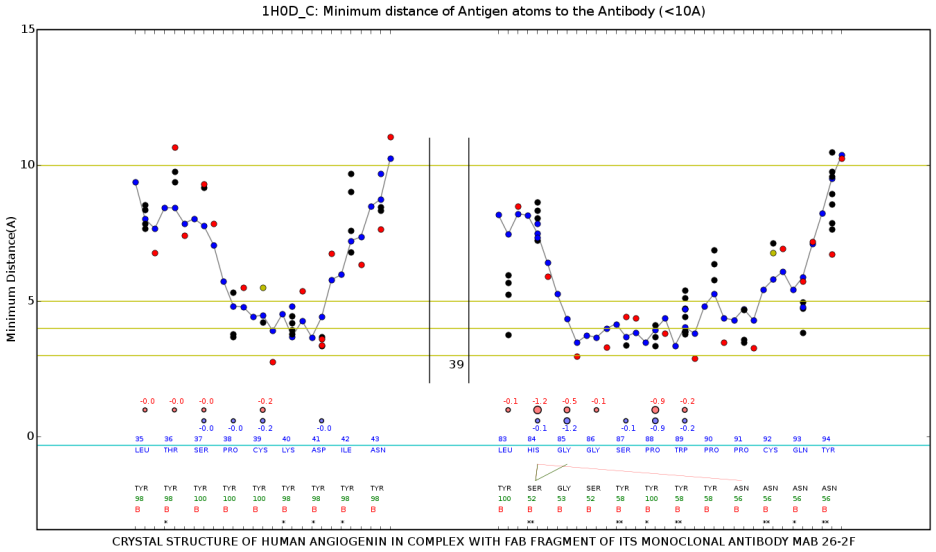

Figure 2.1: **Contact map**. The black curve in the center indicates the antigen backbone and the dots above and below the curve represent the atoms of each antigen amino acids (black dot: carbon, red: oxygen, blue: nitrogen, yellow: sulphur). Each horizontal position corresponds to one amino acid of the antigen. The bottom line designates the amino acids of the antibody. The Y axis represents the closest distance between the atoms on the antigen to the antibody. In this figure, there are two "fingers" with a gap of 39 amino acids between them. Each fingers is approximately 10 amino acids, although we expect that only a subset of amino acids in each finger actually interact with the antibody.

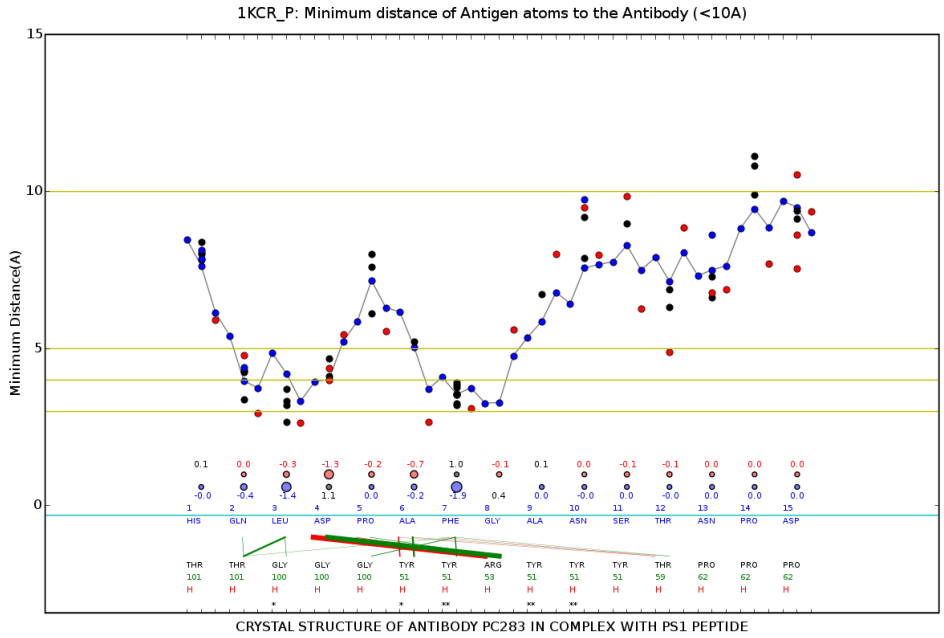

Figure 2.2: **Contact map 2**. Another example of a contact map. There are two ”fingers” with a gap of 39 amino acids between them. Each finger is approximately 10 amino acids, although we expect that only a subset of amino acids in each finger actually interact with the antibody. The original crystal structure is Fab of an Ab PC283 complexed with its corresponding peptide Ag, PS1 (HQLDPAFGANSTNPD), derived from the hepatitis B virus surface Ag was determined.

Chapter 3

Key Residue identification approach - Combinatorial Filtering (CF) algorithm

3.1 Background

We focus on typical mutagenesis experiments where a small number of sequences are available. The limited amount of sequences are usually engineered by point-mutations from a couple of "seed" sequences. Position-specific association methods (PSAM) (e.g. [6][27][28]) that measure correlation or mutual information often fail in identifying key residues in such highly biased datasets, especially when the key residues interact.

Combinatorial algorithms perform better than the PSAM algorithms on this type of highly biased datasets, because they neglect the distributions at each position throughout the data points and are capable of detecting any signal that potentially leads to phenotype changes. Combinatorial algorithms require fewer sequences to establish the statistical power than PSAM does; however, the shortcomings of combinatorial algorithms are that they are very sensitive to data errors and the computational time is usually exponential to the number of data points. The sequences in mutagenesis experiments are usually highly reliable due to the fact that major mutations are engineered; furthermore, we designed soft version of the algorithms to make them robust to

small amounts of errors; the computational time issues can be solved with the assumptions that the number of key residues is usually small and that they are distributed in several continuous regions along the sequences.

An important issue of mutagenesis experiments is that sometimes the data points are insufficient to infer the key residues, and biologists need to design more experiments to draw a conclusion. An Active Learning algorithm systematically searches the candidate space to suggest the most promising experiments in an incremental way to minimize the experimental costs.

In this chapter, we first describe the CF() algorithm, validate its performance with simulated data, and proceed to validate the algorithm with real biological problems. At the end we will describe an Active Learning algorithm and demonstrate its usefulness in a real evidence.

Combinatorial Filtering (CF) algorithm:

The CF() algorithm is essentially a List-then-Eliminate algorithm [41]. CF() represents all hypotheses explicitly, and each hypothesis contains the set of protein residue positions that are assumed to be "Key Residues" (KRs) – those affecting the target phenotype. By the assumption discussed above, there can be up to k key residues, where k is a small positive integer. Therefore, initially there are $O(L^k)$ hypotheses, where L is the length of the protein. Often, prior biological considerations can be brought to bear. In this case, only some regions or positions in the protein will be considered, and L will represent the number of such considered positions. Typically the lengths of RNA virus proteins are $100 < L < 500$.

The CF() algorithm is described in table 3.1. It is deemed to have converged when a single hypothesis remains in H. As described, the algorithm is applicable to binary phenotype as well as to categorical and ranked categorical phenotype (e.g. drug resistance indicated by "low", "intermediate" or "high"; or binding/neutralization phenotype with one of "-", "+", "++", "+++"). The filtering can be performed against every pair of different phenotypes, or a threshold can be established such that filtering is carried out only between sequence pairs whose phenotypes differ by at least that degree (e.g., "+" vs. "+++" but not "+" vs. "++"). The threshold can be chosen to trade off the efficiency of the inference vs. its robustness to noise. Similarly, the algorithm

Combinatorial Filtering Algorithm CF(S, k):	
Input:	
	S: aligned set of protein sequences of length L with their phenotypic labels
	k: integer; assumed number of relevant residue positions
Algorithm:	
	1. Initialize the hypothesis space H to include all k-tuples over $[1..L]$
	2. For each sequence pair (s_i, s_j) with *differing* phenotypic labels:
	2a. Identify all positions where s_i and s_j differ from one another
	2b. Eliminate from H all hypotheses that do not involve *any* of these positions
Output:	
	The hypotheses remaining in H

Table 3.1: **Combinatorial Filtering algorithm**

can be applied to continuous phenotypes by establishing a numerical threshold and carrying out filtering only among sequence pairs whose phenotypes differ by at least that amount.

The CF() algorithm aims to identify the minimum set of interacting key residues that explains the labeled data. It is therefore first called with the smallest possible k value, namely $k = 1$. If the output is the empty set, this means that no hypothesis of size k is consistent with the data. The algorithm is then called again with the next higher value of k. This is repeated until the output is not empty, resulting in the set of all minimal-size hypotheses consistent with the training dataset.

The initial size of the hypothesis space grows polynomially with the alignment length. The degree of the polynomial depends on the number of simultaneously considered key residues. The CF() algorithm described in table 3.1 can handle typical viral protein Multiple Sequence Alignment (MSAs) of up to about 1000 Amino Acids (AAs), and up to 8 KRs. Two variants of CF() algorithms, CF() with "fingerprints" and soft CF(), are designed to extend to large datasets with larger numbers of KRs and reduce the sensitivity to errors.

CF() algorithm for exploiting spatial clustering (Fingerprint) CF() algorithm with fingerprints is a natural extension of the CF() algorithm to consider the clustering of key residues in continuous regions. When the number of key residues is larger than two, we assume that they are clustered into C fingerprints where $C < k$, and the length of each cluster is not larger than

Combinatorial Filtering Algorithm for Fingerprints CFF(S, k):

Input:

S: aligned set of protein sequences of length L with their phenotypic labels
k: integer; assumed number of relevant residue positions
C: integer; assumed maximum number of fingerprints
L: integer; assumed maximum length of each fingerprint
$L1$: integer; assumed minimum length between two fingerprints

Algorithm:

1. Initialize the hypothesis space H to include all k-tuples
   1a. Select C fingerprints, each at most L long with inter distances smaller than L1.
   1b. Combinatorially select k key positions within the C fingerprints
2. For each sequence pair (s_i, s_j) with *differing* phenotypic labels:
   2a. Identify all positions where s_i and s_j differ from one another
   2b. Eliminate from H all hypotheses that do not involve *any* of these positions

Output:

The hypotheses remaining in H

Table 3.2: **Combinatorial Filtering algorithm for Fingerprints**

L. The inter distances between clusters need to be large enough (larger than $L1$) since the clusters form the active sites in 3D spaces and the 1D sequence is folded to form such structures. From the computational point of view, this assumption greatly reduces the number of candidate hypotheses in step 1, and the CF() algorithm can manage large datasets. The algorithm of CF() for fingerprints is shown in table 3.1.

The CF() for "fingerprints" is capable of learning KRs from large datasets with a large number of KRs. The finger parameters can be learned in a cross-validation way and the ones that balance the accuracy and running time are selected.

Another approach to managing large datasets is that while our main implementation is memory-bound, for large datasets with larger numbers of key residues we can design a "lazy mode" alternative implementation by pre-calculating the co-variance matrices of the genotype and then filtering the hypotheses.

Soft version CF() algorithm:

Soft Combinatorial Filtering Algorithm CF(S, k):

Input:

S: aligned set of protein sequences of length L with their phenotypic labels

k: integer; assumed number of relevant residue positions

T: integer; the lowest score for the hypothesis to survive

Algorithm:

1. Initialize the hypothesis space H to include all k-tuples over $[1..L]$
2. For each sequence pair (s_i, s_j) with *differing* phenotypic labels:
   2a. Identify all positions where s_i and s_j differ from one another
   2b. Assign a penalty score -p to the hypothesis when all the positions are the same between the two sequences

Output:

The hypotheses in $\{H, score(H) > T\}$, where T is a tunable threshold

Table 3.3: **Soft Combinatorial Filtering algorithm**

The soft CF() algorithm is modified from the CF() in table 3.1 to reduce the sensitivity to errors in data. One problem of the algorithm in table 3.1 is that any error in the data would potentially alter the solutions. A soft version CF() algorithm allows a small number of errors and the number is tuned by thresholds. The essential modification made in soft CF() is that in step 2b, instead of eliminating the hypothesis entirely, it assigns a penalty score to the hypothesis, and in the output step, a threshold is set to extract final hypotheses in the version space. The soft CF() is shown in table 3.1.

Hypothesis Space Visualization:

It is difficult to manually explore and assess a potentially large set of subsets. To facilitate interactive exploration by a domain expert, we depict the hypothesis space H using a histogram we call a *hypothesis density map*. An example is shown in Figure 3.1. The Y axis denotes the number of hypotheses remaining in H which involve the position of X. We can then use different colors to show the gradual shrinking of H as more pairwise comparisons are performed.

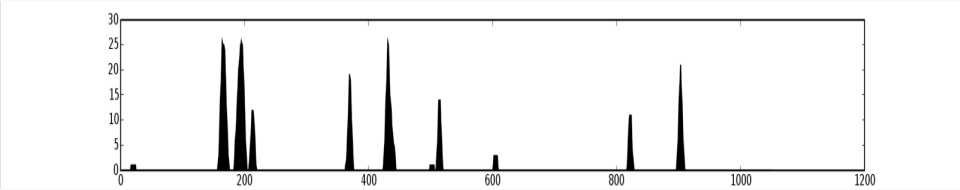

Figure 3.1: **An example "Hypothesis Density Map".** The Y axis shows how many of the remaining hypotheses involve position X.

3.2 Demonstrated CF consistency and convergence

The CF algorithm was first validated on simulated protein sequences with hypothetical target functions. When generating simulated sequences, we matched the position-specific amino acid distributions to those of a real protein dataset, then generated random phenotypic target functions. We used 732 HIV-1 gp160 protein sequences (downloaded from Los Alomos National Lab (LANL)), and assumed a variety of target functions. Each target function contains a small number of key residues (1 to 5), and was used to label the sequences accordingly. Notice that this method enables us to generate as many sequences as needed to test the algorithm's convergence under a variety of conditions.

We repeated this process many times, and in all of these cases the CF algorithm converged to the correct answer. An example run is shown in Figure 3.2, with a target function of three key residues, using 20 '+' labeled and 20 '-' labeled sequences (up to 400 comparisons).

3.2.1 Measured inference efficiency (convergence rate as function of dataset size) of CF algorithm

Inference efficiency: To assess the efficiency of our inference method, we would like to understand the relationship between the complexity of the function to be learned and the number of training examples needed to converge to it. Namely, we would like to know the convergence rate as a function of the number and type of available sequences and the complexity of the genotype-phenotype mapping. This is important because the available number of sequences

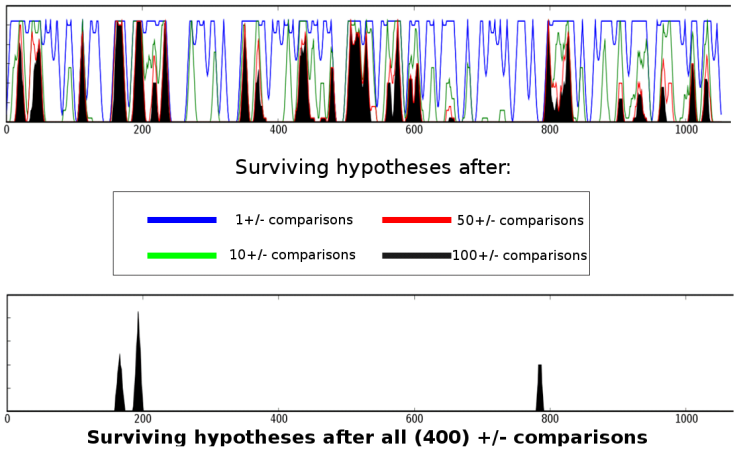

Figure 3.2: **CF consistency and convergence [1]**, visualized via a hypothesis density map. The multi-colored hypothesis density map depicts the gradual shrinking of the hypothesis space H. Once all pairwise comparisons were performed, the three correct key residues are identified by the sharp peaks.

vary for different datasets. Based on the convergence rate we can assess the likelihood of convergence, and whether (and how much) further experimentation will be needed. To do this, we used 911 aligned sequences of Influenza A H3N2 Hemagglutinin protein (HA) downloaded from the National Center for Biotechnology Information (NCBI) website, with an aligned length of 310 (HA1 domain). We then randomly generated putative binary target functions, each depending on a small number (1..5) of key residues. For each such target function, the 911 sequences were labeled accordingly. The CF() algorithm was then run three times, each time assuming a different number (k=1..3) of key residues. CF() was run incrementally, adding one sequence at a time, and the hypothesis space size was tracked. The whole process was repeated 100 times with different target functions to produce a statistically robust result.

Convergence Rate: As described in the introduction, the traditional discovery process uses a single pair of viral strains which are maximally similar at the amino acid level yet exhibit opposite phenotypes. The experimenter then focuses on the residue positions where these two strains differ. It is therefore most meaningful to compare the convergence of the hypothesis space under our algorithm with this method. Figure 3.3 shows the average log reduction in the size of the hypothesis space relative to its size after making only the first pairwise comparison

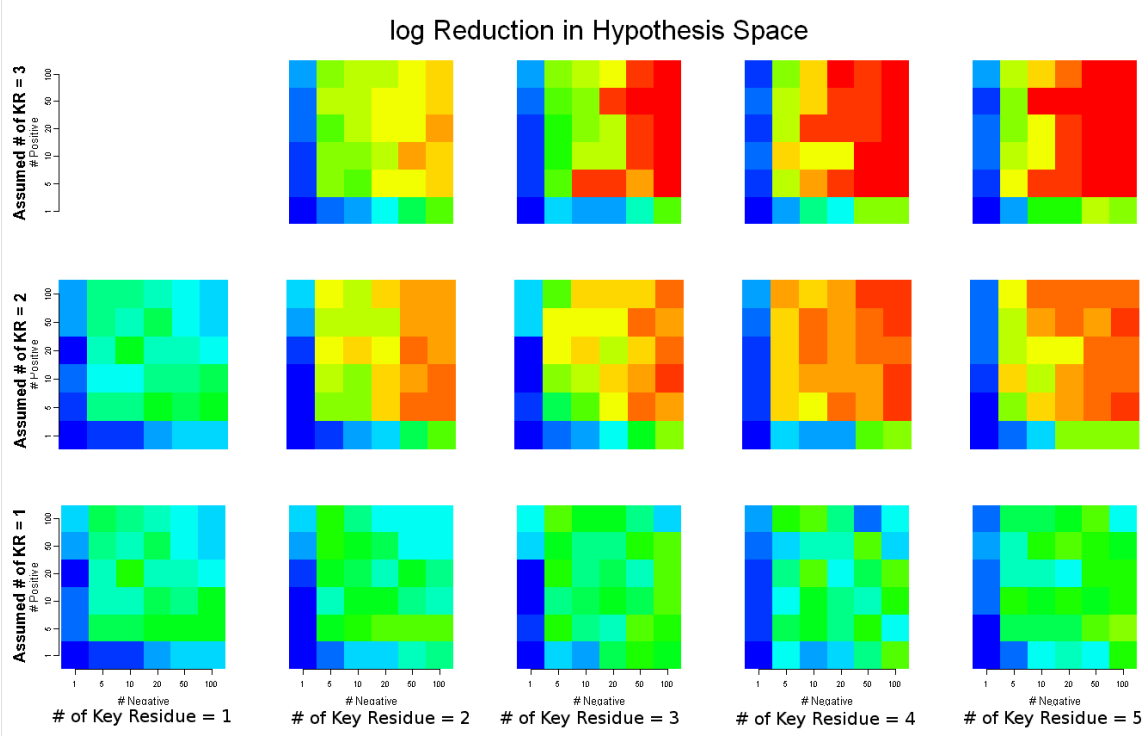

Figure 3.3: **Convergence Rate (credit to Andrew Walsh).** Average log reduction in number of hypotheses as function of the true function complexity (# of key residues), the assumed function complexity (Assumed # of key residues), and the number of positive and negative sequences (tick marks). See text. Blue: no reduction; red, complete convergence.

(that of the maximally similar pair). This is depicted as a function of the number of key residues in the target function (left-to-right), the number of key residues *assumed* by the algorithm (top-to-bottom), and the number of positive and negative sequences (vertical and horizontal rows of small colored squares, respectively, with values of 1, 5, 10, 20, 50, or 100 sequences each). A blue square indicates little or no shrinkage of the space (i.e., same power as the traditional method), whereas a red square indicates complete or nearly complete convergence. Since all the target functions used in this particular run were defined as a disjunction ("OR") of conditions over their key residues, we expect negative sequences to be more informative than positive ones. This is indeed borne out in the figure.

Another way to gauge the efficiency of our method is to let it make a wrong assumption and observe how fast the hypothesis space collapses (namely, shrinks into an empty set). Figure 3.4

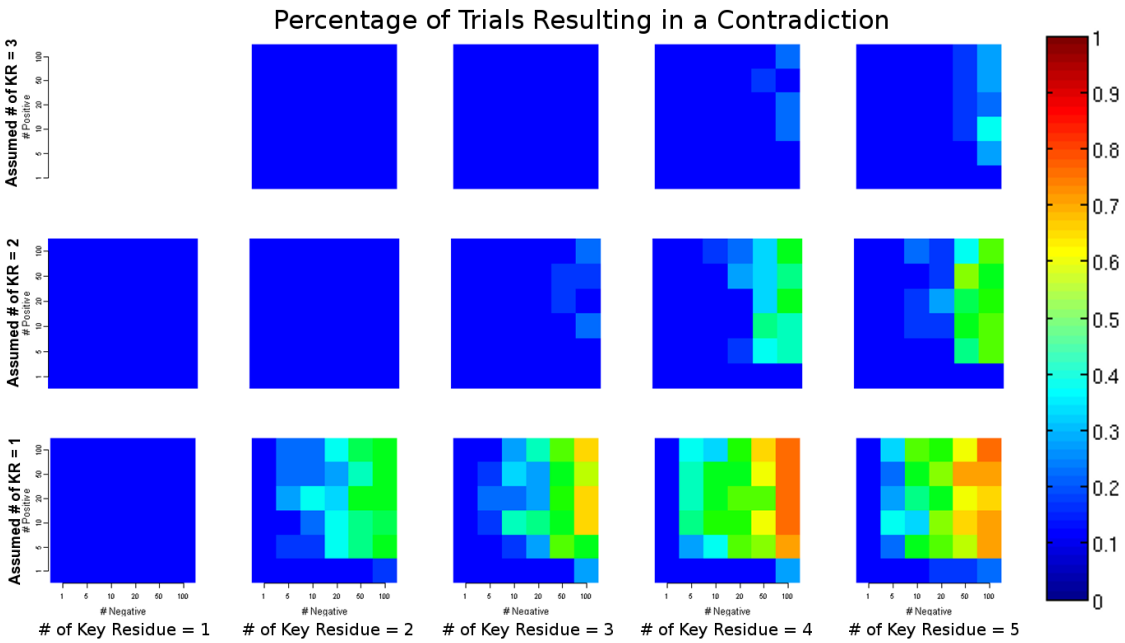

Figure 3.4: **Rate of collapse of the hypothesis space (to an empty set) under incorrect assumptions (credit to Andrew Walsh)**. As more sequences are used for inference, incorrect assumptions are eventually detected. See text.

shows the fraction of trials resulting in a collapse of the hypothesis space (blue=0, red=1). When the assumed number of key residues is equal to or larger than the correct number, naturally no collapse occurs, since the correct hypothesis is guaranteed to be in the space. However, when the assumed number of key positions is smaller than the true number, contradiction will eventually result. This argues in favor of the incremental relaxation of the assumptions discussed earlier (starting with k=1 and increasing it gradually).

Increased inference power: In the conventional discovery method, the positions that differ between the maximally-similar pair of strains constitute the "working set". New reverse-genetics experiments are performed to reduce the size of this working set. One way to gauge the advantage we gain by using *all* the available labeled sequences is to track how many of these "working set" positions are eliminated by the filtering process. Figure 3.5 shows the average relative reduction in this "working set" size. Notice that, under broad conditions, about half (green=0.5) the working set is eliminated before even a single experiment is performed.

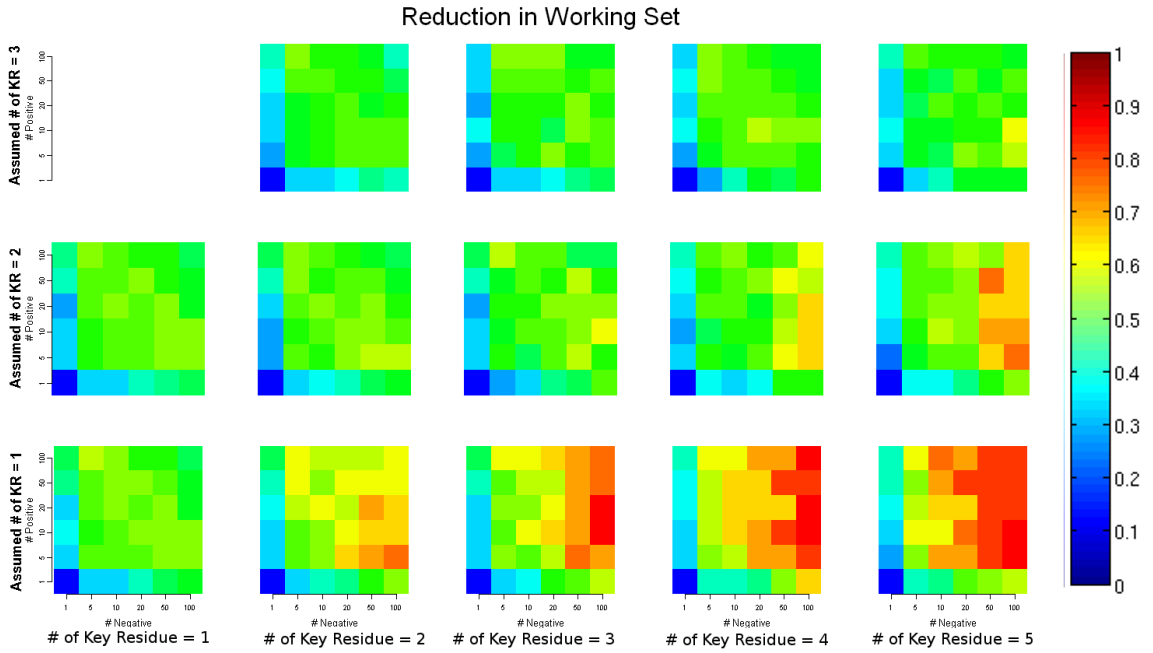

Figure 3.5: **Increased inference power over the traditional method (credit to Andrew Walsh).** Relative reduction in working set positions resulting from the filtering process. "Working set" positions are those that differ between the maximally similar +/- pair. Blue represents the lowest value (0) and red the highest (1). Under a broad set of conditions, about half (green=0.5) of the positions are eliminated without a single new experiment. See text.

(Note: The analysis above shows the *inferential inefficiency* of the traditional method. Another deficiency of the traditional method is its *incompleteness*: since it focuses on the positions that differ between the maximally similar sequence pair, it ignores other positions whose variation may affect the phenotype.)

3.3 Retrospective validation of the CF() algorithm

We retrospectively validated the CF() algorithm by testing it on datasets with real viral protein sequences where the genotype-phenotype mapping is already known and assumed to be correct. We compiled a number of datasets covering several RNA viruses with varying degree of average sequence identity (SI) and a variety of phenotypes, including Avian Flu High/Low pathogenicity (4 mutations in HA proteins changed the pathogenicity from low to high in H5N2 Influenza HA) [37], HIV Env U937 tropism (4 mutations in HIV Env proteins made the HIV unable to infect U937 cells) [40], Influenza H3N2 antigenicity shift (2 mutations in HA shifted the antigenicity of Influenza H3N2) [15], SIV Env neutralizability (2 mutations in SIV Env proteins determined the neutralizability of SIV)[42], FIV tropism in CRFG cells (2 mutations in FIV polymerase PA subunit made it unable to replicate in CRFK cells, SI: 95%)[43]. These conclusions were made from mutagenesis experiments which were chosen empirically or by domain knowledge. We applied our CF() algorithm on the same set of mutagenesis sequences to predict the key residues for the phenotype changes. For all the tests performed, our algorithm converged to the correct answer(s) (Fig. 3.6 **a-e**). In contrast, the conventional position-specific association method, while correctly identifies the key residues in **a**, yields high false positive and false negative rates in the other four datasets (Fig. 3.6 **a-e**).

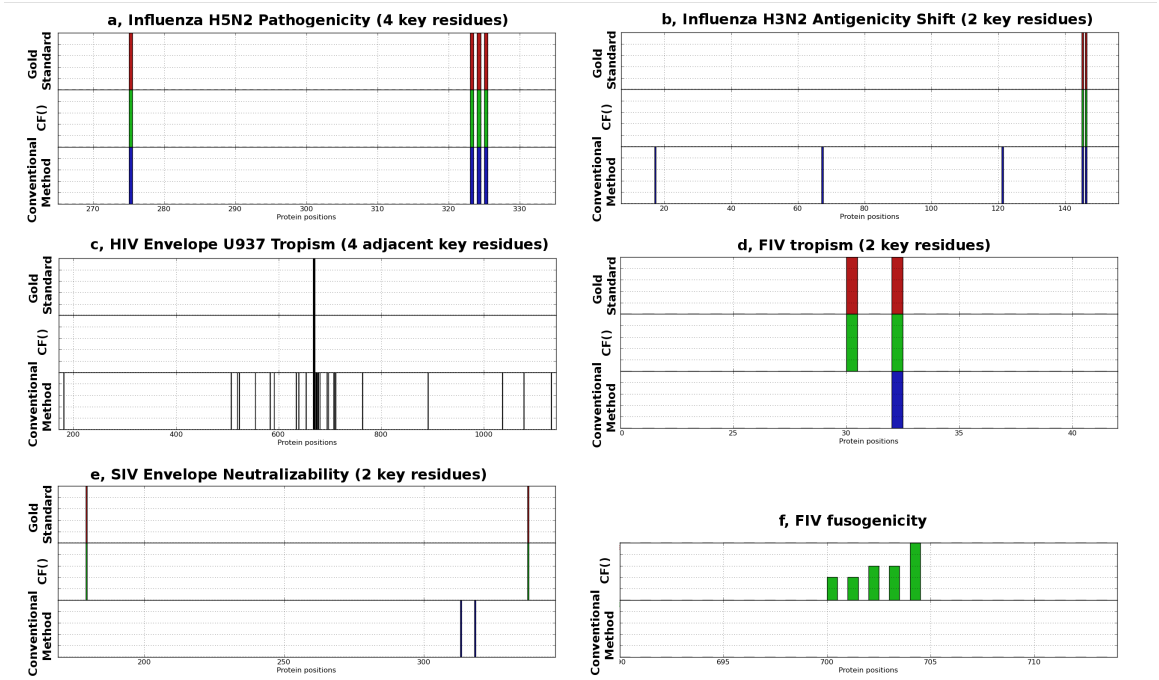

Figure 3.6: **Retrospective and prospective comparisons of the CF algorithm on a variety of datasets [1]**. a, Avian H5N2 high/low pathogenicity; b, Influenza H3N2 antigenicity shift; c, HIV Env U937 tropism; d, FIV polymerase PA CRFK tropism; e, Influenza H3N2 antigenicity shift. Red, gold standard extracted from the literature. Green, predicted by CF(). Blue, predicted by a conventional position-specific association method. f, FIV fusogenicity, for which a gold standard does not yet exist. CF() is able to suggest a set of key residues, but using the conventional method no positions survive cross-validation.

3.3.1 Comparing CF() with position-specific association methods

Position-specific association methods (e.g. [6][27][28]) that measure correlation or mutual information often fail to identify key residue positions when the number of available viral sequences is as small as in typical mutagenesis experiments or when the key residues interact. This is because the moderate statistical association between each such position and the phenotype is often not detectable from the several dozen labeled sequences that are typically available in computational studies or mutagenesis analysis. In contrast, a few dozen sequences are often enough for the CF() algorithm to converge to the correct answer.

To demonstrate this effect systematically, we stochastically generated synthetic sequences from a PSSM derived from a 99 residue long Multiple Sequence Alignment of HIV Protease. This assured that the marginal distribution of each residue position matched that of real HIV Protease data. The generated sequences were then labeled according to the desired target function, and used as the training set to both the CF() algorithm and several position-specific methods as in [6].

As an illustrative example, consider the phenotypic Boolean function (9L AND 36N) OR (70V AND 81A), which expresses two alternative mechanisms that confer some phenotype. This is a typical situation with regard to, say, drug resistance, where alternative mechanisms of resistance often develop in the patient population. When 40 positively-labeled and 40 negatively-labeled randomly selected sequences were used as input, the CF() algorithm converged to the correct set of key residues: (9,36,70,81). In comparison, the position-specific association method described in [6] failed to fully identify the four key residue positions, ranking these positions as #1, #2, #13 and #27 (out of 99). With the cut-off automatically determined as in [6], only 2 positions were captured, yielding a false negative rate of 2/4.

3.4 Applying the algorithms to an unresolved problem in genotype-phenotype mapping

We demonstrate the usefulness of our inference method with an application to an unresolved genotype-phenotype mapping. In [44] the authors attempted to identify the key residues that determine fusogenicity (the ability of the virus to fuse with the host cell) in FIV (Feline Immunodeficiency Virus), focusing on a highly conserved small region of the Env protein called principal immunodominant domain (PID). To do so, they generated random mutations in this domain, performed infectivity assays on the resulting mutants, and sequenced the respective PID domains. This is a commonly used experimental method to infer the key residues and explain the phenotype. In all they generated 24 labeled sequences that can be used to infer the key residues. Since this inference is non-trivial to the human eye, it is not surprising that the authors merely chose to state that "... there were no apparent common features of clones with mutations maintaining and clones with mutations not maintaining envelope fusogenicity ...".

Application of our CF() algorithm showed that a 1-key-residue hypothesis or 2-key-residue hypothesis were not enough to explain the data, resulting in a collapsed hypothesis space. However, the following five 3-key-residue hypotheses were found to be consistent with the data: [704,702,700], [704,703,700], [704,702,701], [704,703,701], [704,703,702] (the density map is shown in Fig 3.6-**f**). Distinguishing between these hypotheses will require additional data. Notice that position 704 occurs in all of them. Notice also that the position-specific association method, on the other hand, fails to identify any of the key residues (no peaks in the bottom half of Fig 3.6-**f**).

We also ran our Active Learning algorithm on this data, starting from the wild-type sequence, using the mutagenesis sequences as our candidate pool, and incrementally adding the next most informative mutated sequence, breaking ties randomly. The algorithm converged to the same hypothesis space shown in Fig 3.6-**f**, but did so using only a fraction of the 24 mutated sequences.

The Active Learning process was repeated 100 times, and required an average of 9.58 ± 2.95 mutants to converge – less than *half* the total number of mutants used by the authors.

3.5 Active Learning (AL)

The number of hypotheses remaining in H will depend on k, on the protein length L, and on the number of sequences and the entropy of their distribution. For biologically typical values of these parameters, when several hundred labeled sequences are available the correct hypothesis may be converged to by this procedure alone, without the need for any new data.

When the existing labeled data is not sufficient to converge, the experimenter needs to generate further labeled data by performing new mutagenesis experiments, reconstituting the virus, and running the phenotypic assay. Typically, many experiments may be considered, e.g. single-point mutagenesis of one of the given strains, crossover between two given strains, etc.

Usually some experiments are potentially more informative than the others, and *active learning* [45][46] algorithms are used to minimize the expected number of experiments to converge. Two possible scenarios of using active learning in key residue identification are 1) a pool of already-sequenced candidate strains is available with their labels are missing, in which case the most informative strain within the pool will be selected and phenotyped (pool-based active learning [47]); and 2) a set of potential mutagenesis experiments are planned, and only the one selected by active learning will be carried out and then phenotyped. Our active learning algorithm is designed to work with both situations (Table 3.4). More generally, the algorithm is made to return the expected informational utility of every candidate sequence. The experimenter then chooses an experiment with maximal utility, or they can choose a different experiment by factoring in financial cost, time, or other external considerations. If a general cost factor can be pre-assigned to each candidate experiment, the algorithm uses it to weigh the expected utility, leading to minimization of the expected total cost to convergence.

The running time of the AL() algorithm is $O(|H| \cdot |S'| \cdot (N_+ + N_-))$, where $|H|$ is the

Active Learning algorithm AL(**S,H,S'**):
Input:
A set **S** of already-available labeled sequences
The set **H** of surviving hypotheses (the output of applying Combinatorial Filtering to **S**)
A set **S'** of candidate sequences (each representing a candidate reverse-genetics experiment to be performed)
Algorithm:
For each s in **S'**
1. Estimate $\hat{p}(ph|s)$: the probability distribution over the possible phenotypic outcomes "ph" for experiment s, (e.g., for binary phenotypes, ph takes on the values "+" and "-".)
2. Calculate the expected entropy of the hypothesis space after performing such an experiment, as given by: $E[Ent(H|s)] = \sum_{ph} [\hat{p}(ph|s) \cdot Ent(H|s, ph)]$, where $Ent(H|s, ph)$ is the entropy of the hypothesis space H after performing experiment s, observing outcome ph, and filtering H based on that new labeled datapoint s and the already-labeled set **S**. Unless a prior is imposed on the hypotheses, we take $Ent(H) = \log(|H|)$.
Output:
the experiment that maximizes the expected informational utility: $s^* = argmin_{s \in S'} E[Ent(H|s)]$

Table 3.4: **The Active Learning algorithm chooses the next most informative experiment**

number of remaining hypoheses, $|S'|$ is the number of candidate sequences, and N_+ and N_- are the number of positively and negatively labeled sequences.

At times a set of related experiments can be performed together at a cost smaller than the sum of their individual costs. In that case the set is added as a single, joint experiment to **S'**, in addition to the component experiments.

Once the next experiment s is chosen, the experimenter performs it, observes the phenotypic outcome, and further filters H based on the new labeled datapoint and the already-labeled set **S**. s is then added to **S**, and $AL()$ is called again with the remaining experiments in **S'**. This process is continued until convergence to a single hypothesis, or until no experiment in **S'** can further reduce H.

Note that, in spite of the use of $\hat{p}(ph|s)$ in $AL()$, the $CF()$ and $AL()$ algorithms remain fundamentally combinatorial, not probabilistic. This is important because we do not want to

assume that the data in **S** is i.i.d. (usually it is not!). $\hat{p}(ph|s)$ is used only as a heuristic to estimate informativeness; it affects the convergence rate but not the convergence itself or the ultimate answer converged to. $\hat{p}(ph|s)$ can be estimated in a variety of ways depending on the setup, domain knowledge, etc. In the experiments reported below we used nonparametric estimation from the labeled data with a kernel based on sequence distance. This notwithstanding, AL() can accommodate a prior on H, and the hard elimination step can be changed to a soft penalty, effectively creating the equivalent of a likelihood function.

3.5.1 Visualization of Active Learning process:

Figure 3.7 shows an example of active learning on SIV Env neutralization data. SIV 17E-Cl and Sivmac239 are two SIV Env proteins with different neutralizability. These two proteins are well studied so they are chosen as the "already-labeled" sequences. Nine positions are different between the two sequences and new sequences are needed to narrow down the solution of the phenotype changes. While this is conventionally done by domain experts, we can instead use Active Learning to choose the most informative sequence. The candidate sequence set S' could be a set of mutagenesis experiments or existing sequences that are going to be phenotyped once they are chosen by the algorithm. In this SIV Env neutralization data, we will use a set of one-point mutation sequences and one-cross-over sequences as the candidate set because these mutation setups are quite common in the lab. In Figure 3.7, the information utility is plotted against each possible mutagenesis sequence. In this example figure, one-cross-over at the 4^{th} and 5^{th} positions yield the highest information gain. Once the next experiment is selected and phenotyped, this step is repeated to pick up the next experiment until a termination condition is met.

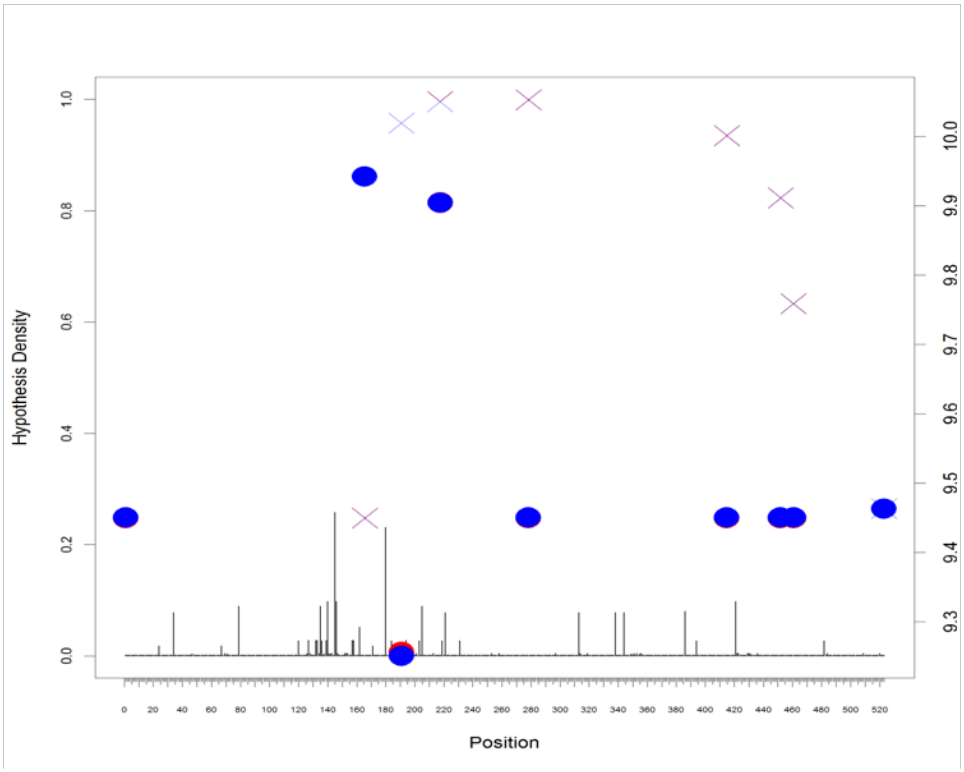

Figure 3.7: **Active Learning (credit to Andrew Walsh).** Expected information gain is plotted against primary sequence position. Possible one-point mutations (circle) and single-point crossovers (cross) are shown above the nine different positions that differ between the two strains. Blue and red colors indicate possible positive and negative outcomes respectively. In this figure, an experiment with a crossover between the fourth and fifth positions yields the highest information gain.

3.5.2 Active Learning on FIV fusogenicity data (needed only half the number of sequences)

We also ran our Active Learning algorithm on this data, starting from the wild-type sequence and incrementally adding the next most informative mutated sequence, breaking ties randomly. The algorithm converged to the same hypothesis space listed above, but did so using only a fraction of the 24 labeled sequences. The Active Learning process was repeated 100 times, and required an average of 9.58 ± 2.95 sequences to converge – less than **half** the total number of sequences used by the authors.

Chapter 4

Genotype-phenotype mapping - Disjunctive Normal Form

4.1 Background

RNA virus genotype changes that happen in the binding active sites modify the structure of functional proteins, and potentially lead to phenotype changes. To learn genotype-phenotype mapping we first characterize the structural binding of viral proteins. For example, drug resistance is often a steric-structural problem, and the physical interactions with inhibitors involve more than one part of the target molecule, e.g. Protease Inhibitors (PIs) bind to four or more binding pockets in the protease substrate cleft of HIV viruses [25]; a variety of active site properties are playing roles in the binding determination, such as residue types, hydrophobicity, charges, secondary structure, cavity volume, cavity depth and area etc. Therefore, multiple, alternative potential mechanisms exist, and each mechanism involves only a small number of mutations since it has to be "discovered" by the virus via random mutations. Therefore, a short Disjunctive Normal Form (DNF, "OR" of "AND") would be an appropriate bias over the hypothesis space under these assumptions.

We propose to use short Disjunctive Normal Form (DNF, "OR" of "AND") to learn genotype-

phenotype mapping also because 1) DNF is a high order boolean function that examines complicated solution space, 2) DNF offers great flexibility and allows identification of unforeseen interactions, 3) DNF is a natural form of knowledge representation for humans to interpret and provides clinical insights and rules to direct further executions, 4) DNF is scalable to large or small datasets. A short DNF increases interpretability and mitigates overfitting bias.

Conceptually DNF is a disjunction of conjunctions where every variable or its negation is represented once in each conjunction. The learning of DNFs is a machine learning technique to infer Boolean function relevant with a class of interest. It has been extensively used in electric circuit design, information retrieval [48], chess [49], and so on.

Considered as a core algorithm in concept learning, DNFs suffer from shortcomings: 1) the learnability of DNFs has been a fundamental and hard problem in computational learning theory for more than two decades, 2) DNFs are sensitive to errors in data, as are all Boolean function learning algorithms, 3) without the constraint of size, DNFs may suffer from a severe overfitting bias. We developed algorithms for accelerating and optimizing DNF learning and applied the techniques to a variety of RNA virus phenotypes.

4.2 DNF learning algorithms

A DNF is a standardization Boolean function, consisting of a disjunction of conjunctions, where the conjunctions consist of one or more positive and negative literals. Any given Boolean function $f : \{0,1\}^d \rightarrow \{0,1\}$ can be converted into an equivalent DNF. The following is an example DNF formula:

$f(x_1, x_2, x_3) = x_1 \wedge x_3 + x_1 \wedge \neg x_2 \wedge x_3 + x_2$

where '$\wedge$' denotes 'AND', '+' denotes 'OR', '$\neg$' denotes negation, and 'x' is a binary literal, indicating whether a particular test "Feature = Value" is true. A DNF formula is essentially a set of Boolean logic if-then rules, describing how the Boolean outcome is calculated based on Boolean inputs.

In Machine Learning, DNFs are traditional binary classifiers that predict Boolean outcomes from instance-based data. The size of DNF functions is two-dimensional: the number of conjunctive clauses and the maximum number of literals in each clause, thus a DNF is usually represented as k-term n-DNF, where k and n are the number of clauses and maximum number of literals respectively. In DNF learning, k and n are usually regularized because, without constraints, k and n tend to become very large, result in overfitting and defeating the purpose of interpretability.

Finding the minimum size DNF formula is a well-known NP-Complete problem [50, 51, 52]. There is no polynomial time learning algorithm, and existing practical solutions usually sacrifice completeness for efficiency. The existing heuristic or approximation approaches fall into deterministic [48, 53, 54] and stochastic algorithms [49, 55]. The deterministic methods include bottom-up schemes (learning clauses first and building DNFs in a greedy way) and top-down schemes (converting DNF learning to a Satisfiability problem). Stochastic methods randomly walk through the solution space to search for clauses but are not guaranteed to yield optimal solutions.

We developed two heuristic algorithms to accelerate the DNF learning by narrowing the solution space under the domain assumptions: standalone DNF learning and monotone DNF learning (MtDL).

1. Converts DNF learning to learning k-DNF where $k \geq 1$ is the maximum size of conjunctive clauses. (Standalone DNF learning algorithm)
2. Exhaustively learns monotone DNF(s) after feature selection (MtDL)
3. Greedily learns DNFs for hard problem settings.
4. Extracts biologically meaningful solutions and better predicts phenotypes from genotypes.

4.2.1 Standalone DNF learning

Valiant [56] showed that for every constant $k \geq 1$, k term DNF can be PAC learned in polynomial time by k-CNF, i.e. CNFs with at most k literals in each clause. K-term DNF learning is essentially a combinatorial problem. The standalone DNF learning algorithm first learns a set of conjunctive clauses deterministically with the maximum clause length of k (Table 4.1), and then converts the DNF learning process to a typical SET-COVER problem (Table 4.2). The DNF learning algorithm is equivalent to finding the minimum number of sets that cover all the positive sequences. The SET-COVER problem is again NP-Complete, by limiting the maximum clause length, for typical RNA virus problem settings the number of clauses is usually manageable and the SET-COVER can be exhaustively completed. Typically, the number of possible clauses of size$k$ is up to L^k, where L is the sequence length. The actual number of clauses that appear in the dataset is much smaller than this number, especially for biologically conserved datasets. After equivalence filtering (see Section 4.2.5), the number of learned clauses is usually on the order of several hundreds in the RNA virus domain. The standalone algorithm can efficiently infer DNFs from small (a couple of sequences) to medium size (hundreds of sequences) datasets, or large conserved datasets.

Clause Learning Algorithm (S, k):
Input:
A set **S** of already-available labeled sequences
k: assumed upper-bound length of clauses (a small positive integer)
Steps:
1. Enumerate all combination of literals to form conjunction clauses
2. Record the set of (positive and negative) sequences that each clause covers (n_j^+, n_j^-)
Output:
The set of clauses C and the corresponding sequence index sets (N^+, N^-)

Table 4.1: **Clause learning algorithm**

Disjunctive Normal Form Learning Algorithm(C, n^+):
Input:
A set **C** of clauses
n^+: the set of positive sequence index to be covered by the clauses
Steps:
1. Equivalent filtering (see Section 4.2.5)
2. Among the clauses that cover only the positive sequences, find a minimum set of clauses that cover all the positive sequences:
2a. start from the clauses that cover the positive sequences which are rarely covered by other clauses
2b. repeat 2a recursively until all the positive sequences n^+ are covered
Output:
The set of the shortest DNFs

Table 4.2: **DNF learning algorithm**

4.2.2 Monotone DNF learning after feature selection (MtDL)

MtDL utilizes feature selection to narrow the feature space and infer DNFs within the space. The choice of feature selector is critical to the MtDL algorithm. The best feature selector needs to guarantee that the selected feature space is a superset of the DNF solution space, and best limit the feature space for efficient learning. The Combinatorial Filtering (CF) algorithm (in Table 3.1) works seamlessly with MtDL as a feature selector. CF() efficiently identifies the smallest set of features that completely explains the differences between classes, such that MtDL is guaranteed to learn DNFs. In this study, MtDL always runs together with CF() to infer DNFs (Table 4.3). Notwithstanding, other feature selection methods, such as LASSO, Logistic Regression with regularization, or dimension reduction methods like PCA, are also good candidate selectors, but in these cases, the coverage threshold might need to be set.

Upon completion of feature selection, the MtDL enumerates literals in the selected features and then combines them into conjunctive clauses (Table 4.3). MtDL differs from the standalone algorithm in that it does not limit the maximum size of clauses but completely considers all $2M$ possible combinations. Take a typical example: once L features are selected in step 1, there should be at most $M = V * L$ possible literals, where V is the size of value space. Because literals from the same feature will not appear in the same conjunctive clause, we do not need to consider

all $2M$ combinations, but only combinatorially choose up to L literals from M. Hence the total number of clauses is at most $N = (MchooseL)$. In reality, depending on the divergence and the amount of the data, the actual number of possible literals is always much smaller. Furthermore, in step 4, the N clauses will be pre-filtered by removing clauses that cover any negative sequences. When the clause pool is ready, in step 5 the algorithm incrementally constructs the combination of clauses to be candidate DNFs and examines the coverage of data points. MtDL starts from one clause, and checks the next larger number if no solution is found. The algorithm terminates when the DNFs exclusively cover all positively labeled data points but not any of the negatively labeled ones.

Monotone DNF Learner (F, S):
Input:
F: A set of selected features (by CF(), for example)
S: the labeled training datasets
Steps:
1. Construct $\{$L$\}$, the list of literals in the features (e.g. 5A).
2. Throw out L that does not cover any positive sequences.
3. Combinatorial construct $\{$Clauses$\}$, the list of conjunctive clauses from $\{$L$\}$, (e.g. $5A \wedge 8C$). The possible combinations are $\|L\|$ chooses $1, 2, .., \|F\|$.
4. Throw out the conjunctive clauses that cover any negative sequences.
5. Incrementally construct $\{$DNF$\}$, the list of disjunctive normal form that covers all positive sequences but no negative sequences: starts from 1 clause, construct DNFs from $\{$Clauses$\}$, try the next larger number if no solution learned.
Output:
The set of the shortest DNFs

Table 4.3: **Monotone DNF learning algorithm**

4.2.3 DNF learning for Fingerprints

Similar to the extension of the CF() algorithm for Fingerprints, the two DNF learning algorithms above can be naturally modified to learn DNFs within continuous clusters. Since both DNF learning algorithms are bottom-up approaches, in the clause construction step, instead of enumerating all combinations, we set cluster limitations similar to that in Table 3.1. DNF learning for Fingerprints can learn DNFs from datasets where the assumption holds, or for very large

datasets to reduce the computational running time.

4.2.4 Greedy versions of both algorithms

The greedy versions of the two DNF learning algorithms are designed to rapidly learn DNFs. In the DNF construction step, the greedy algorithms iteratively select the clause that covers the largest number of the uncovered positive sequences and zero negative sequences until all positive sequences are covered.

4.2.5 Equivalence filtering

Computationally equivalent clauses cover the same set of sequences while differing in their composition literals. Replacing one clause with its equivalent clauses in a DNF will not change the predictions of the DNF on the same training set. Equivalent clauses are very common in clinical datasets; therefore, during DNF learning process equivalent clauses are filtered and only one of them is used as the representative to construct DNFs. By using equivalence filtering the DNF learning running time is greatly reduced. Note that the equivalence filtering is only for computational efficiency purpose. After learning DNFs, all clauses that have equivalent clauses will be expanded to recover all the DNFs.

4.2.6 Avoiding over-fitting and robustness to noise

We used pruning and threshold setting to mitigate over-fitting and improve the algorithms' robustness.

1. **DNF pruning**: similar to the pruning of decision tree, clauses that only cover a small number of sequences may be pruned if removing them results in an increase of the prediction accuracy on the test dataset. The advantages of pruning are:

    - Avoiding over-fitting, because irrelevant clauses are removed,

- Shortening DNFs, which makes them easier to understand and more biologically meaningful and
- Increasing robustness to noise, because pruned DNFs ignore the clauses/literals rendered meaningless by noise.

2. **Threshold setting**: Setting thresholds on the fractions of the sequences that the learned DNF(s) cover: the DNF learning algorithms can be easily modified to terminate when at least a fraction p of the positive sequences are covered, and at most a fraction n of the negative sequences can be covered by the DNFs. The thresholds p and n are determined by cross-validation. Similar to pruning, threshold setting can also avoid over-fitting, learn shorter DNFs and be robust to noise. One advantage of threshold setting over pruning is that the former technique usually achieves better prediction quality.

4.2.7 Extension of literals

The literals can also be extended to negation of one amino acid or a subset of the amino acids.

4.2.8 Extension to multiple class data

The algorithm is applicable to multiple class data by running the algorithm multiple times. Each time, one of the classes is designated as the positive class and the rest are merged as the negative class.

4.3 Demonstrating DNF learning algorithms consistency

The DNF learning algorithms are first validated on simulated viral sequences to assess the consistency and learning efficiency in a practical way. When generating the simulated sequences, we match the position-specific amino acid distributions to those of a real protein dataset, and generate random phenotypic target functions (making sure they did not label the entire dataset

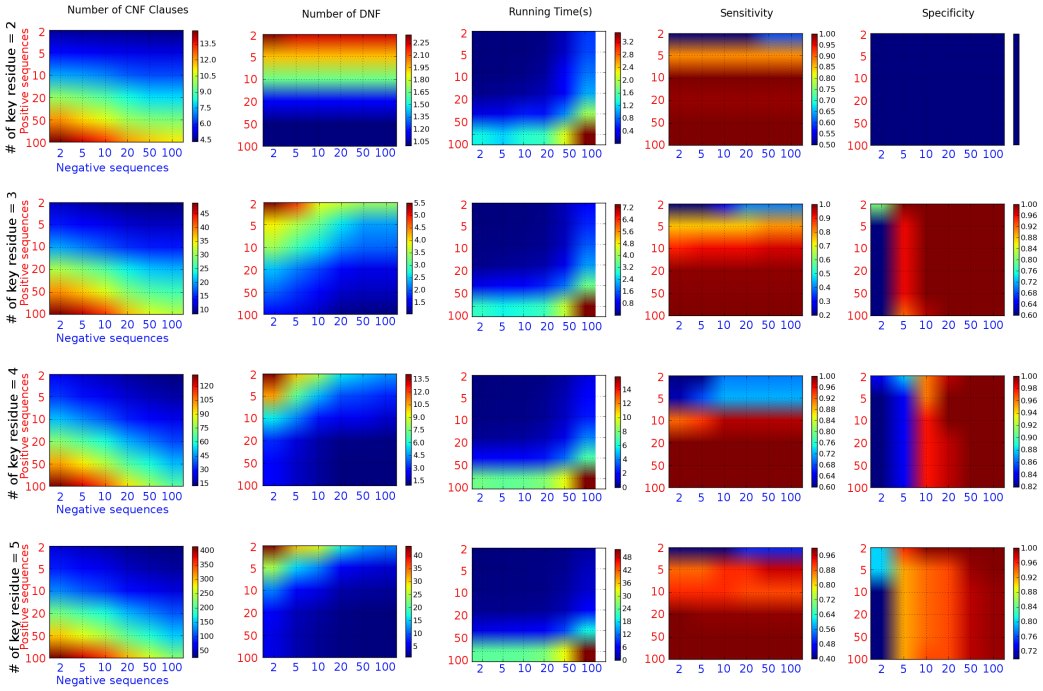

Figure 4.1: **The evaluation of MtDL algorithm on simulated sequences [2].** From left to right, the number of CNF clauses, the number of DNFs, running time, prediction sensitivity and specificity are plotted as functions against the number of key residues assumed in the target function (rows), and the number of positive sequences and negative sequences (vertical columns and horizontal rows of small colored squares). The numerical values of the colors are shown in the colorbar. Take the top left chart for example, when the key residues are assumed to be 2 in the target function, with say 100 positive and 2 negative sequences used, the number of CNF clauses is about 12 (red color means higher value as indicated in the colorbar).

with the same value). We use 732 HIV-1 gp160 protein sequences (downloaded from LANL), and a new hypothetical DNF is proposed in each study, (e.g. $(70a \wedge 9l \wedge 11p \wedge 70t) + (62l \wedge 45m \wedge 36y \wedge 9l) + (62P \wedge 53V \wedge 36s) + (83i \wedge 45I) = +$), and is used to label the sequences accordingly. This method enables us to generate as many sequences as needed to test algorithm convergence under a variety of conditions. We repeated this process many times, and in all of these cases both standalone algorithm and MtDL algorithm converge to the target functions with a moderate number of sequences.

4.4 Measuring inference efficiency (convergence rate as a function of dataset size)

The convergence rate is studied as a function of the number of available sequences and the complexity of the genotype-phenotype mapping functions. This study is important because the available number of sequences varies dramatically in different datasets. Once we know the convergence rate we can assess the likelihood of convergence, and whether (and how much) further experimentation will be needed for DNF learning algorithms to converge.

Five hundred eighty-eight aligned sequences of HIV protease protein are downloaded from the Stanford HIV database. Hypothetical DNFs are randomly generated as target functions, each depending on a small number (2..5) of literals. For each such target function, the 588 sequences were labeled accordingly. The DNF learning algorithm was then run 20 times, each time assuming a different target function to produce a statistically robust result. The evaluation results of MtDL are discussed and shown in the following sections, and the simulation result of the standalone algorithm is similar.

Convergence Rate: Figure 4.1 shows the number of learned DNFs as a function of the number of literals in the target function (# of key residues, top-to-bottom), and the number of positive and negative sequences (vertical and horizontal rows of small colored squares, respectively, with values of 2, 5, 10, 20, 50, 100 sequences each). Blue indicates convergence (i.e. one single DNF is learned given the amount of data, and the learned DNF is exactly the same as the target function), and red indicates not converged yet (additional DNFs are also learned). With this simulated dataset the MtDL algorithm converges using only about 50 positively labeled and 50 negatively labeled sequences.

Running time: DNF learning running time is exponential to the number of sequences and sequence length. RNA viruses tend to have shorter protein lengths with 100-500 amino acids. The constrains on the clause length and the feature selection approaches greatly reduce the clause

pool size and exhaustive learning is now manageable. We also use "equivalent filtering" (section 4.2.5) to accelerate the learning speed. The running time on the simulated sequences is on the order of seconds.

The prediction sensitivity and specificity showed that the algorithms converge with only moderate numbers of sequences. Interestingly, the prediction quality charts are symmetric such that if we flip the labels of the data, the prediction accuracy will remain the same. This is important because although our DNF learning algorithms identify DNFs that only cover the whole positive space, the sequences in both classes contribute equally to the learning.

4.5 Retrospectively validating DNF learning algorithms when ground truth is known

The DNF learning algorithms are retrospectively validated with a variety of phenotypes where the genotype-phenotype mappings are already known and assumed to be correct, including Avian Flu High/Low pathogenicity (four mutations in HA proteins changed the pathogenicity from low to high in H5N2 Influenza HA) [37], Influenza H3N2 antigenicity shift (two mutations in HA shifted the antigenicity of Influenza H3N2) [15], SIV Env neutralizability (two mutations in SIV Env proteins determined the neutralizability of SIV)[42], FIV tropism in CRFG cells (two mutations in FIV polymerase PA subunit made it unable to replicate in CRFK cells)[43]. The MtDL+CF algorithm learns the mapping functions in each dataset and the functions contain exactly the same set of positions as in the known answers (Table 4.5).

In contrast, the conventional position-specific association method (PSAM) we selected as comparison [6] only correctly identifies the positions in one of the four datasets, and yields high false positive and false negative rates in the other three datasets (Table 4.5). For an H5N2 hema Pathogenicity dataset, PSAM identifies the key residues correctly because the sequences are very conservative and only differ in those key residue positions. PSAM tends to have high false

Data set name (# pos/# neg seq)	Golden standard (identified mutations)	DNF(s) learned by MtDL	Positions identified by Traditional method
H3N2 hema Antigenicity shift (490pos/421neg)	145H, 146Q	145H+146Q	18, 67, 122, 145, 146
H5N2 hema Pathogenicity (4pos/11neg)	275K, 275T, 323K, 324R, 325K	323K+324R+325K	275, 323, 324, 325
FIV tropism (3pos/7neg)	30E, 32K	30K∧32E	32
SIV Envelope Neutralizability (8pos/5neg)	179N, 337R	179N+337R	331, 348

Table 4.4: **Comparing DNF learning with position-specific association methods** Retrospective comparisons of the DNF learning algorithm on a variety of datasets.

positive especially when the dataset is biased, and performs poorly with very few sequences or when the key residues interact in a complex fashion.

4.6 DNF learned from HIV resistance datasets

HIV resistance to all available drugs is a persistent problem[12], and some drug resistance mutations are probably present before the start of therapy [13]. A good number of HIV protein sequences have been collected and labeled with the resistance level to 20 anti-viral drugs from the Stanford HIV database, including seven Protease Inhibitor drugs and eleven Reverse Transcriptase Inhibitor drugs. The numerical resistance values were converted to multiple class labels using resistance thresholds from the Stanford HIV database, and the sequences with significant resistant values and susceptible values are selected.

The shortest DNFs learned for each anti-viral drug are shown in Table 4.5. The learned DNFs are very short in terms of the number of clauses, and in most of the cases, two to three clauses are enough to explain the resistance. In spite of the short length, these DNFs show a very high prediction quality of sensitivity and specificity as shown in Table 4.5. For LPV drugs, the 2-term DNF discriminates the resistance and susceptible classes with sensitivity and specificity of 96.1% and 96.5%. The literals 36s, 45m for APV drugs, and 89I for RTV drugs have been identified

by experiments and reported in literatures. These DNFs may also suggest mechanisms for HIV drug resistance that have not been discovered yet.

Drug type	Drug	DNF = sensitivity/specificity; in the DNFs, lowercases mean negation
PI	NFV	$9l + (63l \wedge 9i \wedge 87n) = 0.902/0.834$
	RTV	$(81i \wedge 81v) + 83i + (70a \wedge 70l \wedge 89l \wedge 70t) = 0.981/0.988$
	LPV	$(9l \wedge 53i) + (9l \wedge 45m \wedge 9h \wedge 9m) = 0.961/0.965$
	APV	$(36s \wedge 45m \wedge 36y \wedge 9l) = 0.787/0.961$
	IDV	$(70a \wedge 9l \wedge 11p \wedge 70t) + (62l \wedge 45m \wedge 36y \wedge 9l) + (62P \wedge 53V \wedge 36s) + (83i \wedge 45I) = 0.965/0.982$
	SQV	$(9r \wedge 83i) + (62q \wedge 53i \wedge 89l \wedge 70l) + (45i \wedge 89l \wedge 70t \wedge 70a) + (76V \wedge 89l \wedge 81v \wedge 9l) = 0.899/0.963$
	ATV	$(70a \wedge 9l) + (89l \wedge 76V \wedge 81a) = 0.833/0.910$
NRTI	DDI	$150M + (68s \wedge 68t \wedge 68d \wedge 68n) + (74v \wedge 42n \wedge 74t) = 0.738/0.985$
	AZT	$(73v \wedge 34 - \wedge 214t \wedge 214d) = 0.848/0.988$
	D4T	$(209l \wedge 214d \wedge 34t \wedge 214t) + (68t \wedge 34m \wedge 214d \wedge 214t) + (68T \wedge 117I \wedge 66N) = 0.797/0.956$
	TDF	$(34i \wedge 66N \wedge 214t \wedge 183v) + (68g \wedge 19r \wedge 214Y \wedge 68t) + (34V \wedge 68t \wedge 214f \wedge 214t) = 0.784/0.980$
	ABC	$183V + (214d \wedge 121p \wedge 209l \wedge 82k) + (82k \wedge 66d \wedge 214Y \wedge 180c) = 0.940/0.944$
NNRTI	NVP	$(102k \wedge 102r) + (189g \wedge 102n) + (180y \wedge 100e) = 0.868/1.0$
	DLV	$(210t \wedge 102N) + (180y \wedge 100q \wedge 226F \wedge 210t) = 0.915/0.994$
	EFV	$(102r \wedge 102k) + (102s \wedge 189g) = 0.871/0.997$

Table 4.5: **DNFs learned from HIV drug resistance dataset**

4.7 Improved prediction performance of DNF learning algorithms on the HIV drug resistance problem

We examine the prediction performance of both standalone and MtDL learning algorithms on two well-known benchmark datasets: the HIV drug resistance dataset and the UCI promoter gene dataset (details in section 4.8). Many state-of-the-art machine learning models, including Support Vector Machine, Decision Trees, Neural Networks, Nave Bayes, etc., have been tested on these datasets, and the five-fold cross-validation prediction quality was reported [25]. The five-fold cross-validation prediction accuracies of Protease Inhibitor are shown in Table 4.6. The standalone DNF learning algorithm outperforms other machine learning algorithms in 4 out of the 7 PI datasets (Table 4.6). The result suggests that the exhaustive algorithms achieve better

Prediction accuracy (%)	NFV	SQV	IDV	RTV	APV	LPV	ATV
#pos/#neg sequences	194/211	119/321	115/279	154/244	47/308	103/142	42/111
Standalone DNF	93.5	**91.8**	**91.7**	96.1	**96.1**	88.2	**93.3**
Z-score	74.6	87.3	**91.7**	87.4	92.3	90.5	88.8
NaiveBayes	**95.1**	75.1	78.4	93.2	87.3	92.7	73.1
SVM (svm light)	77.2	74.2	83.4	92.2	87.5	86.3	72.6
DT	94.0	89.0	90.1	**98.6**	91.8	**98.6**	78.5
Winnow	91.1	84.7	89.9	94.6	91.1	94.6	85.9

Table 4.6: **Comparing Standalone DNF learning with published machine learning algorithms on HIV Protease Inhibitor datasets.** The numbers of positively labeled and negatively labeled sequences in the datasets are shown, as well as and the prediction accuracies of 1) Standalone DNF, 2) Z-score [6], 3) Naive Bayes (from Weka), 4) SVM (svm light software, default parameters), 5) Decision Tree (Weka, ID3 algorithm), 6) Winnow (Weka). The highest accuracy of each drug is highlighted in bold

prediction performance, and DNF turns out to be a reasonable bias on the hypothesis space as genotype-phenotype mapping functions for HIV drug resistance. Though Decision Tree (DT) can be converted to DNF, DT is less flexible due to the tree structure, and the roots and branch nodes are prefix in each clauses; However, the DT algorithm does not limit the tree height in the learning algorithm, while our DNF learning aims to learn the shortest DNF, DT outperforms DNF in two of the drug datasets.

4.8 Improved prediction performance on the UCI Promoter Gene dataset

The UCI's promoter gene dataset has been studied with many machine learning models and is used as a benchmark dataset. The goal is to predict whether a DNA sequence contains promoters. The dataset has 53 promoter sequences and 53 non-promoter DNA sequences. In Biology, the promoters are characterized by special motifs at certain positions from the transcription starting location, e.g. "cttgac" motif at $+37$ position indicates a promoter region. However, deriving all such domain theories is impractical and not meaningful. Machine learning algorithms showed promising prediction performance on this dataset (Table 4.7). Among them the knowledge-based

System	Errors	Comments
MtDL + CF	4/106	No domain knowledge required
KBANN	4/106	A hybrid ML system that uses domain knowledge to initialize the network structure
BP	8/106	Std backprop with one hidden layer
O'Neill	12/106	Ad hoc technique from the bio. lit.
Nearest neighbor	13/106	k-nearest neighbor, $k = 3$.
ID3	19/106	Quinlans decision-tree builder

Table 4.7: **Comparing MtDL+CF with published machine learning algorithms on Promoter Gene dataset**

artificial neural network (KBANN) [57] achieves the best accuracy of 4 out of 106 errors in a held-out test manner.

The KBANN model is a hybrid system of both Explanation-based learning (EBL)(a system that incorporates pre-existing knowledge) and the Empirical learning system (learning solely from training examples). In [57] the authors argue that the hybrid system should be superior, in terms of classification accuracy, to empirical learning systems. On the Promoter Gene dataset, KBANN learns a neural network model and translates a set of domain theories to initial the neural network structure. The error rate is the number of wrongly predicted examples in a leave-one-out cross-validation (LOOCV) manner. Three other machine learning algorithms, standard back propagation, Quinlan's ID3, O'Neill's ad hoc partial pattern matching, and the "nearest neighbor" are compared in Table 4.7.

We employed the same LOOCV validation using MtDL algorithm with CF() as the feature selector. Although the prediction performance of MtDL+CF is the same as the best one KBANN, MtDL+CF does not require any pre-existing domain knowledge as KBANN does.

4.9 The feature selector of MtDL

MtDL highly relies on the feature selector whose responsibility is to narrow the solution space to the greatest degree and without any false dismissals. The best feature selector would naturally be a hard version combinatorial algorithm but not a statistical algorithm because statistical

algorithms often yield false dismissals. CF() works seamlessly with the MtDL because DNF functions learned from MtDL algorithm are a proper subset of the hypothesis space of CF(), which can be seen as a general set of hypothesis positions with simple logic between them, e.g. OR or AND. To demonstrate this we chose two other popular feature selectors as a comparison with CF(), Logistic Regression (LR) and K-nearest-neighbor (KNN), and applied MtDL on top of them separately. We still chose the Promoter DNA dataset as the test standard, and LOOCV was applied when testing the prediction accuracy. CF() outperformed the other two as the feature selector for MtDL.

System	**Errors**	**Comments**
MtDL + CF	4/106	False positive: 1; False negative: 3
MtDL + LR	9/106	False positive: 6; False negative: 3
MtDL + KNN	25/106	False positive: 3; False negative: 22

Table 4.8: **Comparing feature selectors with published machine learning algorithms on Promoter Gene** dataset

Chapter 5

Applications to real unresolved problems

After demonstrating the convergence, correctness and efficiency of the algorithms on retrospective datasets, we proceed to test them on prospective data, where the target functions are not known. We have identified two important open genotype-phenotype mapping problems in RNA viruses, where current prediction methods had proven unsatisfactory in spite of the availability of significant amounts of labeled data. We demonstrate the competitiveness of our algorithms, identify specific strains (sequences) where our prediction differs most strongly from that of the currently used methods, and test them empirically in the laboratories of our thesis committee members from the University of Pittsburgh. At the end we also identify a critical clinical problem and prove the utility of our algorithms in other biomedical areas.

Truly prospective evaluation of Active Learning will require dozens of separate experimental sequences, each carried out over several months. Therefore, we will not be able evaluate the active learning algorithm prospectively.

5.1 Antigenicity evolution of Influenza viruses

5.1.1 Background and significance

Influenza viruses are responsible for about 500,000 deaths annually and are a substantial threat to human health. Besides seasonal infections caused by Influenza viruses, four major pandemics over the last 100 years have resulted in about 50 million deaths worldwide. Influenza A and B viruses evolve rapidly and continuously accumulate amino acid changes in the antibody binding (epitope) sites of the surface proteins, resulting in changes in antigenicity. As a result, new antigenic types regularly emerge and rise to predominance, causing worldwide epidemics despite existing vaccination programs.

Influenza viruses' high mutation rates induce antigenically variable pathogens that can escape from immunity induced by prior infection or vaccination. Antibodies against the viral surface glycoprotein hemagglutinin (HA) provide protective immunity to influenza virus infection, which is therefore the primary component of influenza vaccines. However, the antigenic structure of HA has changed significantly over time, a process known as Antigenic Drift, and in most years, the influenza vaccine has to be updated to ensure sufficient efficacy against newly emerging variants. Influenza viruses antigenic properties are characterized using hemagglutination inhibition (HI) assay, a binding assay based on the ability of influenza viruses to agglutinate red blood cells and the ability of animal antisera raised against the same or related strains to block this agglutination.

Retrospective quantitative analyses of the genetic data have revealed important insights into the evolution of influenza viruses. However, the antigenic data are largely unexplored quantitatively because of difficulties in interpretation, even though antigenicity is a primary criterion for vaccine strain selection and is thought to be the main driving force of influenza virus evolution. When antigenic data have been analyzed quantitatively, it has usually been with the methods of, or methods equivalent to, numerical taxonomy. These methods have provided insights; however,

they sometimes give inconsistent results, and do not properly interpret data that are below the sensitivity threshold of the assay.

The antigenic impact of amino acid substitutions in the antigenic evolution of influenza A viruses can reliably be determined by time- and cost-intensive experimental analysis. As an alternative, our key residue identification algorithm can be used to efficiently identify the amino acid changes that impact the Antigenicity Drift, thus providing guidance for further experimental analysis and insights into the underlying mechanisms.

5.1.2 HI types data

HI type data of NewYork State strains and New Zealand strains contain antigenic strain match and genotypes of Influenza Hemagglutinin (HA) proteins from 1989 to 2004. These strains are characterized using the HI assay to determine their antigenicity, and the phenotypes are presented as a predominance worldwide epidemical strain names (HI types), such as Beijing1989, Panama1999, etc. The key residues that cause the antigenic drift are identified by grouping the strains with the same HI types and comparing against each other. The HI types and the number of HA strains are shown in table 5.1.

Table 5.1: **HI type data**

HI type phenotypes (strain name)	**Number of strains**
Beij89	100
Shan93	49
Joha94	94
Wuha95like	11
Sydn97	30
Pana99like	9
Fuji02like	64
Kore02like	20
Well04like	29

5.1.3 Substitutions in antigenic type transitions

Amino acid changes from eight type transitions between adjacent years are identified using the CF() algorithm (Table 5.2). The average number of identified substitutions in each pair is 4.1, and 2.2 of them happen in epitope sites; the average number of substitutions each year (from year 1989 to 2004) is 2.0, and 1.0 in epitope sites. The number of substitutions of each pair is not strongly correlated to the genetic distances.

In the transition Beij89-Shan93, the key residue S63P(E) (mutation from S to P at position 63 in epitope site E) substitution adds an extra sugar ring in the amino acid residue located in epitope site and may modify the epitope site E's 3D structure to alter the antigenicity. This also applies to Q172H(D), Y121H(D), W238R(D) in transition Pana99like-Fuji02like. Results also show that substitutions may involve amino acid charge changes that modify the antibody binding property, including D140N(A) in Beij89-Shan93 transition, D140G(A) in Shan93-Joha94 transition, R213Q(D) in Joha-Wuha95like transition, K172Q(D) in Wuha95like-Sydn97 transition, N142D(A), Q172H(D), Y121H(D), W238R(D) in Pana99like-Fuji02like transition, and D142N(A) in Kore02like-Well04like transition. Some substitutions are hydrophobic to hydrophilic changes that cause the type transition, e.g., V160N(B) in Wuha95like-Sydn97 transition.

Among the eight type transitions, in six of them substitutions happen in one or two epitope sites, suggesting that small numbers of mutations are sufficient to change the structure of the binding sites. In transition Shan93-Joha94, three epitope sites are involved in the changes, though S294N(C) does not change the residue structure or property as strongly as the other substitution. In transition Sydn97-Pana99like, none of the substitutions are in epitope sites; this may suggest that substitutions that are not in the epitope site still affect the epitope binding globally. The substitution positions are not generally shared among different transitions.

Table 5.2: **Substitutions between adjacent groups**

Strain names	**Substitutions (epitope site)**	**Genetic distance***
Beij89-Shan93	S63P(E), D140N(A), N232D, S235Y	9.6
Shan93-Joha94	G291D, P63S(E), S294N(C), D232N, Y235S, D140G(A)	10.1
Joha94-Wuha95like	N278S(C), K151T, N161K, D149N, R213Q(D)	13.2
Wuha95like-Sydn97	K172Q(D), S263C, K108T, V160N(B)	19.4
Sydn97-Pana99like	C263S, T108K	13.4
Pana99like-Fuji02like	N142D(A), Q172H(D), E402G, Y121H(D), W238R(D), G241D	19.6
Fuji02like-Kore02like	H121Y(D), P243S	4.3
Kore02like-Well04like	K161N, D142N(A), S243P, D204Y	8.5

*Genetic distance is defined as the average amino acids changes between two groups of Hemagglutinin proteins.

5.1.4 Fixed and emerged substitutions

We define fixed substitutions in antigenic type transitions as the substitutions of type transitions that remain as the only amino acid after the substitutions happen; on the other hand, emerged substitutions temporarily emerge in only one type and disappear in other types.

The full list of fixed substitutions is compiled in figure 5.1.4. Fixed substitutions are common and occur in all of the eight type transitions. More than five substitutions are fixed in transition Jona94-Wuha95like, Wuha95like-Sydn97, and Pana99like-Fuji02like, while only one fixed substitution occur in the other transitions. On average, 2.8 fixed substitutions are observed in each transition and 1.4 per year. There can be multiple fixed substitutions happened at one position throughout different HI types, e.g., position 213 is amino acid "R" in Beij89, Shan93, and Joha94, and it is mutated to amino acid "Q" from Wuha95like to Well04like. For position 142 and 161, the substitutions regress to the previously fixed amino acids, i.e. "A" $\Rightarrow$ "B", and then mutate back to "A". These fixed substitutions may not play a role in the type transitions, and are most likely hitchhikers. Fifteen out of the twenty two fixed substitutions are located in the epitope sites. Fixed substitutions H91Q and W238R between Pana99like-Fuji02like are huge structure changes in the residues; D140N, K151T, R231Q, D149N, K78E, E174K, K172Q, H91Q are the charge changes in residues; S294N and N278S are perhaps hitchhikers due to the small differences between the two amino acids properties.

	140	**137**	**294**	**278**	151	**213**	161	149	**208**	**78**	**174**	**172**	**160**	66	99	**91**	**142**	**238**	241	402	372	204
Beij89	D	I	S	N	K	R	N	D	T	K	E	K	V	R	E	H	N	W	G	E	T	D
Shan93	N	I	S	N	K	R	N	D	T	K	E	K	V	R	E	H	N	W	G	E	T	D
Joha94	G	T	S/N	N	K	R	N	D	T	K	E	K	V	R	E	H	N	W	G	E	T	D
Wuha95like	G/S	T	N	S	T	Q	K	N	T	K	E	K	V	R	E	H	N	W	G	E	T	D
Sydn97	S	N	N	S	T	Q	K	N	I	E	K	Q	N/I/T	R	E	H	N	W	G	E	T	D
Pana99like	S	N	N	S	T	Q	K	N	I	E	K	Q	N	R/G	E	H	N	W	G	E	T	D
Fuji02like	S	N	N	S	T	Q	K	N	I	E	K	H	N	G	K	Q	D	R	D	G	T	D
Kore02like	S	N	N	S	T	Q	K	N	I	E	K	H	N	G	K	Q	D	R	D	G	T/I	D
Well04like	S	N	N	S	T	Q	N/S	N	I	E	K	H	N	G	K	Q	N	R	D	G	I	Y

Figure 5.1: **Fixed substitutions in the eight type transitions.** The rows are chronologically HI types, and columns are positions along the HA proteins sorted by the time when the substitutions happen. Bolded positions indicate epitope sites. The amino acids in light background are the original amino acids, and dark colors are fixed substitutions. Transition states are marked in light grey. Three positions, 140, 137 and 172 involve fixed substitutions twice and the final fixed substitutions are marked in black.

The full list of emerged substitutions is compiled in figure 5.1.4. Each antigenic type is compared against the rest of types to identify the unique mutations. The result shows that each group has at least one unique emerged substitution, e.g., 61G for type Wuha95like, 175F for type Pana99like. Eight out of night types have only one or two emerged substitutions, except for Shan93 which has four such substitutions. Some substitutions may be compensatory mutations to retain function, and others may be hitchhikers carried along by chance. Five of the unique substitutions occur in the epitope sites. On average, 1.5 emerged substitutions happen in each HI type, and 0.45 in epitope sites; 1 emerged substitutions happen each year, and 0.33 in epitope sites. The sites and substitutions identified by our method may be of particular relevance for influenza A (H3N2) virus antigenic evolution, which has not been described before.

	140	232	235	**63**	61	263	108	**175**	**121**	243	**142**	204	377
Beij89	D	N	S	S	S	S	K	Y	Y	S	N	D	T
Shan93	N	D	Y	P	S	S	K	Y	Y	S	N	D	T
Joha94	G	N	S	S	S	S	K	Y	Y	S	N	D	T
Wuha95like	G/S	N	S	S	G	S	K	Y	Y	S	N	D	T
Sydn97	S	N	S	S	S	C	T	Y	Y	S	N	D	T
Pana99like	S	N	S	S	S	S	K	F	Y	S	N	D	T
Fuji02like	S	N	S	S	S	S	K	Y	H	P	N	D	T
Kore02like	S	N	S	S	S	S	K	Y	Y	S	D	D	T
Well04like	S	N	S	S	S	S	K	F	Y	P	N	D/Y	T/I

Figure 5.2: **Emerged substitutions in the nine HI types.** The HI types are sorted chronologically in rows, and positions along the HA proteins are sorted in columns by when the substitutions happen. Bolded positions are epitope sites. Each HI type contains at least one emerged substitutions marked in dark grey.

The fixed and emerged substitutions are identified using the "soft version" CF() algorithm

(table 3.1). We used the threshold of 95%, allowing 5% errors and outliers in the data. In the fixed substitutions learning, all the HI types are sorted chronologically, and split into two groups by a year threshold, i.e. the types before year 1990 as class '+' and types after 1990 as class '-'. This is repeated using different year thresholds until all the combinations are studied. On the other hand, in the emerged substitutions learning, each time one type is selected as class '+' and all the rest types are merged as class '-'.

Another interesting study is to learn the unique substitutions in one type against the types right before and after this type, e.g. type 'Shan93' against 'Beij89' plus 'Joha94'. This study shows the emerged substitutions in a shorter time range. The results are shown in table 5.3.

Table 5.3: **Unique substitutions in a shorter time range**

Strain name	**Emerged substitutions**
Beij89	140D, 232N, 235S
Shan93	140N, 232D, 235Y, 63P
Joha94	140G
Wuha95like	61G
Sydn97	66G, 21V, 160N, 263C, 107T
Pana99like	65S/D, 175N, 183A
Fuji02like	243P
Kore02like	142D
Well04like	377I, 243P, 204Y, 175F

5.2 The Neuraminidase Inhibitor (NAI) resistance

Influenza Neuraminidase protein (NA) is an integral type II membrane glycoprotein. It has 1453 nucleotides encoding 454 amino acids. Neuraminidase promotes the release of influenza from infected cells and accelerates virus spread by cleaving stalic acid [34]. Because of the essential role of NA in influenza replication and its highly conserved active site, NA inhibitors (NAIs) become promising drugs [34]. Zanamivir and Oseltamivir are NAIs that have shown efficacy against influenza A and B viruses [58].

After oseltamivir treatment NAI resistant viruses were infrequent in clinical trials with estimated resistance rates varying from 0.4% to 1% in the adult population [59]. This is much

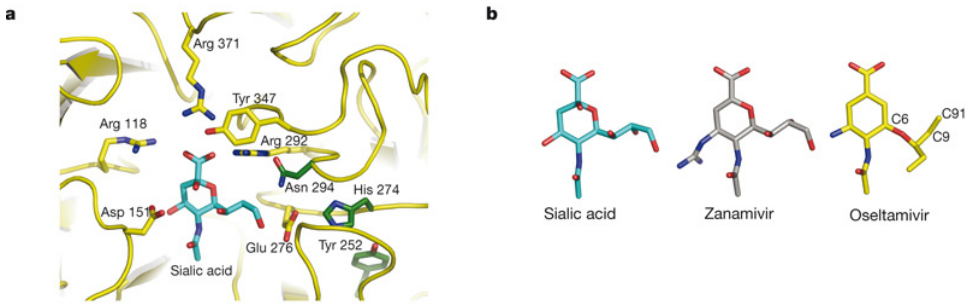

Figure 5.3: **Structure of N1 neuraminidase complexes and NAIs [3]**. Structure of N1 neuraminidase complexes and NAIs [3]. **a**, sialic acid (colored blue) docked into the active site of wild-type N1 Neuraminidase (ribbons colored yellow) from superposition of the sialic acid complex of N2. **b**, the structures of sialic acid (carbons colored blue), zanamivir (carbons colored grey) and oseltamivir (carbons colored yellow) are shown in similar orientations with selected carbon atoms numbered.

less than the resistance to M2 ion-channel inhibitors which is 92.3% amount Human Influenza A (H3N2) viruses in United States [60], and neuraminidase inhibitor resistance tends to revert when the drug pressure is removed [61]. However, it is reported that the resistance to oseltamivir is getting more prevalent, especially in Europe (25%) [14]. While oseltamivir resistance had become more prevalent for seasonal H1N1, the 2009 H1N1 pandemic strain appeared to be Oseltamivir sensitive [62, 63].

To learn the drug resistance associated mutations, influenza strains are isolated and tested with phenotypic/genotypic assays. The resistance to NAIs is mostly determined by the mutations on neuraminidase proteins; however some mutations on hemagglutinin are also correlated [64]. Once drug resistant strains are identified, the neuraminidase proteins are sequenced, and the associated mutations are identified. Mutagenesis experiments are carried out on neuraminidase to identify such resistant mutations to understand the mechanism [3][61][65].

A number of both wild-type and genetically engineered neuraminidase variants have been reported in the literature, often including their sequences and NAI-resistance phenotype. And yet, no accurate prediction algorithms have yet been devised to date. I have compiled a dataset of about 300 such influenza neuraminidase proteins, including 48 engineered sequences and about 250 clinical or in vitro isolates. The hemagglutinin sequences of these variants are not always

available in their entirety, a fact that will complicate our analysis.

5.2.1 Compiled Influenza sequences

The NAI resistance data are collected from the literature reporting NAI experiment. Isolated or mutagenesis Influenza virus resistance to NAIs, Zanamivir and Oseltamivir, were represented as IC50. The following sequences were compiled:

1. Phenotype: the sequence phenotype need to be explicitly reported as 'resistant' or 'susceptible' to at least one of the NAI drugs. The sequences with IC50 or folds of resistance reduction reported, but no cutoff provided were discarded.
2. Genotype: the neuraminidase protein sequences are provided directly or the mutagenesis experiments as well as the wildtype sequences are both provided, or the sequences can be retrieved given the strain name.

We focused on the neuraminidase genotype of Influenza A and neuraminidase subtype N1 and N2. The subtype of Influenza A viruses were represented as HxN1 (N1 neuraminidase subtype, any hemagglutinin type) and HxN2 (N2 neuraminidase subtype, any hemagglutinin type. The number of sequences is shown in parentheses (table 5.4).

Table 5.4: **Drug resistance data size:**

Zanamivir drug:
HxN1 resistant (3) vs. HxN1 susceptible (66)
HxN2 resistant (18) vs. HxN2 susceptible (52)
Oseltamivir drug:
HxN1 resistant (15) vs. HxN1 susceptible (47)
HxN2 resistant (2) vs. HxN2 susceptible (62)

5.2.2 The ground truth of resistant mutations identified by domain expert

We did a thorough literature search and compiled the drug resistant mutations that have been identified and proven to be key residues. These serve as ground truth when comparing the pre-

diction performance of CF() and 'position-specific' algorithms.

HxN1 resistant to Zanamivir

H131N and G254R [66] are identified in two influenza H1N1 NA proteins isolated in patients. The resistance level was determined as 'low resistance' (the resistance IC50 is smaller than mean of IC50 value plus three times of standard deviation). The sequences were retrieved from WHO and tested using a commercially available NA-Star kit. V121A was identified in a chicken H5N1 NA protein. The position was highly conserved.

HxN2 resistant to Zanamivir

E119A, E119D, E119G [66]are three Zanamivir-selected mutant from "A/turkey/ Minnesota/833/80" (H4N2).

R152K **??** is mutagenesis from A/Tokyo/67(H3N2) sequence. The WT sequence as downloaded and made zanamivir sensitive sequence. The resistant sequence is manually generated from the WT sequence.

Fourteen sequences [66] are also collected from three consecutive seasons (2004-2007). Ten of them are determined to have 'low resistance' to Zanamivir. Within these 14 sequences, 143V, 387N and 439L always occur together.

HxN1 resistant to Oseltamivir

Six strains with H274Y [66] mutation are also collected from three consecutive seasons (2004-2007). Another six strains with H274Y [67] are isolated clinically. V116A is identified in A/chicken/Vietnam/486A/2004(H5N1) NA protein. The position is highly conserved.

I117V and I314V are co-occurring in A/Chicken/Indonesia/Wates/77/2005 NA protein. 46S and 206V are co-occurring in A/Hanoi;30408/2005 [68], isolated from a 14-year-old Vietnamese girl. The mutation reported in the literature was H274Y.

HxN2 resistant to Oseltamivir

One strain with R293K [66] are Zanamivir-selected mutant from A/turkey/ Minnesota/833/80 (H4N2). Another R293K [69] is clinically isolated influenza virus A/Sydney/5/97(H3N2).

5.2.3 Comparison between the ground truth, CF(), DNF learning, and the traditional method (Z-score)

We retrospectively validate the performance of CF(), DNF learning, and one traditions method "z-score" algorithm with the ground truth. The DNF learning algorithm learned almost exactly the same function as the ground truth, whereas CF() has slightly higher false positive and false negative rates in Zanamivir HxN1 and Oseltamivir HxN2 datasets. The traditional method failed in learning the correct answer in all of the four datasets (Table 5.5).

Table 5.5: **NAI retrospective validation**

Data	**Ground truth**	**Key residues by CF()**	**DNF learning**	**Traditional method (Z-score)**
Zanamivir - HxN1	H131N, G254R, V121A	121, 254, 354	121A + 131N + 254R = +	254, 441, 443, 445, 465, 466, 477, 478, 479, 480, 481
Zanamivir - HxN2	E119A, E119D, E119G, R152K	119, 152	119A + 119D + 119G + 143V + 152K = +	18, 23, 30, 42, 93, 143, 151, 216, 221, 309, 387, 439
Oseltamivir - HxN1	H274Y, V116A, I117V, I314V, 46S, 206V	274, 116, 117, 314	46S + 116A + 117V + 275Y = +	17, 23, 52, 64, 105, 275, 3666, 386, 418
Oseltamivir - HxN2	R293K	293	293K = +	43, 127, 172, 199, 291, 293, 333, 402

5.3 Using Data-driven Rules to Predict Mortality in Severe Community Acquired Pneumonia

The CF() and DNF learning algorithms can be naturally extended to other domains where similar assumptions apply. Prediction of outcome in the intensive care is in great need of computational tools to optimize hospital performance benchmarking, yet their use by clinicians is limited by the complexity of available tools and amount of data required. A large set of instance-based patient data are collected from patients admitted to an acute care hospital with community acquired pneumonia. Early detection of patients at high risk of developing organ dysfunction and death from the data has proved challenging, yet it is critical to allocate the resources and guide decisions regarding active treatment and withdrawal of care. In the early prediction setting, the features are limited to those available immediately when the patients are hospitalized, including demography, cytokine data, genetic information, and early bedside measurements. DNF learning algorithms can fast learn an intuitive set of prediction rules using the limited size of features, and thus is of greater practical usefulness than currently available prediction tools in this population.

5.3.1 Introduction and background

Among inflammatory illnesses, pneumonia often presents as sepsis, defined as infection accompanied by systemic signs and symptoms of infection [70], including rapid heart rate, rapid respiratory rate, and fever. Approximately 750,000 patients develop severe sepsis each year in the US, with a hospital mortality rate of 28.6%, or 215,000 deaths per year [71]. A significant number of these patients have pneumonia [72]. Interventions for severe sepsis that decrease morbidity and mortality could profoundly impact public health [73]. There is ample pre-clinical and clinical evidence that immunomodulation improves the outcome of patients at higher risks of death, yet pre-clinical data and simulation have also indicated that harm may ensue from targeting some subpopulations of patients [74, 75, 76]. Early detection of patients at high risk of developing

organ dysfunction and death has proved challenging.

Tools to predict the outcomes of critical illness have been developed for three decades [77, 78, 79, 80, 81]. Most of these prediction tools are logistic regression models, presumably because of their popularity and ease of interpretation of odds ratios associated with predictors of outcome. Yet, logistic regression is intolerant of missing data, does not readily deal with correlated data, and it may be difficult to quickly generate a prediction for the non-expert. A desirable prediction tool should possess the following properties: discrimination (the ability to classify the outcome of patients who will develop hospital mortality and who will not), learnability (the ability to achieve the discrimination from moderate quantity of data and few features, especially in the early detection of critical care where fewer data are available), completeness (explore the solution space as completely as possible under appropriate assumptions), transparency (not behave as a "black box"), and having the ability to be easily interpretable by the end-user, typically a non-expert.

We propose to use short Disjunctive Normal Form (DNF; "OR" of "AND") as an appropriate representation of the hypothesis space to predict critical care outcomes because 1) DNF is a high order boolean function that examines potentially complicated relationships between predictors and outcomes, 2) DNF offer great flexibility and allows identification of unforeseen interactions between predictors, 3) DNF is a natural form of knowledge representation for humans to interpret and they provide clinical insights and clear rules to assist in decision making, 4) DNF is scalable to large or small datasets. A short DNF increases interpretability of the rules and mitigates overfitting bias. The aim of this study was to illustrate the ability of DNF to predict hospital and 90-day mortality within 2 days of admission in patients with community acquired pneumonia.

Related work

Previous models have been limited by retrospective design, [82, 83, 84, 85] the dependence on large hospitalization data [82, 83, 84, 85, 86, 87, 88], the lack of interpretability of complex

models [85], restricted applicability to single study sites [84, 87, 89], and bias to certain patient populations [84, 85, 87]. Time dependent techniques as alternatives to standard Cox proportional hazard models [90] and dynamic microsimulation [91] have also been published [92] [93]. Both microsimulation and Markov transition kernels derived in these publications are learned from population-level inference and are not instance-based (i.e. patient-specific). We also have the intuition that, outside the framework of a clinical study, clinical data are collected on the basis of perceived clinical need and thus missingness is highly likely not random. Accordingly, there is a very good case to be made that models based on instances might perform better than population models.

5.3.2 Materials and Methods

The GenIMS study cohort

Patients with community acquired pneumonia (CAP), a common cause of sepsis, were recruited as part of the Genetic and Inflammatory Markers of Sepsis (GenIMS) study, a large, multicenter study of subjects presenting to the EDs of 28 teaching and non-teaching hospitals in 4 regions in the United States (Western Pennsylvania, Connecticut, Michigan, and Tennessee) between November 2001 and November 2003. Eligible subjects were > 18 years and had a clinical and radiologic diagnosis of pneumonia, as per the criteria of Fine, et al. [90]. Further details on inclusion and exclusion criteria are provided elsewhere [91]. The GenIMS study was approved by the Institutional Review Boards of the University of Pittsburgh and all participating sites. The current study used fully de-identified data and was approved by the University of Pittsburgh IRB.

Of the 2320 patients enrolled, we restricted our analysis to 1815 subject admitted to the hospital and with measurements of serum inflammatory markers data on enrollment day. Our primary outcomes were all-cause mortality at hospital discharge and at 90 days after enrollment.

Measurements

The dataset included demographic information, diagnostic information as to bacterial etiology and anatomical site of sepsis, admission APACHE III as an indicator of overall disease severity [92], organ level physiologic variables to quantify organ dysfunction, routine laboratory markers, and interventions. Relevant to our analysis, the inflammatory markers IL-6, IL-10, tumor necrosis factor (TNF), and lipopolysaccharide binding protein (LBP) were collected on days 1, 2, 3, 4, 5, 6, 7, 8, 15, 22 and 30 while patients were still in the intensive care unit. An extended set of coagulation studies was collected on day 1, as well as an array of fluorescent antibody cell sorting (FACS) markers to quantify different immune cell populations on day 1. Finally, DNA information on 27 single nucleotide polymorphism (SNP), each segregating the study population in non-overlapping binary or ternary genotypic categories, was also collected. There were chosen because they were previously shown or suspected to have prognostic value in sepsis [93, 94, 95, 96].

Model hierarchy and benchmark classifiers

We construct a hierarchy of models 1 to 8 incrementally including features pertaining to different domains of data (Table 5.6). Model 8 is the most complete model containing all available features; Model 7 is a complete set of features, but restricted to data available only on day 1 of hospital, while Models 1 to 6 include selective domains of features. No data beyond day 2 post-enrollment were included in the predictions.

To compare the performance of the DNF learning algorithm, a number of other classifiers were constructed. These include simple Logistic Regression, Naive Bayes, SVM, Multi-layer Perceptron (Neural Network), and tree-based algorithms, (e.g. Random Tree, and Random Forest). Prior to classification, all continuous data were discretized in terciles (age), or quartiles (all analytes and APACHE score). For each model, two feature selection algorithms (information gain ranking and chi-square ranking) were run to select a maximum of 15 predictor variables

(features). Feature selection was applied using 10-fold cross-validation to mitigate overfitting. Benchmark classifiers used the union of feature sets identified by the selection algorithms.

Table 5.6: **Predictors (features) inluded in the different models.**

Model	**Features included**
Model 1	Demographics (age, sex, race, chronic disease), Macrophysiology (APACHI II score, Severe sepsis on enrollment)
Model 2	Demographics, physiology, Day 1 cytokines
Model 3	Demographics, physiology, SNP profile
Model 4	Demographics, physiology, day 1 cytokines, SNP profile
Model 5	Demographics, physiology, day 1 cytokines, SNP profile, coagulation data
Model 6	Demographics, physiology, FACS
Model 7	Demographics, physiology, day 1 cytokines, SNP profile, coagulation data, FACS
Model 8	Demographics, physiology, all available cytokines, SNP profile, coagulation data, FACS

Early prediction capabilities of the classifiers

For each model, we attempted classification using the following classifiers: Nave Bayes (NB), Neural Network (NN), logistic regression (LOG), boosted logistic regression (BL), support vector machines (libSVM and SMO implementations in Weka 3.5.7), Random Forest and Random trees algorithms, and finally the KStar lazy classifier. Their performance on each model was compared, using a 10-fold cross-validation strategy.

Prior to classification, all continuous data was discretized in terciles (age), or quartiles (all analytes and APACHE score). Two feature selection algorithms (information gain ranking and chi-square ranking) were run to select a maximum of 15 predictor variables (features). These algorithms were also run using a 10-fold cross-validation procedure.

Performance metrics

We evaluate the models ability to discriminate outcome by received operating characteristics (ROC) area under the curve. Sensitivity and specificity are also provided. We computed the Brier score as a global measure of calibration. For DNF, we also adapted the Hosmer-Lemeshow H-statistic (AHL) to binary outcomes [97]. Because DNF learning outcomes are either 0s or 1s, we created five bins including a geometrically larger number of predicted deaths. We randomly choose predicted survivors to complete the bins which comprised an approximately equal number of patients. The AHL was then computed as a chi-squared statistic across the five bins [98]. For the probability-based models, e.g., Logistic Regression and SVM, we use their binary outcomes instead of the continuous probability to compute the AHL statistics scores. All metrics are reported in the entire population and in the external validation cohort.

Results

Patient characteristics

All 1815 patients had demographic, disease severity and at least two inflammatory markers measured on day 1. The number of patients where different domains of data were available varied and was least for FACS (Figure 5.4). This distribution strongly determined the hierarchy of models examined. A complete description of cohort demographics and physiology has been published [91].

Predictors identified by benchmark classifiers

Clinical markers of severity (APACHE score and number of failing organ systems) were the strongest predictors of both hospital and 90-day mortality. Of demographic features, only age and the presence of chronic illness were included in most predictive models. Most SNPs examined were uncorrelated to 90-day mortality, but IL6M174 (GG), L100M1048 (G/T) and MIFM173

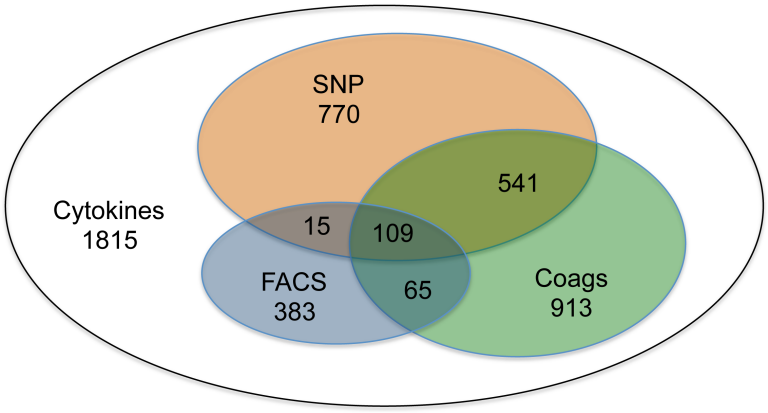

Figure 5.4: **Cohort sizes across different domains of data [4].**

(GG) were consistently predictive, even in multivariate models. IL18M137 was less consistently associated with outcome. Features also consistently selected in the hierarchy of models included monocyte positivity for CD-14 and CD-120a, and monocytic and granulocytic positivity for toll-like receptor (TLR)-2. Although it could be that the 10-fold cross-validation procedure admitted significant overfitting (N=124), it is an interesting hypothesis that the profile of activation of immune cells conveys as much or more information than cytokines and SNP polymorphisms.

5.3.3 DNF learning algorithm consistency and efficiency

The DNF learning algorithms are first validated on simulated clinical data to assess the consistency and learning efficiency in a practical way. When generating the simulated data a new hypothetical DNF is proposed in each study, e.g. $(age > 3 \wedge Sepsis > 1 + Nil6\_2 > 1 = mortality$, see section 5.3.4 for the explanation of DNF functions) and the data is randomly produced following the features distributions in the real patient data and labeled according to the target function. When learning the DNFs the positively labeled and negatively labeled data are shuffled and used in an incremental way to monitor the solution space size against the number of data used. The step is repeated 20 times and the DNF learning algorithm converge to a single DNF solution with 50 positively labeled data and 50 negatively labeled data, and more importantly the learned single DNF is consistent with the target function in all of the studies.

5.3.4 DNF learning algorithm prediction performance

The DNF learning prediction quality is first evaluated by its discrimination. The ROC curve (Figure 5.5) is generated upon tuning the sensitivity/specificity weights in the optimization objective function. The AUC for hospital mortality dataset in Model 8 is 0.937, which is very similar to the performance obtained with Model 7, suggesting that serum inflammatory markers levels after day 1 do not contribute much to the predictive ability. This is a meaningful result as hospital mortality is by and large determined by data obtained on the first admission day.

90-day mortality is considerably more difficult to predict than hospital mortality with the AUC decreasing to 0.785. We again compare the performance on Model 7, Model 8, and also add day 2 serum inflammatory marker levels to Model 7, without significant improvement in predictive ability (Figure 5.5). The DNF learning algorithm outperforms other benchmark classifiers built from Model 7 and Model 8 (Table 5.6), even if Model 8 contains a much more complete set of features; however Naive Bayes and Logistic Regression model prediction performance are lower than that of Model 7 because these two models lack regularization; Random tree and Random Forests' implementations we used do not implement pruning and result in severe overfitting issues; on the other hand, Boosted Logistic and DNF naturally implements regularizations and perform as well as Model 7 (Table 5.6).

When removing features from Model 7 (Models 1 to 6), the DNF learning accuracy decreases (Table 5.6). DNF learning also outperforms other classifiers on Model 6, suggesting that models which include FACS data perform well despite the modest size of the cohort. For less rich Models 1 to 5, the performances of DNF and benchmark classifiers were comparable, suggesting that richness of the set of features contributes more to the predictive ability of DNF compared to other classifiers. This conjecture could be examined in computational experiments. Interestingly, Logistic Regression-based classifiers performed consistently better than other benchmark classifiers through Model 5 (Table 5.6).

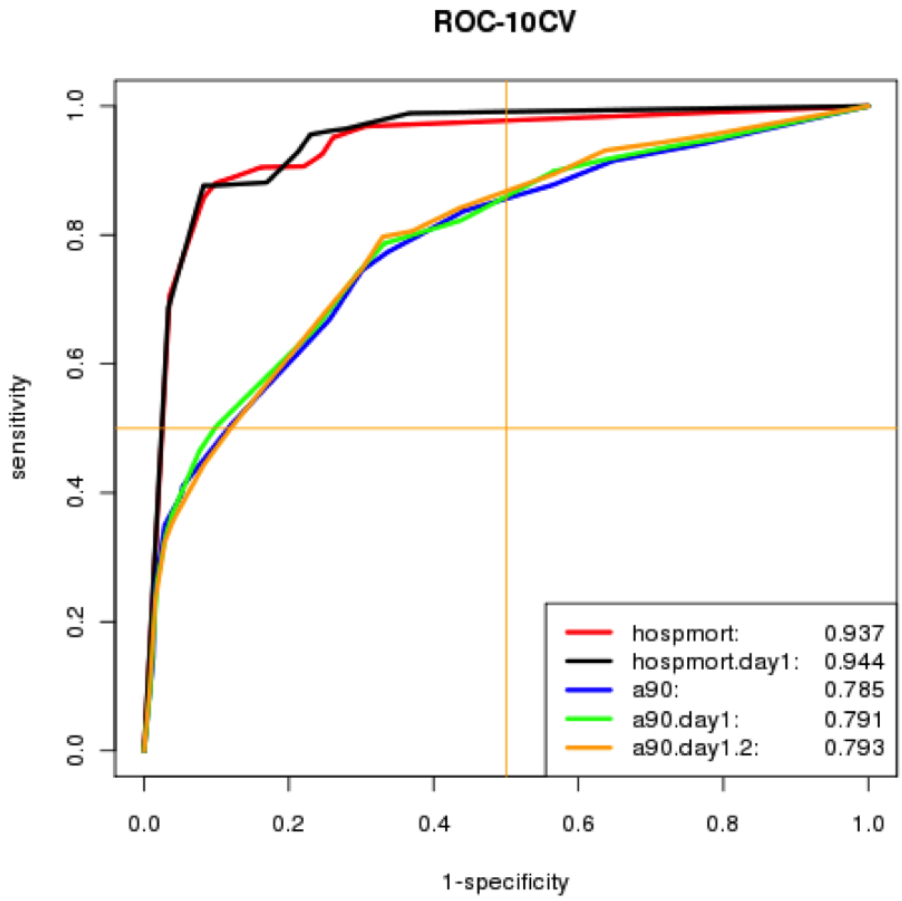

Figure 5.5: **Prediction performance of DNF learning on hospital mortality and 90-day mortality data [4].** The 10-fold cross validation is applied to access the prediction performance of DNF learning on the two datasets, and compare the performance when using the whole feature set and only day 1 and/or day 2 cytokine.

DNF learning algorithm external validation

To evaluate the external validity of predictions from DNF learning, we developed models using patients from a random subset of 27 hospitals, comprising approximately two-thirds of the patients. The prediction performance of DNF rules are then tested on patients from the remaining six hospitals, where the numbers of patients per hospital varied between 1 to 343.

Using 90-mortality as the outcome of interest the DNF learning ROC achieves 0.789 which is similar to that we learned in cross-validation over the entire cohort when using all the features.

Table 5.7: **Comparative performance of models on predicting 90-day mortality.**

Model	NB	SVM	NN	LOG	BL	RT	RF	DNF
Model 1	.740	.623	.746	.748	**.755**	.705	.743	.752
Model 2	.733	.625	.690	.747	**.752**	.673	.681	.740
Model 3	.709	.578	.742	**.762**	.763	.670	.696	.755
Model 4	.745	.630	.733	**.762**	.755	.662	.711	.749
Model 5	.770	.678	.728	**.774**	.766	.650	.654	.756
Model 6	.739	.682	.696	.728	.739	.690	.699	**.759**
Model 7	.783	.747	.751	.785	.766	.701	.715	**.791**
Model 8	.704	.744	.756	.723	.768	.575	.628	**.785**

NB-Naive Bayes, SVM-Support vector machine, NN-neural network, LOG-Logistic regression, BL-Boosted logistic regression, RT-Random tree, RF-Random forest, DNF-Disjunctive Normal Form learning.

The external validation performance of DNF learning compared advantageously with that of benchmark models (Table 5.8). Of note, DNF learning was the best calibrated model (AHL=9.06, p=0.06 with 4 df).

Table 5.8: **Comparative performance of models on predicting 90-day mortality.**

Scores	NB	SVM	NN	LOG	BL	RT	RF	DNF
ROC	.747	.752	.757	.738	.748	.655	.698	**.789**
1 - Brier Score	.712	.750	.844	.822	.867	.792	.874	**.891**

NB-Naive Bayes, SVM-Support vector machine, NN-neural network, LOG-Logistic regression, BL-Boosted logistic regression, RT-Random tree, RF-Random forest, DNF-Disjunctive Normal Form learning.

Specific rules learned from the data

The DNF learning algorithm simultaneously optimizes the prediction quality and minimizes the length of DNF functions, because without constraining the function length, the DNF functions can be complicated and lead to severe over-fitting problems. The DNF learning algorithms aim to learn the shortest functions (see section 4.1 for the definition of the function length), i.e. the most generic functions extracted from the data that can discriminate the mortality outcomes. The DNF learned to predict hospital mortality is:

$$(Ssday1 > 1)\ \textbf{\textit{OR}}\ ((Ssday1 > 0)\ \textit{AND}\ (Npct > 1)\ \textit{AND}\ (NIL6\_2 > 1)) = + \tag{5.1}$$

Where $Feature = t$ means the value of the feature falls into group *t*; $Feature > t$ means the feature value is larger than that of group t. Recall that the feature values are discretized into 3 to 5 groups, and the group values are indexed from 0 to $N - 1$ where N is the number of groups. The full explanation of literals appeared in this study is shown in Table 5.9.

Function (1) indicates that if either one of two conditions is satisfied, the outcome is predicted to be hospital death, where the two conditions are 1) $Ssday1$ value is larger than 1 (failure in more than one organ system), or 2) $Ssday1$ value is larger than 0 AND $Npct$ value is larger than 1 AND $NIL6\_2$ value (quartile of IL-6 levels on the second day) is larger than 1. The positive symbol on the right side of function (1) is positive label, i.e., hospital mortality. Since all the DNF predict positive class, the '+' symbol on the right side is replaced with the sensitivity/specificity metrics of the DNF. For representation purposes a DNF will be written as DNF = sensitivity/specificity, and the above function is now:

$$(Ssday1 > 1)\ \textbf{\textit{OR}}\ ((Ssday1 > 0)\ \textit{AND}\ (Npct > 1)\ \textit{AND}\ (NIL6\_2 > 1)) = 93.6\%/82.3\% \tag{5.2}$$

This DNF contains 2 terms of 4 literals covering 3 different features: $Ssday1$, $Npct$, and $NIL6\_2$, comprising only 3% of all features available in the data, suggesting that DNF functions discriminate the outcomes by only using a small fraction of the feature sets ($< 10\%$ features in all cases).

The prediction procedure implied by a DNF (3) is illustrated in Figure 5.6. The prediction procedure of DNF is represented in three layers: the top layer is the DNF itself; the middle layer is the clause level; and the bottom layer is the final outcome. Red color rectangles indicate

that patient data is above the threshold and a severity condition is met; green rectangles indicate that patient data is below and the condition is not met. Three example patients are shown. For patient A, $Ssday1$, $Npct$ and $NIL6\_2$ are all above the threshold and results in a positive Clause 2 so the predicted outcome is mortality. For patient B, Clause 2 is negative due to the low $Npct$ (procalcitonin in the lowest quartile); however high $Ssday1$ turns on Clause 1 and predicts mortality too. Patient C has high $Npct$ but it is not sufficient to turn on either Clause 1 or 2 and she is therefore predicted to survive.

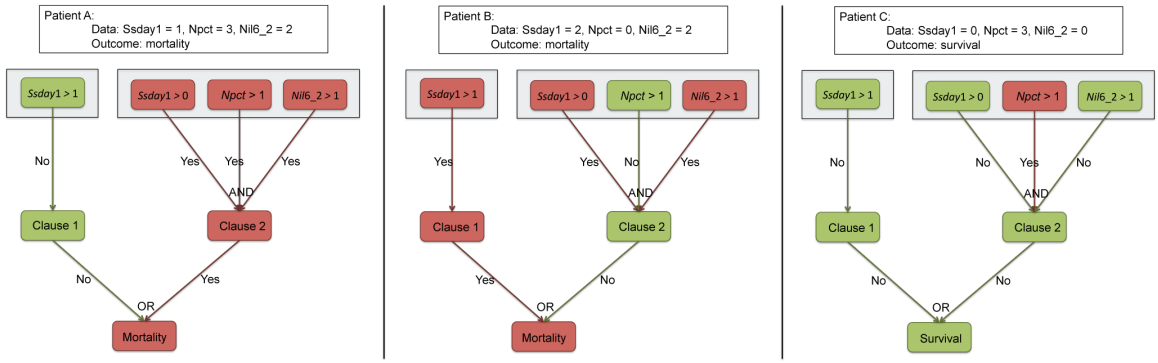

Figure 5.6: **Interpreting DNF models on three patients [4].** The prediction procedure of DNF is represented in three layers: the top layer is the DNF itself; the middle layer is the clause level; and the bottom layer is the final outcome. Red color rectangles indicate that patient data is above the threshold and a severity condition is met; green rectangles indicate that patient data is below and the condition is not met. Three example patients are shown. For patient A, $Ssday1$, $Npct$ and $NIL6\_2$ are all above the threshold and results in a positive Clause 2 so the predicted outcome is mortality. For patient B, Clause 2 is negative due to the low $Npct$ (procalcitonin in the lowest quartile); however high $Ssday1$ turns on Clause 1 and predicts mortality too. Patient C has high $Npct$ but it is not sufficient to turn on either Clause 1 or 2 and she is therefore predicted to survive.

The DNF learned from the data are shown in Table 5.10. For hospital mortality, $Ssday1$ is a strong predictor. A high level of $Ssday1$ is associated with high risk of mortality. $IL6$ and $IL10$ are strong predictors too, and appear to be consistently predictive, which can possibly support the concept that total inflammation, as opposed to a balance between pro-inflammation and anti-inflammation, is predictive of outcome [90]. $IL6$ on day 2 turns out to be a strong predictor, yet needs two other conditions to also be present (Equation (1) in Table 5.10). In Model 7, $IL10\_1$

is selected instead, and it needs 3 other conditions too: $age > 1$, $Ssday1 > 0$ and $IL1R901827$ SNP is not A/G (Equation (2) in Table 5.10).

To predict 90-day mortality, the number of terms in DNF increases to 5, and the sensitivity decreases to 80%, suggesting that $Ssday1$ is not as strong a predictor of 90-day mortality as it is of hospital mortality. In Model 8, $Ssday1$ combines with $Npct$ factor to form a single clause, and in Model 7 it needs $Nap3$. Higher $Nap3$ is also an indication of high death risk. Interestingly although SNP generally has low correlation with the 90-day mortality, $IL1R895495$ and $IL1R901827$ are learned in the DNF.

The highest discriminator of poor outcome was the day 1 to day 2 trend in the product of IL-10 and IL-6. Trends in day 1 to day 2 TNF, IL-10, IL-6, were also retained in the models. This is a very interesting, and somewhat refreshing observation, raising the hypothesis that interventions significantly impacting early cytokine profiles might indicate biological activity resulting in more favorable long-term outcome..

In the external validation, the DNF learned from the development set is:

$$
\begin{aligned}
&(Npct > 1 \textbf{ AND } Ssday1 > 0 \textbf{ AND } Nap3 > 1) \textbf{ OR} \\
&(Nap3 = 4 \textbf{ AND } IL1R895495\_AG! = G/A \textbf{ AND } Npct > 1 \textbf{ AND } Nfcd120a < 4) \textbf{ OR} \\
&(chronic\_t > 0 \textbf{ AND } NIL6\_3 > 1 \textbf{ AND } IL1R895495\_AG! = G/A \textbf{ AND } Ssday1 > 0) = 81.8\%/71.4\%
\end{aligned}
\tag{5.3}
$$

The first two clauses are similar to those learned in Table 5, which indicated that 1) the process of DNF learning is robust in identifying predictive rules if data used in development is consistent with population data, and that 2) correlations in data may allow similar, but not identical rules, when different development sets are selected.

Table 5.9: **DNF literals explanation**

literal	**meaning**	**value type**	**num of value groups**
$Ssday1$	Presence of some organ dysfunction on day 1 [49]	integer	5*
$Npct$	Quartile of procalcitonin [50]	integer	5
$NIL6\_2$	Quartile of the inflammatory marker IL-6 on the second day of admission	integer	5
$IL1R901827_A15$	Genetic polymorphism of IL-1 receptor antagonist protein	Gene	3
$Nage$	Quartile of age	integer	5
$NIL10\_1$	Quartile of the inflammatory marker IL-10 on the day of admission	integer	5
$IL1R895495\_AG$	Genetic polymorphism of IL-1 receptor antagonist protein	Gene	5
$Nap3$	Quartile of APACHE III score	integer	5
$chronic_t$	Burden of chronic illness, as determined by the Charlson index [51]	integer	5
$Nfactor$	Quartile of coagulation Factor IX activity	integer	5

Note*: when missing values present in the data, they are treated as a literal, but they are never selected in the DNF learning.

Table 5.10: **DNF of the patient mortality**

Hospmort mortality (model 8)	$(Ssday1 > 1)$ ***OR*** $((Ssday1 > 0)$***AND***$(Npct > 1)$***AND***$(NIL6\_2 > 1)) = 93.6/82.3$
Hospmort mortality (model 7)	$((Ssday1 > 1)$***OR***$((Ssday1 > 0)$***AND***$(IL1R901827\_A15! = A/G)$***AND***$(Nage > 1)$***AND***$(NIL10\_1) > 1)) = 91.0/89.3$
90-day mortality (model 8)	$((Ssday1 > 0)$***AND***$(Npct > 1))$***OR*** $((IL1R895495\_AG ! = A/G)$***AND***$(Npct > 1)$***AND***$(Nap3 = 4)$***AND***$(chronic\_t > 0)) =70.1/76.4$
90-day mortality (model 7)	$((Ssday1 > 0)$***AND***$(Nap3 > 0)$***AND***$(Npct > 0))$***OR***$(IL1R901827\_A15 ! = A)$***AND***$(Nap3 = 4)$***AND***$(Npct > 0)$***AND***$(Nfactor\_0 > 0)) =69.6/76.6$

5.3.5 Discussion of the learned DNFs

We present a new class of models, DNF learning, which produce data-driven rules predicting mortality in patients hospitalized with severe community acquired pneumonia (see Appendix for details). A distinctive feature of DNF, compared to commonly presented prediction models, is that the resulting rules are readily interpreted by clinicians and can be used to enhance clinical decision making in a variety of contexts. These rules are created under the assumption that DNF are an appropriate representation of the manner data relate to outcome in severe community acquired phenomena. In other words, several alternative (disjunctions) mechanisms can contribute to the outcome, each mechanisms represented by the conjunction of conditions. The assumption is clinically plausible and important as we develop algorithms to compute the DNF, because it reduces the hypothesis space greatly and makes the computational hard problem solvable in reasonable time. We demonstrated learning efficiency and consistency on simulated sequences, showed the strength of the methods in learning meaningful mapping functions and showed superior prediction accuracy compared to other machine learning methods on real clinical data.

The use of DNF as a prediction tool has several strengths. Prediction rules are intuitive and easy to apply at the bedside (Figure 5.6). They could be easily interfaced with the electronic health record. Because a rule is comprised of separate disjunctive statements, each or which can be true or false, its veracity can typically be assessed even if partial data is available, and very soon following an initial assessment of the patient. A popular mortality prediction model, APACHE [99], requires 24 hours of observation before formulating a prediction. Another popular tool, MPM [100], uses information available upon initial encounter, but is less accurate and requires many more data elements to formulate a prediction. Prediction models not based on logistic regression are essentially black-box classifiers which provide little insight as to which feature drives the prediction. In this regard, DNF are very transparent in their use of data to generate a prediction.

We aimed to learn the minimum size DNF in spite of the fact that the exact learning task

is NP-complete [50, 51]. Compared to existing heuristic algorithms that only focus on learning time and learnability [48, 49, 53, 54, 55], we exploit domain knowledge and develop efficient exhaustive algorithms to learn the shortest DNF. We also applied a number of techniques to accelerate the DNF learning process (see Appendix for details), including setting the maximum length of clauses in standalone algorithm, using feature selector (CF) in MtDL to narrow down the searching space, equivalence filtering of the clauses, and extending both algorithms to greedy versions. This enables the algorithms to run efficiently on large datasets. The DNF learning algorithms are also powerful in extracting DNF from only a small numbers of sequences where the data are reliable.

The approach achieves equivalent or higher prediction performance compared to a set of state-of-the-art machine learning models, and unveils insights unavailable with standard methods. For example, we have shown that although predictive on their own, the added benefit of genetic and cytokine data over physiology and demographics-based classifiers was not spectacular in identifying poor long-term outcome. It also appears that, if one were to choose between a serum assay and a DNA profile (or SNP screen) as an early predictor of outcome, both convey comparable information with the possible exception of the product of serum levels of $IL6$ and $IL10$, plausibly a (quite naive) integrator of the magnitude of the inflammatory response. There are no currently available point-of-care kits to measure cytokine panels reliably, although a rapid kit exists for IL-6. The same is true of SNP profiling. Our exploration suggests that we probably do not need both a cytokine and SNP profile at this time, but the jury is certainly not out. Yet, it cannot be anticipated that such detailed physiotyping will be commonly performed at the bedside in the foreseeable future. Therefore, it would seem appropriate to expand data available to the DNF algorithms to include a larger overlap with data used by currently available mortality prediction tools. Indeed, one could conceive of DNF rules as representing phenotypes, confined to data that is already available, and that could be refined if more data were available to develop a more complete set of rules. The level of sophistication with which these phenotypes would be described would increase from purely clinical, to phenotypes characterized by a combination of

clinical, laboratory, and genetic markers.

Our exploration was limited to 27 SNPs and 3 cytokines, and several leukocytic surface markers in a subset of the population therefore our representation of the cellular and genetic component to physiotyping is very limited. Other analytes are now becoming available in this database, including SNPs for coagulation genes, which are definitely strong predictors of outcome. This can be understood mechanistically when that considering excessive activation of coagulation, with subsequent microthrombosis and perfusion deficit, is a plausible cause of cellular energetic failure with ensuing organ dysfunction [80].

It can be argued that 90-day mortality is an inappropriate outcome and that one would expect early physiotyping to perform better on predicting outcome on a shorter time scale. However, it is apparent, especially in this dataset that our current concept of what constitute acute illness extends well beyond the intensive care unit, or a specific hospitalization episode [101, 102]. It makes entire sense that wider genetic screens might be more predictive than early physiology in teasing late death. Different classes of predictive models are required to tease out time-varying hazard ratios [103]. Such a study would be a natural extension of this work. It could also be argued that predicting mortality does not mean the ability to predict response to treatment, a holy grail of acute care medicine. Any signal in the possible effectiveness of immunomodulatory therapies has been observed in the sickest individuals. [104, 105], suggesting the relevance of more detailed physiotyping in the prediction of the response to treatment. This is also suggested by in silico studies [76]. The DNF formulation can generally be applied to a variety of outcomes of clinical interest. For example, enrollment and decision points in clinical trials are often criteria-based. The applications of data-driven rules computed from DNF learning to the profiles of patients currently screened or enrolled in clinical trials could be quite helpful to assist clinical trial design, enrich enrollment, or eventually adapt design based on observed response.

In conclusion, we presented DNF as a novel prediction tool which perform comparably or better than currently available tools to predict outcome in patients with hospitalized community acquired pneumonia, and which presents the added advantage to be criteria-based and easily im-

plemented as a decision support system at the bedside. We believe DNF are generally applicable to a range of clinically relevant patient-centered outcomes. Despite its apparent simplicity, DNF do require the input of expert quantitative scientists to develop and implement.

Chapter 6

Conclusions

6.1 Conclusions and future work of key residue identification

We described a commonly used process for identifying key protein residue positions affecting a given viral phenotype, and argued that it is inefficient, incomplete, and unreliable. We then introduced a combinatorial filtering algorithm to systematically infer these positions using all available labeled data. We demonstrated the consistency of this algorithm, and described its use under incremental relaxation of constraints.

For cases when new data are needed to fully converge to an answer, we introduced an active learning algorithm to help choose the most informative experiment from a set of candidates to minimize the expected total laboratory time or financial cost. When additional strains are available or can be acquired from other labs, they need only be phenotyped and/or sequenced, both of which are typically much cheaper and faster than creating new mutants via reverse genetics. This enables us to make use of existing resources to optimize the identification process. The active learning algorithm can suggest which of these should be phenotyped and/or sequenced and in what order, to further minimize cost. Otherwise, the active learning algorithm suggests the next mutant to be generated.

The soft CF() algorithm can be readily generalized to tolerate errors in the data. Another

way to achieve this is via a Bayesian framework: a (possibly uniform) prior is defined over the hypotheses, and a cost (negative likelihood) function is defined based on the estimated probability that two sequences that were empirically found to have differing phenotypes in fact have the same phenotype. Possible sequencing errors are handled similarly, via a cost function. Then, in step 2b 3.1, hypotheses are merely penalized instead of eliminated and are ranked by their posterior (the product of their prior and likelihoods) at the end. We chose to avoid this approach for the time being because (1) the Bayesian framework must make assumptions regarding the data distribution and sampling method, whereas naturally occurring sets of biological sequences are hardly ever acquired by uniform sampling and often their method of acquisition is unknown; and (2) in many datasets, including the ones we discuss above, both the sequences and the phenotypic labels are considered reliable - indeterminate values are usually omitted from the analysis. In an Active Learning scenario, experimental results are typically much more reliable than those of high throughput setups. Nonetheless, we did use the Bayesian framework in a limited way in the Active Learning algorithm discussed above.

The CF() algorithm outperforms conventional position specific association methods on mutagenesis data especially when only a moderate number of data are available, and are naturally suited for active learning. It is also potentially applicable to other types of genotype-phenotype mapping inference.

Our goal for the CF() algorithm has been to aid biological investigation by identifying the key residue positions, which is the more combinatorially demanding part of the full genotype-phenotype mapping problem. Since this is our focus, we compared our method to other methods designed to do the same (e.g. [6][27][28]). We note that any conventional classifier (e.g. KNN or logistic regression) can be applied to the hypothesized positions selected by our algorithm. Given the limited data, such techniques would be ineffective if they were to be applied to the full set of protein positions, because of the so-called “curse of dimensionality”.

6.2 Conclusions of Genotype-Phenotype Mapping using DNF learning

As discussed in section 4.1, we propose to use short Disjunctive Normal Form (DNF, "OR" of "AND") as the appropriate bias over the hypothesis space because 1) DNF is a high order Boolean function that examines complicated solution space, 2) DNF offers great flexibility and allows identification of unforeseen interactions, 3) DNF is a natural form of knowledge representation for humans to interpret and provides clinical insights and rules to direct further executions, 4) DNF is scalable to large or small datasets. A short DNF increases interpretability and mitigates overfitting bias.

The shortcomings shared by Boolean function learning algorithms are the difficult learnability and inefficient running time. While existing heuristic algorithms sacrifice completeness for efficiency, we focus on the completeness and designed two efficient DNF learning algorithms utilizing biologically plausible assumptions. We applied a number of techniques to accelerate the DNF learning process, including setting the maximum lengths of clauses in the Standalone algorithm, using a feature selector (CF) in MtDL to narrow down the searching space, using equivalence filtering of the clauses, and extending both algorithms to greedy versions. These methods enable the algorithms to run on very large datasets. Notwithstanding, as shown in the result section, the DNF learning algorithms are also powerful in extracting DNFs from only a small numbers of sequences.

The mutagenesis RNA virus data are highly reliable. For the high throughput viral sequences with low quality alignment we use pruning and set thresholds to tolerate errors. The algorithms show superior prediction performances to the other state-of-the-art approaches on two benchmark datasets, and unveil biologically meaningful explanations that are unavailable to other statistical models. The DNF learning algorithms can be extensively used on other domains where similar assumptions hold.

Appendix A

Supplementary materials

A.1 RNA virus phenotypes

- Drug resistance: the development of effective antiviral drugs is an important biomedical scientific achievement of the late 20th century. However, drug resistance is a major factor contributing to therapy failure. The genetic basis of drug resistance is an RNA virus' high mutation rate and very high replication rate. Researchers have estimated that each single mutation in the 9-kbp (kilobase pair) viral genome appears once daily in each infected individual in Influenza. Some mutations lead to a slightly altered 3D protein structure that enables the viral enzyme to fulfill its task even in an inhibitor's presence (see Figure A.1). These mutants have a selective advantage under drug pressure and become dominant in the virus population. So, persistent viral replication due to subinhibitory drug levels or host immune failure leads to the evolution of drug-resistant variants and consequently to therapy failure.

  Drug resistance is defined as a reduced susceptibility to a drug in a laboratory culture system and is expressed as an altered IC50 or IC90 (drug concentration required to inhibit viral growth by 50% or 90% respectively). This phenotype is caused by specific mutations in the viral genome, which leads to a structure change of drug target proteins. The high

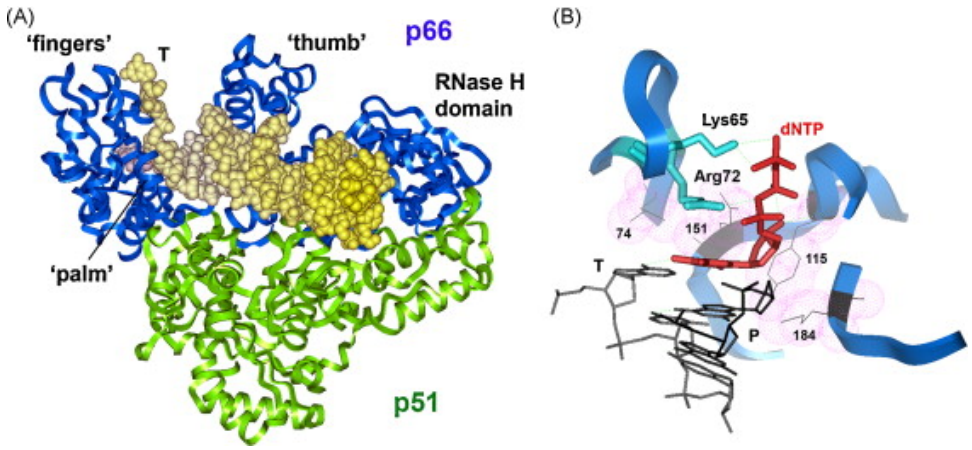

Figure A.1: **Crystal structure of the ternary complex of HIV-1 RT** [5]. Crystal structure of the ternary complex of HIV-1 RT, double-stranded DNA and incoming dNTP (A). Ribbon representations of the backbones of the p66 and p51 subunits are shown in blue and green respectively. The incoming dNTP (in purple) is located in the palm subdomain of the p66 subunit. (B) shows the dNTP binding site with the side chains of Lys65 and Arg72 making hydrogen bonds with the phosphate groups of the incoming nucleotide. Van der Waals surface of the side-chains of residues 74 (Leu), 115 (Tyr), 151 (Gln) and 184 (Met) are shown in pink. T and P stand for template and primer, respectively. Atomic coordinates were obtained from PDB file 1RTD (Huang et al., 1998)

rate of RNA virus genome mutations results in high antiviral resistance. Some resistance exists in untreated infected people already.

- Neutralizability: A neutralizing antibody is one which not only binds to the virus but in so doing prevents or at least interferes with viral replication, typically via steric hindering of viral attachment to the host cell. Not all antibodies neutralize virus, and not all the neutralizing antibody neutralize all variants of the protein. A small number of mutations can render a viral antigen not neutralizable. A neutralizing Abs ability to neutralize depends on the Ag:Ab binding affinity. (Neutralization can also be affected by more global conformational change in the protein.) For HIV, research has focused on vaccination development and after more than two decades of limited success with vaccines, researchers are returning to the importance of antibody-mediated neutralization.

- Antigenecity: Much of the burden of infectious disease today is caused by antigenically

variable pathogens that can escape from immunity induced by prior infection or vaccination [106] RNA viruses utilize **Antigenic drift** to escape the immune system. When the viruses constantly mutate, they produce new forms of these antigen. When the new form is sufficiently different from the old antigen, it will no longer bind to the receptor and the virus evade the immune system. Human Influenza viruses cause annual epidemics due to antigenic drifts in the hemmagglutinin protein [107].

A.2 Predictors identified by benchmark models of patient hospital mortality with Severe CAP

Consistently selected features in the hierarchy of models included monocyte positivity for CD-14 CD-120a, and monocytic and granulocytic positivity for toll-like receptor (TLR)-2 receptors. Note that the choice of fluorescent antibody was based on expected activation by pathogen associated molecular patterns. Although it could be that the 10-fold cross-validation procedure admitted significant overfitting (N=124), it is an interesting hypothesis that the profile of activation of immune cells conveys as much or more information than cytokines and SNP polymorphisms.

Markers of severity were the strongest predictors of 90-day mortality (table 5.5). Of demographic features, only age tercile and the presence of chronic illness were included in most predictive models. Most SNPs examined were uncorrelated to 90-day mortality, but IL6M174 (GG), L100M1048 (G/T) and MIFM173 (GG) were consistently predictive, even in multivariate models. IL18M137 was less consistently associated.

Bibliography

[1] Wu C, Walsh A, Rosenfeld R. Identification of Viral Protein Genotypic Determinants using Combinatorial Filtering and Active Learning. IEEE-BIBE. 2010;. (document), 1.2.2, 3.2, 3.6

[2] Wu C, Walsh AS, Rosenfeld R. Genotype Phenotype Mapping in RNA Viruses - Disjunctive Normal Form Learning. Pacific Symposium On Biocomputing. 2011;16:62–73. (document), 4.1

[3] Collins PJ, Haire LF, Lin YP, Liu J, Russell RJ, Walker PA, et al. Crystal structures of oseltamivir-resistant influenza virus neuraminidase mutants. Nature. 2008;453:1258–1262. (document), 5.3, 5.2

[4] Wu C, Rosenfeld R, Clermont G. Using Data-Driven Rules to Predict Mortality in Severe Community Acquired Pneumonia. PLoS ONE. 2014;9:4. (document), 5.4, 5.5, 5.6

[5] Menendez-Arias L. Molecular basis of human immunodeficiency virus drug resistance: An update. Antiviral Research. 2010;85:210–231. (document), A.1

[6] Hanenhalli SS, Russell RB. Analysis and prediction of functional sub-types from protein sequence alignments. JMB. 2000;303:61–76. (document), 1.2.2, 3.1, 3.3.1, 4.5, 4.6, 6.1

[7] Beerenwinkel N, Lengauer T, Selbig J, Schmidt B, Walter H, Korn K, et al. Geno2pheno: Interpreting Genotypic HIV Drug Resistance Tests. IEEE. 2001 11;16(6):35–41. 1.1, 1.2.2

[8] World Health Organization PH. World Health Organization, Geneva, 2010; 2009. Avail-

able from: http://www.who.int/csr/don/2010_08_06/en/index.html. 1.1

[9] Tsai KN, Chen GW. Influenza genome diversity and evolution. Microbes and Infection. 2011;p. 1–10. 1.1

[10] Drake JW, Holland JJ. Mutation rates among RNA viruses. PNAS. 1999;96:13910–13913. 1.1

[11] Knipe D, Howley P. Fields Virology. Lippincott Williams and Wilkins; 2006. 1.1

[12] Pillay D, Zambon M. Antiviral Drug Resistance. BMJ. 1998 3;317:660–662. 1.1, 4.6

[13] Shafer RW. Genotypic Testing for HIV-1 Drug Resistance. http://hivinsiteucsfedu/. 2004;. 1.1, 4.6

[14] Rameix-Welti MA, Enouf V, Cuvelier F, Jeannin P, van der Werf S. Enzymatic Properties of the Neuraminidase of Seasonal H1N1 Influenza Viruses Provide Insights for the Emergence of Natural Resistance to Oseltamivir. PLoS Pathogens. 2008;4:1–5. 1.1, 5.2

[15] Jin H, Zhou H, Liu H, Chan W, Adhikary L, Mahmood K, et al. Two residues in the hemagglutinin of A/Fujian/411/02-like influenza viruses are responsible for antigenic drift from A/Panama/2007/99. Viology. 2005;336:113–119. 1.2.1, 2.1, 3.3, 4.5

[16] Pettit SC, Henderson GJ, Schiffer CA, Swanstrom R. Replacement of the P1 Amino Acid of Human Immunodeficiency Virus Type 1 Gag Processing Sites Can Inhibit or Enhance the Rate of Cleavage by the Viral Protease. Journal of Virology. 2002;76:10226–10233. 1.2.1

[17] Hiramatsu K, Tadano M, Men R, Lai CJ. Mutational Analysis of a Neutralization Epitope on the Dengue Type 2 Virus (DEN2) Envelope Protein: Monoclonal Antibody Resistant DEN2/DEN4 Chimeras Exhibit Reduced Mouse Neurovirulence. Viology. 1996;224:437–445. 1.2.1, 2.1

[18] Beerenwinkel N, Schmidt B, Walter H, Kaiser R, Lengauer T, Hoffmann D, et al. Diver-

sity and complexity of HIV-1 drug resistance: A bioinformatics approach to predicting phenotype from genotype. PNAS. 2002 6;99(12):8271–8276. 1.2.2

[19] DiRienzo1 G, DeGruttola1 V, Larder B, Hertogs K. Nonparametric Methods to predict HIV drug susceptibility phenotype from genotype. Statistics in Medicine. 2003;22:2785–2798. 1.2.2

[20] Foulkes AS, DeGruttola V. Characterizing the Relationship Between HIV-1 Genotype and Phenotype: Prediction-Based Classification. biometrics. 2002;58:145–156. 1.2.2

[21] Draghici S, Potter RB. Predicting HIV drug resistance with neural networks. Bioinformatics. 2003 1;19(1):98–107. 1.2.2

[22] Wang D, Larder B. Enhanced Prediction of Lopinavir Resistance from Genotype by Use of Artificial Neural Networks. The Journal of Infectious Diseases. 2003 3;188:653–660. 1.2.2

[23] Sevin AD, DeGruttola V, Nijhuis M, Schapiro JM, Foulkes AS, Para MF, et al. Methods for Investigation of the Relationship between Drug-Susceptibility Phenotype and Human Immunodeficiency Virus Type 1 Genotype with Applications to AIDS Clinical Trials Group 333. The Journal of Infectious Diseases. 2000;182:59–67. 1.2.2

[24] Wang K, Jenwitheesuk E, Samudrala R, Mittler JE. Simple linear model provides highly accurate genotypic predictions of HIV-1 drug resistance. Antiviral Therapy. 2004;9:343–352. 1.2.2

[25] Rhee SY, Taylor J, Wadhera G, Ben-Hur A, Brutlag DL, Shafer RW. Genotypic predictors of human immunodeficiency virus type 1 drug resistance [article]. PNAS. 2006;10(1073):53–82. 1.2.2, 4.1, 4.7

[26] Beerenwinkel N, Sing T, Lengauer T, Rahnenfuhrer J, Roomp K, Savenkov I, et al. Computational methods for the design of effective therapies against drug resistant HIV strains. Bioinformatics. 2005 8;21(21):3943–3950. 1.2.2

[27] Mirny LA, Gelfand MS. Using orthologous and paralogous proteins to identify specificity determining residues in bacterial transcrption factors. JMB. 2002;321:7–20. 1.2.2, 3.1, 3.3.1, 6.1

[28] Pazos F, Rausell A, Valencia A. Phylogeny-independent detection of functional residues. Bioinformatics. 2006;22:1440–1448. 1.2.2, 3.1, 3.3.1, 6.1

[29] Quinlan JR. Induction of Decision Trees. In: Shavlik JW, Dietterich TG, editors. Readings in Machine Learning. Morgan Kaufmann; 1990. Originally published in Machine Learning 1:81–106, 1986. 1.2.2

[30] Clark P, Boswell R. Rule Induction with CN2: Some Recent Improvements. Proceedings of the Fifth European Working Session on Learning. 1991;482:151–163. 1.2.2

[31] Kooperberg C, Ruczinski I, LeBlanc ML, Hsu L. Sequence Analysis using Logic Regression. Genetic Epidemiology. 2001;21:626–631. 1.2.2

[32] Kooperberg C, Ruczinski I. Identifying Interacting SNPs Using Monte Carlo Logic Regression. Genetic Epidemiology. 2005;28:157–170. 1.2.2

[33] Belshe RB, Smith MH, Hall CB, Betts R, , Hay AJ. Genetic Basis of Resistance to Rimantadine Emerging during Treatment of Influenza Virus Infection. Journal of virology. 1988 5;62:1508–1512. 2.1

[34] Gubareva LV, Kaiser L, Hayden FG. Influenza virus neuraminidase inhibitors. The Lancet. 2000 3;355:827–835. 2.1, 5.2

[35] Moscona A. Neuraminidase Inhibitors for Influenza. The new england journal of medicine. 2005;353:1363–1373. 2.1

[36] Andre S, Seed B, Eberle J, Schraut W, Bultmann N, Haas J. Increased Expression and Immunogenicity of Sequence-Modified Human Immunodeficiency Virus Type 1 gag Gene. Journal of Virology. 2000;74(6):2628–2635. 2.1

[37] Garcia M, Crawford JM, Latimer JW, Rivera-Cruz E, Perdue ML. Heterogeneity in the

haemagglutinin gene and emergence of the highly pathogenic phenotype among recent H5N2 avian influenza viruses from Mexico. Journal of General Virology. 1996;77:1493–1504. 2.1, 3.3, 4.5

[38] Kwong PD, Doyle ML, Casper DJ, Cicalak C, Leavitt SA, Majeed S, et al. HIV-1 evades antibody-mediated neutralization through conformational masking of receptor-binding sites. Nature. 2002 12;420:678–682. 2.1

[39] Kalia V, Sarkar S, Gupta P, , Montelar RC. Antibody Neutralization Escape Mediated by Point Mutations in the Intracytoplasmic Tail of Human Immunodeficiency Virus Type 1 gp41. American Society for Microbiology. 2005 2;79(4):2097–2107. 2.1

[40] Cordonnier A, Montagnier L, Emerman M. Single amino-acid changes in HIV envelope affect viral tropism and receptor binding. Nature. 1989 8;340:571–574. 2.1, 3.3

[41] Mitchell TM. Machine Learning. McGraw Hill Higher Education; 1997. 3.1

[42] Cole K, Ross T. Antibody neutralization. Current HIV Res. 2007;5:505–506. 3.3, 4.5

[43] Verschoor EJ, Boven LA, Blaak H, Vliet A, Horzinek MC, , et al. A Single Mutation within the V3 Envelope Neutralization Domain of Feline Immunodeficiency Virus Determines Its Tropism for CRFK Cells. American Society for Microbiology. 1995 8;69(8):4752–4757. 3.3, 4.5

[44] Pancino G, Sonigo P. Retention of Viral Infectivity after Extensive Mutation of the Highly conserved Immunodominant Domain of the Feline Immunodeficiency Virus Envelope. Journal of virology. 1997 6;71(6):4339–4346. 3.4

[45] Cohn DA, Ghahramani Z, Jordan MI. Active Learning with Statistical Models. Journal of Artificial Intelligence Research. 1996;4:129–145. 3.5

[46] Donmez P, Carbonell JG. Proactive Learning: Cost-Sensitive Active Learning with Multiple Imperfect Oracles. In Proceedings of the 17th ACM Conference on Information and Knowledge Management. 2008;. 3.5

[47] Lewis DD, Catlett J. Heterogeneous Uncertainty Sampling for Supervised Learning. In: Proceedings of the International Conference on Machine Learning (ICML); 1994. p. 148–156. 3.5

[48] Sanchez SN, Triantaphyllou E, Chen J, Liao TW. An incremental learning algorithm for constructing Boolean functions from positive and negative examples [article]. Computer & Operations Research. 2002;29(12):1677–1700. 4.1, 4.2, 5.3.5

[49] Ruckert U, Kramer S. Stochastic local search in k-term DNF learning. Proc 20th International Conf on Machine Learning. 2003;p. 648–655. 4.1, 4.2, 5.3.5

[50] Brayton. Logic Minimization Algorithms for VLSI Minimization [article]. Kluwer Academic Publisher. 1985;. 4.2, 5.3.5

[51] Gimpel. A method of producing a Boolean Function having an arbitrarily prescribed prime implicant table [article]. IEEE Transactions on Computers. 1965;. 4.2, 5.3.5

[52] Yang L, Blum A, Carbonell JG. Learnability of DNF with Representation-Speci?c Queries. Innovations in Theoretical Computer Science. 2013;. 4.2

[53] Triantaphyllou E, Soyster AL. On the minimum number of logical clauses inferred from examples [article]. Computers & Operations Research. 1999;23(8):783–799. 4.2, 5.3.5

[54] Kamath. A continuous approach to inductive inference [article]. Mathematical Programming. 2005;57(1-3):215–238. 4.2, 5.3.5

[55] Ruckert U, Kramer S, Raedt LD. Phase transitions and stochastic local search in k-term dnf learning. Springer Verlag. 2002;p. 405–417. 4.2, 5.3.5

[56] Valiant LG. A theory of the learnable. Proceedings of the sixteenth annual ACM symposium on Theory of computing table of contents. 1984;p. 436–445. 4.2.1

[57] Towell GG, Shavlik JW, Noordewier MO. Refinement of Approximate Domain Theories by Knowledge-Based Neural Networks. AAAI-90 Proceedings. 1991;. 4.8

[58] Ferraris O, Lina B. Mutations of neuraminidase implicated in neuraminidase inhibitors

resistance. Journal of Clinical Virology. 2008 3;41:13–19. 5.2

[59] Monto AS, McKimm-Breschkin JL, Macken C, Hampson AW, Hay A, Klimov A, et al. Detection of Influenza Viruses Resistant to Neuraminidase Inhibitors in Global Surveillance during the First 3 Years of Their Use. Antimicrobial Agents and Chemotherapy. 2006;50:2395–2402. 5.2

[60] Bright RA, Shay DK, Shu B, Cox NJ, Klimor AI. Adamantane Resistance Among Influenza A Viruses Isolated Early During the 2005-2006 Influenza Season in the United States. The Journal of the American Medical Association. 2008;295(8):891–894. 5.2

[61] Yen HL, Herlocher LM, Hoffmann E, Matrosovich MN, Monto AS, Webster RG, et al. Neuraminidase Inhibitor-Resistant Influenza Viruses May Differ Substantially in Fitness and Transmissibility. Antimicrobial Agents and Chemotherapy. 2005;49:4075–4084. 5.2

[62] Mai LQ, Wertheim HFL, Duong TN, van Doorn HR, Hien NT, Horby P. A Community Cluster of Oseltamivir-Resistant Cases of 2009 H1N1 Influenza. The New England Journal of Medicine. 2010;362:86–87. 5.2

[63] Baz M, Abed Y, Papenburg J, Bouhy X, ?ve Hamelin M, Boivin G. Emergence of Oseltamivir-Resistant Pandemic H1N1 Virus during Prophylaxis. The New England Journal of Medicine. 2009;361:2296–2297. 5.2

[64] Abed Y, Bourgault AM, Fenton RJ, Morley PJ, Gower D, Owens IJ, et al. Characterization of 2 Influenza A(H3N2) Clinical Isolates with Reduced Susceptibility to Neuraminidase Inhibitors Due to Mutations in the Hemagglutinin Gene. The Journal of Infectious Diseases. 2002;186:1074–1080. 5.2

[65] Zurcher T, Yates PJ, Daly J, Sahasrabudhe A, Walters M, Dash L, et al. Mutations conferring zanamivir resistance in human influenza virus N2 neuraminidases compromise virus fitness and are not stably maintained in vitro. Journal of Antimicrobial Chemotherapy. 2006;58:723–732. 5.2

[66] Sheu TG, Deyde VM, Okomo-Adhiambo M, Garten RJ, Xu X, Bright RA, et al. Surveillance for Neuraminidase Inhibitor Resistance among Human Influenza A and B Viruses Circulating Worldwide from 2004 to 2008. American Society for Microbiology. 2008;52:3284–3292. 5.2.2, 5.2.2, 5.2.2, 5.2.2

[67] Rameix-Welti MA, Enouf V, Cuvelier F, Jeannin P, van der Werf S. Enzymatic Properties of the Neuraminidase of Seasonal H1N1 Influenza Viruses Provide Insights for the Emergence of Natural Resistance to Oseltamivir. PLOS Pathogens. 2008;4. 5.2.2

[68] Le. Isolation of drug-resistant H5N1 virus. Nature. 2005;437. 5.2.2

[69] Carr J, Ives J, Kelly L, Lambkin R, Oxford J, Mendel D, et al. Influenza virus carrying neuraminidase with reduced sensitivity to oseltamivir carboxylate has altered properties in vitro and is compromised for infectivity and replicative ability in vivo. Antiviral Research. 2002;54:79–88. 5.2.2

[70] Levy MM, Fink MP, Marshall JC, Abraham E, Angus D, Cook D, et al. 2001 SCCM/ESICM/ACCP/ATS/SIS International Sepsis Definitions Conference. Crit Care Med. 2003;31(4):1250–1256. Available from: http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?cmd=Retrieve&db=PubMed&dopt=Citation&list_uids=12682500. 5.3.1

[71] Angus DC, Linde-Zwirble WT, Lidicker J, Clermont G, Carcillo J, Pinsky MR. Epidemiology of severe sepsis in the United States: Analysis of incidence, outcome, and associated costs of care. CritCare Med. 2001 Jul;29(7):1303–1310. 5.3.1

[72] Kaplan V, Clermont G, Griffin M, Kasal J, Watson S, Linde-Zwirble WT, et al. Pneumonia, still the old man's friend? JAMA Internal Medicine. 2003;163:317–323. 5.3.1

[73] Dellinger RP, Carlet JM, Masur H, Gerlach H, Calandra T, Cohen J, et al. Surviving Sepsis Campaign guidelines for management of severe sepsis and septic shock. Crit Care Med. 2004 Mar;32(3):858–873. Available from: PM:15090974. 5.3.1

[74] Feldmann M, Bondeson J, Brennan FM, Foxwell BMJ, Maini RN. The rationale for the current boom in anti-TNFalpha treatment. Is there an effective means to define therapeutic targets for drugs that provide all the benefits of anti-TNFalpha and minimise hazards? AnnRheumDis. 1999;58:Suppl 1:I27–I31. 5.3.1

[75] Read RC. Experimental therapies for sepsis directed against tumour necrosis factor. JAntimicrobChemother. 1998;41:Suppl A: 65–69. 5.3.1

[76] Clermont G, Bartels J, Kumar R, Constantine G, Vodovotz Y, Chow C. In silico design of clinical trials: a method coming of age. Crit Care Med. 2004;32:2061–2070. 5.3.1, 5.3.5

[77] Keegan MT, Gajic O, Afessa B. COMPARISON OF APACHE III AND IV, SAPS 3 AND MPM0III, AND INFLUENCE OF RESUSCITATION STATUS ON MODEL PERFORMANCE. Chest. 2012;142:851–858. 5.3.1

[78] Vincent JL, Taccone F, Schmit X. Classification, incidence, and outcomes of sepsis and multiple organ failure. ContribNephrol. 2007;156:64–74. 5.3.1

[79] Sakr Y, Krauss C, Amaral ACKB, Rea-Neto A, Specht M, Reinhart K, et al. Comparison of the performance of SAPS II, SAPS 3, APACHE II, and their customized prognostic models in a surgical intensive care unit. BrJ Anaesth. 2008;101:798–803. 5.3.1

[80] e Silva VTC, de Castro I, Liano F, Muriel A, Rodr?guez-Palomares JR, Yu L. Performance of the third-generation models of severity scoring systems (APACHE IV, SAPS 3 and MPM-III) in acute kidney injury critically ill patients. Nephrology, Dialysis, Transplantation. 2011;26:3894–3901. 5.3.1, 5.3.5

[81] S L, F L, A M, R C, C F, A S, et al. Can generic scores (Pediatric Risk of Mortality and Pediatric Index of Mortality) replace specific scores in predicting the outcome of presumed meningococcal septic shock in children. Critical Care Medicine. 2001;29:1239–1246. 5.3.1

[82] Daley J, Jencks S, Draper D, Lenhart G, Thomas N, Walker J. Predicting Hospital-

Associated Mortality for Medicare Patients. JAMA. 1988;260:3617–3624. 5.3.1

[83] Keeler EB, Kahn KL, Draper D, Sherwood MJ, Rubenstein LV, Reinisch EJ, et al. Changes in Sickness at Admission Following the Introduction of the Prospective Payment System. JAMA. 1990;264:1962–1968. 5.3.1

[84] NY K, al Hamdan A, EM I, al Idrissi HY, al Bayari TH. Community acquired acute bacterial and atypical pneumonia in Saudi Arabia. Thorax. 1992;47:115–118. 5.3.1

[85] MJ F, BH H, JR L, DE S, RA S, LA W, et al. Comparison of a disease-specific and a generic severity of illness measure for patients with community-acquired pneumonia. J Gen Intern Med. 1995;10:359–368. 5.3.1

[86] MJ F, DN S, DE S. Hospitalization decision in patients with community-acquired pneumonia: a prospective cohort study. Am J Med. 1990;89:713–721. 5.3.1

[87] TJ M, H D, L Y. Community-acquired pneumonia requir- ing hospitalization: 5-year prospective study. Rev Infect Dis. 1989;11:586–599. 5.3.1

[88] MJ F, JJ O, D A. Prognosis of patients hospitalized with community-acquired pneumonia. Am J Med. 1990;88:1N–8N. 5.3.1

[89] A O, J H, L G. Aetiology, outcome and prognostic factors in community-acquired pneumonia requiring hospitalization. Eur Respir J. 1990;3:1105–1113. 5.3.1

[90] MJ F, TE A, DM Y, BH H, LA W, DE S, et al. A prediction rule to identify low-risk patients with community acquired pneumonia. N Engl J Med. 1997;336:243–50. 5.3.1, 5.3.2, 5.3.4

[91] Kellum JA, Kong L, Fink MP, Weissfeld LA, Yealy DM, Pinsky MR, et al. Understanding the inflammatory cytokine response in pneumonia and sepsis: results of the Genetic and Inflammatory Markers of Sepsis (GenIMS) Study. ArchInternMed. 2007 Aug;167(15):1655–1663. Available from: PM:17698689. 5.3.1, 5.3.2, 5.3.2

[92] Clermont G, Angus DC. Severity scoring systems in the modern intensive care unit.

AnnAcadMedSingapore. 1998 May;27(3):397–403. 5.3.1, 5.3.2

[93] Agnese DM, Calvano JE, Hahm SJ, Coyle SM, Corbett SA, Calvano SE, et al. Human toll-like receptor 4 mutations but not CD14 polymorphisms are associated with an increased risk of gram-negative infections. Journal of Infectious Diseases. 2002;186(10):1522–1525. Available from: PM:12404174. 5.3.1, 5.3.2

[94] Poynter SE, Wong HR. Role of gene polymorphisms in sepsis. Pediatr Crit Care Med. 2002;3(4):382–4. Available from: http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?cmd=Retrieve&db=PubMed&dopt=Citation&list_uids=12813318. 5.3.2

[95] Stuber F. Effects of genomic polymorphisms on the course of sepsis: is there a concept for gene therapy? J Am Soc Nephrol. 2001;12 Suppl 17:60–4. Available from: http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?cmd=Retrieve&db=PubMed&dopt=Citation&list_uids=11251034. 5.3.2

[96] Wunderink RG, Waterer GW, Cantor RM, Quasney MW. Tumor necrosis factor gene polymorphisms and the variable presentation and outcome of community-acquired pneumonia. Chest. 2002;121(3 Suppl):87S. Available from: PM:11893712. 5.3.2

[97] Lemeshow S, Hosmer DW. A review of goodness of fit statistics for use in the development of logistic regression models. American Journal of Epidemiology. 1982;115:92–106. 5.3.2

[98] Clermont G, Angus DC, DiRusso SM, Griffin M, Linde-Zwirble WT. Predicting hospital mortality for patients in the intensive care unit: a comparison of artificial neural networks with logistic regression models. Crit Care Med. 2001;29:291–296. 5.3.2

[99] Zimmerman JE, Kramer AA, McNair DS, Malila FM. Acute Physiology and Chronic Health Evaluation (APACHE) IV: hospital mortality assessment for today's critically ill patients. Crit Care Med. 2006;34:1297–1310. 5.3.5

[100] Vasilevskis EE, Kuzniewicz MW, Cason BA, Lane RK, Dean ML, Clay T, et al. Mortality

Probability Model III and Simplified Acute Physiology Score II. Chest. 2009;136:89–101. 5.3.5

[101] Angus DC, Laterre PF, Helterbrand J, Ely EW, Ball DE, Garg R, et al. The effect of drotrecogin alfa (activated) on long-term survival after severe sepsis. CritCare Med. 2004 Nov;32(11):2199–2206. 5.3.5

[102] Kaplan V, Griffin MS, Watson RS, Linde-Zwirble WT, Kasal J, Clermont G, et al. Do elderly survivors of community-acquired pneumonia remain at increased risk of death after hospital discharge? In: American Journal of Respiratory & Critical Care Medicine. vol. 163; 2001. p. A253. 5.3.5

[103] Kasal J, Jovanovic Z, Clermont G, Weissfeld LA, Kaplan V, Watson RS, et al. Comparison of Cox and Gray's survival models in severe sepsis. Crit Care Med. 2004 Mar;32(3):700–707. Available from: PM:15090950. 5.3.5

[104] Bernard GR, Vincent JL, Laterre PF, LaRosa SP, Dhainaut JF, Lopez-Rodriguez A, et al. Efficacy and safety of recombinant human activated protein C for severe sepsis. NEnglJMed. 2001 Mar;344(10):699–709. 5.3.5

[105] Panacek EA, Marshall JC, Albertson TE, Johnson DH, Johnson S, MacArthur RD, et al. Efficacy and safety of the monoclonal anti-tumor necrosis factor antibody F(ab')2 fragment afelimomab in patients with severe sepsis and elevated interleukin-6 levels. Crit Care Med. 2004 Nov;32(11):2173–2182. Available from: PM:15640628. 5.3.5

[106] Smith DJ, Lapedes AS, de Jong JC, Bestebroer TM, Rimmelzwaan GF, Osterhaus AD, et al. Mapping the Antigenic and Genetic Evolution of Influenza Virus. Science. 2004;305. A.1

[107] Franciscus A. New HCV Antivirals and Drug Resistance. HCV Advocate. 2009;. A.1