Modeling Cancer Phylogeny Tree Generation

By Stefan Ivanovic

General Task

- The Goal is modeling the probability of different cancer evolution phylogenies.
- Given a phylogeny tree, the model can determine the probability of that phylogeny tree.
- For now, only SNVs are allowed in the phylogeny for simplicity, however, the same general method applies to very diverse phylogeny problems including CNAs, migrations, etc.
- This can be used to determine the high probability trees from the set of possible phylogenetic trees.
- This can also be used to directly understand the patterns in cancer evolution, including evolutionary pathways and the fitness effects of mutations.

Precise Task

- The goal is to maximize the probability of seeing the data given of the model.
- Define P(T, M) as the probability of tree T given model parameters M.
- Let C_i be the set of phylogeny trees possible for patient i (and assume they all generate the data for patient i with equal probability).
- One therefore needs to maximize the below expression.

$$\prod_i \sum_{T \in C_i} P(T, M)$$

The Model

- The model iteratively adds mutations to partially completed trees until the tree is complete.
- Let there be M mutations in the data set.
- Let T be a partially completed tree with N clones.
- Let C_i be an M dimensional vector where C_i[j] = 1 if clone i has mutation j, and C_i[j] = 0 otherwise.
- Let f be a function that inputs and outputs M dimensional vectors.
- The probability that mutation j occurs on clone i in the next 1 time unit is exp(f(C_i)[j]).
- Equivalently, the probability that the next mutation to occur is mutation j on clone i is softmax([f(C_1) | …| f(C_N)])[i, j].

The Model Part 2

- The probability of a tree T occurring is sum of the probability of all evolution processes that generate that tree.
- For example, the tree with edges (Root, A), (Root, B) could be generated in two ways: first having mutation A then mutation B, or first having mutation B then mutation A.
- The probability of generating a tree can be optimized without directly knowing the probability of generating that tree by using reinforcement learning.
- In fact, in cases where the patient has many many possible trees (for instance when bulk sequencing is used), the probability of generating a tree in that set can be optimized without even explicitly representing the set of trees.

The Model Part 3

- The function f, which determines the probability of new mutations, is chosen to be a simple neural network with 2 layers and L hidden neurons.
- Keeping L independent of the number of mutations, allows the number of parameters to only grow linearly in the number of mutations.
- Therefore, this model is much more resistant to overfitting than many existing models.
- For example, a model with one parameter per possible tree edge has the number of parameters grow quadratically in M, and therefore has a higher risk of overfitting.

Some Technical Details of Two Training Methods

- The default reinforcement learning method is extremely efficient when there are very large number of possible trees per patient (even too many trees to explicitly enumerate, as long as there is a procedure to determine if a tree is possible).
- However, the default reinforcement learning method sometimes has difficulty when only a tiny percentage of evolution processes generate possible trees (< 1 in 1,000,000).
- In this case, one can modify the sampling procedure to only sample evolution processes that generate a given tree. One can then do this for every tree in the data set, and modify the reinforcement learning loss function to correct for this.
- This slows down the training, but has no other downsides.

Examples:

It converges to the optimal solution on all the below examples, and many more:

- P percentage of patients have the chain [1, 2, 3], and (1-P) are [3, 2, 1]
- All patients have the bulk frequencies [0.9, 0.4, 0.3], [0.9, 0.3, 0.4], which is only consistent with the tree [(Root, 1), (1, 2), (1, 3)].
- All patients have a chain [A, B], where A and B are chosen randomly from separate sets of 100 mutations (giving 10,000 possibilities), and there are only 1000 training data points (the solution is approximately optimal).
- All patients have 10 mutations, 9 of which are made on the root, but one is randomly chosen to be added to an already mutated clone.
- One third of patients have each of the bulk frequency measurements [0.9, 0.8, 0.7], [0.3, 0.2, 0.1], [0.1, 0.2, 0.3], with the optimal solution of 50% the tree [(Root, 1), (Root, 2), (Root, 3)] and 50% the chain [1, 2, 3].

RECAP simulated data

- The model should be flexible enough to apply to tasks outside the domain of task which it was designed for. Therefore, I tested it on the simulated data from the RECAP paper from professor El-Kebir's research.
- In these data sets, a small number of true trees were generated, and each patient is assigned one of these true trees.
- Then, each patient has bulk sequencing simulated, generating many possible trees.
- The task is to predict the true tree from the set of possible trees.
- This process does not follow the realistic assumptions of our model (for instance, it is unrealistic to have the trees [(Root, A), (A, B), (A, C)] and [(Root, A), (Root, B), (Root, C)] but never [(Root, A), (A, B), (Root, C)]).
- Our model is not designed for clustering (where one knows a priori there are a small number of unique true trees), and must be made compatible with this.

RECAP simulated data details

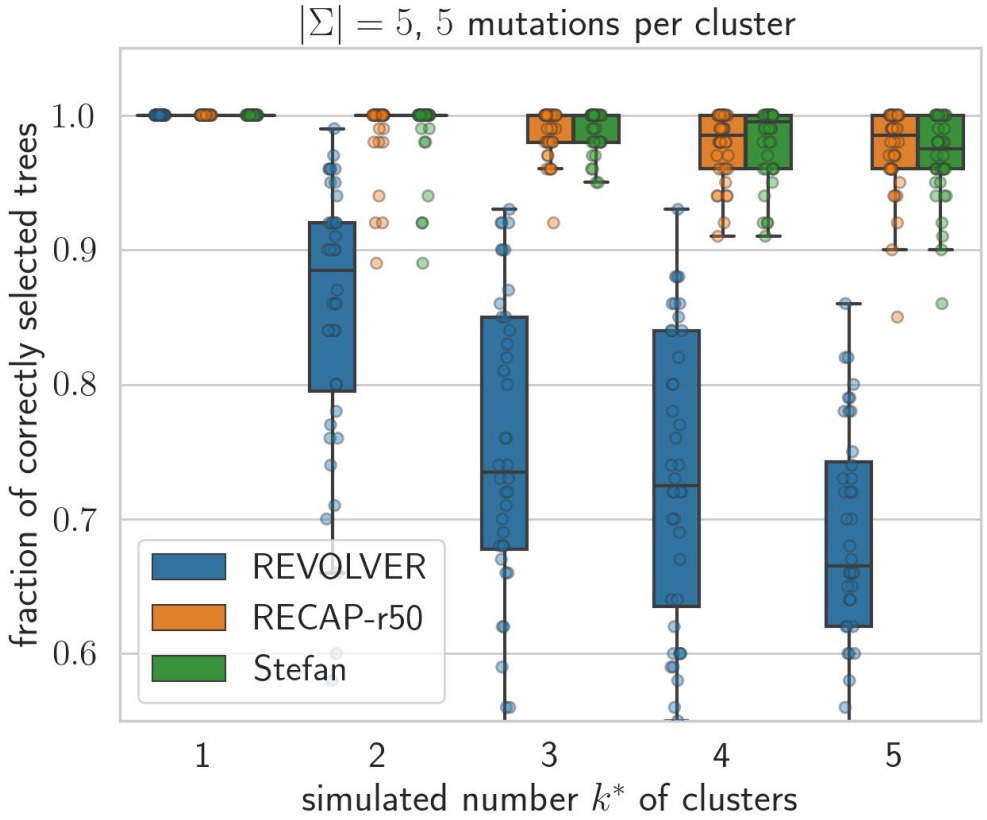
- There are two groups of data sets I used, one with 5 mutations in total and 5 mutations per patient, and one with 12 mutations total and 7 mutations per patient.
- Each of these groups of data sets has 200 individual data sets.
- The individual data sets have between 1 and 5 clusters (unique true trees).
- They also have either 50 or 100 patients.

Adapting for Clustering

- The predictions of our model need to be adapted to the knowledge that there is only a small number of unique true trees (clusters).
- Let T be a list of trees predicted for each patient, and let P(T) be the probability the model assigns to all of these trees being chosen.
- Define S(T) as the list of unique trees in T.
- Initialize T as the list of trees that maximize P(T).
- Define T_i as the list of trees that maximize P(T_i), with S(T_i) = S(T) - S(T)[i]
- Set T to the T_i that maximizes P(T_i), and repeat the previous step.
- Continue until it is impossible to continue since eliminating an additional tree would eliminate all possible trees for some patient.

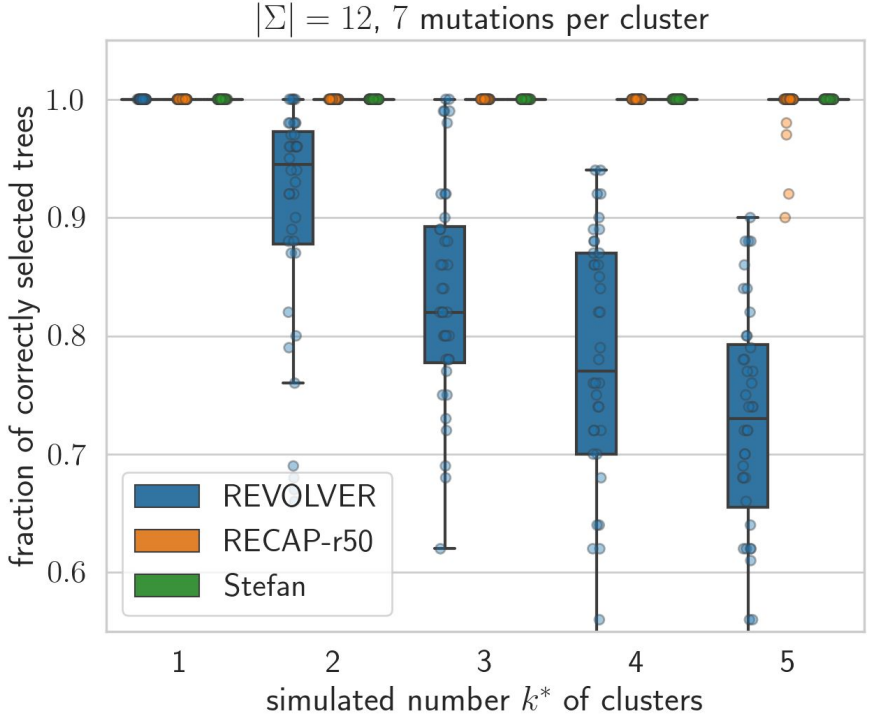
Results: 5 Mutations accuracy

- The plot has a dot for every data set.
- The y axis is prediction accuracy, and the x axis is number of clusters.
- RECAP and my method both have an accuracy of 98.6%.



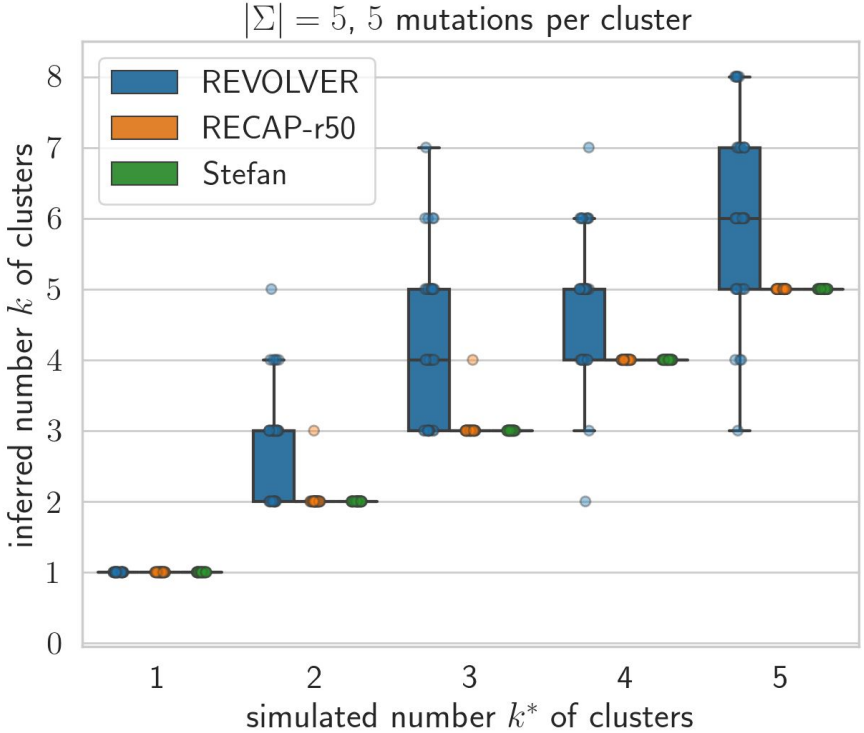
Results: 12 Mutations accuracy.

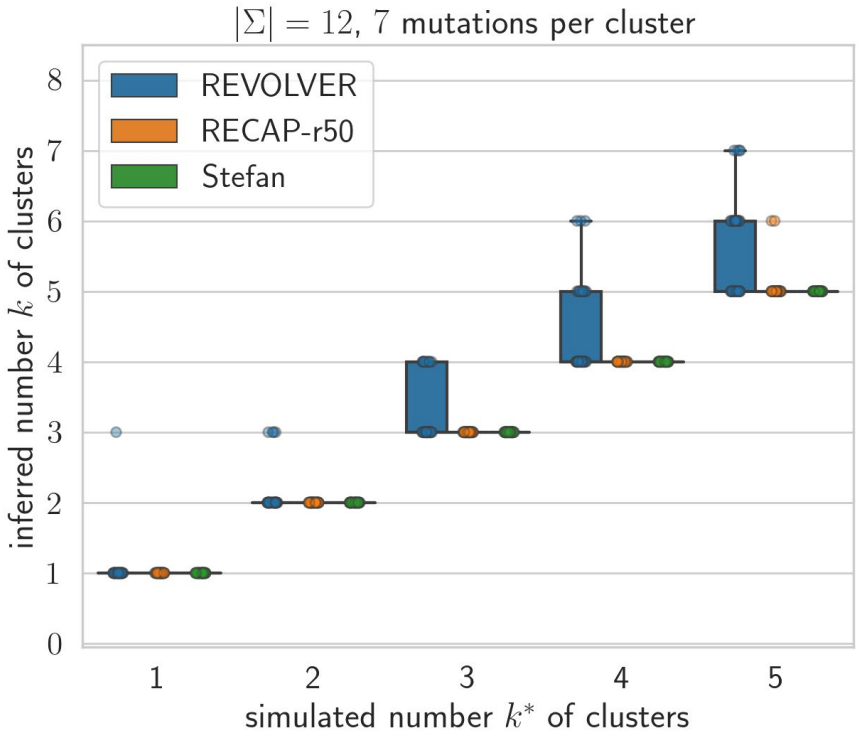
- RECAP has almost 100% accuracy, and my method has exactly 100% accuracy.
- RECAP has a few incorrect predictions when the cluster size is 5.



Predicted Cluster Number

- RECAP almost always predicts the correct number of clusters. My method identifies the correct clusters 100% of the time.



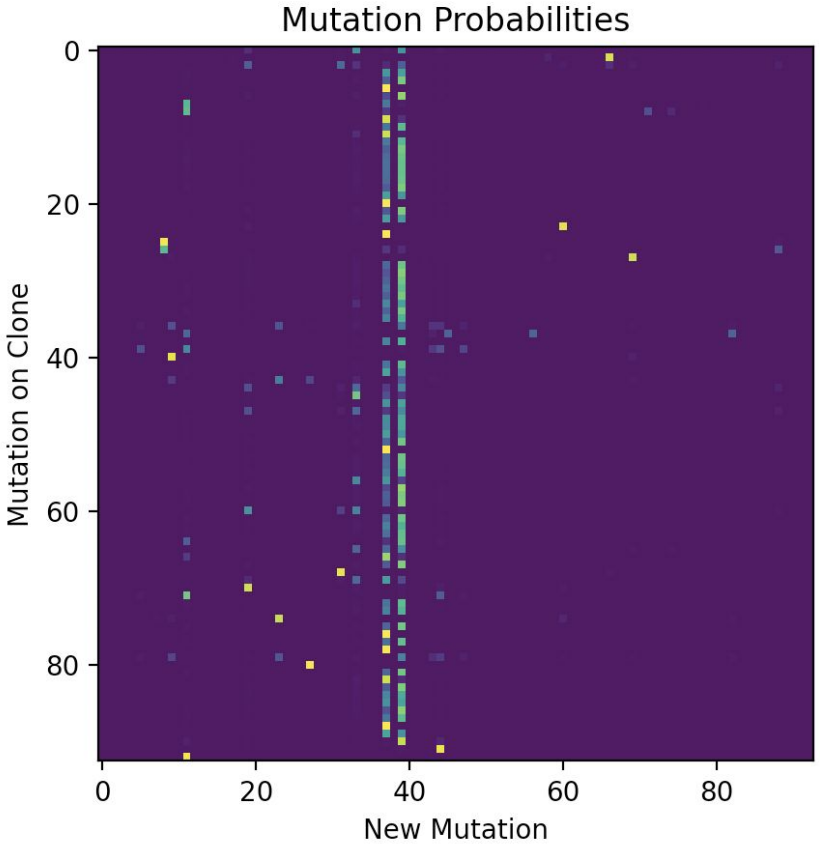


Real Data

- I manually transcribed 111 trees for 77 patients from the data in “Clonal Evolution of Acute Myeloid Leukemia Revealed by High-Throughput Single-Cell Genomics”
- There are typically less than 10 mutations per patient, and often less than 5.

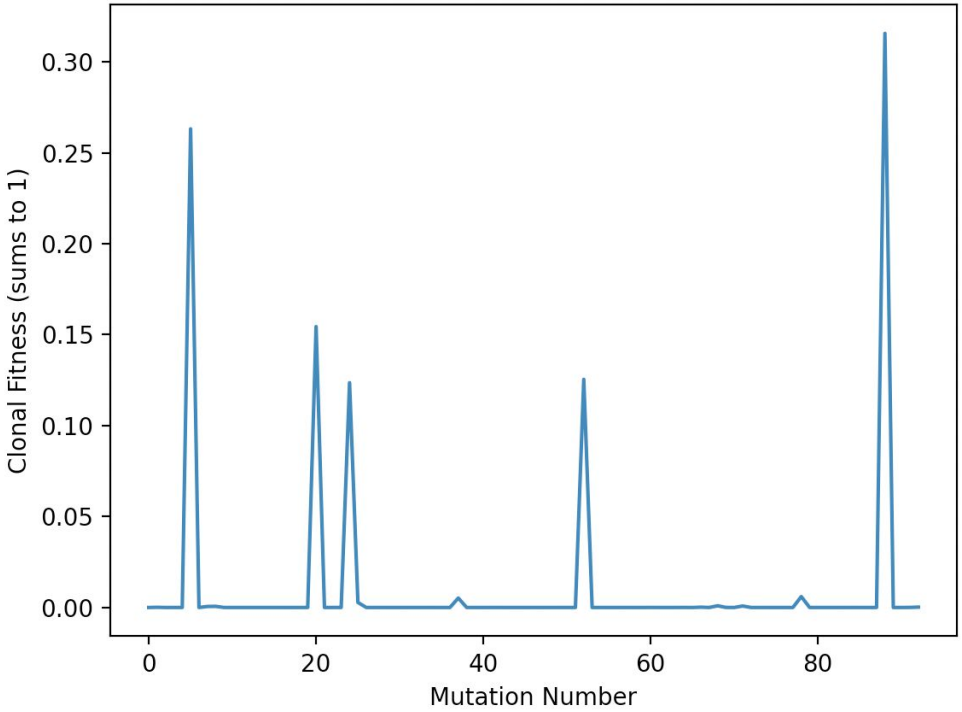
Real Data: Co-occurence Patterns

- The plot shows the probability of each mutation on a clone that has exactly one mutation.
- The Y axis is the existing mutation and the X axis is the new mutation.
- Two of the new mutations have very high probabilities for many clones.
- There is a clear pattern of some driver mutations causing other mutations.
- The pattern is very asymmetric. Mutation A causing mutation B does not imply mutation B causes mutation A.



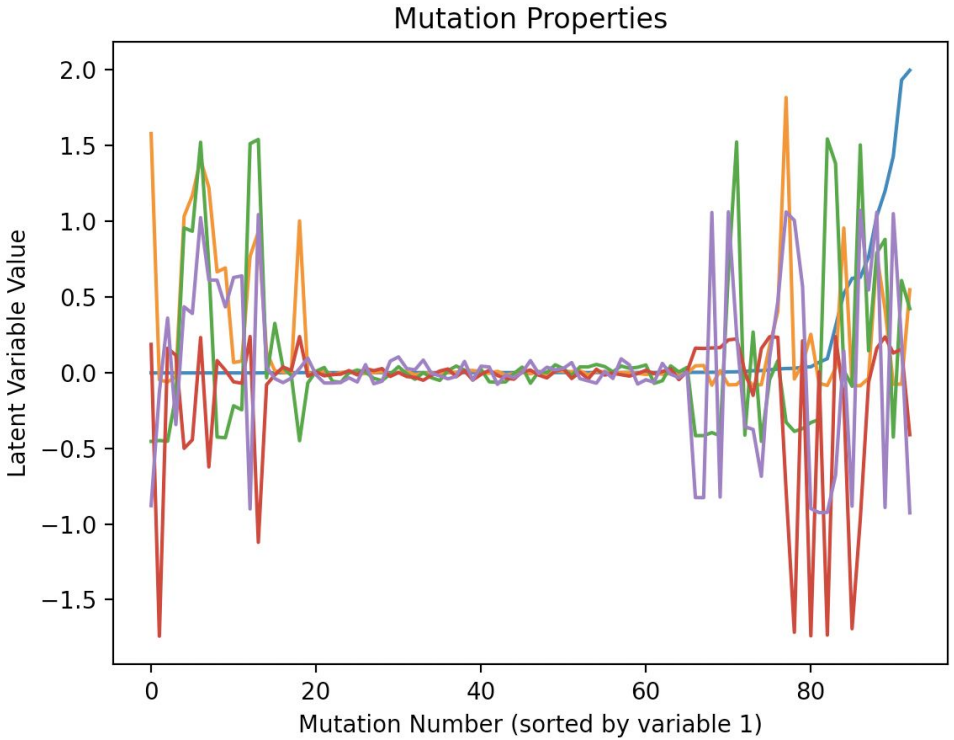
Cancer fitness

- The relative fitness of a tumor can be defined by the probability that if a mutation occurs, it will occur on that clone.
- The plot shows the relative fitness of all of the tumors which only contain one mutation.
- 5 of the clones are much more fit than all of the other clones combined. Those mutations are likely very adaptive.



Low Dimensional Representations

- The model creates low dimensional internal representations of the clones.
- The plot shows the 5 dimensional representation of all of the clones with exactly one mutation.
- The clones are sorted by the first dimension of the representation.
- As one can see, around half of the mutations are similar/unimportant.
- This representation can be used for many different cancer related tasks.



Conclusion

- In conclusion, this modern has many desirable properties.
- It is flexible, resistant to overfitting, uses realistic assumptions, performs very well on simulated data, is interpretable, detects a variety of interesting patterns, and creates potentially useful low dimensional representations.
- There are two main directions for future work:
- 1. Better utilizing the predictions of the model.
- 2. Expanding the model to account for more complex evolutionary behavior such as CNAs.

Technical detail 1:

- The actual optimization for this problem is done using the below gradient.

$$\frac{d}{dM} log(\prod_{i} \sum_{T \in C_i} P(T, M)) = \sum_{i} \frac{d}{dM} \log(\sum_{T \in C_i} P(T, M))$$

$$= \sum_{i} \sum_{T \in C_i} \{(\frac{d}{dM} P(T, M)) / \sum_{T \in C_i} P(T, M)\}$$

Technical detail 2:

- In reinforcement learning, typically the model assigned probability and the sample probability are assumed to be the case. However, it doesn't have to be if one is careful about the difference (note: negative sign dropped for simplicity)

$$L = R\log(P)$$

$$\frac{dL}{dM} = R\frac{dP}{dM}\frac{1}{P} = R\frac{dP_{model}}{dM}\frac{1}{P_{sample}}$$

Technical detail 3:

- It is unrealistic to have the first two trees but not the third.