DBDM Assignment 4: Gene prediction in bacteria E.Coli using Hidden Markov Models.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Alexey Gritsenko | 0940755 |  |  | Jelmer Cnossen | XXXXXX |

# Abstract

Someone will have to write something here...

So far seem to be done:

* Statistics

# Introduction

With advances in sequencing technologies more and more individual genomes of different species are sequenced. This provides overwhelming amounts of information to be processed, as genomes have to be annotated in order to recognize genes. Finding genes in a genome is a cumbersome task that requires carrying out biological experiments, which cost time and money. In order to aid scientists various algorithms are designed to predict genes in a given genome. One such method is also used in Data Mining and Machine Learning and is called Hidden Markov Models (HMM).

Our task is to predict genes in Escherichia Coli bacteria using Hidden Markov Models. The general algorithm to be used is:

1. Construct an HMM.
2. Train the model on a part of E.Coli genome.
3. Assess performance of the given model.

Every part of this approach will be further addressed in detail.

## Hidden Markov Models

A Hidden Markov Model (HMM) is a stochastic model that captures the statistical properties of observed real world data. A good HMM accurately models the real world source of the observed data and has the ability to simulate the source. Machine Learning techniques based on HMMs have been successfully applied to problems including speech recognition, optical character recognition, and as we will examine problems in computational biology.

A generalized HMM consists of a finite set of states, a set of transition probabilities between these states, an output alphabet emitted by these states and a set of emission probabilities. Emission probabilities define the distribution of symbol that may be emitted from each state. HMM consists of 2 stochastic processes – transition between states of HMM and emission of alphabet symbols. The states are hidden and observed only though symbols emitted (hence the name HMM).

# Our approach

In order tackle the problem of predicting genes in E.Coli DNA we have devised several Hidden Markov Models. Some of them were experimental, others were designed for production. A difference between models that is essential for us is the great difference in complexity (number of hidden states).

Main focus in our work was given to working on two models:

1. A model with simple intergenic sub-model inside;
2. A model with complex intergenic sub-model inside.

We have chosen MATLAB as our HMM framework, as it integrates all features we need, including DNA processing functions (Bioinformatics toolbox), HMM functions (Statistics toolbox) and Gene Bank interface (Bioinformatics toolbox). We have developed a set of MATLAB scripts and classes that aid construction of these models. These scripts performed most of the work automatically, including statistics and probability estimation, model construction, training and testing.

## Data used

We did not use datasets provided with the assignment for our model. Instead we used Gene Bank annotated genomes though MATLAB interface.

We’ve introduced a small MATLAB script to work with gene bank data. This script works as a caching proxy: checks if a gene bank file exists on hard drive and if not – downloads it from gene bank repository.

The whole dataset (5478 genes, 5528445 base pairs per strand) was split into two contigs – training and testing datasets. The dataset was split in such way that training set contained 25% (1369 genes, 1413243 base pairs per strand) of the original data and testing set - 75% (4109 genes, 4115202 base pairs per strand) of the data. Data was split into contigs in such way, that no genes were broken apart.

## Statistical report

HMM use statistical methods to model real data. That’s why gathering statistics about the data is the first step in HMM construction.

We will present statistics used in our models: codon statistics, nucleotide statistics and stop/start codon statistics. Statistics are presented separately for training and testing datasets. When appropriate, statistics were gathered on both direct and complementary strands of DNA.

Table (2) in Appendix presents nucleotide statistics separately for training, testing and original datasets. For each dataset statistics are gathered for genic, intergenic regions and for whole dataset for reference; relative frequencies are given. Non-traditional nucleotides were not counted by the statistics, as they are not used by the model; for genic and intergenic regions only regions/genes with no non-traditional nucleotides were counted. By traditional nucleotides we understand A, C, G and T (Adenine, Cytosine, Guanine and Thymine respectively). For genic region statistics only genes with traditional start and stop codons were counted, start and stop codons were disregarded from statistics. Codons TAA, TGA, TAG are considered traditional stop codons and codons ATG and GTG are considered traditional start codons. Such restrictions are made because HMM we devised use a separate mechanism to account for frame shifts and do not take into account possibility of untraditional start or stop codons.

Note, that statistics for all nucleotides are roughly the same (around ) and differ between training and test contigs. This difference is not important for genic model, as nucleotide distribution is used there only in insertion states, however such difference can potentially influence performance of a complex intergenic model.

Table (3) in Appendix gives codon usage statistics for training and testing contigs. These statistics were gathered without division into genic and intergenic region. Table (4) in Appendix gives codon used statistics for the whole genome. Tables (3) and (4) and given for reference only, as these statistics are not incorporated in models we use. Number of regions used to gather statistics in training contig is: 1162 (of 1369) for genic regions and 1001 (of 1147) for intergenic regions; for testing set these numbers are: 3111 (of 4109) for genic regions and 2713 (of 3444) for intergenic regions.

Table (5) in Appendix gives codon usage statistics in genic regions of training and testing contigs. These statistics are used to construct genic sub-model of HMMs described. To gather these statistics only genes with traditional start, stop codons, length devisable by 3 and traditional nucleotides were considered. Codon statistics for intergenic regions are omitted as those are not requested by assignment, nor required for model construction (intergenic models presented use nucleotide emissions instead of codon emissions).

Table (6) in Appendix presents start and stop codon statistics for training, testing contig and Table (7) - for whole genome. These statistics were again gathered only for ‘traditional’ genes. Statistics from this Table (6) are used to construct the base model described later.

## Base model

The general model for HMMs we described and implemented is given in Figure (1). The model consists of genic and intergenic sub-models. Each sub-model is a separate part of HMM and hence submodels are interchangeable; our implementation contains 2 genic submodels and 2 intergenic submodels.

Figure 1: general HMM structure outline. Filled circle in the center is a central state that does not emit anything and is used to connect sub-models together in an easy way; each dashed rectangle represents a sub-model.

Model of coding region (Genic model)

Stop codons

Intergenic model

Start codons

HMM described in Figure (1) models only one strand of the DNA, so it has to applied twice (for original strand and it’s reverse complement).

### Start and stop codon sub-models

Start and stop codon models are likely to be all very similar, however there is a freedom of choosing which codons to model. We have decided to model only traditional start and stop codons. This has been done in order to keep the model simple and not to overfit training contig data. Figures (2) and (3) present stop and start codon models respectively.

Figure 2: stop codon sub-model. Rectangles with a single solid border emit nucleotides with probabilities proportional to lengths of corresponding bar; rectangles with double border do not emit anything and are included for easier sub-model linking; strength of the lines is proportional to transition probability between states; arrow shows transition direction.

Begin

A

G

C

T

A

G

C

T

A

G

C

T

A

G

C

T

A

G

C

T

End

Figure 3: start codon model. For description see Figure (2).

A

G

C

T

A

G

C

T

A

G

C

T

End

Begin

Transition and emission probabilities for these two models are estimated automatically and can be derived as relative frequencies from Table (6) in Appendix.

### Genic sub-model

By genic sub-model we understand a HMM that models data in a gene between its start and stop codons, hence models coding region of the gene. Various approaches may be considered for modelling this region:

1. Modelling structure of the region using single nucleotide emission or using codon emission;
2. Modelling structure with regard of previously emitted states (introduces conditional probabilities) or without it.

This assignment suggests modelling this region using codon emissions without introducing conditional probabilities. This choice can be supported by several reasons, main of which are:

1. Codon emission better reflects what actually happens in nature – genes are translated into proteins using triplets of nucleotides (codons) to match an amino acid;
2. Introducing conditional probabilities would greatly increase complexity (in terms of number of states) of the model and could also overfit training contig. However it could allow for more careful modelling of genes and could implicitly introduce modelling of protein domains.

A schematic illustration of proposed genic model is given in Figure (4).

Figure 4: schematic structure of genic model. Green circle is the central state of the genic model, it does not emit anything (it can be identified as central state from Figure (1). Each rectangle represents a codon-emitting sub-model.

AAA

AAG

AAC

TTT

Model from Figure (4) contain a total of 61 codons (all possible codons without stop codons – E.Coli genes contain a total of 73105 start codons and only 2 stop in the middle), each codon has a codon-emitting sub-model associated with it. Transition probability from central node to each codon sub-model is determined using statistics from Table (5). Codon sub-model is described in detail in Figure (5) for codon AAG.Figure 5: codon-emitting sub-model of genic model. For description see Figure (2); states and transitions depicted with dashed lines allow nucleotide insertions and deletions.

End

Begin

A

G

C

T

A

G

C

T

A

G

C

T

A

G

C

T

A

G

C

T

A

G

C

T

A

G

C

T

As mentioned in Statistical report section, for statistics and model construction of genic region we have only considered genes with traditional start/stop codons, traditional nucleotides and traditional length. However not all gene have traditional length – this is caused by frame shifts. As suggested in [1] we used deletions and insertions in codon-emitting model in order to account for this phenomenon. Probability of a deletion or insertion is taken to be equal to . In Figure (5) states and transitions that allow insertions and deletions are depicted using dashed lines. Diamonds represent nucleotide insertion states; nucleotides probabilities for insertion states are determined automatically and are equal to nucleotide relative frequencies from Table (2) in Appendix.

### Testing genic sub-model

Modular structure of HMM allows testing its components. In order to test genic sub-model we have decided to find the most probable path through this sub-model for training contig genes. To do this we have extended the model with start and stop codon sub-models entering and exiting central state respectively: probability of transition from start codon model to genic model is 1, but transition probability from central state to stop codon model in whole HMM will play a role of transition probability from genic model to intergenic model; [1] suggests to approximate this probability as number of intergenic regions divided by number of codons in a typical E.Coli contig.

To calculate the probability of transition from genic model to intergenic model we have found number on intergenic regions in our testing conting and number of codons on this contig (both direct and complement); the transition probability was then . Since by definition emission probabilities must sum to 1 (as do transition probabilities from central node to codon-emitting sub-models in genic sub-model), we’ve multiplied each transition probability from central node to codon-emitting sub-model by factor .

To find most probable path through the model we utilized Viterbi algorithm. If the produced set of states for then gene started with start codon states and stoped with stop codon states, we concluded that the model correctly models this gene. Results of this experiment are given in Table (1) and Table (2), tests were performed only on genes with traditional start and stop codons and traditional nucleotides.

|  |  |  |  |
| --- | --- | --- | --- |
|  | With nucleotide insertions and deletions | | |
| Length is not multiple of 3 | Length is multiple of 3 | Both |
| Total | 0 | 1162 | 1162 |
| Correct | 0,  N/A | 1162,  100% | 1162,  100% |

Table : genic sub-model (with nucleotide insertions and deletions) test results.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Without nucleotide insertions and deletions | | |
| Length is not multiple of 3 | Length is multiple of 3 | Both |
| Total | 0 | 1162 | 1162 |
| Correct | 0,  N/A | 1162,  100% | 1162,  100% |

Table : genic sub-model (without nucleotide insertions and deletions) test results.

No genes of length that is not a multiple of 3 with traditional start/stop codons were present in the training conting (it seems that most genes in E.Coli of length that is not a multiple of 3 have non-traditional start/stop codons), hence zeros are present in both tables.

Both genic sub-models were able to predict correct paths for all genes, however only genic sub-model with insertions and deletions is capable of modeling frame shifts, hence we will use it.

## Intergenic sub-model

The last part of base model left to define is intergenic sub-model. As in [1] we have tried 2 intergenic sub-models – complex and simple one.

### Simple model

Simple intergenic model consists of a single state (except begin and end states) from which transition to the same state is possible, see Figure (6).

Figure 6: simple intergenic model. For description see Figure (2).

End

Begin

A

G

C

T

Emission probabilities for central state of intergenic model are estimated from intergenic regions of the training contig and can be found in Table (2) in Appendix.

The tricky part of this model is setting transition probabilities from central node. We have approximated probability of stay in central intergenic state as , where is average length of an intergenic region in a typical E.Coli DNA (as transition from the intergenic model is done once after the intergenic region has been fully constructed). Note, that [1] does not mention how this transition probability should be approximated.

### Complex model

Complex intergenic sub-model consists of 2 more sub-models of similar structure, but different size. These sub-models are short and long intergenic sub-model, they are designed to model short (of length 1-14 nucleotides) and long (of length 10 or more nucleotides) intergenic regions respectively.

General structure of complex intergenic model can be found on Figure (7).

Figure 7: general structure of complex intergenic model. See Figures (1) and (2) for description.

End

Begin

Short intergenic sub-model

Long intergenic sub-model

Transition probabilities to short and long intergenic regions are proportional to number of intergenic regions of given length; total number of intergenic regions in training contig is 1147, 154 of which have length of 1-14 nucleotides long and and 1054 have length 10 or more nucleotides.

Figure (9) in Appendix presents common structure for long and short sub-models. This model was inspired by [1], however it is probably different as the paper did not give enough details to restore complex intergenic model used in it ([1] does not mention directions and initial transition probabilities between states). Our idea behind model on Figure (9) is that intergenic regions have some kind of structure (for example intergenic regions could be silenced genes or contain common nucleotide sequences like Shine-Delgarno sequence) and this general structure can be modeled by states depicted as squares (with single solid border), some insertions and deletions are possible in intergenic regions as a result of random mutations, and states depicted as circles (deletions) and diamonds (insertions) allow this possibility. Emission probabilities for all states that emit nucleotides are estimated as codon usage frequencies in intergenic region and can be found in Table (5) in Appendix. Complex intergenic model does was not design to model any kind of structure explicitly, instead the idea is to learn possible structures in intergenic regions of genome by mean of HMM training procedures. This is why we did not pay too much attention to setting transition probabilities – for every state transition probabilities to all other states are set equal (hence if from some state we can transition to other states, then transition probabilities are set to ).

#### Short and long intergenic models

Despite of the possible difference between our complex intergenic model and the one described in [1] we have adopted model lengths suggested there – length of short model was set to 9 and length of long model was set to 44.

## Putting all together

We have described all parts of the general HMM shown in Figure (1). In order to construct the final model all sub-models must be connected into a single model using mentioned transition probabilities between sub-models. In case a sub-model requires training it has to be trained before it is merged into a final model.

Final HMMs had the following number of states: simple model has 436 states and complex model has 543 states.

## Complexity estimations

When models are ready, prior to feeding them to corresponding algorithms it is wise to estimate approximate runtime of these algorithms on our models and amount of input data we have.

We have to main algorithms that we utilize for purposes of this assignment:

1. Viterbi algorithm – finds most probably path (through hidden states of Markov model) using dynamic programming;
2. Baum-Welch algorithm [3] – used in HMM training; modifies transition and emission probabilities in such way that they maximize models probability of emitting given string.

For every symbol in training sequence and every hidden state Viterbi algorithm calculates probability of emitting this symbol by given state state. MATLAB implementation of this algorithm checks all possible transitions (even those with probability 0 and transition to nodes which have 0 probability of emitting given symbol from sequence); this gives us time complexity of and space complexity of , where – number of states in HMM and – length of sequences we would like to find the path for.

Baum-Welch is special case of generalized iterative expectation-maximization algorithm that updates transition and emission probabilities based on backward and forward sequence emission probabilities. For each iteration Baum-Welch algorithm requires operations, where - number of states in HMM and – sum of lengths of all training sequences; number of iterations may vary, but default maximum limit in MATLAB is 500.

With and this task becomes hardly feasible (mainly because of very big number of states). This means that some work-around must be found in ordered to solve the problem in the amount of time given.

## Performance assessment

In order to test performance of our model we need to apply them on E.Coli genome to determine positions of probable genes in the genome. After running Viterbi algorithm for a given sequence and model we do not get annotated genome as output. What we get is actually a set of most probable hidden states for each emitted nucleotide in the sequence. Since we know which states belong to genic part and which – to intergenic, we can then extract genes (as start and stop positions) from states that were given my Viterbi algorithm.

Having annotated genes does not solve all the problems, in fact it introduces some new ones – we now have to match genes our programme has output with real genes (also given as positions of first and last nucleotides) to see how well our models can predict genes. We do not expect our model to predict genes perfectly moreover we expect to have situations when a predicted gene overlaps partially with several annotated genes. In order to build one-to-one correspondence we have to choose only one gene of all genes that overlap with the gene we predicted. We decided to simply choose gene which overlaps the most (we calculate length of overlap divided by sum of lengths of both genes without overlap to score gene overlaps; this gives a value of 1 for a perfect match).

Having constructed one-to-one correspondence between predicted genes and real genes we can score our prediction the following way: for each overlapping gene we add overlap fraction values for each gene (calculated as described above) for all genes that overlap and subtract 1 for each false positive or false negative gene). This allows comparing different models and seeing which work best.

Such ranking system also presents us and *Assignment Problem* (AP) which can be solved in several ways:

1. Finding exact optimal solution - if we treat it as Min-Cost-Max-Flow problem;
2. Finding good approximation to exact solution - by means of Kuhn algorithm.

However since solving AP is beyond the scope of this assignment, we have solved this problem using a very greedy algorithm – choosing free pairs of overlapping genes in decreasing order of their overlap value.

# Model training

The first question we asked ourselves was “Do our models need any training at all”? They were built using accurately gathered statistics and should already be able to model E.Coli genome pretty well. While answering “no” may be correct for HMM with a simple intergenic model inside, it is definitely incorrect for the complex model because complex model was designed to estimate it’s parameters by means of training.

First thing that must be address is how exactly we should train our model. The easiest way would be to train as a single model. Unfortunately such approach is incorrect because training the model as whole means training it on whole genome and our models were not meant to mimic whole genome of E.Coli. In fact we have made clear assumptions (such as taking in account only traditional start/stop codons, traditional nucleotides), which prevent us from training our model on whole genome. Moreover E.Coli genome has some sequencing errors and uncertainties; regions with such errors must be either skipped or dealt with in some other way (randomly replacing uncertain nucleotide to one of its equivalents) – skipping prevents us from training on whole genome and random ‘correction’ would introduce unwanted bias into the model. Conclusion is that model should be trained by training its sub-models.

## Training sub-models

First of all we have to decide what sub-models should be trained. An obvious choice would be – genic and intergenic sub-models, since there is nothing to train in start and stop codon sub-models.

Next we have to decide how exactly we should train these sub-models. The scheme is going to be a bit different for genic and intergenic sub-models.

To train genic sub-model we took sub-model from Figure (4) and connected it to start and stop codon sub-models from Figures (2) and (3) just like we did for testing genic sub-model. Newly built model is an autonomous unit that generates genes statistically very close to those of E.Coli. This model should now be trained on sequences it aims to predict – genes with standard nucleotides, standard start/stop codons and length divisible by 3.

We have used same approach for training intergenic sub-models: extend model with stop and start codons, train it on intergenic regions starting with traditional stop codons and ending with traditional start codon of appropriate length (critical for long and short intergenic sub-models). If we do not extended the sub-model with stop and start codon sub-models some transition probabilities will turn into 1, moreover these sub-models are required for building whole model afterwards. (for Simple intergenic sub-model has been trained on 959 regions; short complex intergenic sub-model – on 70 regions and long complex intergenic sub-model – on 915 regions (some regions are used to train both sub-models).

For training our models we used Baum-Welch algorithm implemented in MATLAB. Since this algorithm maximizes probability of emitting given sequences it would always change transition probabilities from stop/start codon sub-models to genic and intergenic sub-models. Such behavior created numerous problems when trained sub-models were put together to constitute a final model – each sub-model had different transition and emission probabilities of start/stop codons. To resolve these conflicts we kept stop/start codon transition and emission probabilities from genic model (no matter if it was trained or not) and restored original transition probabilities for complex intergenic model (problems with transition probabilities arose there as short and long intergenic models were trained separately).

MATLAB implementation of Baum-Welch and Viterbi algorithms does not allow setting initial states easily. Instead it requires addition of a silent state with index 1 which has non-zero (and in our case equal) transition probabilities to actual start node. In our code this node has name 'entry\_point’. Starting state for genic sub-model is first state of start codon sub-model and starting state for intergenic sub-model is first state of stop codon sub-model.

# Implementation

This assignment required a lot of coding and maximum work automation, as we were working with bigger amounts of data, than a human being can handle on its own. It became clear to us very soon that a lot of experimentation will be required to produce a working HMM; we also found the structure of the HMM to be modular, as it can easily be split into separate sub-model (e.g. genic, intergenic, ‘dumb’ inergenic, short and long intergenic). These 2 properties suggested the use of objected-oriented approach to implementing all required functionality.

We have developed a set of classes and scripts in MATLAB in order to aid every step of HMM construction and application. Following table gives a quick description of every MATLAB file devised.

|  |  |
| --- | --- |
| File name | Description |
| HMM.m | Base class for HMM representation. |
| HMM\_Genic.m | Class representing genic sub-model of HMMs. |
| HMM\_Intergenic.m | Base class for intergenic sub-models of HMMs. |
| HMM\_Intergenic\_Dumb.m | Class representing simple intergenic model. |
| HMM\_Intergenic\_Short.m | Class representing short intergenic sub-model of complex intergenic sub-model. |
| HMM\_Intergenic\_Long.m | Class representing long intergenic sub-model of complex intergenic sub-model. |
| HMM\_Merge.m | Class used to merge two sub-models into a new model. |
| get\_bank\_file.m | Caching proxy-script for downloading data from gene bank. |
| load\_ecoli.m | Loads E.Coli genome. |
| split\_contig.m | Splits given sequence with annotated genes into training and testing contigs in given proportion (cuts sequence in an intergenic region). |
| get\_statistics.m | Script for gathering nucleotide and codon usage statistics in genic and intergenic regions. Used by HMM classes for automated model construction. |
| build\_hmm.m | Script that constructs various HMMs using HMM classes representation. |
| test\_hmm.m | Script for automated HMM testing. Reports a score for given HMM and testing contig. |
| test\_genic\_model.m | Assesses performance of genic sub-model. See “Testing genic sub-model” section. |
| my\_hmmviterbi.m | Modified version of Viterbi algorithm provided with MATLAB. |
| my\_hmmtrain.m | Modified version of Baum-Welch algorithm provided with MATLAB. |
| my\_hmmdecode.m | Modified version of algorithm scripts calculating forward and backward probabilities (used in MATLAB). |
| nucleotide\_stats\_report.m | Script generating nucleotide statistics for this report. |
| codon\_stats\_report.m | Script for automated generation of codon usage statistics and their representation in form of tables using CSV files. |
| codon\_stats\_report\_genic.m | Script for automated generation of codon usage statistics in genic regions and their representation in form of tables using CSV files. |
| start\_codon\_stats\_check.m | Reports statistics for start and stop codons that are in the middle of genes. |

Table : MATLAB m-file descriptions.

Full code description is not given, instead code written for purposes of this assignment has been extensively commented (each function has a description, lists all input and output arguments).

## Class hierarchy

All HMMs we have considered in this assignment are rather complex. In order to maintain our code readable and to avoid potential bugs we have developed a set of classes in MATLAB that incorporate all necessary functionality for construct and applying HMMs that finds genes in E.Coli. Potentially these classes can be modified to allow convenient work with more general HMMs. Figure (7) presents class hierarchy, description of every class can be found in Table (3) and detailed information of class functions can be found in comments in corresponding m-files.

Figure 8: HMM class hierarchy.

HMM Genic

HMM Merge

HMM

HMM Intergenic Dumb

HMM Intergenic Short

HMM Intergenic Long

HMM Intergenic

## Implementation tips

In this section we describe hard or unexpected situations, which arose in this assignment.

As mentioned above MATLAB implementation of Viterbi and Baum-Welch algorithms is too slow to be used in this assignment. For this reason we have introduced a couple of modifications to these algorithms.

### Viterbi algorithm

Computational complexity of Viterbi algorithm arises from the fact that it tries out all possible transitions between states. Such behavior can be justified for models which can be presented as dense graphs (models in which from every state a transition to most other states has non-zero probability); however in our case big parts of the model are linear (one state must always be followed by another) and average number of transitions from a state is around 3. This means that most of transitions tried out by the algorithm lead to 0 resulting probability and do not affect the resulting solution.

For this reason we have changed MATLAB implementation of Viterbi algorithm in such way, that it would only check states transition to which has non-zero probability. To do this we had to switch from adjacency matrix representation of HMM in Viterbi algorithm to adjacency list representation.

We have also introduced a heuristic to Viterbi algorithm which bans transitions to states which have zero probability of emitting coming symbol in the sequence.

Above modifications helped decreasing complexity of Viterbi algorithm to , where - length of sequence and – number of transitions between states (number of edges).

### Baum-Welch algorithm

Just like Viterbi algorithm implementation in MATLAB, implementation of Baum-Welch algorithm is too slow to be applied on such big models as in this assignment. The reason is that MATLAB provides implementation for dense models, while our models are sparse and require a different approach. We have introduced multiple changes to MATLAB’s HMM code in attempt to speed it up. All changes were directed towards changing the algorithm to work with adjacency lists instead of adjacency matrices. As result of our modifications we were able to reduce complexity of the algorithm to per iteration, where - number of transitions between states and – sum of lengths of all training sequences.

### Silent states

All models schematically described above used silent or help states which do not emit anything. Those were used for convenience only and must not be present in real model (definition of an HMM does not allow such nodes and algorithms will not give correct results while silent states are present).

Unfortunately constructing such big models as in this assignment without silent nodes can be extremely hard. For this reason we used a small trick – we constructed the model with silent nodes present and cut them out only when it was required. Pseudocode of an algorithm cutting these nodes out is given below.

|  |
| --- |
| **Input:** - transition matrix (adjacency matrix with transition probabilities); - emission matrix (values in a given row define emission probabilities for given state).   1. {dimensionality of square matrix } 2. {number of columns in matrix } 3. **for** **to** **do**    1. **for** **to** **do**    2. **end**    3. **if** **then** {found a silent node, cut out}       1. **for** **to** **do**          1. **for** **to** **do**             1. **if** **then**             2. **end**          2. **end**       2. **end**       3. {removes a row and a column with given index from given matrix}       4. {removes a row with given index from given matrix}    4. **end** 4. **end**   **Output:** transition matrix ; emission matrix . |

Algorithm : pseudocode for cutting out silent nodes from HMM.

Note that Algorithm (1) cuts out only a single silent node. In order to cut out all nodes it has to be modified or applied iteratively.

It is important to note that not all silent states must be cut out (sometimes silent states must be actually added). For correct functioning Viterbi and Baum-Welch algorithms require addition of silent start state (with index 1) with has non-zero transition probabilities to actual start nodes of HMM. This makes construction and separate training of sub-models a tedious task.

### Memory constraints

Memory limit is an important factor which influences design and application of algorithms. In our case it influenced testing of our models. Viterbi algorithm when applied to whole genome required more RAM than there was available on our computers. For this reason we had to split data given to Viterbi algorithm into separate contigs and test our models separately for each contig. Splitting of genome into contigs is done implicitly using recursion (in *test\_hmm* script).

# Results

We designed a number of HMMs to address the problem of finding genes and to see which one of them performs best we performed several experiments. For reasons mentioned before for genome testing input sequence was dynamically divided into separate contigs and results computer for each contig separately were then merged together. Experiment details are given in Table (4) and results – in Tables (5) and (6).

Testing was based on HMM Viterbi algorithm which requires definition of initial states. We have taken start state(s) of intergenic sub-model(s) as initial states. Hence we assumed that genome start with an intergenic region, not a gene. This is in fact true since algorithms we used to cut genome into contigs make sure that no genes are cut in the middle. Models were applied separately on training and testing contigs.

|  |  |  |  |
| --- | --- | --- | --- |
| # | Model | Trained genic | Trained intergenic |
| 1 | HMM with simple intergenic model | no | no |
| 2 | HMM with simple intergenic model | no | yes |
| 3 | HMM with simple simple intergenic model | yes | no |
| 4 | HMM with simple intergenic model | yes | yes |
| 5 | HMM with complex intergenic model | no | yes |
| 6 | HMM with complex intergenic model | yes | Yes |

Table : experiment description. Here # - id of experiments, Model - textual descritpion of model used, Trained (inter)genic - tells whether corresponding model has been trained. For experiment results see Tables (5), (6).

Note, that we have tried different combinations of trained and untrained model parts.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| # | Post-proc. | Perfect | Part 80% | Part 50% | Part 0% | False + | False - | Score |
| 1 | no | 395 | 228 | 79 | 341 | 1137 | 326 | -1460 |
| yes |  |  |  |  |  |  |  |
| 2 | no | 161 | 90 | 68 | 225 | 183 | 825 | -1006 |
| yes |  |  |  |  |  |  |  |
| 3 | no |  |  |  |  |  |  |  |
| yes |  |  |  |  |  |  |  |
| 4 | no |  |  |  |  |  |  |  |
| yes |  |  |  |  |  |  |  |
| 5 | no |  |  |  |  |  |  |  |
| yes |  |  |  |  |  |  |  |
| 6 | no |  |  |  |  |  |  |  |
| yes |  |  |  |  |  |  |  |

Table : model performance on training set per experiment. Here # - experiment ID; Perfect – number of perfectly detected genes, Part 80% - number of gene matches that have score bigger or equal to 0.8; Part 50% - number of gene matches that have score bigger or equal to 0.5; Part 0% number of gene matches that have score bigger than 0; False + - number of false positive genes found; False - - number of false negative genes (missed genes); Score – final score of the model (rounded to nearest integer to fit table).

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| # | Post-proc. | Perfect | Part 80% | Part 50% | Part 0% | False + | False - | Score |
| 1 | no | 981 | 549 | 325 | 1220 | 3213 | 1034 | -4244 |
| yes |  |  |  |  |  |  |  |
| 2 | no | 405 | 255 | 205 | 769 | 547 | 2475 | -3019 |
| yes |  |  |  |  |  |  |  |
| 3 | no |  |  |  |  |  |  |  |
| yes |  |  |  |  |  |  |  |
| 4 | no |  |  |  |  |  |  |  |
| yes |  |  |  |  |  |  |  |
| 5 | no |  |  |  |  |  |  |  |
| yes |  |  |  |  |  |  |  |
| 6 | no |  |  |  |  |  |  |  |
| yes |  |  |  |  |  |  |  |

Table : model performance on testing set per experiment. For description see Table (5).

# Conclusion

Do we conclude that experiments were a success?

# References

[1] A. Krogh, I. Saira Mian, D. Haussler, “A hidden Markov model that finds genes in E.Coli DNA”, Nucleic Acids Research, Vol. 22, pp. 4768-4778, 1994;

[2] L.R. Rabiner, “A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition”, Proceedings of the IEEE, Vol. 77, No. 2, pp 257-286, February 1989;

[3] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains", Ann. Math. Statist., vol. 41, no. 1, pp. 164–171, 1970.

# Appendix

In this section we mostly present various statistics which give a feel of the data we are working with and are required for HMM construction. The general rule is that everything that was too big or irrelevant to be placed in the text can be found here.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Training contig | | | Testing contig | | | Whole genome | | |
| Genic | Inter. | All | Genic | Inter. | All | Genic | Inter. | All |
| A | 243400,  0.2453 | 201902,  0.2524 | 693414,  0.2454 | 615101,  0.2476 | 358006,  0.2607 | 2042995,  0.2486 | 858501,  0.2470 | 559908,  0.2576 | 2736409,  0.2478 |
| C | 241273,  0.2432 | 204630,  0.2558 | 719432,  0.2546 | 595992,  0.2399 | 337426,  0.2457 | 2065963,  0.2514 | 837265,  0.2409 | 542056,  0.2494 | 2785395,  0.2522 |
| G | 272121,  0.2743 | 184797,  0.2310 | 719432,  0.2546 | 669116,  0.2694 | 309165,  0.2251 | 2065963,  0.2514 | 941237,  0.2708 | 493962,  0.2273 | 2785395,  0.2522 |
| T | 235333,  0.2372 | 208717,  0.2609 | 693414,  0.2454 | 603908,  0.2431 | 368842,  0.2686 | 2042995,  0.2486 | 839241,  0.2414 | 577559,  0.2657 | 2736409,  0.2478 |
| Total | 992127 | 800046 | 2825692 | 2484117 | 1373439 | 8217916 | 3476244 | 2173485 | 11043608 |

Table : nucleotide usage statistics for training and testing contings and for whole genome. Separate results are given for genic and intergenic parts of the contigs.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | Training contig | | | | | | Testing contig | | | | | |
|  |  | 1 | 2 | 3 | 4 | 5 | 6 | 1 | 2 | 3 | 4 | 5 | 6 |
| AAA | | 11238,  0.0239 | 11207,  0.0238 | 11174,  0.0237 | 10708,  0.0227 | 10912,  0.0232 | 10774,  0.0229 | 33051,  0.0241 | 32752,  0.0239 | 33194,  0.0242 | 33389,  0.0243 | 33604,  0.0245 | 33395,  0.0243 |
| AAC | | 8243,  0.0175 | 7890,  0.0167 | 7998,  0.0170 | 8299,  0.0176 | 8424,  0.0179 | 8278,  0.0176 | 24204,  0.0176 | 24496,  0.0179 | 24603,  0.0179 | 23675,  0.0173 | 23956,  0.0175 | 23916,  0.0174 |
| AAG | | 6557,  0.0139 | 6660,  0.0141 | 6591,  0.0140 | 5790,  0.0123 | 5925,  0.0126 | 5897,  0.0125 | 18830,  0.0137 | 18123,  0.0132 | 18263,  0.0133 | 18743,  0.0137 | 18966,  0.0138 | 19368,  0.0141 |
| AAT | | 8318,  0.0177 | 8053,  0.0171 | 8395,  0.0178 | 8437,  0.0179 | 8466,  0.0180 | 8555,  0.0182 | 25050,  0.0183 | 25317,  0.0185 | 25652,  0.0187 | 25226,  0.0184 | 25335,  0.0185 | 24987,  0.0182 |
| ACA | | 5733,  0.0122 | 6055,  0.0129 | 5691,  0.0121 | 6243,  0.0133 | 6395,  0.0136 | 6531,  0.0139 | 18059,  0.0132 | 18139,  0.0132 | 18161,  0.0132 | 17510,  0.0128 | 17235,  0.0126 | 17544,  0.0128 |
| ACC | | 7638,  0.0162 | 6903,  0.0147 | 7221,  0.0153 | 8062,  0.0171 | 8303,  0.0176 | 7964,  0.0169 | 22160,  0.0162 | 22203,  0.0162 | 22192,  0.0162 | 20993,  0.0153 | 21314,  0.0155 | 21057,  0.0154 |
| ACG | | 7372,  0.0156 | 7093,  0.0151 | 6931,  0.0147 | 7090,  0.0151 | 7730,  0.0164 | 7418,  0.0157 | 21305,  0.0155 | 21004,  0.0153 | 21494,  0.0157 | 20793,  0.0152 | 20691,  0.0151 | 21133,  0.0154 |
| ACT | | 4876,  0.0104 | 5022,  0.0107 | 4980,  0.0106 | 5215,  0.0111 | 5315,  0.0113 | 5075,  0.0108 | 14950,  0.0109 | 15106,  0.0110 | 15161,  0.0111 | 14684,  0.0107 | 14562,  0.0106 | 14897,  0.0109 |
| AGA | | 5896,  0.0125 | 5967,  0.0127 | 5758,  0.0122 | 5615,  0.0119 | 5498,  0.0117 | 5569,  0.0118 | 16943,  0.0124 | 16962,  0.0124 | 16685,  0.0122 | 17010,  0.0124 | 16880,  0.0123 | 16943,  0.0124 |
| AGC | | 8534,  0.0181 | 7651,  0.0162 | 7956,  0.0169 | 8075,  0.0171 | 8355,  0.0177 | 8180,  0.0174 | 23540,  0.0172 | 23532,  0.0172 | 23700,  0.0173 | 23353,  0.0170 | 22807,  0.0166 | 23153,  0.0169 |
| AGG | | 5715,  0.0121 | 5646,  0.0120 | 5598,  0.0119 | 4742,  0.0101 | 4840,  0.0103 | 5038,  0.0107 | 14832,  0.0108 | 14598,  0.0106 | 14470,  0.0105 | 15510,  0.0113 | 15406,  0.0112 | 15489,  0.0113 |
| AGT | | 5215,  0.0111 | 5075,  0.0108 | 5315,  0.0113 | 4876,  0.0104 | 4980,  0.0106 | 5022,  0.0107 | 14684,  0.0107 | 14897,  0.0109 | 14562,  0.0106 | 14950,  0.0109 | 15161,  0.0111 | 15106,  0.0110 |
| ATA | | 6591,  0.0140 | 6325,  0.0134 | 6354,  0.0135 | 6699,  0.0142 | 6757,  0.0143 | 6939,  0.0147 | 20232,  0.0147 | 19469,  0.0142 | 19559,  0.0143 | 19175,  0.0140 | 19465,  0.0142 | 19921,  0.0145 |
| ATC | | 8490,  0.0180 | 7991,  0.0170 | 8190,  0.0174 | 8797,  0.0187 | 9289,  0.0197 | 8828,  0.0187 | 25598,  0.0187 | 25583,  0.0187 | 26199,  0.0191 | 24560,  0.0179 | 25113,  0.0183 | 25028,  0.0182 |
| ATG | | 8486,  0.0180 | 8108,  0.0172 | 8132,  0.0173 | 7166,  0.0152 | 7558,  0.0160 | 7322,  0.0155 | 22521,  0.0164 | 22134,  0.0161 | 22652,  0.0165 | 23391,  0.0171 | 23349,  0.0170 | 23473,  0.0171 |
| ATT | | 8437,  0.0179 | 8555,  0.0182 | 8466,  0.0180 | 8318,  0.0177 | 8395,  0.0178 | 8053,  0.0171 | 25226,  0.0184 | 24987,  0.0182 | 25335,  0.0185 | 25050,  0.0183 | 25652,  0.0187 | 25317,  0.0185 |
| CAA | | 6906,  0.0147 | 7014,  0.0149 | 7321,  0.0155 | 7932,  0.0168 | 7695,  0.0163 | 7654,  0.0162 | 22630,  0.0165 | 22983,  0.0168 | 22520,  0.0164 | 22229,  0.0162 | 21853,  0.0159 | 21581,  0.0157 |
| CAC | | 6039,  0.0128 | 5979,  0.0127 | 6462,  0.0137 | 7983,  0.0169 | 7417,  0.0157 | 7471,  0.0159 | 20641,  0.0150 | 20273,  0.0148 | 20118,  0.0147 | 18494,  0.0135 | 18683,  0.0136 | 19237,  0.0140 |
| CAG | | 10568,  0.0224 | 10432,  0.0221 | 11309,  0.0240 | 11218,  0.0238 | 11274,  0.0239 | 10789,  0.0229 | 30956,  0.0226 | 31016,  0.0226 | 31414,  0.0229 | 30005,  0.0219 | 30766,  0.0224 | 30282,  0.0221 |
| CAT | | 7166,  0.0152 | 7322,  0.0155 | 7558,  0.0160 | 8486,  0.0180 | 8132,  0.0173 | 8108,  0.0172 | 23391,  0.0171 | 23473,  0.0171 | 23349,  0.0170 | 22521,  0.0164 | 22652,  0.0165 | 22134,  0.0161 |
| CCA | | 7727,  0.0164 | 8113,  0.0172 | 7611,  0.0162 | 9460,  0.0201 | 9447,  0.0201 | 9548,  0.0203 | 25490,  0.0186 | 25696,  0.0187 | 26070,  0.0190 | 24052,  0.0175 | 24321,  0.0177 | 23750,  0.0173 |
| CCC | | 4374,  0.0093 | 4388,  0.0093 | 4236,  0.0090 | 5842,  0.0124 | 5999,  0.0127 | 5847,  0.0124 | 15022,  0.0110 | 15349,  0.0112 | 15233,  0.0111 | 13597,  0.0099 | 13465,  0.0098 | 13653,  0.0100 |
| CCG | | 8868,  0.0188 | 9098,  0.0193 | 8779,  0.0186 | 9328,  0.0198 | 9398,  0.0199 | 9571,  0.0203 | 25442,  0.0185 | 25426,  0.0185 | 25888,  0.0189 | 25252,  0.0184 | 24716,  0.0180 | 25073,  0.0183 |
| CCT | | 4742,  0.0101 | 5038,  0.0107 | 4840,  0.0103 | 5715,  0.0121 | 5598,  0.0119 | 5646,  0.0120 | 15510,  0.0113 | 15489,  0.0113 | 15406,  0.0112 | 14832,  0.0108 | 14470,  0.0105 | 14598,  0.0106 |
| CGA | | 6962,  0.0148 | 7014,  0.0149 | 7054,  0.0150 | 6735,  0.0143 | 6420,  0.0136 | 6827,  0.0145 | 20075,  0.0146 | 20146,  0.0147 | 19573,  0.0143 | 20491,  0.0149 | 20232,  0.0147 | 20177,  0.0147 |
| CGC | | 10300,  0.0219 | 10662,  0.0226 | 10738,  0.0228 | 11687,  0.0248 | 11545,  0.0245 | 11899,  0.0253 | 33105,  0.0241 | 32305,  0.0236 | 32986,  0.0240 | 31432,  0.0229 | 31605,  0.0230 | 31366,  0.0229 |
| CGG | | 9328,  0.0198 | 9571,  0.0203 | 9398,  0.0199 | 8868,  0.0188 | 8779,  0.0186 | 9098,  0.0193 | 25252,  0.0184 | 25073,  0.0183 | 24716,  0.0180 | 25442,  0.0185 | 25888,  0.0189 | 25426,  0.0185 |
| CGT | | 7090,  0.0151 | 7418,  0.0157 | 7730,  0.0164 | 7372,  0.0156 | 6931,  0.0147 | 7093,  0.0151 | 20793,  0.0152 | 21133,  0.0154 | 20691,  0.0151 | 21305,  0.0155 | 21494,  0.0157 | 21004,  0.0153 |
| CTA | | 2634,  0.0056 | 2546,  0.0054 | 2453,  0.0052 | 2551,  0.0054 | 2532,  0.0054 | 2750,  0.0058 | 8096,  0.0059 | 8053,  0.0059 | 7794,  0.0057 | 7993,  0.0058 | 7785,  0.0057 | 7823,  0.0057 |
| CTC | | 4029,  0.0086 | 4080,  0.0087 | 4154,  0.0088 | 4661,  0.0099 | 4928,  0.0105 | 4787,  0.0102 | 13417,  0.0098 | 13119,  0.0096 | 13159,  0.0096 | 12223,  0.0089 | 12537,  0.0091 | 12465,  0.0091 |
| CTG | | 11218,  0.0238 | 10789,  0.0229 | 11274,  0.0239 | 10568,  0.0224 | 11309,  0.0240 | 10432,  0.0221 | 30005,  0.0219 | 30282,  0.0221 | 30766,  0.0224 | 30956,  0.0226 | 31414,  0.0229 | 31016,  0.0226 |
| CTT | | 5790,  0.0123 | 5897,  0.0125 | 5925,  0.0126 | 6557,  0.0139 | 6591,  0.0140 | 6660,  0.0141 | 18743,  0.0137 | 19368,  0.0141 | 18966,  0.0138 | 18830,  0.0137 | 18263,  0.0133 | 18123,  0.0132 |
| GAA | | 8782,  0.0186 | 8980,  0.0191 | 9110,  0.0193 | 8108,  0.0172 | 8101,  0.0172 | 7810,  0.0166 | 24226,  0.0177 | 24560,  0.0179 | 25018,  0.0182 | 25325,  0.0185 | 25545,  0.0186 | 25184,  0.0184 |
| GAC | | 5691,  0.0121 | 5737,  0.0122 | 5883,  0.0125 | 5804,  0.0123 | 5520,  0.0117 | 5408,  0.0115 | 15843,  0.0115 | 16534,  0.0121 | 15972,  0.0116 | 16027,  0.0117 | 16309,  0.0119 | 15781,  0.0115 |
| GAG | | 4661,  0.0099 | 4787,  0.0102 | 4928,  0.0105 | 4029,  0.0086 | 4154,  0.0088 | 4080,  0.0087 | 12223,  0.0089 | 12465,  0.0091 | 12537,  0.0091 | 13417,  0.0098 | 13159,  0.0096 | 13119,  0.0096 |
| GAT | | 8797,  0.0187 | 8828,  0.0187 | 9289,  0.0197 | 8490,  0.0180 | 8190,  0.0174 | 7991,  0.0170 | 24560,  0.0179 | 25028,  0.0182 | 25113,  0.0183 | 25598,  0.0187 | 26199,  0.0191 | 25583,  0.0187 |
| GCA | | 9306,  0.0198 | 9916,  0.0210 | 9457,  0.0201 | 9689,  0.0206 | 9453,  0.0201 | 9896,  0.0210 | 28067,  0.0205 | 27952,  0.0204 | 27971,  0.0204 | 27692,  0.0202 | 27438,  0.0200 | 27312,  0.0199 |
| GCC | | 8978,  0.0191 | 8707,  0.0185 | 8797,  0.0187 | 10183,  0.0216 | 9906,  0.0210 | 9947,  0.0211 | 27209,  0.0198 | 27361,  0.0199 | 26899,  0.0196 | 25503,  0.0186 | 25958,  0.0189 | 26131,  0.0190 |
| GCG | | 11687,  0.0248 | 11899,  0.0253 | 11545,  0.0245 | 10300,  0.0219 | 10738,  0.0228 | 10662,  0.0226 | 31432,  0.0229 | 31366,  0.0229 | 31605,  0.0230 | 33105,  0.0241 | 32986,  0.0240 | 32305,  0.0236 |
| GCT | | 8075,  0.0171 | 8180,  0.0174 | 8355,  0.0177 | 8534,  0.0181 | 7956,  0.0169 | 7651,  0.0162 | 23353,  0.0170 | 23153,  0.0169 | 22807,  0.0166 | 23540,  0.0172 | 23700,  0.0173 | 23532,  0.0172 |
| GGA | | 6351,  0.0135 | 6585,  0.0140 | 6391,  0.0136 | 5645,  0.0120 | 5460,  0.0116 | 5469,  0.0116 | 16866,  0.0123 | 16890,  0.0123 | 16349,  0.0119 | 17570,  0.0128 | 17149,  0.0125 | 17674,  0.0129 |
| GGC | | 10183,  0.0216 | 9947,  0.0211 | 9906,  0.0210 | 8978,  0.0191 | 8797,  0.0187 | 8707,  0.0185 | 25503,  0.0186 | 26131,  0.0190 | 25958,  0.0189 | 27209,  0.0198 | 26899,  0.0196 | 27361,  0.0199 |
| GGG | | 5842,  0.0124 | 5847,  0.0124 | 5999,  0.0127 | 4374,  0.0093 | 4236,  0.0090 | 4388,  0.0093 | 13597,  0.0099 | 13653,  0.0100 | 13465,  0.0098 | 15022,  0.0110 | 15233,  0.0111 | 15349,  0.0112 |
| GGT | | 8062,  0.0171 | 7964,  0.0169 | 8303,  0.0176 | 7638,  0.0162 | 7221,  0.0153 | 6903,  0.0147 | 20993,  0.0153 | 21057,  0.0154 | 21314,  0.0155 | 22160,  0.0162 | 22192,  0.0162 | 22203,  0.0162 |
| GTA | | 5649,  0.0120 | 5451,  0.0116 | 5623,  0.0119 | 5102,  0.0108 | 5279,  0.0112 | 5216,  0.0111 | 15617,  0.0114 | 15051,  0.0110 | 15699,  0.0114 | 15780,  0.0115 | 15809,  0.0115 | 15733,  0.0115 |
| GTC | | 5804,  0.0123 | 5408,  0.0115 | 5520,  0.0117 | 5691,  0.0121 | 5883,  0.0125 | 5737,  0.0122 | 16027,  0.0117 | 15781,  0.0115 | 16309,  0.0119 | 15843,  0.0115 | 15972,  0.0116 | 16534,  0.0121 |
| GTG | | 7983,  0.0169 | 7471,  0.0159 | 7417,  0.0157 | 6039,  0.0128 | 6462,  0.0137 | 5979,  0.0127 | 18494,  0.0135 | 19237,  0.0140 | 18683,  0.0136 | 20641,  0.0150 | 20118,  0.0147 | 20273,  0.0148 |
| GTT | | 8299,  0.0176 | 8278,  0.0176 | 8424,  0.0179 | 8243,  0.0175 | 7998,  0.0170 | 7890,  0.0167 | 23675,  0.0173 | 23916,  0.0174 | 23956,  0.0175 | 24204,  0.0176 | 24603,  0.0179 | 24496,  0.0179 |
| TAA | | 6881,  0.0146 | 6957,  0.0148 | 6749,  0.0143 | 6987,  0.0148 | 6803,  0.0144 | 6995,  0.0148 | 20794,  0.0152 | 21413,  0.0156 | 20415,  0.0149 | 20941,  0.0153 | 20677,  0.0151 | 20910,  0.0152 |
| TAC | | 5102,  0.0108 | 5216,  0.0111 | 5279,  0.0112 | 5649,  0.0120 | 5623,  0.0119 | 5451,  0.0116 | 15780,  0.0115 | 15733,  0.0115 | 15809,  0.0115 | 15617,  0.0114 | 15699,  0.0114 | 15051,  0.0110 |
| TAG | | 2551,  0.0054 | 2750,  0.0058 | 2532,  0.0054 | 2634,  0.0056 | 2453,  0.0052 | 2546,  0.0054 | 7993,  0.0058 | 7823,  0.0057 | 7785,  0.0057 | 8096,  0.0059 | 7794,  0.0057 | 8053,  0.0059 |
| TAT | | 6699,  0.0142 | 6939,  0.0147 | 6757,  0.0143 | 6591,  0.0140 | 6354,  0.0135 | 6325,  0.0134 | 19175,  0.0140 | 19921,  0.0145 | 19465,  0.0142 | 20232,  0.0147 | 19559,  0.0143 | 19469,  0.0142 |
| TCA | | 7986,  0.0170 | 8569,  0.0182 | 7926,  0.0168 | 9128,  0.0194 | 8735,  0.0185 | 9649,  0.0205 | 26169,  0.0191 | 25638,  0.0187 | 25440,  0.0185 | 24738,  0.0180 | 24261,  0.0177 | 24660,  0.0180 |
| TCC | | 5645,  0.0120 | 5469,  0.0116 | 5460,  0.0116 | 6351,  0.0135 | 6391,  0.0136 | 6585,  0.0140 | 17570,  0.0128 | 17674,  0.0129 | 17149,  0.0125 | 16866,  0.0123 | 16349,  0.0119 | 16890,  0.0123 |
| TCG | | 6735,  0.0143 | 6827,  0.0145 | 6420,  0.0136 | 6962,  0.0148 | 7054,  0.0150 | 7014,  0.0149 | 20491,  0.0149 | 20177,  0.0147 | 20232,  0.0147 | 20075,  0.0146 | 19573,  0.0143 | 20146,  0.0147 |
| TCT | | 5615,  0.0119 | 5569,  0.0118 | 5498,  0.0117 | 5896,  0.0125 | 5758,  0.0122 | 5967,  0.0127 | 17010,  0.0124 | 16943,  0.0124 | 16880,  0.0123 | 16943,  0.0124 | 16685,  0.0122 | 16962,  0.0124 |
| TGA | | 9128,  0.0194 | 9649,  0.0205 | 8735,  0.0185 | 7986,  0.0170 | 7926,  0.0168 | 8569,  0.0182 | 24738,  0.0180 | 24660,  0.0180 | 24261,  0.0177 | 26169,  0.0191 | 25440,  0.0185 | 25638,  0.0187 |
| TGC | | 9689,  0.0206 | 9896,  0.0210 | 9453,  0.0201 | 9306,  0.0198 | 9457,  0.0201 | 9916,  0.0210 | 27692,  0.0202 | 27312,  0.0199 | 27438,  0.0200 | 28067,  0.0205 | 27971,  0.0204 | 27952,  0.0204 |
| TGG | | 9460,  0.0201 | 9548,  0.0203 | 9447,  0.0201 | 7727,  0.0164 | 7611,  0.0162 | 8113,  0.0172 | 24052,  0.0175 | 23750,  0.0173 | 24321,  0.0177 | 25490,  0.0186 | 26070,  0.0190 | 25696,  0.0187 |
| TGT | | 6243,  0.0133 | 6531,  0.0139 | 6395,  0.0136 | 5733,  0.0122 | 5691,  0.0121 | 6055,  0.0129 | 17510,  0.0128 | 17544,  0.0128 | 17235,  0.0126 | 18059,  0.0132 | 18161,  0.0132 | 18139,  0.0132 |
| TTA | | 6987,  0.0148 | 6995,  0.0148 | 6803,  0.0144 | 6881,  0.0146 | 6749,  0.0143 | 6957,  0.0148 | 20941,  0.0153 | 20910,  0.0152 | 20677,  0.0151 | 20794,  0.0152 | 20415,  0.0149 | 21413,  0.0156 |
| TTC | | 8108,  0.0172 | 7810,  0.0166 | 8101,  0.0172 | 8782,  0.0186 | 9110,  0.0193 | 8980,  0.0191 | 25325,  0.0185 | 25184,  0.0184 | 25545,  0.0186 | 24226,  0.0177 | 25018,  0.0182 | 24560,  0.0179 |
| TTG | | 7932,  0.0168 | 7654,  0.0162 | 7695,  0.0163 | 6906,  0.0147 | 7321,  0.0155 | 7014,  0.0149 | 22229,  0.0162 | 21581,  0.0157 | 21853,  0.0159 | 22630,  0.0165 | 22520,  0.0164 | 22983,  0.0168 |
| TTT | | 10708,  0.0227 | 10774,  0.0229 | 10912,  0.0232 | 11238,  0.0239 | 11174,  0.0237 | 11207,  0.0238 | 33389,  0.0243 | 33395,  0.0243 | 33604,  0.0245 | 33051,  0.0241 | 33194,  0.0242 | 32752,  0.0239 |
| Others | | 382,  0.0008 | 380,  0.0008 | 379,  0.0008 | 382,  0.0008 | 379,  0.0008 | 380,  0.0008 | 3438,  0.0025 | 3454,  0.0025 | 3438,  0.0025 | 3438,  0.0025 | 3438,  0.0025 | 3454,  0.0025 |
| Total | | 471081 | 471080 | 471080 | 471081 | 471080 | 471080 | 1371734 | 1371733 | 1371733 | 1371734 | 1371733 | 1371733 |

Table : codon usage statistics for training and testing contigs. Statistics are given separately for each reading frame: fames 1-3 are in normal direction and frames 4-6 are in reverse direction on complementary strand.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Whole genome | | | | | | | | | | | | |
|  | 1 | 2 | 3 | 4 | 5 | 6 |  | 1 | 2 | 3 | 4 | 5 | 6 |
| AAA | 44289,  0.0240 | 43959,  0.0239 | 44368,  0.0241 | 44097,  0.0239 | 44516,  0.0242 | 44169,  0.0240 | GAC | 21534,  0.0117 | 22271,  0.0121 | 21855,  0.0119 | 21831,  0.0118 | 21829,  0.0118 | 21189,  0.0115 |
| AAC | 32447,  0.0176 | 32386,  0.0176 | 32601,  0.0177 | 31974,  0.0174 | 32380,  0.0176 | 32194,  0.0175 | GAG | 16884,  0.0092 | 17252,  0.0094 | 17465,  0.0095 | 17446,  0.0095 | 17313,  0.0094 | 17199,  0.0093 |
| AAG | 25387,  0.0138 | 24783,  0.0134 | 24854,  0.0135 | 24533,  0.0133 | 24891,  0.0135 | 25265,  0.0137 | GAT | 33357,  0.0181 | 33856,  0.0184 | 34402,  0.0187 | 34088,  0.0185 | 34389,  0.0187 | 33574,  0.0182 |
| AAT | 33368,  0.0181 | 33370,  0.0181 | 34047,  0.0185 | 33663,  0.0183 | 33802,  0.0183 | 33542,  0.0182 | GCA | 37373,  0.0203 | 37868,  0.0205 | 37428,  0.0203 | 37381,  0.0203 | 36891,  0.0200 | 37208,  0.0202 |
| ACA | 23792,  0.0129 | 24194,  0.0131 | 23852,  0.0129 | 23753,  0.0129 | 23630,  0.0128 | 24075,  0.0131 | GCC | 36187,  0.0196 | 36068,  0.0196 | 35696,  0.0194 | 35686,  0.0194 | 35864,  0.0195 | 36078,  0.0196 |
| ACC | 29798,  0.0162 | 29106,  0.0158 | 29413,  0.0160 | 29055,  0.0158 | 29617,  0.0161 | 29021,  0.0157 | GCG | 43119,  0.0234 | 43265,  0.0235 | 43150,  0.0234 | 43405,  0.0236 | 43724,  0.0237 | 42967,  0.0233 |
| ACG | 28677,  0.0156 | 28097,  0.0152 | 28425,  0.0154 | 27883,  0.0151 | 28421,  0.0154 | 28551,  0.0155 | GCT | 31428,  0.0171 | 31333,  0.0170 | 31162,  0.0169 | 32074,  0.0174 | 31656,  0.0172 | 31183,  0.0169 |
| ACT | 19826,  0.0108 | 20128,  0.0109 | 20141,  0.0109 | 19899,  0.0108 | 19877,  0.0108 | 19972,  0.0108 | GGA | 23217,  0.0126 | 23475,  0.0127 | 22740,  0.0123 | 23215,  0.0126 | 22609,  0.0123 | 23143,  0.0126 |
| AGA | 22839,  0.0124 | 22929,  0.0124 | 22443,  0.0122 | 22625,  0.0123 | 22378,  0.0121 | 22512,  0.0122 | GGC | 35686,  0.0194 | 36078,  0.0196 | 35864,  0.0195 | 36187,  0.0196 | 35696,  0.0194 | 36068,  0.0196 |
| AGC | 32074,  0.0174 | 31183,  0.0169 | 31656,  0.0172 | 31428,  0.0171 | 31162,  0.0169 | 31333,  0.0170 | GGG | 19439,  0.0105 | 19500,  0.0106 | 19464,  0.0106 | 19396,  0.0105 | 19469,  0.0106 | 19737,  0.0107 |
| AGG | 20547,  0.0111 | 20244,  0.0110 | 20068,  0.0109 | 20252,  0.0110 | 20246,  0.0110 | 20527,  0.0111 | GGT | 29055,  0.0158 | 29021,  0.0157 | 29617,  0.0161 | 29798,  0.0162 | 29413,  0.0160 | 29106,  0.0158 |
| AGT | 19899,  0.0108 | 19972,  0.0108 | 19877,  0.0108 | 19826,  0.0108 | 20141,  0.0109 | 20128,  0.0109 | GTA | 21266,  0.0115 | 20502,  0.0111 | 21322,  0.0116 | 20882,  0.0113 | 21088,  0.0114 | 20949,  0.0114 |
| ATA | 26823,  0.0146 | 25794,  0.0140 | 25913,  0.0141 | 25874,  0.0140 | 26222,  0.0142 | 26861,  0.0146 | GTC | 21831,  0.0118 | 21189,  0.0115 | 21829,  0.0118 | 21534,  0.0117 | 21855,  0.0119 | 22271,  0.0121 |
| ATC | 34088,  0.0185 | 33574,  0.0182 | 34389,  0.0187 | 33357,  0.0181 | 34402,  0.0187 | 33856,  0.0184 | GTG | 26477,  0.0144 | 26708,  0.0145 | 26100,  0.0142 | 26680,  0.0145 | 26580,  0.0144 | 26252,  0.0142 |
| ATG | 31007,  0.0168 | 30242,  0.0164 | 30784,  0.0167 | 30557,  0.0166 | 30907,  0.0168 | 30795,  0.0167 | GTT | 31974,  0.0174 | 32194,  0.0175 | 32380,  0.0176 | 32447,  0.0176 | 32601,  0.0177 | 32386,  0.0176 |
| ATT | 33663,  0.0183 | 33542,  0.0182 | 33802,  0.0183 | 33368,  0.0181 | 34047,  0.0185 | 33370,  0.0181 | TAA | 27675,  0.0150 | 28370,  0.0154 | 27164,  0.0147 | 27928,  0.0152 | 27480,  0.0149 | 27905,  0.0151 |
| CAA | 29536,  0.0160 | 29997,  0.0163 | 29841,  0.0162 | 30161,  0.0164 | 29548,  0.0160 | 29235,  0.0159 | TAC | 20882,  0.0113 | 20949,  0.0114 | 21088,  0.0114 | 21266,  0.0115 | 21322,  0.0116 | 20502,  0.0111 |
| CAC | 26680,  0.0145 | 26252,  0.0142 | 26580,  0.0144 | 26477,  0.0144 | 26100,  0.0142 | 26708,  0.0145 | TAG | 10544,  0.0057 | 10573,  0.0057 | 10317,  0.0056 | 10730,  0.0058 | 10247,  0.0056 | 10599,  0.0058 |
| CAG | 41524,  0.0225 | 41448,  0.0225 | 42723,  0.0232 | 41223,  0.0224 | 42040,  0.0228 | 41071,  0.0223 | TAT | 25874,  0.0140 | 26861,  0.0146 | 26222,  0.0142 | 26823,  0.0146 | 25913,  0.0141 | 25794,  0.0140 |
| CAT | 30557,  0.0166 | 30795,  0.0167 | 30907,  0.0168 | 31007,  0.0168 | 30784,  0.0167 | 30242,  0.0164 | TCA | 34155,  0.0185 | 34207,  0.0186 | 33366,  0.0181 | 33866,  0.0184 | 32996,  0.0179 | 34309,  0.0186 |
| CCA | 33217,  0.0180 | 33809,  0.0183 | 33681,  0.0183 | 33512,  0.0182 | 33768,  0.0183 | 33298,  0.0181 | TCC | 23215,  0.0126 | 23143,  0.0126 | 22609,  0.0123 | 23217,  0.0126 | 22740,  0.0123 | 23475,  0.0127 |
| CCC | 19396,  0.0105 | 19737,  0.0107 | 19469,  0.0106 | 19439,  0.0105 | 19464,  0.0106 | 19500,  0.0106 | TCG | 27226,  0.0148 | 27004,  0.0147 | 26652,  0.0145 | 27037,  0.0147 | 26627,  0.0144 | 27160,  0.0147 |
| CCG | 34310,  0.0186 | 34524,  0.0187 | 34667,  0.0188 | 34580,  0.0188 | 34114,  0.0185 | 34644,  0.0188 | TCT | 22625,  0.0123 | 22512,  0.0122 | 22378,  0.0121 | 22839,  0.0124 | 22443,  0.0122 | 22929,  0.0124 |
| CCT | 20252,  0.0110 | 20527,  0.0111 | 20246,  0.0110 | 20547,  0.0111 | 20068,  0.0109 | 20244,  0.0110 | TGA | 33866,  0.0184 | 34309,  0.0186 | 32996,  0.0179 | 34155,  0.0185 | 33366,  0.0181 | 34207,  0.0186 |
| CGA | 27037,  0.0147 | 27160,  0.0147 | 26627,  0.0144 | 27226,  0.0148 | 26652,  0.0145 | 27004,  0.0147 | TGC | 37381,  0.0203 | 37208,  0.0202 | 36891,  0.0200 | 37373,  0.0203 | 37428,  0.0203 | 37868,  0.0205 |
| CGC | 43405,  0.0236 | 42967,  0.0233 | 43724,  0.0237 | 43119,  0.0234 | 43150,  0.0234 | 43265,  0.0235 | TGG | 33512,  0.0182 | 33298,  0.0181 | 33768,  0.0183 | 33217,  0.0180 | 33681,  0.0183 | 33809,  0.0183 |
| CGG | 34580,  0.0188 | 34644,  0.0188 | 34114,  0.0185 | 34310,  0.0186 | 34667,  0.0188 | 34524,  0.0187 | TGT | 23753,  0.0129 | 24075,  0.0131 | 23630,  0.0128 | 23792,  0.0129 | 23852,  0.0129 | 24194,  0.0131 |
| CGT | 27883,  0.0151 | 28551,  0.0155 | 28421,  0.0154 | 28677,  0.0156 | 28425,  0.0154 | 28097,  0.0152 | TTA | 27928,  0.0152 | 27905,  0.0151 | 27480,  0.0149 | 27675,  0.0150 | 27164,  0.0147 | 28370,  0.0154 |
| CTA | 10730,  0.0058 | 10599,  0.0058 | 10247,  0.0056 | 10544,  0.0057 | 10317,  0.0056 | 10573,  0.0057 | TTC | 33433,  0.0181 | 32994,  0.0179 | 33646,  0.0183 | 33008,  0.0179 | 34128,  0.0185 | 33540,  0.0182 |
| CTC | 17446,  0.0095 | 17199,  0.0093 | 17313,  0.0094 | 16884,  0.0092 | 17465,  0.0095 | 17252,  0.0094 | TTG | 30161,  0.0164 | 29235,  0.0159 | 29548,  0.0160 | 29536,  0.0160 | 29841,  0.0162 | 29997,  0.0163 |
| CTG | 41223,  0.0224 | 41071,  0.0223 | 42040,  0.0228 | 41524,  0.0225 | 42723,  0.0232 | 41448,  0.0225 | TTT | 44097,  0.0239 | 44169,  0.0240 | 44516,  0.0242 | 44289,  0.0240 | 44368,  0.0241 | 43959,  0.0239 |
| CTT | 24533,  0.0133 | 25265,  0.0137 | 24891,  0.0135 | 25387,  0.0138 | 24854,  0.0135 | 24783,  0.0134 | Others | 3820,  0.0021 | 3834,  0.0021 | 3817,  0.0021 | 3820,  0.0021 | 3817,  0.0021 | 3834,  0.0021 |
| GAA | 33008,  0.0179 | 33540,  0.0182 | 34128,  0.0185 | 33433,  0.0181 | 33646,  0.0183 | 32994,  0.0179 | Total | 1842815 | 1842814 | 1842814 | 1842815 | 1842814 | 1842814 |

Table : codon usage statistics for whole genome. Statistics are given separately for each reading frame: fames 1-3 are in normal direction and frames 4-6 are in reverse direction on complementary strand.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Train | Test |  | Train | Test |  | Train | Test |  | Train | Test |
| AAA | 11357,  0.0343 | 29209,  0.0353 | CAA | 4727,  0.0143 | 12663,  0.0153 | GAA | 13054,  0.0395 | 33163,  0.0401 | TAA | 0,  0.0000 | 0,  0.0000 |
| AAC | 7184,  0.0217 | 17293,  0.0209 | CAC | 3145,  0.0095 | 7690,  0.0093 | GAC | 6622,  0.0200 | 15309,  0.0185 | TAC | 4142,  0.0125 | 9939,  0.0120 |
| AAG | 3866,  0.0117 | 9376,  0.0113 | CAG | 10031,  0.0303 | 23672,  0.0286 | GAG | 6453,  0.0195 | 15650,  0.0189 | TAG | 0,  0.0000 | 0,  0.0000 |
| AAT | 6274,  0.0190 | 16636,  0.0201 | CAT | 4223,  0.0128 | 10997,  0.0133 | GAT | 10752,  0.0325 | 27560,  0.0333 | TAT | 5476,  0.0166 | 14224,  0.0172 |
| ACA | 2780,  0.0084 | 7008,  0.0085 | CCA | 2765,  0.0084 | 7209,  0.0087 | GCA | 6941,  0.0210 | 17369,  0.0210 | TCA | 2702,  0.0082 | 7146,  0.0086 |
| ACC | 7634,  0.0231 | 18023,  0.0218 | CCC | 1933,  0.0058 | 4709,  0.0057 | GCC | 8227,  0.0249 | 20382,  0.0246 | TCC | 2982,  0.0090 | 7247,  0.0088 |
| ACG | 4879,  0.0148 | 12148,  0.0147 | CCG | 7466,  0.0226 | 17663,  0.0213 | GCG | 10601,  0.0321 | 26007,  0.0314 | TCG | 2861,  0.0087 | 7242,  0.0087 |
| ACT | 2909,  0.0088 | 7703,  0.0093 | CCT | 2409,  0.0073 | 6186,  0.0075 | GCT | 5222,  0.0158 | 12961,  0.0157 | TCT | 2823,  0.0085 | 7429,  0.0090 |
| AGA | 1132,  0.0034 | 2657,  0.0032 | CGA | 1334,  0.0040 | 3395,  0.0041 | GGA | 3123,  0.0094 | 7548,  0.0091 | TGA | 0,  0.0000 | 2,  0.0000 |
| AGC | 5408,  0.0164 | 12915,  0.0156 | CGC | 6733,  0.0204 | 17069,  0.0206 | GGC | 9313,  0.0282 | 22500,  0.0272 | TGC | 2055,  0.0062 | 5550,  0.0067 |
| AGG | 703,  0.0021 | 1581,  0.0019 | CGG | 2241,  0.0068 | 5129,  0.0062 | GGG | 4016,  0.0121 | 9521,  0.0115 | TGG | 5142,  0.0155 | 12599,  0.0152 |
| AGT | 3023,  0.0091 | 8104,  0.0098 | CGT | 6559,  0.0198 | 16774,  0.0203 | GGT | 7822,  0.0237 | 19856,  0.0240 | TGT | 1740,  0.0053 | 4654,  0.0056 |
| ATA | 1899,  0.0057 | 5154,  0.0062 | CTA | 1247,  0.0038 | 3576,  0.0043 | GTA | 3719,  0.0112 | 9184,  0.0111 | TTA | 4410,  0.0133 | 12550,  0.0152 |
| ATC | 7878,  0.0238 | 19902,  0.0240 | CTC | 3459,  0.0105 | 8846,  0.0107 | GTC | 4929,  0.0149 | 12288,  0.0148 | TTC | 5198,  0.0157 | 13159,  0.0159 |
| ATG | 8175,  0.0247 | 20541,  0.0248 | CTG | 17193,  0.0520 | 41168,  0.0497 | GTG | 8667,  0.0262 | 20557,  0.0248 | TTG | 4067,  0.0123 | 11098,  0.0134 |
| ATT | 9803,  0.0296 | 25637,  0.0310 | CTT | 3832,  0.0116 | 9993,  0.0121 | GTT | 5955,  0.0180 | 15455,  0.0187 | TTT | 7494,  0.0227 | 19264,  0.0233 |
|  |  |  |  |  |  |  |  |  | Total | 330709 | 828039 |

Table : codon usage statistic in genic regions of training and test contigs.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Training contig | | | | | | | | Testing contig | | | | | | | |
| Start codons | | | | Stop codons | | | | Start codons | | | | Stop codons | | | |
| ATG | GTG | TTG | Total | TAA | TGA | TAG | Total | ATG | GTG | TTG | Total | TAA | TGA | TAG | Total |
| 987,  0.8494 | 152,  0.1308 | 23,  0.0198 | 1162 | 674,  0.5800 | 388,  0.3339 | 100,  0.0861 | 1162 | 2541,  0.8168 | 496,  0.1594 | 74,  0.0238 | 3111 | 1868,  0.6005 | 980,  0.3150 | 263,  0.0845 | 3111 |

Table : start and stop codon frequencies for training and testing contigs.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Whole genome | | | | | | | |
| Start codons | | | | Stop codons | | | |
| ATG | GTG | TTG | Total | TAA | TGA | TAG | Total |
| 3528,  0.8256 | 648,  0.1516 | 97,  0.0227 | 4273 | 2542,  0.5949 | 1368,  0.3201 | 363,  0.0850 | 4273 |

Table : start and stop codon frequencies for whole genome.

Figure 9: general structure of complex intergenic model. The length of model is adjustable. See Figures (1) and (2) for description.

End

A

G

C

T

A

G

C

T

A

G

C

T

A

G

C

T

A

G

C

T

A

G

C

T

A

G

C

T

A

G

C

T

Begin

A

G

C

T