Brovine: Mammary Gland Gene Database

Therin C. Irwin
tcirwin@calpoly.edu
Cal Poly, San Luis Obispo

June 21, 2013

Abstract

Brovine is used by the Animal Science department at Cal Poly to catalog and analyze genetic information. Brovine, or the Mammary Gland Gene Database, is a system used to store and categorize genetic information which is gathered through experimentation and through TESS, a web application that lets users search through catalogs of similar genetic information. This document describes the purpose, use, and maintenance of Brovine.

Contents

Acknowledgements

I would like to thank Dr. Alexander Dekhtyar for advising this project, Dr. Dan Peterson for making the project possible, and Sterling Hirsh, Trevor DeVore, and Ryan Schroeder for working on the first iteration of Brovine.

1 Biological Overview

To give a brief overview of the scope of the problem the researchers are attempting to solve, we first discuss the biology of the processes they are studying. Physiological function in multicellular organisms requires different cells to have different specific functions and characteristics, and the attainment of these specialized characteristics is called differentiation. Differentiation is achieved in part when proteins, called "transcription factors" interact with regions of DNA called "promoter regions." When transcription factors bind to the promoter regions of DNA, the transcription of the gene is "activated," which means that the gene is expressed in the cell. Differentiation involves a myriad of regulatory events that lead to structural and functional organization of the genome to allow expression and regulation of the appropriate set of proteins for the specialized functions of that cell type.

Proteins are made up of amino acids, whose order is determined by a sequence of nucleotides in DNA. The proteins that characterize a cell's phenotype are strings of amino acids bound together whose order is coded by the sequence of nucleotides in a messenger RNA (mRNA) that is transcribed from DNA in the nucleus. The process of transcription (production of the mRNA from DNA sequence) of these protein coding genes is regulated in part by the interaction of regulatory proteins called transcription factors (TF) with specific sequences (called regulatory elements) within a region of DNA adjacent to the protein coding gene called the promoter (Figure 1).

Understanding the regulatory events that lead to differentiation of different cell types is a critical step in understanding normal biological function as well as understanding what errors underlie dysfunctions such as cancer. The researchers have been studying the differentiation of mammary epithelial cells by identifying a subset of the proteins whose abundance changes during the transition from a mammary precursor cell to a functional mammary cell. The promoter region of each of the genes coding for these proteins has been analyzed to identify many potential regulatory elements (hundreds) that may have contributed to their differential expression. It is likely that few, if any, of these elements were actually involved, making critical analysis of the probability of their involvement very important. The sheer number of potential elements makes analysis "by hand" impractical, and is the reason for constructing Brovine.

1.1 How Experiments are Performed

Dr. Peterson's approach is to study what changes occur in mammary cells from different sources during the transformation from one state to another state (e.g. pregnancy to lactation in mice, or undifferentiated to differentiated in the MAC-T bovine cell line). This is done by comparing global protein expression in the two states and identifying a subset of the proteins

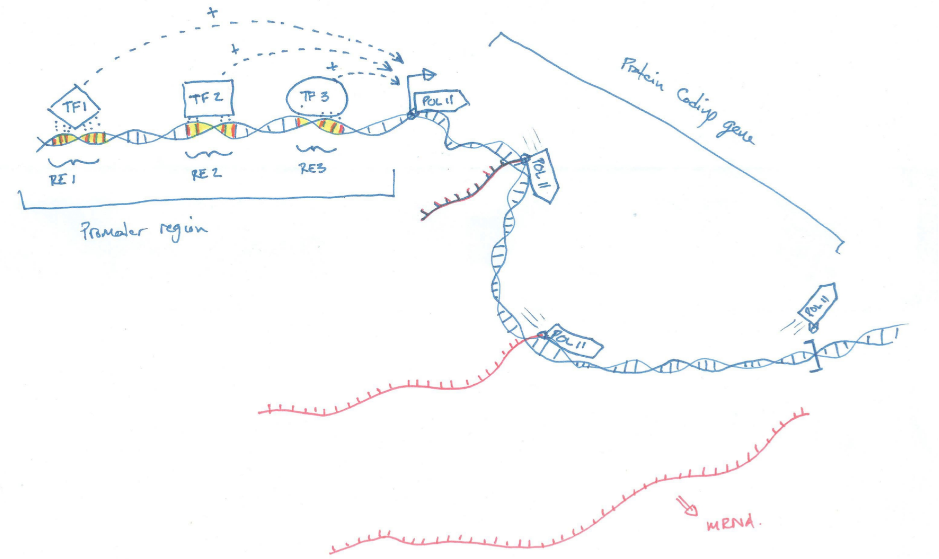


Figure 1: Schematic showing the promoter region of a gene with three regulatory elements (RE) each bound by their respective transcription factor (TF) contributing to the activation of transcription of the gene.

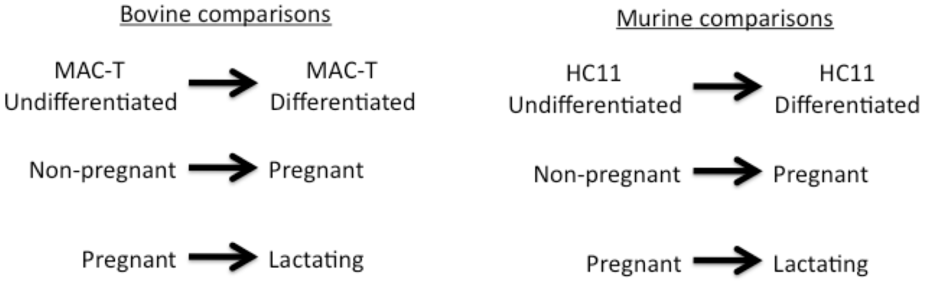


Figure 2: Several types of differentiation (cell transforming from one type to another) that will be used.

that change in abundance with the transformation from one state to the other. The data that we will begin with is from a cell line (MAC-T) from one species (cow, or "Bovine"). For each species, we will also be including more than one comparison based on the different sources of cells we are comparing. For example, within the Murine species, we will be comparing pregnant to lactating, nonpregnant to pregnant, and undifferentiated to differentiated HC11 cell line (3 separate comparisons). In figure 2, each arrow corresponds to a comparison. From each comparison, we generate a list of proteins that change in abundance as well as their direction and magnitude of change (though we are not including magnitude in our database). These proteins determine the genes present in Brovine - they are the genes the researchers are interested in studying.

The gene promoters: Remember that each protein is the result of a gene being "expressed", and that the gene expression is regulated in part by sequences in the gene promoter interacting with activating proteins or "transcription factors". Once the list of proteins that changed in abundance (up or down) in our comparison is obtained, the researchers search for the gene for each protein in a genome database and find its promoter sequence. The researchers then copy 2000 bases of that promoter sequence for analysis of the sequence to identify potential regulatory sequence elements within the 2000 base sequence.

Regulatory sequence search (TESS): The researchers then use an online program called Transcription Element Search System (TESS) that analyzes their input DNA sequence (the 2000 base promoter) and identifies short sequences that have been known to interact with a transcription factor to activate gene expression. The results of this search can be downloaded in a Microsoft Excel for each gene promoter that is used as a query. These Excel documents are the documents that are inserted into Brovine for analysis. To learn more about these Excel files, view the PDF [4].

2 Using Brovine

Brovine uses a log-in system to keep genetic data private and to keep malicious users from editing data. To start using Brovine, you need to get an account. Currently, the registration for Brovine is closed, so an account must be created for you by an administrator. To get an account, contact the person in charge of Brovine.

After you get an account and log in, you will be presented with the Experiment Hierarchy page, as well as a navigation bar at the top of the page. Click the Navigation link to see a list of views available. Each view is explained in section 2.5.

Clicking on your display name in the navigation bar will display a menu of options that affect your user account. If you have sufficient access to Brovine, the Upload link will also appear in this menu. The settings link on this menu allows you to change your password or your display name. Finally, the log out link is in this menu.

2.1 Navigating Through Data

Most pages contain many tables where genetic data is displayed. These tables can be clicked with the mouse to select rows of data that you would like to see more information about. For example, on the Experiment Hierarchy page if you select Bovine in the Species table, the Comparison table next to it will be populated only with comparison types involving Bovine species. Additionally, some tables in Brovine allow the user to select multiple table rows at the same time. In this case, hold down Ctrl (Windows) or Command (Mac) to select multiple rows of data.

In general, the flow of each page is from left to right, then top to bottom. This means that to use each view, you must start with the table in the top-left corner and work right and then down to get more specific genetic data.

2.2 View Features

Quality Filter: Some pages, like the Transcription Factor Search page, contain some extra search options for regulatory sequences. These search options are labeled Regulatory Sequence Filter Options, and they let you search using minimum and maximum values for quality filters (La, La/, Lq, Ld), position of factor in promoter sequence, and sense.

Table Search: Each table in Brovine is equipped with a search box that searches through all of the data in the table, displaying only rows which contain each word you search for. The search box is located above each table.

Regulation Filter: Some pages, such as the Gene Summary page, let you filter genes by regulation type. This box will appear above the table that it filters, next to a “Filter by Regulation” label.

Export Buttons: Tables that have an “Export” button below them can be exported as a CSV file.

2.3 Editing and Adding Data

To edit or hide data in Brovine, see the Experiment Hierarchy page.

Adding Data to Brovine is accomplished with the Upload button under your user menu (click your display name in the navigation bar to access this menu). Once on the Upload Data page, click the “Select Files” link to choose files to upload into Brovine. To select multiple files, hold down the Shift key and select another file after selecting the first one. Currently, only CSV spreadsheets from the TESS system are supported. Be sure to include all three TESS files related to each gene, or Brovine will not upload any data about that gene.

2.4 Account Management

Each user of Brovine must have an account in order to access any views or to edit or hide any data. There are several different types of access, or privileges, which may or may not be granted to a user. Visitors to Brovine can only view the Help pages.

2.4.1 Privilege Types

Admin: This type of user can browse, edit, and hide data in Brovine. In the future, this type of account will be able to add and remove other user accounts.

Modify: This type of user can browse the data in Brovine and can also hide or edit data using the Experiment Hierarchy page. They can also add new data using the Upload page.

Read: This type of user can do nothing except browse the data available in Brovine.

2.4.2 Adding and Removing User Accounts

Currently, user accounts can only be added or removed by Brovine’s software developer.

2.4.3 Changing your Password or Display Name

After logging in, a user can change his or her password or display name using the Settings page. This page is linked in the menu under your display name.

2.5 View Descriptions

2.5.1 Experiment Hierarchy

See figure 3. How to use this view:

I. Select a Species from 1. All comparisons which are about the selected species will populate the next table.

II. Select a Comparison from 2. All experiments which use the selected comparison will be present in the next table.

III. Select an Experiment from 3. All genes in the selected experiment will be present in the next table.

IV. Select a Gene from 4 All of the gene's transcription factors will appear in the next table.

V. Select a Transcription Factor from 5. All regulatory sequences which match the transcription factor selected will appear in the next table.

VI. Select a Regulatory Sequence from 6. The Sequence Info section will appear, with all information about the sequence you've selected, as well as similar sequences and matching factors in 7 and 8.

There is a regulation filter for 4 and a quality filter for 6.

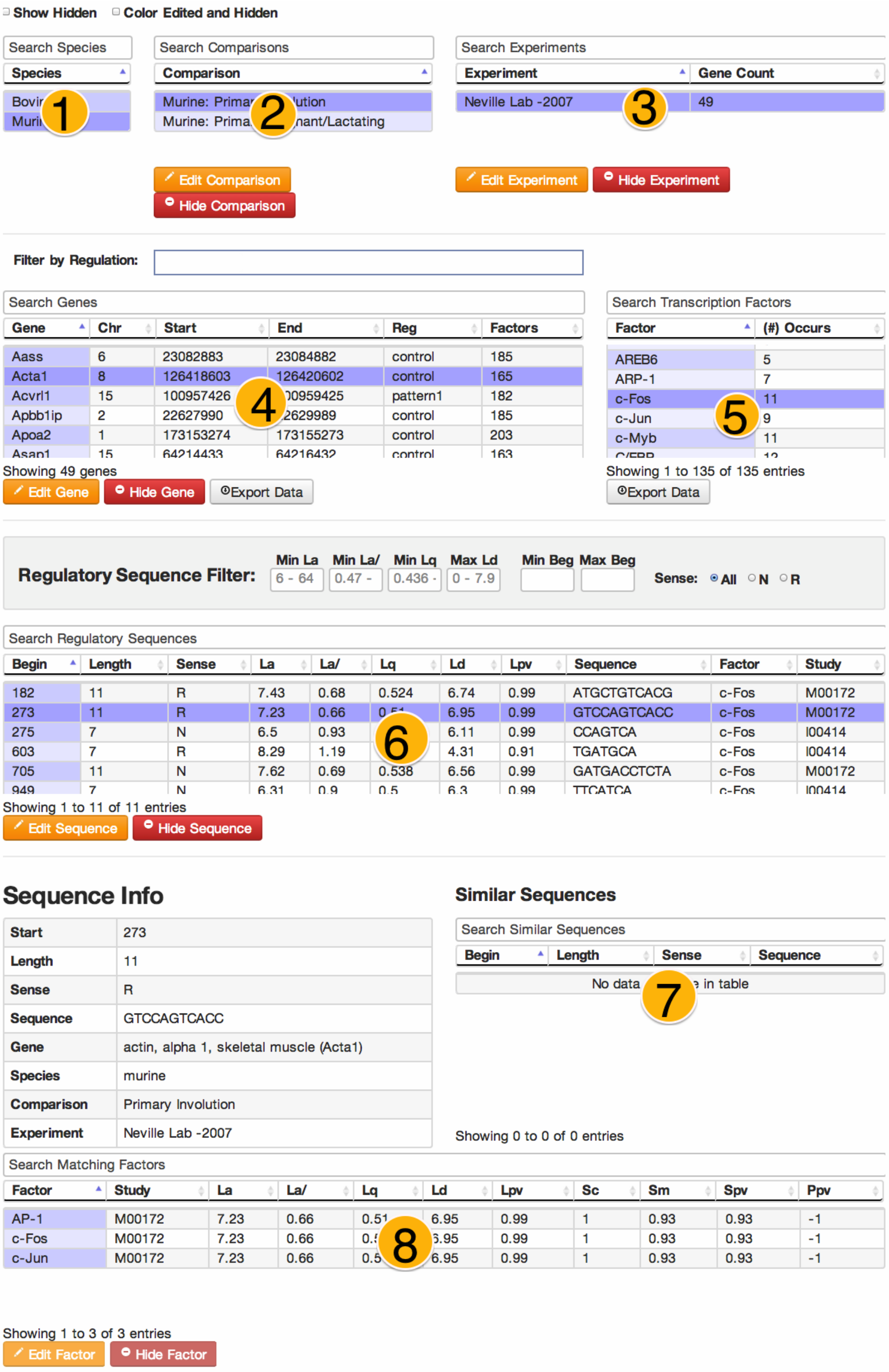
On this page, comparisons, experiments, genes, and regulatory sequences can be edited using the "Edit" and "Hide" buttons. Additionally, a user can choose to see previously hidden rows ("Show Hidden" above 1) or color edited and hidden rows differently than the rest ("Color Edited and Hidden"). Edited rows will show up yellow and hidden rows will show up red.

2.5.2 Transcription Factor Search

See figure 4. How to use this view:

I. Select one or more Species from 1 . All comparisons for any selected species will populate the next table.

II. Select one or more Comparisons from 2. All experiments which use any of the selected comparisons will be present in the next table.



□ Show Hidden □ Color Edited and Hidden

Search Species

Species
Bovir
Muri

Search Comparisons

Comparison
Murine: Primar lution
Murine: Prima nant/Lactating

Search Experiments

Experiment	Gene Count
Neville Lab -2007	49

Filter by Regulation:

Search Genes

Gene	Chr	Start	End	Reg	Factors
Aass	6	23082883	23084882	control	185
Acta1	8	126418603	126420602	control	165
Acvrl1	15	100957426	00959425	pattern1	182
Apbb1ip	2	22627990	2629989	control	185
Apoa2	1	173153274	173155273	control	203
Asap1	15	64214433	64216432	control	163

Showing 49 genes

Search Transcription Factors

Factor	(#) Occurs
AREB6	5
ARP-1	7
c-Fos	11
c-Jun	9
c-Myb	11
C/EBP	12

Showing 1 to 135 of 135 entries

Regulatory Sequence Filter: Min La 6 - 64 Min La/ 0.47 - Min Lq 0.436 - Max Ld 0 - 7.9 Min Beg Max Beg Sense: ● All ○ N ○ R

Search Regulatory Sequences

Begin	Length	Sense	La	La/	Lq	Ld	Lpv	Sequence	Factor	Study
182	11	R	7.43	0.68	0.524	6.74	0.99	ATGCTGTCACG	c-Fos	M00172
273	11	R	7.23	0.66	0.51	6.95	0.99	GTCCAGTCACC	c-Fos	M00172
275	7	N	6.5	0.93		6.11	0.99	CCAGTCA	c-Fos	I00414
603	7	R	8.29	1.19		4.31	0.91	TGATGCA	c-Fos	I00414
705	11	N	7.62	0.69	0.538	6.56	0.99	GATGACCTCTA	c-Fos	M00172
949	7	N	6.31	0.9	0.5	6.3	0.99	TTCATCA	c-Fos	I00414

Showing 1 to 11 of 11 entries

Sequence Info

Start	273
Length	11
Sense	R
Sequence	GTCCAGTCACC
Gene	actin, alpha 1, skeletal muscle (Acta1)
Species	murine
Comparison	Primary Involution
Experiment	Neville Lab -2007

Similar Sequences

Search Similar Sequences

Begin	Length	Sense	Sequence
No data e in table			

Showing 0 to 0 of 0 entries

Search Matching Factors

Factor	Study	La	La/	Lq	Ld	Lpv	Sc	Sm	Spv	Ppv
AP-1	M00172	7.23	0.66	0.51	6.95	0.99	1	0.93	0.93	-1
c-Fos	M00172	7.23	0.66	0.5	6.95	0.99	1	0.93	0.93	-1
c-Jun	M00172	7.23	0.66	0.5	6.95	0.99	1	0.93	0.93	-1

Showing 1 to 3 of 3 entries

Figure 3: The experiment hierarchy page.

III. Select one or more Experiments from 3. All genes in any of the selected experiments will be present in the next table.

IV. Select one or more Genes from 4 All transcription factors present in any gene selected will appear in the next table.

V. Select a Transcription Factor from 5. All regulatory sequences which match the transcription factor selected will appear in the next table.

VI. Select a Regulatory Sequence from 6. The Sequence Info section will appear, with all information about the sequence you've selected, as well as similar sequences and matching factors in 7 and 8.

2.5.3 Transcription Factor Subtract

Finds transcription factors that are in any gene selected from 4 and which are not in any gene selected in 5. For example, imagine gene A has transcription factors M, N, and O, while gene B has transcription factors O, P, and Q. If A was selected in 4 and B selected in 5, then 6 would show M and N only.

See figure 5. How to use this view:

I. Select one or more Species from 1 . All comparisons for any selected species will populate the next table.

II. Select one or more Comparisons from 2. All experiments which use any of the selected comparisons will be present in the next table.

III. Select one or more Experiments from 3. All genes in any of the selected experiments will be present in the next table.

IV. Select one or more Genes from 4 . The transcription factors present in these genes will be included in 6.

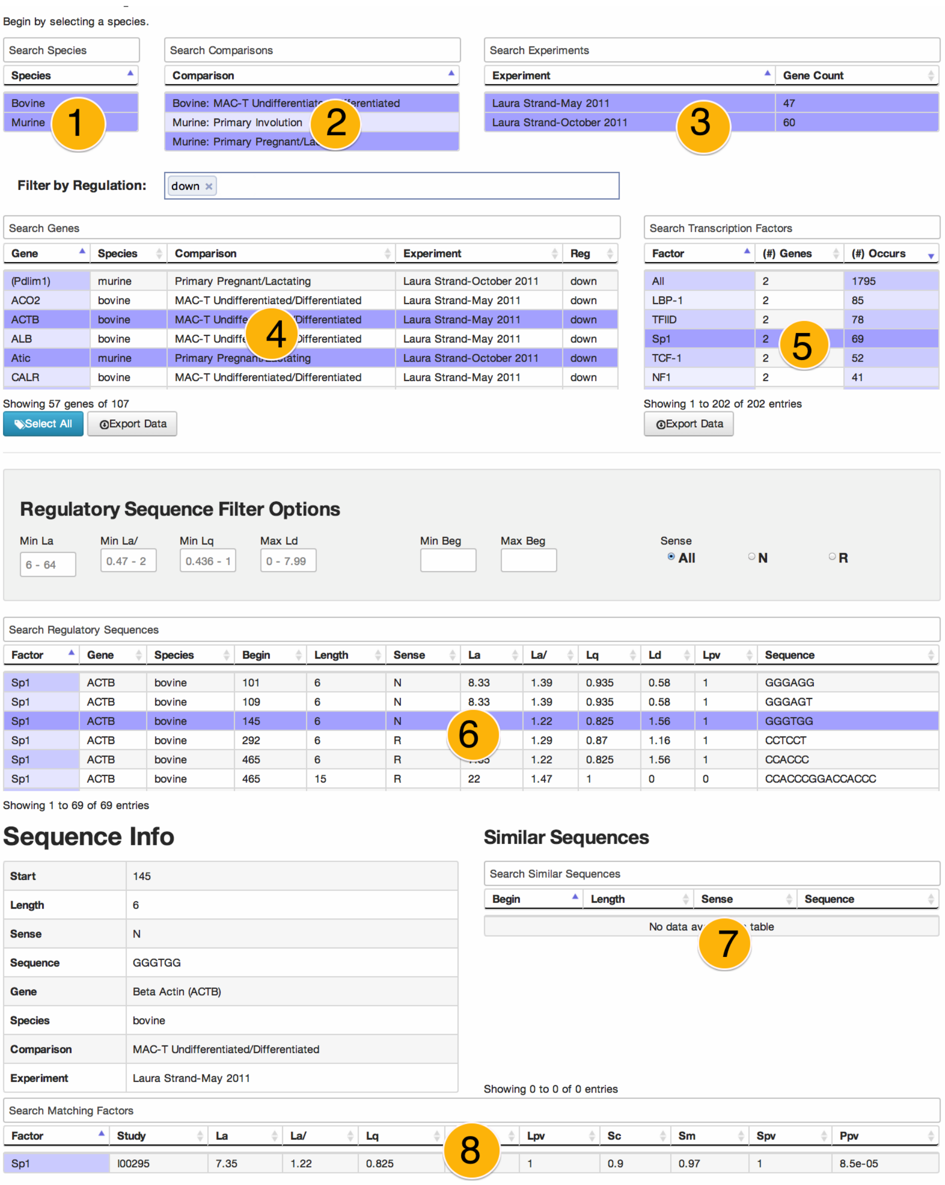
V. Select one or more Genes from 5 . The transcription factors present in these genes will be excluded, by name, from 6.

2.5.4 Gene Summary

See figure 6. How to use this view:

I. Select a Gene from 1.

II. All experiments with the Gene that was selected will appear in 2.



Begin by selecting a species.

Search Species

Species
Bovine
Murine

Search Comparisons

Comparison
Bovine: MAC-T Undifferentiat ferentiated
Murine: Primary Involution
Murine: Primary Pregnant/La

Search Experiments

Experiment	Gene Count
Laura Strand-May 2011	47
Laura Strand-October 2011	60

Filter by Regulation: down ×

Search Genes

Gene	Species	Comparison	Experiment	Reg
(Pdlim1)	murine	Primary Pregnant/Lactating	Laura Strand-October 2011	down
ACO2	bovine	MAC-T Undifferentiated/Differentiated	Laura Strand-May 2011	down
ACTB	bovine	MAC-T Undiffe Differentiated	Laura Strand-May 2011	down
ALB	bovine	MAC-T Undiffe Differentiated	Laura Strand-May 2011	down
Atic	murine	Primary Pregnan tating	Laura Strand-October 2011	down
CALR	bovine	MAC-T Undifferentiated/Differentiated	Laura Strand-May 2011	down

Showing 57 genes of 107

Select All Export Data

Search Transcription Factors

Factor	(#) Genes	(#) Occurs
All	2	1795
LBP-1	2	85
TFIID	2	78
Sp1	2	69
TCF-1	2	52
NF1	2	41

Showing 1 to 202 of 202 entries

Export Data

Regulatory Sequence Filter Options

Min La: 6 - 64 Min La/: 0.47 - 2 Min Lq: 0.436 - 1 Max Ld: 0 - 7.99 Min Beg: Max Beg:

Sense ● All ○ N ○ R

Search Regulatory Sequences

Factor	Gene	Species	Begin	Length	Sense	La	La/	Lq	Ld	Lpv	Sequence
Sp1	ACTB	bovine	101	6	N	8.33	1.39	0.935	0.58	1	GGGAGG
Sp1	ACTB	bovine	109	6	N	8.33	1.39	0.935	0.58	1	GGGAGT
Sp1	ACTB	bovine	145	6	N		1.22	0.825	1.56	1	GGGTGG
Sp1	ACTB	bovine	292	6	R		1.29	0.87	1.16	1	CCTCCT
Sp1	ACTB	bovine	465	6	R		1.22	0.825	1.56	1	CCACCC
Sp1	ACTB	bovine	465	15	R	22	1.47	1	0	0	CCACCCGGACCACCC

Showing 1 to 69 of 69 entries

Sequence Info

Start	145
Length	6
Sense	N
Sequence	GGGTGG
Gene	Beta Actin (ACTB)
Species	bovine
Comparison	MAC-T Undifferentiated/Differentiated
Experiment	Laura Strand-May 2011

Similar Sequences

Search Similar Sequences

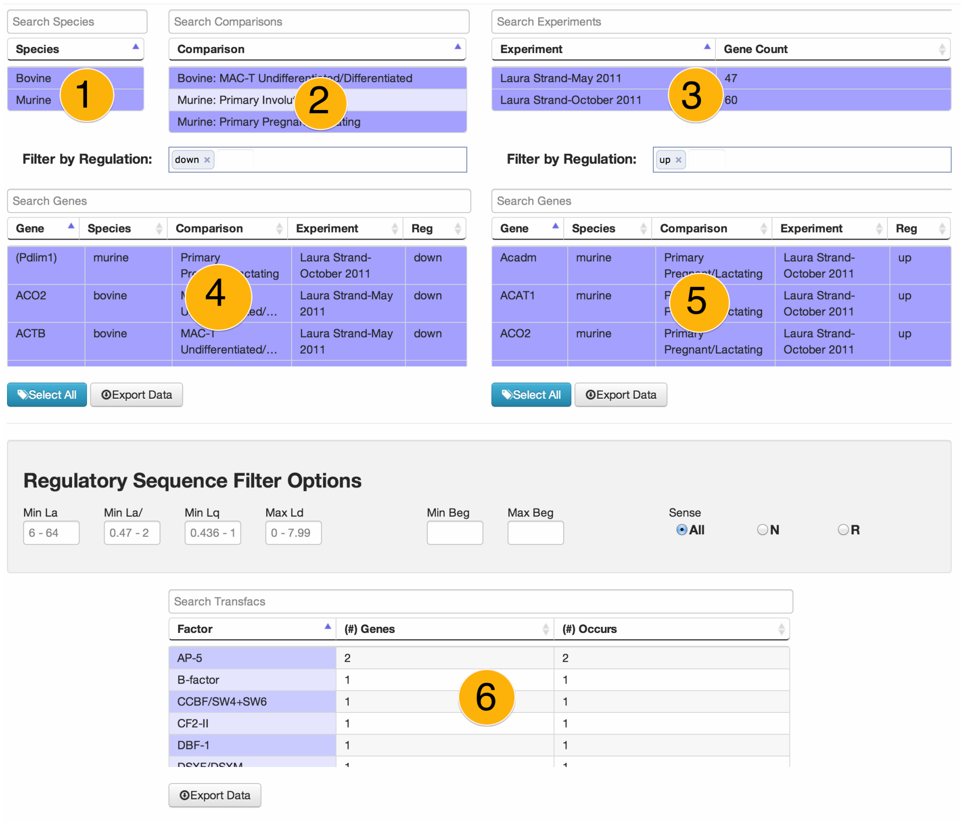
Begin	Length	Sense	Sequence
No data av table			

Showing 0 to 0 of 0 entries

Search Matching Factors

Factor	Study	La	La/	Lq		Lpv	Sc	Sm	Spv	Ppv
Sp1	I00295	7.35	1.22	0.825		1	0.9	0.97	1	8.5e-05

Figure 4: The transcription factor search page.



Search Species

Species
Bovine
Murine

Search Comparisons

Comparison
Bovine: MAC-T Undifferen ed/Differentiated
Murine: Primary Involu
Murine: Primary Pregna ting

Search Experiments

Experiment	Gene Count
Laura Strand-May 2011	47
Laura Strand-October 2011	60

Filter by Regulation: down ×

Filter by Regulation: up ×

Search Genes

Gene	Species	Comparison	Experiment	Reg
(Pdlim1)	murine	Primary Pr ctating	Laura Strand-October 2011	down
ACO2	bovine	M U ed/…	Laura Strand-May 2011	down
ACTB	bovine	MAC-T Undifferentiated/…	Laura Strand-May 2011	down

Select All · Export Data

Search Genes

Gene	Species	Comparison	Experiment	Reg
Acadm	murine	Primary Preg t/Lactating	Laura Strand-October 2011	up
ACAT1	murine	P F ctating	Laura Strand-October 2011	up
ACO2	murine	Primary Pregnant/Lactating	Laura Strand-October 2011	up

Select All · Export Data

Regulatory Sequence Filter Options

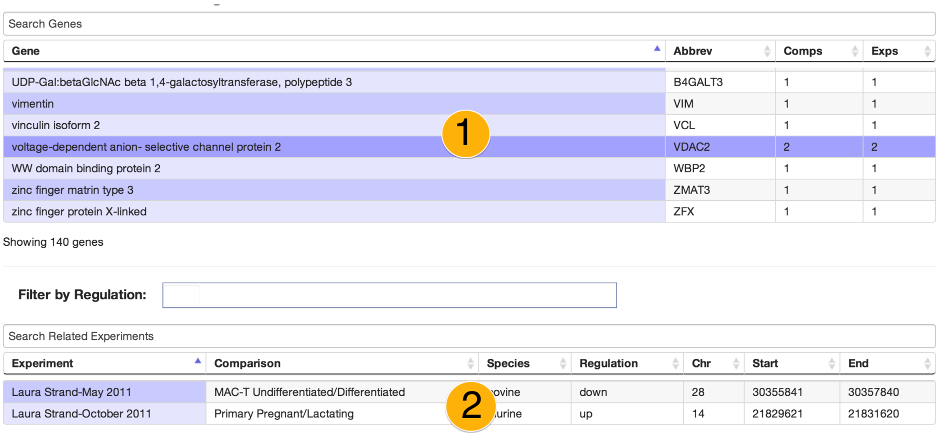
Min La	Min La/	Min Lq	Max Ld	Min Beg	Max Beg	Sense
6 - 64	0.47 - 2	0.436 - 1	0 - 7.99			● All ○ N ○ R

Search Transfacs

Factor	(#) Genes	(#) Occurs
AP-5	2	2
B-factor	1	1
CCBF/SW4+SW6	1	1
CF2-II	1	1
DBF-1	1	1
DSXF/DSXM	1	1

Export Data

Figure 5: The transcription factor subtract page.



Search Genes

Gene	Abbrev	Comps	Exps
UDP-Gal:betaGlcNAc beta 1,4-galactosyltransferase, polypeptide 3	B4GALT3	1	1
vimentin	VIM	1	1
vinculin isoform 2	VCL	1	1
voltage-dependent anion- selective channel protein 2	VDAC2	2	2
WW domain binding protein 2	WBP2	1	1
zinc finger matrin type 3	ZMAT3	1	1
zinc finger protein X-linked	ZFX	1	1

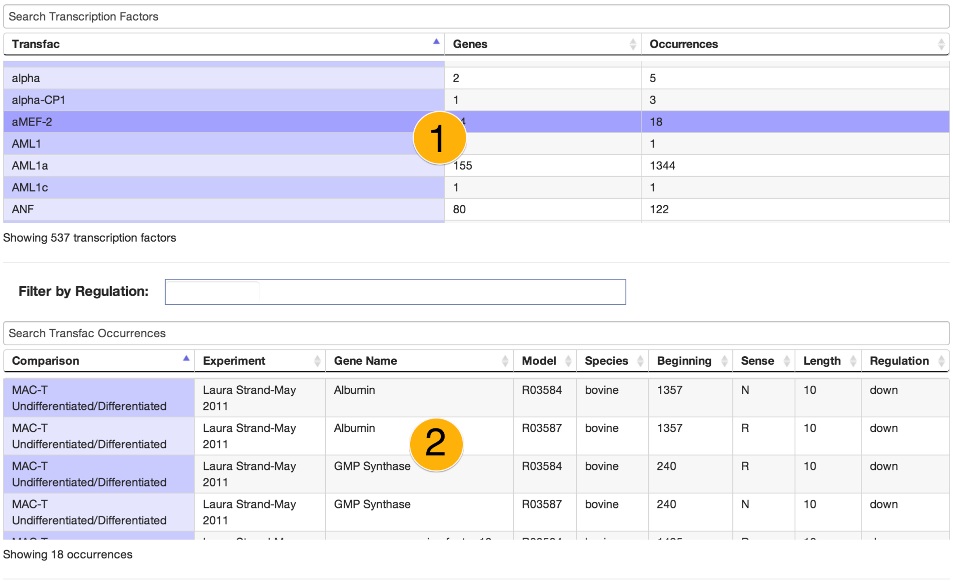
Showing 140 genes

Filter by Regulation:

Search Related Experiments

Experiment	Comparison	Species	Regulation	Chr	Start	End
Laura Strand-May 2011	MAC-T Undifferentiated/Differentiated	ovine	down	28	30355841	30357840
Laura Strand-October 2011	Primary Pregnant/Lactating	urine	up	14	21829621	21831620

Figure 6: The gene summary page.



Search Transcription Factors

Transfac	Genes	Occurrences
alpha	2	5
alpha-CP1	1	3
aMEF-2	[illegible]	18
AML1	[illegible]	1
AML1a	155	1344
AML1c	1	1
ANF	80	122

Showing 537 transcription factors

Filter by Regulation:

Search Transfac Occurrences

Comparison	Experiment	Gene Name	Model	Species	Beginning	Sense	Length	Regulation
MAC-T Undifferentiated/Differentiated	Laura Strand-May 2011	Albumin	R03584	bovine	1357	N	10	down
MAC-T Undifferentiated/Differentiated	Laura Strand-May 2011	Albumin	R03587	bovine	1357	R	10	down
MAC-T Undifferentiated/Differentiated	Laura Strand-May 2011	GMP Synthase	R03584	bovine	240	R	10	down
MAC-T Undifferentiated/Differentiated	Laura Strand-May 2011	GMP Synthase	R03587	bovine	240	N	10	down

Showing 18 occurrences

Figure 7: The transcription factor summary page.

2.5.5 Transcription Factor Summary

See figure 7. How to use this view:

I. Select a Transcription Factor from 1.

II. All genes with the Transcription Factor that was selected will appear in 2.

2.5.6 Gene Search

See figure 8. How to use this view:

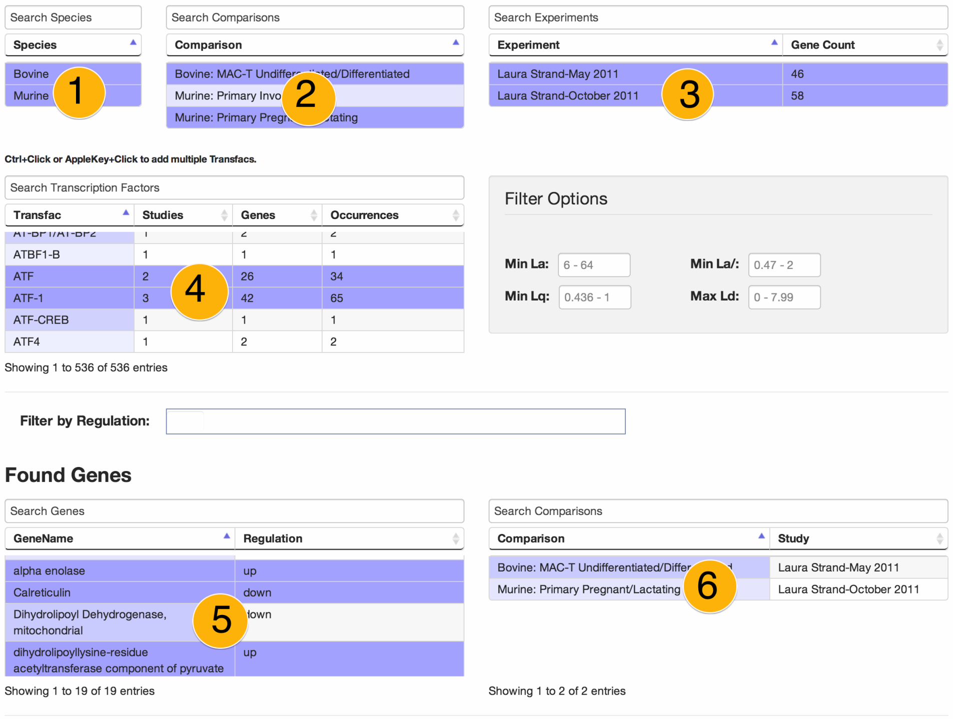
I. Select one or more Species from 1 . All comparisons for any selected species will populate the next table.

II. Select one or more Comparisons from 2. All experiments which use any of the selected comparisons will be present in the next table.

III. Select one or more Experiments from 3. All genes in any of the selected experiments will be present in the next table.

IV. Select one or more Transcription Factors from 4. Genes which have any of the selected transcription factors will be shown in the next table.

V. Select one or more Genes from 5 . Experiments which contain all of the genes selected will be shown in 6 .



Search Species

Species
Bovine
Murine

Search Comparisons

Comparison
Bovine: MAC-T Undiffer ted/Differentiated
Murine: Primary Invo
Murine: Primary Pregn tating

Search Experiments

Experiment	Gene Count
Laura Strand-May 2011	46
Laura Strand-October 2011	58

Ctrl+Click or AppleKey+Click to add multiple Transfacs.

Search Transcription Factors

Transfac	Studies	Genes	Occurrences
AT-BP1/AT-BP2	1	2	2
ATBF1-B	1	1	1
ATF	2	26	34
ATF-1	3	42	65
ATF-CREB	1	1	1
ATF4	1	2	2

Showing 1 to 536 of 536 entries

Filter Options

Min La: 6 - 64

Min La/: 0.47 - 2

Min Lq: 0.436 - 1

Max Ld: 0 - 7.99

Filter by Regulation:

Found Genes

Search Genes

GeneName	Regulation
alpha enolase	up
Calreticulin	down
Dihydrolipoyl Dehydrogenase, mitochondrial	down
dihydrolipoyllysine-residue acetyltransferase component of pyruvate	up

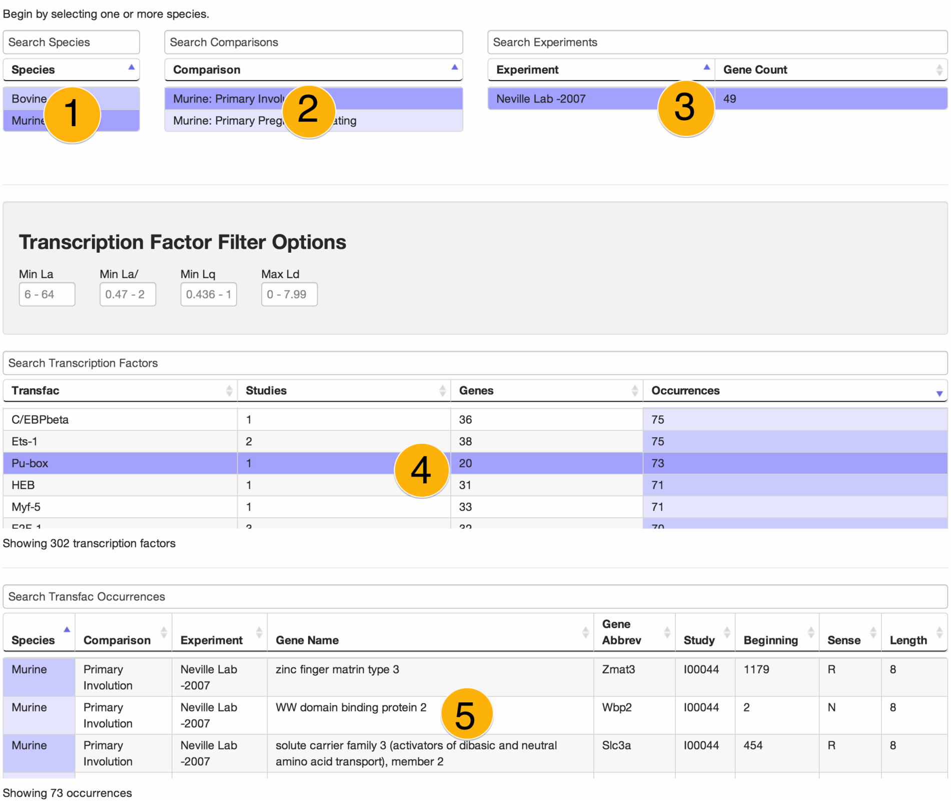
Showing 1 to 19 of 19 entries

Search Comparisons

Comparison	Study
Bovine: MAC-T Undifferentiated/Differ	Laura Strand-May 2011
Murine: Primary Pregnant/Lactating	Laura Strand-October 2011

Showing 1 to 2 of 2 entries

Figure 8: The gene search page.



Begin by selecting one or more species.

Search Species

Species
Bovine
Murine

Search Comparisons

Comparison
Murine: Primary Involu
Murine: Primary Preg ating

Search Experiments

Experiment	Gene Count
Neville Lab -2007	49

Transcription Factor Filter Options

Min La: 6 - 64
Min La/: 0.47 - 2
Min Lq: 0.436 - 1
Max Ld: 0 - 7.99

Search Transcription Factors

Transfac	Studies	Genes	Occurrences
C/EBPbeta	1	36	75
Ets-1	2	38	75
Pu-box	1	20	73
HEB	1	31	71
Myf-5	1	33	71

Showing 302 transcription factors

Search Transfac Occurrences

Species	Comparison	Experiment	Gene Name	Gene Abbrev	Study	Beginning	Sense	Length
Murine	Primary Involution	Neville Lab -2007	zinc finger matrin type 3	Zmat3	I00044	1179	R	8
Murine	Primary Involution	Neville Lab -2007	WW domain binding protein 2	Wbp2	I00044	2	N	8
Murine	Primary Involution	Neville Lab -2007	solute carrier family 3 (activators of dibasic and neutral amino acid transport), member 2	Slc3a	I00044	454	R	8

Showing 73 occurrences

Figure 9: The transcription factor popularity page.

2.5.7 Transcription Factor Popularity

See figure 9. How to use this view:

I. Select a Species from 1. All comparisons which are about the selected species will populate the next table.

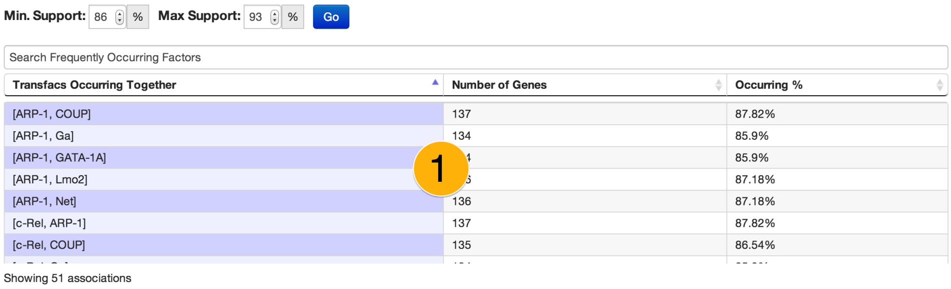
II. Select a Comparison from 2. All experiments which use the selected comparison will be present in the next table.

III. Select an Experiment from 3. All genes in the selected experiment will be present in the next table.

IV. Select a Transcription Factor from 4. All occurrences of the transcription factor selected, from any gene in the experiment selected, will be displayed in 5.

2.5.8 Frequent Transcription Factors

This view allows users of Brovine to discover the most commonly occurring transcription factors or sets of transcription factors for all genes in the



Min. Support: 86 % Max Support: 93 %

Search Frequently Occurring Factors

Transfacs Occurring Together	Number of Genes	Occurring %
[ARP-1, COUP]	137	87.82%
[ARP-1, Ga]	134	85.9%
[ARP-1, GATA-1A]	134	85.9%
[ARP-1, Lmo2]	136	87.18%
[ARP-1, Net]	136	87.18%
[c-Rel, ARP-1]	137	87.82%
[c-Rel, COUP]	135	86.54%

Showing 51 associations

Figure 10: The frequent transcription factor search page.

database. You can select the minimum and maximum percentage of genes a set can show up in, to limit the results of the search. For example, the factors AP-2alphaA and AP-2alphaB occur together in approximately 86% of all genes currently in the database - so if you choose 85% as the minimum and 92% as the maximum, this set of transcription factors would appear. Upon reaching this view, 1 will have a loading spinner, indicating that the data is being loaded from Brovine.

Note: Select the Minimum and Maximum percentages wisely. Choosing too wide of a range, or selecting maximum greater than 96%, may not yield any results. Try to choose a range close to the default (85 - 95%).

See figure 10. How to use this view:

I. If 1 is loading data, wait until data is loaded. If it has been loading for a long time, try refreshing the page.

II. Select minimum and maximum percentage above 1, then click Go.

III. All sets of transcription factors which are present in a percentage of genes within the range will be displayed in 1.

3 Technical Overview

The Brovine gene database is built upon the CodeIgniter framework on the backend, an MVC framework written in PHP. This runs on an Apache HTTP server, and it uses a MySQL database to store all of the genetic data. On the front end, the application is entirely in Javascript, using several open source projects, including JQuery, DataTables, and TokenInput.

Each page, which represents a view which the customers find useful, contains tables that display the genetic data to the user, as well as let the user drill down to more specific data. For example, on each page there are tables which show the species and experiments the customers have uploaded to the application. Customers can select one or more species, and Brovine will filter the experiment table to show only experiments with the selected species.

The source code for Brovine and the Frequent Itemset Generator are available on GitHub under the MIT License. [3][1]

3.1 Server-Side

CodeIgniter is a powerful MVC framework that Brovine uses. Almost all of the data transfer is done using AJAX, which gives the user the effect of a seamless desktop application with few complete page reloads.

Brovine's MySQL data store is a highly joined set of tables that represent the complicated client data. Most queries require a join of four or five tables. Details about the database implementation are explained on the SQL schema help page.

Genetic data, which is represented by a series of CSV files, is uploaded using the Uploadify plugin. Uploadify is a Flash plugin that provides safe, seamless upload of data files to a server with minimal effort required of the user. It also lets the user track the progress of the file uploads, upload multiple files at one time, and cancel any uploads if necessary. Brovine stores temporary files from Uploadify into the `/brovine/genedata-uploads` folder while the system converts the files into data usable by Brovine.

The data itself is received from TESS as a set of Excel files. There is a set of example files downloadable as a zip file. Each gene that the researcher analyzes has its own set of 3 files which must all be uploaded to Brovine if the gene is to be committed into the system:

- **Job parameters:** contains information about the experiment conducted, and the gene; populates the experiment, comparison_type, and gene tables.
- **Sequences:** contains information about the promoter sequence for the gene; populates the promoter_sequence table
- **HITS1:** contains information about the regulatory elements discovered in prior research; populates the regulatory_sequences, factor_matches, and study_pages tables.

3.2 Client-Side

The following libraries are used to enhance the user experience:

- **JQuery[1]:** a powerful Javascript client library
- **DataTables[2]:** a JQuery plugin that provides extensive table support

[1] http://jquery.com

[2] http://datatables.net

- **TokenInput[3]:** a small but powerful JQuery plugin which enhances text boxes. The plugin searches through a set of pre-defined strings given a user input, which the user can then select.
- **Bootstrap[4]:** a front-end HTML5 framework that makes web design simpler by offering basic styles for tables, lists, navigation, layout, and more.

Each user view is a set of tables that allow the user to drill down to specific data points they want to see. For example, on the Transcription Factor Summary page, the user is interested in finding all genes in which a specific transcription factor occurs. So the first table lets the user select a transcription factor by name, and the second table shows all genes which have the selected factor.

Each table is populated on the back end by a method in the ajax controller, which is located at `/brovine/application/controllers/ajax.php`. The DataTables library handles searching (via the search box above each table), sorting, and filtering on the client side.

There are other Brovine features worth mentioning. The Filter by Regulation box is an example of a TokenInput text box. The purpose of this text box is to allow users to filter their results by regulation. As the user starts typing, the TokenInput library sends whatever prefix they have typed to the `getRegHints` method on the ajax controller, which attempts to match their prefix term with any of the regulation types in the database.

3.3 Cache Control

Another feature of Brovine is the cache control mechanism. All static files (JS and CSS) have a timestamp appended to their name using the Apache `mod_rewrite` module. The timestamp is the last modify time on the file, so each time a static file is modified, the browser will think it is a brand new file and download the new version. This eliminates issues where cached versions of static files are used by the client's browser.

3.4 Javascript Architecture

Each view has its own javascript file - for example, the ExperimentHierarchy javascript code is held inside experimentHierarchy.js. All Javascript code is located in the `/brovine/js` folder.

Here is a list of views and their corresponding Javascript code:

- **Experiment Hierarchy:** `experimentHierarchy.js`

[3] http://loopj.com/jquery-tokeninput

[4] http://twitter.github.io/bootstrap

- **Transcription Factor Search:** `tfSearch.js`
- **Transcription Factor Subtract:** See the next section
- **Gene Summary:** `geneSummary.js`
- **Transcription Factor Summary:** `tfSummary.js`
- **Gene Search:** `geneSearch.js`
- **Transcription Factor Popularity:** `tfPop.js`
- **Frequent Transcription Factors:** `experimentHierarchy.js`

In addition to each view's javascript code, there are helper scripts. The file `common.js` holds methods useful for all of the views. It contains the updateSelectList and updateMultiSelectList methods, which set up event handlers and handle the AJAX calls for each table.

The file `scripture.js` is responsible for the local download functionality which is present on some Brovine views - the Transcription Factor Subtract page, for example. Instead of generating a file on the server and sending it back to the user, this Javascript code generates a `data:octet-stream` link which lets the user extract the data without making another AJAX call.

The file `upload.js` talks to the Uploadify flash software to notify the user about the status of file uploads on the Upload page.

3.5 New Javascript Architecture

The creators of Brovine recognized that the current Javascript code is a huge mess, but under the weight of deadlines found no time to change the design. However, a significant effort has been made to create a system that is more reasonable. This effort is located in `/brovine/js/commonjs`. Currently, the Transcription Factor Subtract page is the only view to use the new Javascript architecture.

This new architecture uses CommonJS Modules and browserbuild to create a modular Javascript system that greatly increases code reuse and maintainability within Brovine. Javascript files that are shared among pages belong in the `lib` folder, while view-specific files belong in the `init-pages` folder. Each Brovine view still has its own Javascript file, but the file size per page is much smaller. Finally, Browserbuild[5] lets us concatenate and minify all necessary files into one unit, which reduces load time for the user.

[5] https://github.com/LearnBoost/browserbuild

3.6 Server Configuration

Brovine is currently hosted on a VM managed by the Cal Poly Computer Science department. Check with the CSL sysadmins to get access to the box via the shell. Its web address is http://brovine.csc.calpoly.edu. On the VM is the usual LAMP stack plus Java:

- MySQL.
- PHP.
- Apache.
- phpMyAdmin. Unnecessary; simply for database manipulation.
- Java Runtime Client (JRE).

The machine hosting Brovine needs to have a Java runtime client installed to run the Frequent Itemset Generator - see section 4. The generator runs as a standalone Java client that calculates the most common transcription factors among the genes selected. The data generated is displayed on the Frequent Transcription Factors page.

On the box, the server root is located at `/var/www/html` .

3.7 Apache Configuration

Brovine and CodeIgniter use the `mod_rewrite` package to edit incoming URLs. This enables CodeIgniter to shorten the final URL that the user sees and uses to access the service. It also enables Brovine to serve versioned CSS and JS documents, which stops browsers from using outdated files (see section 3.3).

3.8 MySQL Configuration

The MySQL configuration for Brovine is simple - just create the database using SQL schema and import the database backup. The username password, server name, and port that are used for Brovine are stored in the `passwd.php` file, which is not uploaded to the repository. There is an example of the file in the repository's README.

3.9 FAQ

What are we looking to get out of this project? - We're really looking for the ability of something to compare "the list of stuff" from one gene to another (how strong of a match)

For our purposes, what is a gene? - A gene is the 2000 base pairs that we get, for our purposes (even though this is not actually the case, the 2000 base pairs are the promoter region in front of the gene)

Could a factor have the same Beg, Sns, Len and a different Regulatory Sequence? - No

Can there be a different beg/len for the opposite sns that would still match? - No

What are L factors in the Factor Match table used for? - These are different measures of the probability of this seq actually interacting with this factor.

4 Frequent Itemset Generator

The frequent itemset generator is a Java service that finds statistically significant sets of frequent items in large datasets. For example, let's say we have a database of supermarket transactions, or baskets, each containing a number of grocery items. If we want to find the frequent grocery items in this database, we're finding the grocery items which are most frequently purchased together: the "frequent itemsets."

Generally we want the sets of items to occur in at least m transactions. We'll call this the minimum support number. The minimum support number allows us to specify how frequent an itemset must be to be included in the final output - obviously, a set of items that does not occur in any transactions, or even a small number of transactions, is insignificant to the user of the system. The minimum support generally must be tuned for each specific dataset to determine what will give the user plenty of data, but not so much data that the output becomes excessive.

To this end, we also include the maximum support number, which caps the number of possible transactions any set in the final output can have. Imagine that at a specific supermarket, nearly every customer buys milk when they buy anything else. Then, it is not very significant to include milk in the final output, since the item is so commonly purchased. Thus we can use the maximum support to filter out items like this. As with the min support, this number will also vary wildly between datasets. If you end up using the max support metric (less than 100% of course) for a dataset, this indicates that you should probably use the FPGrowth algorithm, as this algorithm runs much quicker on datasets with a large number of similar items between transactions.

4.1 Use

This service was built to find the genetic transcription factors (proteins) which most frequently occur together in one gene. However, you can add your own dataset if you want to use it for a different purpose. To edit and recompile the Frequent Itemset Generator, you must have Groovy[6] installed. Use `make` to recompile.

4.2 Customizability

- `class BasketIterator` : Implement BasketIterator to create your own dataset.
- `class ItemsetGenerator` : Implement ItemsetGenerator to create your own. algorithm.
- file `lib/passwd.groovy` : Password file - holds database configuration information.

4.3 API Reference

Get request: `get [minSup:decimal:0-1] [maxSup:decimal:0-1]`

Returns a list of the itemsets between the min and max support values. time is always 0 on FAILURE. maxSup must be less than minSup, obviously. Both are decimal values indicating the percent support an itemset must have to be included.

Returns:

```
{
    'res': "(SUCCESS | FAILURE)":string,
    'item-cnt': "integer indicating the number of unique items↙
        ↪ ":integer,
    'reason': "Reason for failure if FAILURE, request type if ↙
        ↪ SUCCESS.":string,
    'message': "Explanation of failure iff 'res' == 'FAILURE↙
        ↪ '.":string,
    'time': "integer indicating the time it took to find ↙
        ↪ itemsets":integer,
    'data': "map with the itemsets and their frequency counts; ↙
        ↪ ex: [[one, two]: 138]":map
}
```

Set request: `set [BasketIterator] [ItemsetGenerator]`

Changes the algorithm and the dataset used when computing frequent item sets.

[6] http://groovy.codehaus.org/

If the request is successful, res will be set to SUCCESS and reason will be “SET”. Any subsequent queries by client will use the BasketIterator and ItemsetGenerator specified. Both Set values must be fully-qualified Java class names.

Returns:

```
{
    'res': "(SUCCESS | FAILURE)":string,
    'reason': "Reason for failure if FAILURE, request type if SUCCESS.":string,
    'message': "Explanation of failure iff 'res' == 'FAILURE'.":string,
    'time': 0,
}
```

4.4 Available Algorithms

Apriori algorithm: Iterates through every transaction (supermarket basket) and candidate to find the most frequent sets of items. A candidate is any one subset of items in the entire set of items (in our example, every item in the supermarket). So we start with candidates with one item (Bread, milk, cheese), then we look for baskets with two items ([Bread, milk], [milk, cheese], [cheese, bread]), and so on until we’ve searched every subset. At worst case, this algorithm is exponential, as there are 2^n subsets for n items. But the trick is that we drop from consideration all larger subsets containing an infrequent item. For example, if we know that bread is in $m - 1$ baskets (one less than our minimum support) then we know bread is *never* a frequent item, so we can drop [Bread, milk], [cheese, Bread], and [bread, milk, cheese] from our consideration. This optimization significantly speeds up the algorithm for reasonable minimum support values (if you set the minimum support to 0, every item set will be frequent and you’ll have to iterate through every set).

FPGrowth algorithm: Based on the FPTree data structure [5]. This algorithm builds a tree representing your dataset. The more items each transaction has in common, the smaller the tree will be and thus the faster this algorithm will process all frequent itemsets. See chapter 6 of [5] for an overview and description of the algorithm.

5 Development Environment Setup

Development on a local machine is required for those who are performing major changes to the site. Using the VM to test is not acceptable for a live system, where users could be using the features you’re testing! Follow these steps to set up Brovine on your local box.

5.1 Prerequisites

- A Linux or Mac machine. Windows will probably work, but I haven't tried.
- A working LAMP stack. For Mac users, I've heard MAMP is a good solution.
- The Java Runtime Environment (JRE) installed.

5.2 Install Brovine

1. Clone the code for Brovine into your development directory.

   ```
   git clone https://github.com/brovine-developers/↙
       ↳ teambrovine
   ```

2. Edit Apache's `httpd.conf` to support CodeIgniter's clean URLs. You have to enable `mod_rewrite`, which lets you define rules in `.htaccess` files that modify incoming request URLs:

   ```
   LoadModule rewrite_module /usr/lib/apache2/modules/↙
       ↳ mod_rewrite.so
   ```

3. Start Apache and MySQL. Check that you can reach the log in page of Brovine. If you can't, try changing the file attributes and group. Each folder in Brovine should be in the group which the Apache HTTPD user is in. This is generally `_www`:

   ```
   chown -R :_www /brovine
   ```

   All folders should have rwx access for group; files at least r:

   ```
   chmod -R g+rwx /brovine
   ```

4. Create a database named `brovine` in MySQL.
5. Make sure at least one user has the following privileges to the database: insert, update, delete, select, index (preferrably not root)
6. Copy the sample `passwd.php` file from the repository's README into the `/brovine/application/config` directory.
7. Edit the file to match the settings you used in the previous step.
8. Get the SQL schema (section 6.4) and the database backup. [2]
9. Import both files into the `brovine` database you created earlier.

10. Insert a new user into the `users` table using a SHA1 hash of your password of choice, username, display name, and privilege set to `20`, which gives you administrator access to Brovine.

11. Log in to Brovine and check that all of the required tables are present and populated (see the SQL schema description for more information).

5.3 Install Frequent Itemset Generator

This standalone service generates data for the Frequent Transcription Factors page.

1. Clone the frequent itemset generator code. into your development directory.

   ```
   git clone https://github.com/brovine-developers/freq-itemset-gen
   ```

2. Start the service: `make start`

On startup, the service opens port 8100 and waits for messages - see the API for how to get data or test the service manually.

6 SQL Schema

Table and column descriptions and an ER diagram of the database. All tables have an auto-increment ID field which were omitted from the column descriptions.

6.1 Table Descriptions

Table Name	Used For
comparison_types	Stores all differentiation types the researchers are studying. The key is species, celltype, transition.
experiments	Stores each experiment that the researcher performs, each of which has a distinct comparison type and species. The key is a automatically assigned experiment id.
genes	Stores a gene that the researcher studies, which can be present in multiple experiments. Keyed by gene name.
regulatory_sequences	Stores a sequence of nucleotides that affect the expression of a specific gene. See the glossary entry for more details.
factor_matches	Stores transcription factors that other researchers have studied. Each transcription factor can be associated with multiple regulatory sequences, and each regulatory sequence can match multiple factor matches.
study_pages	Stores research paper references that the factor matches were retrieved from. Not currently used in the Brovine system.
promoter_sequences	Stores the sequence of nucleotides which contain all possible regulatory sequences for a specific gene.
apriori_staging	Stores temporary data for the Frequent Itemset Generator.
users	Stores user data for Brovine - usernames, password hashes, etc.

6.2 ER Diagram

See figure 11 for an entity-relationship diagram of the data modeled with Brovine, as well as the relationship names and table unique keys.

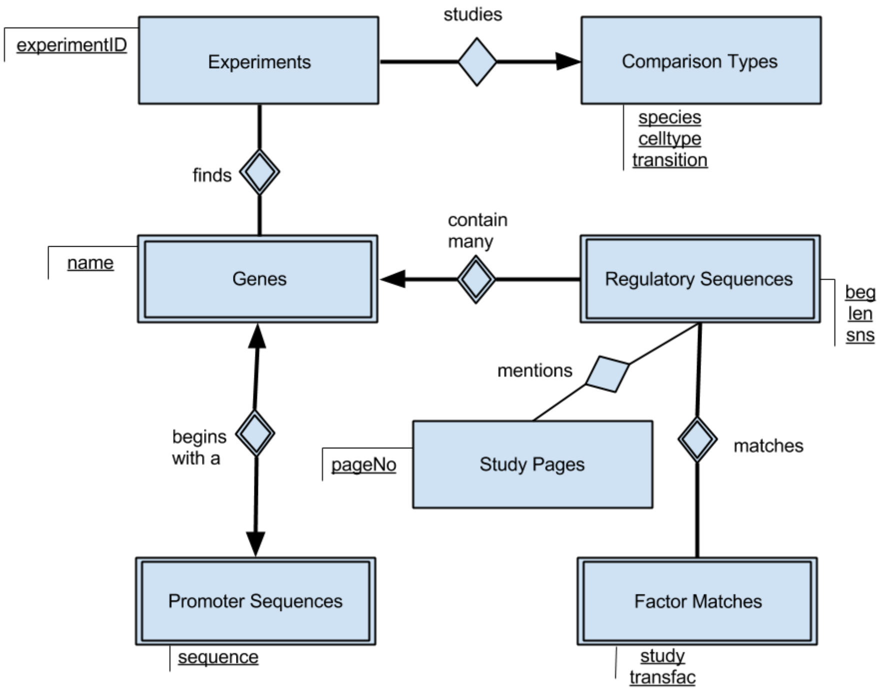


Figure 11: An E-R diagram modeling Brovine's database. Underlined labels indicate the unique keys of each table. In MySQL, each table also has a primary auto-increment key to simplify table joins.

6.3 Table Schemas

6.3.1 Comparison Types

Column Name	Type	Description
species	varchar	The species that the researcher is studying.
celltype	varchar	The differentiation (from one cell type to another) that the researcher is studying.

6.3.2 Genes

Column Name	Type	Description
genename	varchar	the name of this gene.
chromosome	int	the chromosome which this gene is on.
start	int	the start nucleotide of this gene on the chromosome.
end	int	the end nucleotide of this gene on the chromosome.
experimentid	int	the experiment that this gene was studied on.
geneabbrev	varchar	the abbreviation of the gene name.
regulation	varchar	the expression of the gene in the experiment.

6.3.3 Factor Matches

Column Name	Type	Description
seqid	varchar	indicates the sequence which this factor matches.
study	varchar	the biological study which this factor match was obtained from (indirectly through TESS).
transfac	varchar	the name of this factor
la	double	a factor quality indicator (how reliable the relationship between this factor and its related sequence are).
la_slash	double	a factor quality indicator.
lq	double	a factor quality indicator.
lq	double	a factor quality indicator.
ld	double	a factor quality indicator.
lpv	double	a factor quality indicator.
sc	double	a factor quality indicator.
sm	double	a factor quality indicator.
spv	double	a factor quality indicator.
ppv	double	a factor quality indicator.

6.3.4 Experiments

Column Name	Type	Description
label	varchar	The experimenter-designated name for the experiment.
comparisontypeid	int	Specifies the comparison type and species that this experiment is performed on.
tessjob	varchar	The TESS job number assigned to the experiment's results.
experimenter_email	varchar	Email address of the experimenter who performed the experiment.

6.3.5 Regulatory Sequences

Column Name	Type	Description
beginning	int	start nucleotide on the promoter sequence where this regulatory sequence begins.
length	int	number of nucleotides in this regulatory sequence.
sense	char	direction of regulation of this sequence
geneid	int	gene which this sequence regulates.

6.3.6 Promoter Sequences

Column Name	Type	Description
geneid	int	the gene ID of this promoter sequence.
sequence	varchar	the 2000 base pair nucleic acid sequence that is the promoter region for this gene.

6.3.7 Study Pages

Column Name	Type	Description
pageno	varchar	the page identifier for the study.
seqid	int	the sequence which this page is referenced in.

6.3.8 Users

Column Name	Type	Description
username	varchar	login name for the user.
password	varchar	the SHA1 hash of the user's password.
display_name	varchar	when the user is logged in, this name is displayed in the navigation bar.
privileges	int	indicates what privileges the user have. Read = 0, Write = 10, Administrator = 20. For more information on privileges, see the account management page.

6.4 Code Listing

```
# Dump of table apriori_staging

DROP TABLE IF EXISTS ‘apriori_staging‘;

CREATE TABLE ‘apriori_staging‘ (
  ‘id‘ int(11) NOT NULL AUTO_INCREMENT,
  ‘geneid‘ int(11) NOT NULL,
  ‘tf_cart‘ text CHARACTER SET latin1 NOT NULL,
  PRIMARY KEY (‘id‘)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin;


# Dump of table comparison_types

DROP TABLE IF EXISTS ‘comparison_types‘;

CREATE TABLE ‘comparison_types‘ (
  ‘comparisontypeid‘ int(11) unsigned NOT NULL AUTO_INCREMENT,
  ‘species‘ varchar(64) COLLATE utf8_bin NOT NULL,
  ‘celltype‘ varchar(255) COLLATE utf8_bin NOT NULL,
  ‘hidden‘ tinyint(1) NOT NULL,
  ‘date_edited‘ int(11) NOT NULL,
  PRIMARY KEY (‘comparisontypeid‘),
  UNIQUE KEY ‘species‘ (‘species‘,‘celltype‘)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin;


# Dump of table experiments

DROP TABLE IF EXISTS ‘experiments‘;

CREATE TABLE ‘experiments‘ (
  ‘experimentid‘ int(11) unsigned NOT NULL AUTO_INCREMENT,
  ‘label‘ varchar(255) COLLATE utf8_bin NOT NULL,
  ‘tessjob‘ varchar(255) COLLATE utf8_bin NOT NULL,
  ‘comparisontypeid‘ int(11) unsigned NOT NULL,
  ‘experimenter_email‘ varchar(255) COLLATE utf8_bin NOT NULL,
  ‘storage_time‘ text COLLATE utf8_bin,
  ‘search_transfac_strings‘ tinyint(1) DEFAULT NULL,
  ‘search_my_site_strings‘ tinyint(1) DEFAULT NULL,
  ‘selected‘ tinyint(1) DEFAULT NULL,
  ‘search_transfac_matrices‘ tinyint(1) DEFAULT NULL,
  ‘search_imd_matrices‘ tinyint(1) DEFAULT NULL,
  ‘search_cbil_matrices‘ tinyint(1) DEFAULT NULL,
  ‘search_jaspar_matrices‘ tinyint(1) DEFAULT NULL,
  ‘search_my_weight_matrices‘ tinyint(1) DEFAULT NULL,
  ‘combine_with‘ text COLLATE utf8_bin,
  ‘factor_attr_1‘ text COLLATE utf8_bin,
  ‘matches‘ text COLLATE utf8_bin,
  ‘use_core_positions‘ tinyint(1) DEFAULT NULL,
  ‘max_mismatch‘ int(11) DEFAULT NULL,
  ‘min_log_likelihood‘ int(11) DEFAULT NULL,
  `min_strlen` int(11) DEFAULT NULL,
  `min_lg` double DEFAULT NULL,
  `group_selection` text COLLATE utf8_bin,
  `max_lg` int(11) DEFAULT NULL,
  `min_core` double DEFAULT NULL,
  `min_matrix` double DEFAULT NULL,
  `secondary_lg` int(11) DEFAULT NULL,
  `count_significance` double DEFAULT NULL,
  `pseudocounts` double DEFAULT NULL,
  `use_at` double DEFAULT NULL,
  `explicit_acgt` text COLLATE utf8_bin,
  `handle_ambig` text COLLATE utf8_bin,
  `hidden` tinyint(1) NOT NULL,
  `date_edited` int(11) NOT NULL,
  PRIMARY KEY (`experimentid`),
  UNIQUE KEY `label` (`label`),
  KEY `comparisontypeid` (`comparisontypeid`),
  CONSTRAINT `experiments_ibfk_1` FOREIGN KEY (`↙
      ↪ comparisontypeid`) REFERENCES `comparison_types` (`↙
      ↪ comparisontypeid`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin;


# Dump of table factor_matches

DROP TABLE IF EXISTS `factor_matches`;

CREATE TABLE `factor_matches` (
  `matchid` int(11) unsigned NOT NULL AUTO_INCREMENT,
  `seqid` int(11) unsigned NOT NULL,
  `study` varchar(255) COLLATE utf8_bin NOT NULL,
  `transfac` varchar(32) COLLATE utf8_bin NOT NULL,
  `la` double NOT NULL,
  `la_slash` double NOT NULL,
  `lq` double NOT NULL,
  `ld` double NOT NULL,
  `lpv` double NOT NULL,
  `sc` double NOT NULL,
  `sm` double NOT NULL,
  `spv` double NOT NULL,
  `ppv` double NOT NULL,
  `hidden` tinyint(1) NOT NULL,
  `date_edited` int(11) NOT NULL,
  PRIMARY KEY (`matchid`),
  UNIQUE KEY `seqid` (`seqid`,`study`,`transfac`),
  KEY `tfKey` (`transfac`,`study`,`seqid`),
  CONSTRAINT `factor_matches_ibfk_1` FOREIGN KEY (`seqid`) ↙
      ↪ REFERENCES `regulatory_sequences` (`seqid`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin;


# Dump of table genes

DROP TABLE IF EXISTS `genes`;
```

```

CREATE TABLE ‘genes‘ (
  ‘geneid‘ int(11) unsigned NOT NULL AUTO_INCREMENT,
  ‘genename‘ varchar(255) COLLATE utf8_bin NOT NULL,
  ‘chromosome‘ smallint(2) NOT NULL,
  ‘start‘ int(11) NOT NULL,
  ‘end‘ int(11) NOT NULL,
  ‘experimentid‘ int(11) unsigned NOT NULL,
  ‘geneabbrev‘ varchar(32) COLLATE utf8_bin NOT NULL,
  ‘regulation‘ varchar(20) COLLATE utf8_bin NOT NULL,
  ‘hidden‘ tinyint(1) NOT NULL,
  ‘date_edited‘ int(11) NOT NULL,
  PRIMARY KEY (‘geneid‘),
  UNIQUE KEY ‘experimentid‘ (‘experimentid‘,‘genename‘),
  CONSTRAINT ‘genes_ibfk_1‘ FOREIGN KEY (‘experimentid‘) ↙
      ↳ REFERENCES ‘experiments‘ (‘experimentid‘)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin;


# Dump of table promoter_sequences

DROP TABLE IF EXISTS ‘promoter_sequences‘;

CREATE TABLE ‘promoter_sequences‘ (
  ‘geneid‘ int(11) unsigned NOT NULL,
  ‘sequence‘ text COLLATE utf8_bin NOT NULL,
  PRIMARY KEY (‘geneid‘),
  CONSTRAINT ‘promoter_sequences_ibfk_1‘ FOREIGN KEY (‘geneid‘)↙
      ↳ REFERENCES ‘genes‘ (‘geneid‘)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin;


# Dump of table regulatory_sequences

DROP TABLE IF EXISTS ‘regulatory_sequences‘;

CREATE TABLE ‘regulatory_sequences‘ (
  ‘seqid‘ int(11) unsigned NOT NULL AUTO_INCREMENT,
  ‘beginning‘ int(11) NOT NULL,
  ‘length‘ int(11) NOT NULL,
  ‘sense‘ char(1) COLLATE utf8_bin NOT NULL,
  ‘geneid‘ int(11) unsigned NOT NULL,
  ‘hidden‘ tinyint(1) NOT NULL,
  ‘date_edited‘ int(11) NOT NULL,
  PRIMARY KEY (‘seqid‘),
  UNIQUE KEY ‘geneid‘ (‘geneid‘,‘beginning‘,‘length‘,‘sense‘),
  CONSTRAINT ‘regulatory_sequences_ibfk_1‘ FOREIGN KEY (‘geneid↙
      ↳ ‘) REFERENCES ‘genes‘ (‘geneid‘)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin;


# Dump of table study_pages

DROP TABLE IF EXISTS ‘study_pages‘;
```

```
CREATE TABLE `study_pages` (
  `pageno` char(7) COLLATE utf8_bin NOT NULL,
  `seqid` int(11) unsigned NOT NULL,
  PRIMARY KEY (`seqid`,`pageno`),
  UNIQUE KEY `pageno` (`pageno`,`seqid`),
  CONSTRAINT `study_pages_ibfk_1` FOREIGN KEY (`seqid`) ↙
      ↳ REFERENCES `regulatory_sequences` (`seqid`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin;


# Dump of table users

DROP TABLE IF EXISTS `users`;

CREATE TABLE `users` (
  `id` int(11) unsigned NOT NULL AUTO_INCREMENT,
  `username` varchar(20) CHARACTER SET utf8 NOT NULL DEFAULT ↙
      ↳ '',
  `password` varchar(128) CHARACTER SET utf8 NOT NULL DEFAULT ↙
      ↳ '',
  `date_created` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `privileges` tinyint(4) NOT NULL DEFAULT '0',
  `display_name` varchar(60) CHARACTER SET utf8 NOT NULL ↙
      ↳ DEFAULT '',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin;
```

7 Glossary

promoter region: a region of DNA that initiates transcription of a particular gene.

differentiation: the process by which a less specialized cell become a more specialized cell type.

gene: a set of instructions that code for a specific protein or function within an organism.

transcription factor: a regulatory protein that binds to a specific sequence of DNA and controls the number of mRNA that are transcribed.

regulatory sequence: a segment of DNA which is capable of controlling the expression of a gene within an organism.

protein: one or more chains of amino acids that perform various functions for cells.

transcription: the process of creating messenger RNA by copying the DNA strand. mRNA subsequently exits the nucleus and is translated into a working protein.

activation: refers to the initiation of transcription of a particular gene.

mammary gland: an organ in female mammals that produces milk to feed young offspring.

messenger RNA (mRNA): RNA molecules that carry genetic information from the nucleus, where DNA is, to the cytoplasm, where proteins are made.

sense: the direction of translation that is associated with a regulatory sequence. Can either be normal or reverse.

References

[1] Therin Irwin. Frequent Itemset Generator - GitHub Repository. https://github.com/brovine-developers/freq-itemset-gen, June 2013.

[2] Therin Irwin, Sterling Hirsh, Ryan Schroeder, and Trevor Devore. Brovine Database SQL Schema. http://brovine.csc.calpoly.edu/files/brovine_schema.sql, March 2012.

[3] Therin Irwin, Sterling Hirsh, Ryan Schroeder, and Trevor Devore. Brovine - GitHub Repository. https://github.com/brovine-developers/teambrovine, June 2013.

[4] Dan Peterson. Gene Promoter Database Description. http://brovine.csc.calpoly.edu/files/project-goals.pdf, January 2012.

[5] Pang-Ning Tan et al. *Introduction to data mining*. Pearson Education, 2007.