Interactive GFF3 File Viewer Process Book

Andrew Miller: andrew.c.miller@utah.edu u0702603
Michael Watkins: michael.watkins@utah.edu, u1147057
Project Repository: https://github.com/acmiller015/Interactive-GFF3-Viewer

Overview and Motivation

This process book outlines the design and development of the Interactive GFF3 File Viewer web application. The project objectives and scope are described in detail in the Project Proposal section below and will briefly be described here. The Interactive GFF3 File Viewer serves as a visualization and exploration tool for bioinformaticians. The aim of this app is to improve the users' overall understanding of GFF3 file contents while also enabling them to perform a detailed investigation of the file's contents. GFF3 files store genome annotations in tab-delimited columns and can be extremely large. Manually scrolling through these lines is not feasible and some software tools have been created to calculate summaries of the file contents. However, none of these utilize visualization.

Both of the team members (Andrew Miller and Michael Watkins) are PhD students in the Biomedical Informatics department and have a primary interest in the field of genomics. Both team members work in the Eilbeck lab which maintains the Sequence Ontology (SO). SO is very closely related to the GFF3 standard as both are used in the process of genome annotation. Karen Eilbeck, the creator of SO, suggested to the team members that GFF3 files were becoming more widely used but that there was a lack of tools which analyze those files.

Related Work

The Integrative Genomics Viewer (IGV) is a genome browser which is used to visually analyze many different types of genomic files.[1] It allows you to upload a GFF3 file and gives you the initial view seen in Figure 1.

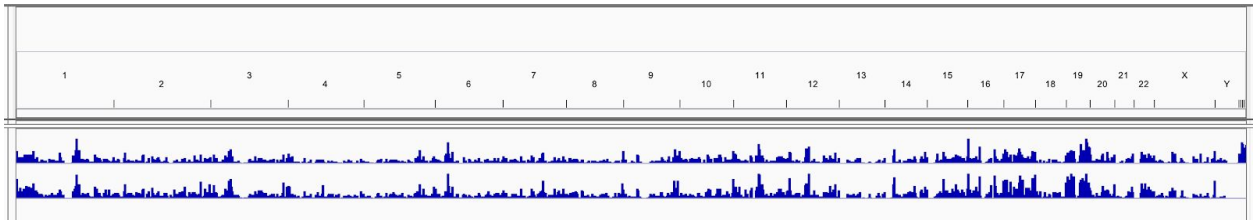

Figure 1 - IGV initial view

This is an extremely high level view of the entire genome. It does use visualization but doesn't allow you to gauge how many types are in the file itself. It also requires a lot of manual zooming to get to the level of detail that actually show the features from the GFF3 file, as shown in Figure 2.

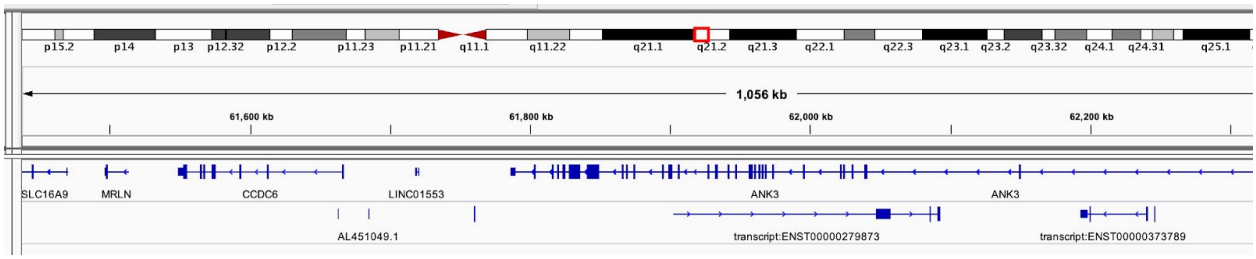

Figure 2 - IGV zoomed-in view

However, since most researchers are interested in the quantity of features, the team decided that a viewer which presented more of a list view with effective filtering would be more useful than a genome-level view which required manual zooming to find relevant features.

Another tool, Blast2Go, is very similar to IGV. It is a genome-level view of features in the file which allows you to scroll through genomic locations and see the density of features at those locations.[2] However, it doesn't provide file-wide summaries of type totals or allow you to easily filter the features you want to investigate. If you don't know where the location is then you can't view the information.

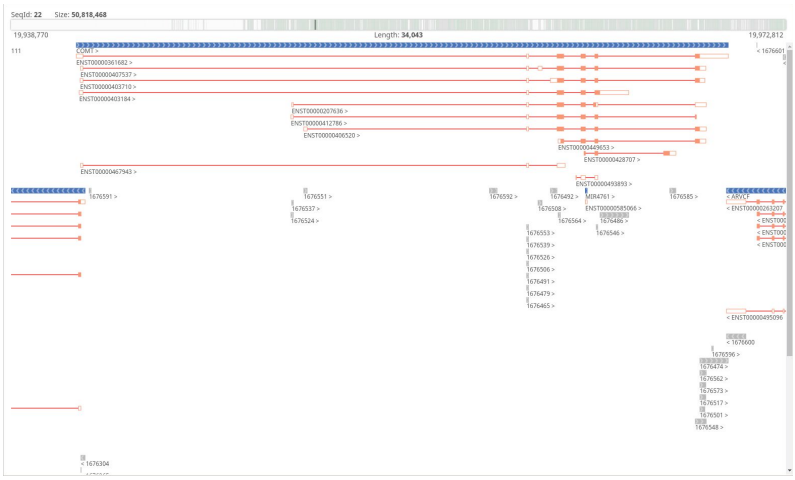

Figure 3 - Blast2Go example view

While these tools are useful in some cases, they are not conducive to hypothesis generation. They allow you to investigate areas of interest, but don't assist you in finding out why certain areas may be interesting in the first place. The team wanted to create a visualization tool which would help the researcher identify patterns or other indications of interest based on the contents of the GFF3 file.

Initial Brainstorming Session (10/19/18)

To identify key pieces of information in GFF3 files and gain a deeper understanding of the file specification we organized a brainstorming session between our team and Karen Eilbeck, a professor in the Department of Biomedical Informatics. In this session we identified key pieces of information that would be useful for the user and potential ways to visualize that information. A picture of the session is contained in Figure 4.

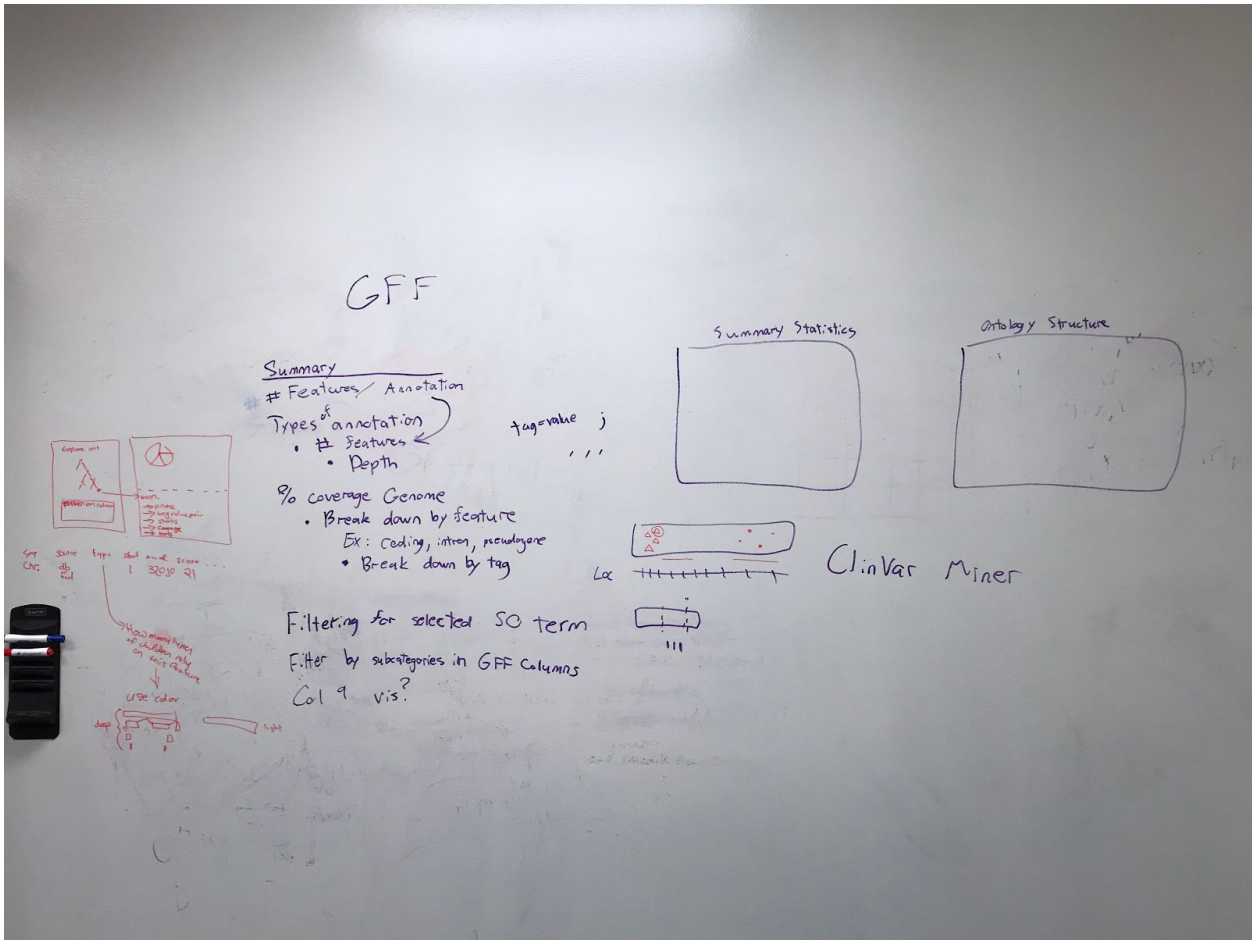

Figure 4 - Brainstorming Session Picture

Project Proposal - 10/26/18

Background and Motivation

The General Feature Format (GFF) specification was created to allow researchers to store annotated genomic features in a standardized way. It is a tab-delimited representation which lists a feature as well as various details about that feature. Examples of genomic features are gene, CDS, mRNA, etc. Details about a genomic feature would include the start and end location of that feature on a genomic sequence, the program which assigned that feature to the genome, the quality of that feature, and the controlled vocabulary which the feature is drawn from. Version 3 of this specification (GFF3) was developed as a solution to many common extensions that were previously being applied ad hoc by different researchers. It is an effort to increase the level of standardization in genomic research while providing sufficient flexibility.

Genomic features are drawn from controlled vocabularies such as the Sequence Ontology. This means that groups of features may have hierarchical relationships which can go unnoticed in a tab-delimited file format. By using various visualization marks and channels, these files could be represented in a way which would allow a researcher to more easily assess the quality of their

file and be able to visually detect patterns or relations in the features which would not be otherwise apparent.

Although GFF3 files are becoming increasingly important in applied genomics, little has been done to assist researchers in exploring their features. Other tools primarily focus on visualizing the start and end positions of a feature and lack effective summary visualizations of the entire file and the relationship between different features. Creating an interactive GFF3 file viewer can enable researchers to quickly understand the file content which can serve as an informative step in many applied genomics pipelines.

Project Objectives

We want to explore how different modes of visualization and interaction affect the ability of researchers to draw conclusions from the contents of GFF3 files. The following questions represent what we aim to explore during the course of this project:

Will showing visual representations of the relationships between features provide additional insights into their biological interactions?

- Benefits: the genomic features in GFF3 files come from ontologies which means that relationships between them are currently available. However, there are no tools which take advantage of these relationships or use them to assist the researcher. We anticipate that visualization of these relationships will provide the researcher with better insight of their data.

Is a tree view the best way to show those relationships?

- Benefits: tree visualizations have been used to represent ontologies because it allows a user to infer function based on knowledge about relationships to parent or children nodes. This type of visualization could be a good fit for genomic features which many times are interconnected.

What kinds of summary metrics can we synthesize and display that will give the user a quick way to gauge the quality of the file as a whole?

- Benefits: Sifting manually through an extremely long tab-delimited file is a labor-intensive process. Being able to dynamically calculate global summaries about the contents of a file can give a researcher a quick glimpse at their file. They can see if their file contains what they think it should contain. For example, typically RNA sequencing experiments require rRNA be depleted from the sample before library prep and sequencing. An unusually high number of rRNA features would indicate that the rRNA depletion step wasn't performed effectively and may warrant additional sample collection before further analysis.

What fields will be the most useful to implement filtering on?

- Benefits: Obviously the ability to filter will be an important component, but should all fields have filters? Are there some fields which should be automatically filtered right when the viewer is loaded? There may be some fields which should be filtered in

tandem. For example, the quality score of a feature is specific to the tool which generated it (i.e. not all scores can be directly compared). So, filtering by quality should automatically filter the features by the tool used to generate them. Identifying other such relationships has much potential benefit.

Are there any extraneous fields which can be shown via tooltips so that focus can be directed toward important fields?

- Benefits: Tab-delimited files require the use of a different tools to re-order the data such as command line tools or table-like programs such as Microsoft Excel. However, in these programs all of the columns are included which may clutter the column of interest. An interactive viewer gives us the potential to shift the paradigm and determine if some fields should be cut out of the immediate visualization and only be shown on-hover as a tooltip. This would reduce the amount of data in the main visualization which can help highlight patterns which may otherwise be obfuscated.

Data

Many types of sequencing data will need to be tested to ensure our application is generalizable to different sequencing files that comply with the GFF3 file format specification. There are many publicly available GFF3 files from the following sites. Ensembl provides many different species, NCBI has a vast number of human files, VectorBase contains samples of vertebrate vectors from pathogens, and Tair provides samples for a common model plant. We anticipate that this wide variety of test data will help us avoid inadvertently tailoring the browser to a specific type of organism/sample type such as human samples, or Ensembl samples.

Ensembl gene sets: https://uswest.ensembl.org/info/data/ftp/index.html
NCBI RefSeq Reference Genome Annotations:
https://www.ncbi.nlm.nih.gov/projects/genome/guide/human/
VectorBase:
https://www.vectorbase.org/downloads?field_status_value=Current&field_download_file_type_tid=470&page=8
Tair:
https://www.arabidopsis.org/download/index-auto.jsp?dir=%2Fdownload_files%2FGenes%2FTAIR10_genome_release%2FTAIR10_gff3

Data Processing

This project is somewhat atypical in that the main point is not to explore a certain dataset, but a certain specification. Our interactive viewer will be compatible with any GFF3 compliant file, despite what the contents of that file may be. As a results of that, our viewer must be dynamic and able to handle empty fields. Additional processing will be done on the file when loaded in order to generate the aforementioned summary statistics. We will need to process how many of each feature there are and other statistics included in the summary. Data processing will be limited to ensuring that the files we use are GFF3 compliant which is mitigated since public databases currently contain GFF3 compliant files.

Prototype Development:

Prototype 1:

The graph for the Sequence Ontology will be described first as it is a common visualization in both Prototype 1 and Prototype 2. The Sequence Ontology visualization will take a feature from the loaded GFF3 file and create a tree that shows it's location with respect to other features in the ontology. Furthermore, when the user clicks on a feature on the graph, the corresponding table on the left will be filtered to only contain features corresponding to the feature selected in the graph. The user can also increase the number of levels shown in the tree.

The file data display on the left will incorporate basic ascending and descending sorting for each of the columns. The user can also "remove" columns of data by right clicking on a column. The column will then be replaced by an ellipsis (...) for the headed and a thin column will be separating the adjoining columns.

The radar plot in the top right corner displays the coverage of a selected feature, i.e. how much of the chromosome is composed of that specific feature. With the concentric circles indicating percent coverage.

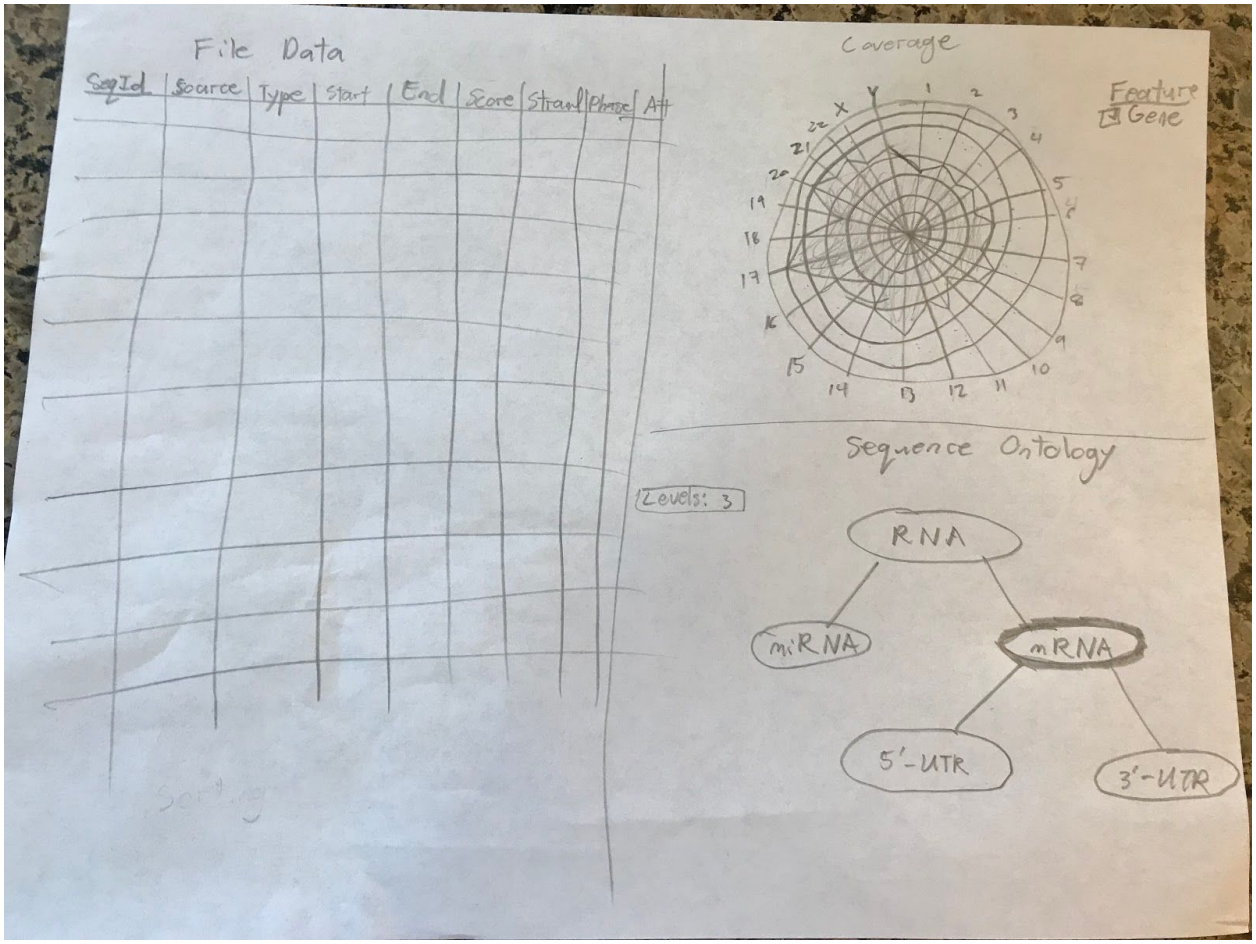

Figure 5 - Prototype 1.

Prototype 2:
Summary statistics are displayed in the top panel. On the left of this panel a list of features are displayed with corresponding checkboxes. The corresponding pie charts are then updated on the right. The first pie chart shows the relative abundance of the selected features for the entire file. An other category is added to each pie chart to account for features that have not been selected to avoid making the display too cluttered. The subsequent pie charts correspond to chromosomes. Each chromosome pie chart can be substituted for another one allowing the user to compare multiple chromosomes at the same time.
Similar to Prototype 1, the Sequence Ontology tree is included as one of the visualizations. When a feature is selected for on the tree, the corresponding feature is highlighted on the feature position display on the right.
The feature position display is similar to other visualizations, such as the UCSC Genome Browser, that display the feature by genomic location. The button at the top of the visualization allows the user to select a chromosome which will update the positional chart below for the selected features in the top panel.

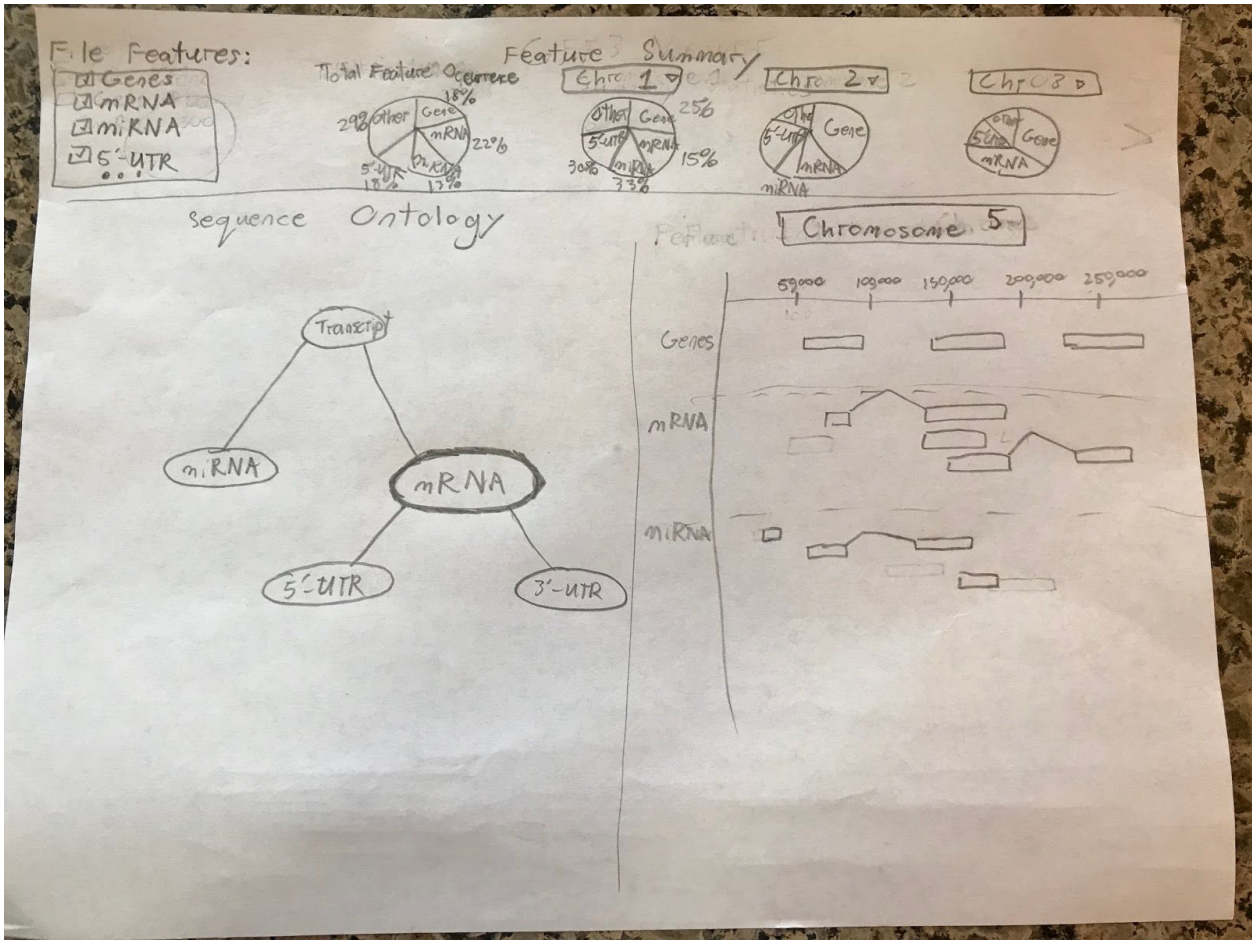

Figure 6 - Prototype 2

Prototype 3

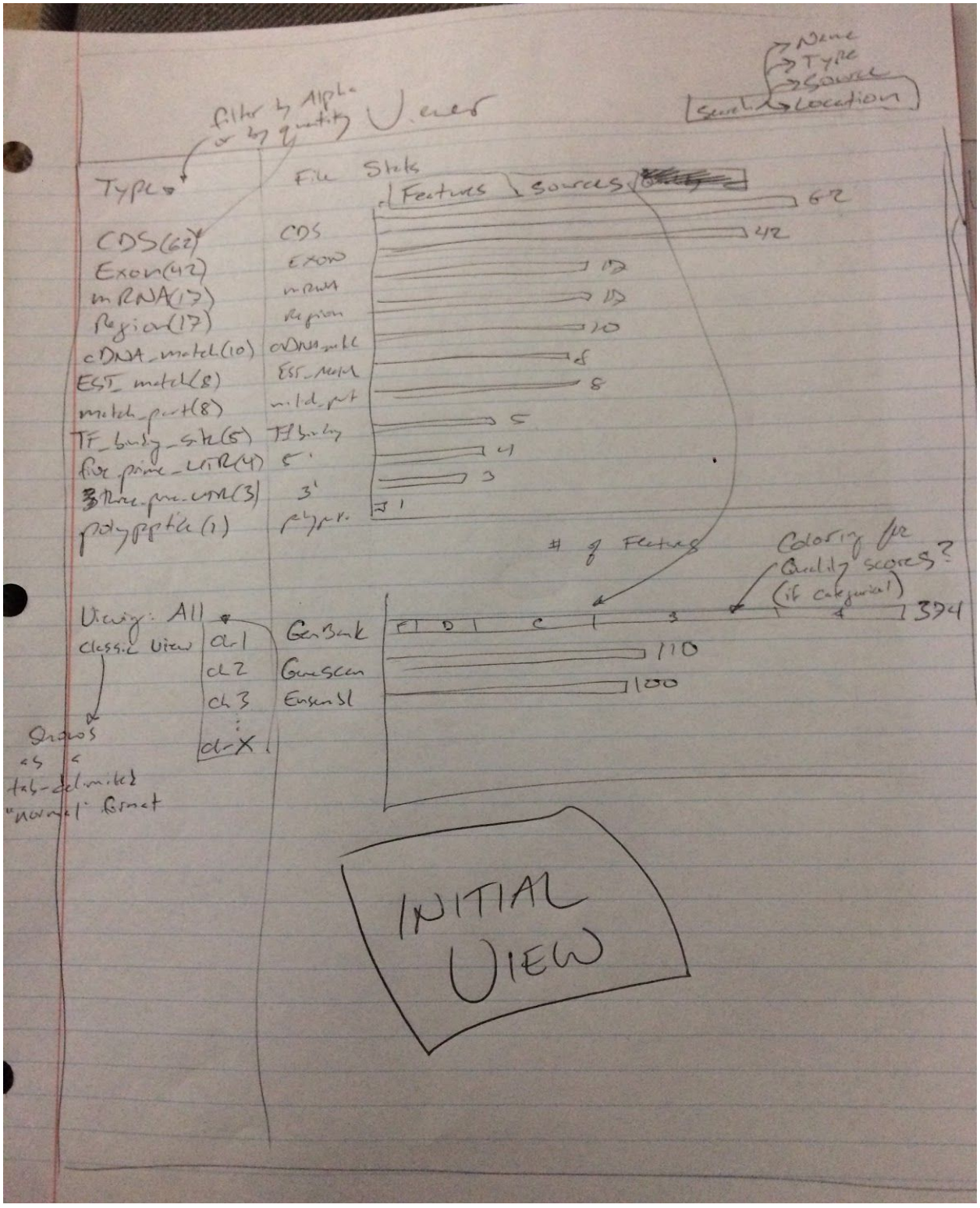

Figure 7 - Prototype 3 Initial View

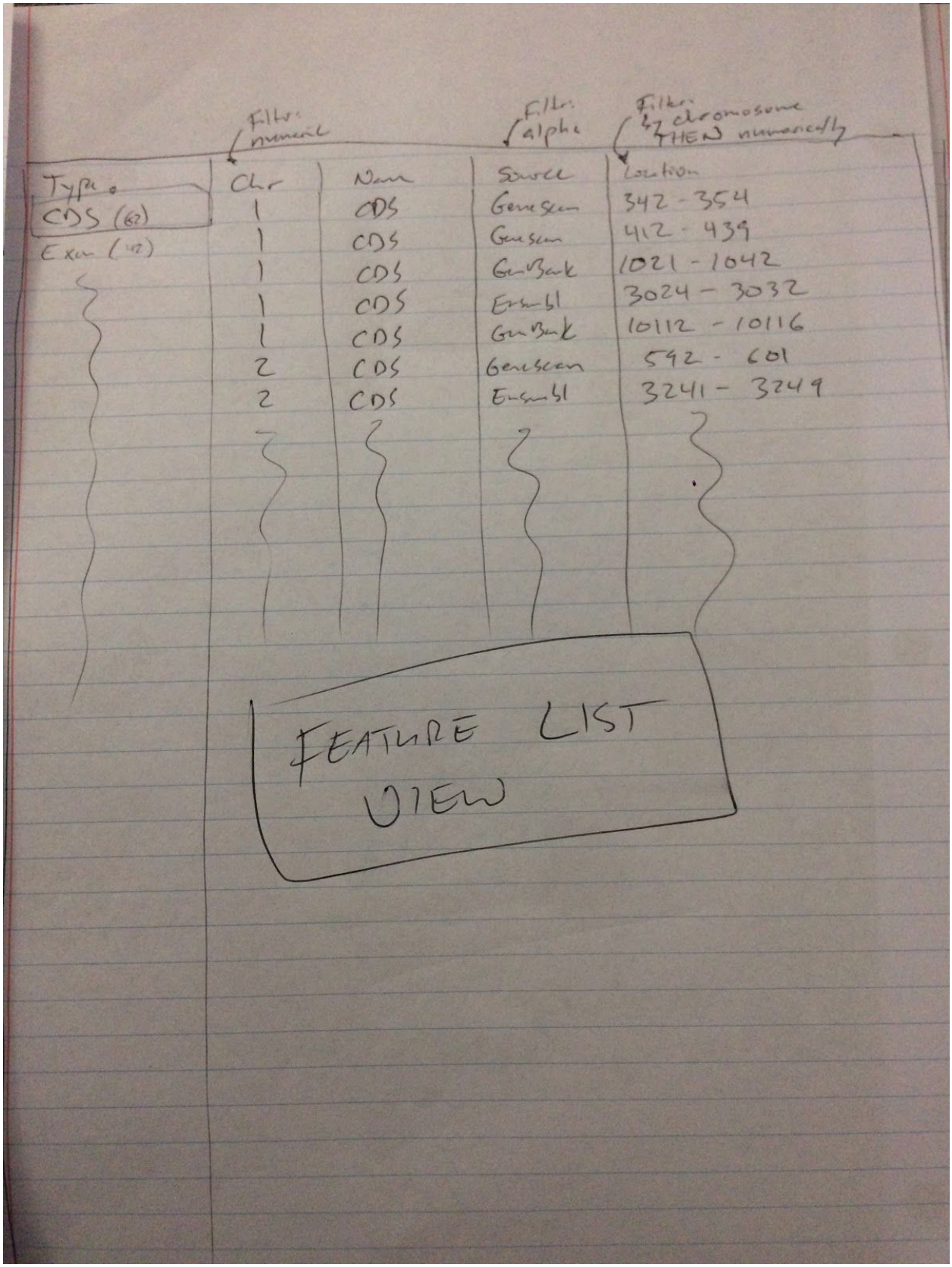

Figure 8 - Prototype 3 Feature List View

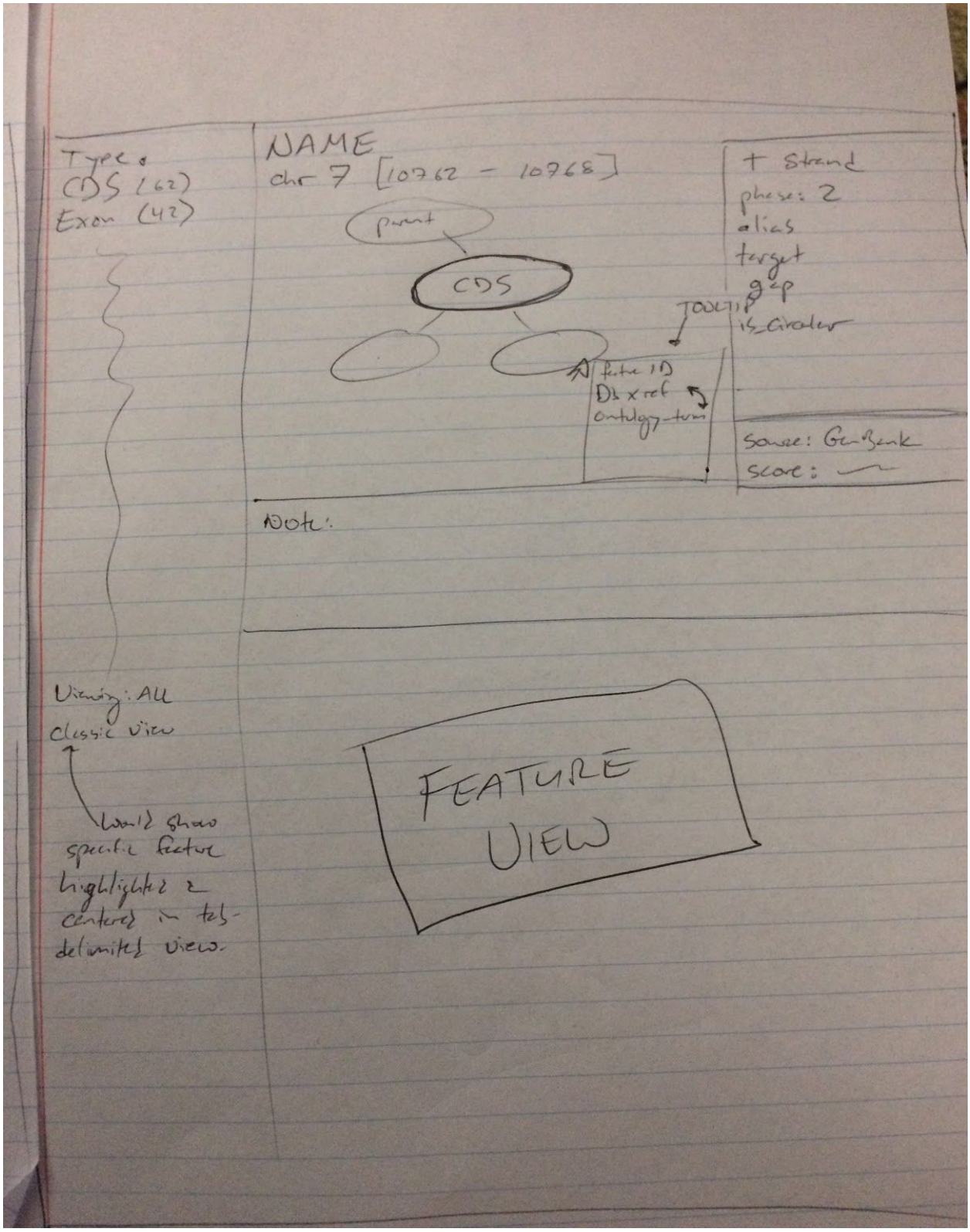

Figure 9 - Prototype 3 Feature View

Initial View: this default view would show file-wide summary statistics with a fixed panel on the left showing the types and how many of each there are in the file. This panel can be filtered alphabetically or by quantity. The default is to view features from all chromosomes together but this can be changed on that panel to view only for a specific chromosome. Since this panel is fixed through each of the three views, this chromosome selection is always easily available to be changed. There is also a “classic” view option which would show the file in its original tab-delimited form. A search bar would be able to search by feature name, feature type, feature source, or feature location. This would be an optional feature if we have time but the view (not included) would probably resemble the feature list view. The statistics would take the form of 2 bar charts. These are available through tabs. The default would be to show the number of each feature type in the file on the bar chart. This is a visual form of the data in the left panel. The other tab would be to view a bar chart of the sources where the features came from. This would show the source and a bar chart of how many features came from that source. The bars themselves can be colored to visually encode the quality of the features they supplied, or can also be color coded to show the types of features it supplied. The latter would require a legend. Clicking on a feature type from the left panel would trigger the feature list view.

Feature List View: this view will highlight the feature type being viewed on the left panel and would bring it to the top. The list would include 4 fields which we are hoping to be descriptive enough to differentiate between features (this needs to be further evaluated), and are chromosome, name, source, and location. The name will be the same for everything in the list but reinforces the idea that they are viewing a list of specific features and may merit the redundancy. Filtering can be performed numerically on the chromosomes, alphabetically on the sources, and for the location would be first by chromosome and then numerically. The left panel will still be available for them to designate a certain chromosome and avoid a lot of scrolling. Clicking the “classic” view option on the left panel from the feature list view will show the features of that type all highlighted and filtered to be together in the tab-delimited file. Clicking on a feature in the feature list would trigger the feature view.

Feature View: this view gives the fine-grained details specific to that feature. Name, chromosome, and location are displayed prominently at the top. The ontological information associated with that feature will be used to generate a tree structure showing relationships to other features in the file. Hovering over these features will prompt a tooltip showing a minimally viable amount of identifying data. Clicking on other elements in the tree will trigger the feature view for that feature. A right panel will show other details for the feature such as strand, phase, alias, target, gap, and is_circular which all may not be included for each feature. We are hoping that a right panel that lists them very basically will make it easier to handle missing fields since those fields are the most commonly missing ones. Below the tree and panel will be a note field for notes associated with that feature. Clicking on the “classic” view option on the left panel from the feature view will show that feature in its original place in the tab-delimited file but will center it on the screen and highlight it.

Realization Design:

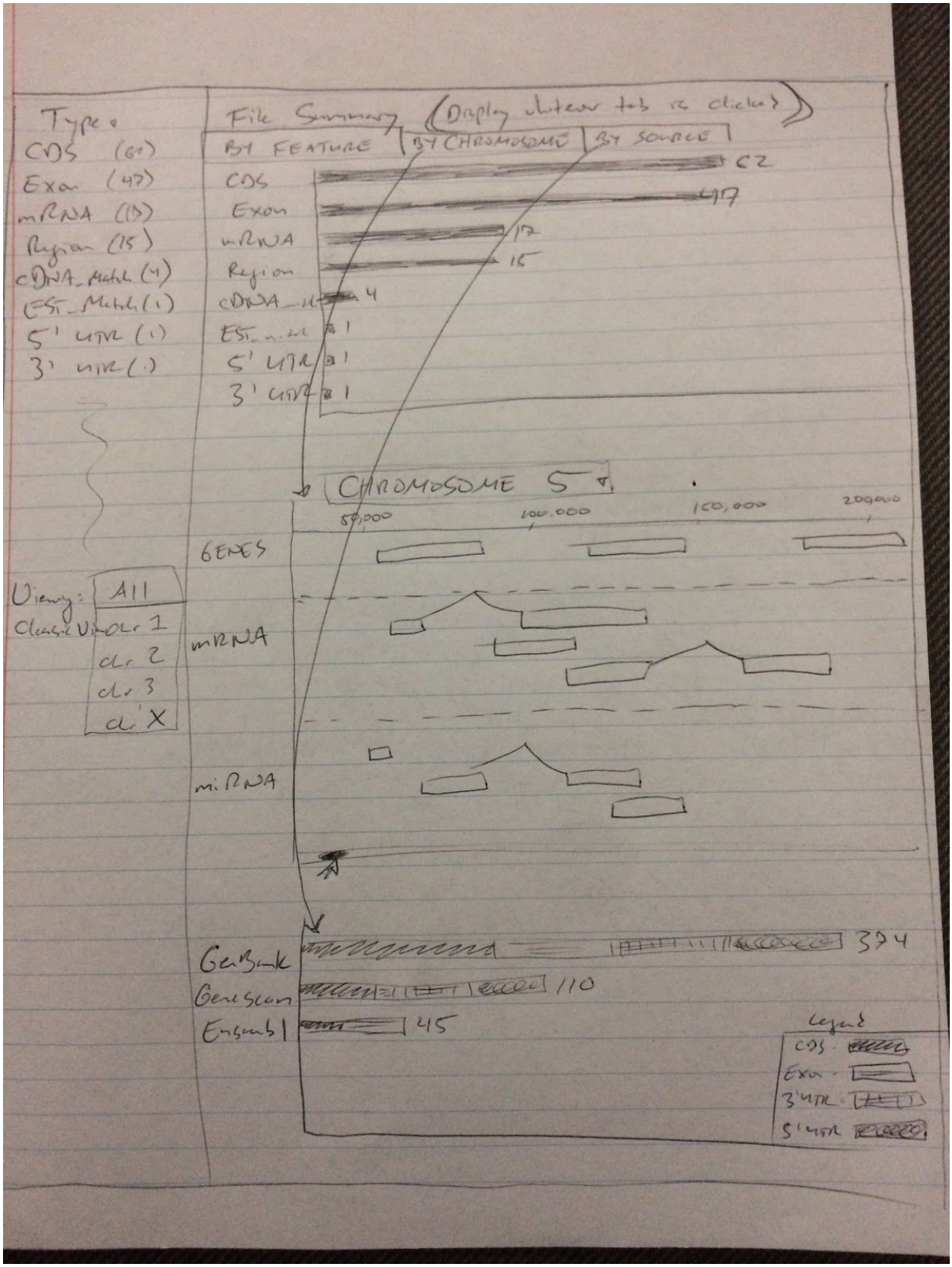

Figure 10 - Final Design Initial View

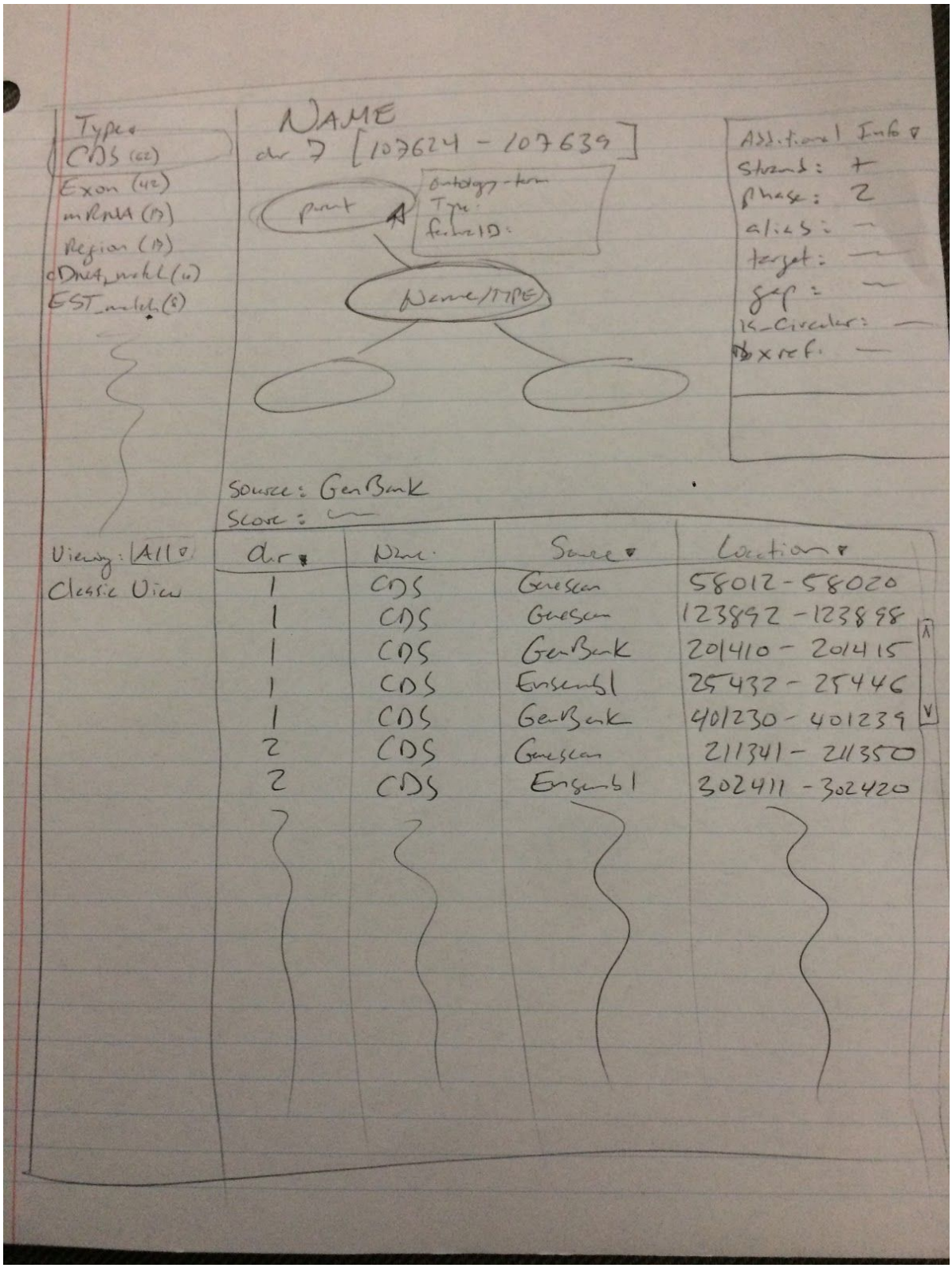

Figure 11 - Final Design Feature View

Our final design has two views. The first view is the landing view which will be launched after a file is uploaded. It will provide, file-level statistics and summaries. A fixed left panel will display all of the detected features and their amounts. It also has an option to filter by chromosome or view the “classic” representation of the file (i.e. tab-delimited rows). There will be three visualization options made accessible through tabs. The default will be a bar chart representation of the data in the left panel. Each type will be represented as a row in a graph with the length of the bar representing the amount of features of that type in the file. Another tab will replace the bar chart with a location-based visualization which shows each feature as a rectangle spanning its location on the chromosome selected at the top of the visualization. The user can scroll sideways along the entire length of the chromosome. Each feature will have its own row in the visualization. The third tab will replace the location-based visualization with another bar chart representing the sources of the features. Each source will be a row in the chart and the length of the bars will represent the number of features from that source These bars will be color-coded to give a visualization for how many of each type of feature came from that source. A legend will be provided. A click on any feature or type will activate another view giving details on that feature or type. This view will also have the same fixed left bar giving totals for the types. This view will have a top view and a bottom view. The bottom view will be a list of all the features of the given type. This list will display the chromosome, feature name, source, and location. As the user selects a feature from the list it will populate the upper view which gives feature-level details. This includes the name, chromosome, and location, which will be prominently displayed, as well as a tree visualization of associated ontology terms. On the right will be a list which will be collapsed by default but which can be expanded to give the user the rest of the details from the file for that feature. The source and score for that feature will be displayed below the tree.

Must-Have Features

- Filter-enabled feature detail display.
- Basic feature tree.
- Descriptive summary statistics.

Optional Features

- Customizable filtering combinations.
- Ability to expand/shrink portions of the feature tree on-click.
- Search bar
- “Classic” view

Project Schedule

10/28 - 11/3

[Mandatory peer feedback 10/30]

- Architecture planned and file structure created.
- Test data and use cases gathered.
- Feature details parsed and given basic display.

11/4 - 11/10

[Project milestone due 11/9]

- Basic tree created for feature relationships.

11/11 - 11/17

- More robust display for feature details.
    - Filters
    - File-wide metrics

11/18 - 11/24

- More robust display for the feature tree.
    - Collapse / expand
    - Tooltips

11/25 - 11/30

[Project due 11/30]

- Polish and submit.

Initial Questions

- What are the most useful information summaries we can generate for the contents of GFF3 files.
- How can we encode features by chromosome?
- What information can we include in the tree representation of the features?
- Which fields are most useful to enable filtering on?
- How can we keep the viewer from feeling too foreign to users who are used to classic methods of viewing the file?

Data

As mentioned in the proposal, our project requires very little data cleanup. One problem that affected us throughout the design process was that public GFF3 files tend to be very large. In order to get test files which had good representation of types per chromosome, we used human genome reference GFF3 files. These are gold standard files which contain aggregated features from many individuals. It gives a nice breadth of data but they are a lot bigger than standard GFF3 files. We pulled them from the Ensemble's Human page.[3] Since the files were so large, we went through and did some manual trimming where we deleted features whose types and sources were already well-represented in the file. This was done via Python script. This left us with a concise test file which had many different feature types and sources.

Exploratory Data Analysis

The biggest help we had in terms of exploratory analysis was the GFF3 specification which is hosted on the SO website.[4] This spec gave detailed explanations of each field contained in a valid GFF3 file. This was important because since we were only using a single manually-curated test file, we would otherwise be prone to unforeseen errors from just coding to a single file. The spec told us possible values each field could have as well as how to parse the files which would be uploaded to our viewer.

Peer Review Feedback - 10/30/18:
James Brisette (his second group member was not present)

James was really helpful. Even though our project is very specific to genomics, and he didn't have much background in it, he gave us some very practical suggestions on how to improve. It was almost better that he didn't have a lot of background because he was able to give us high-level fundamental issues to think about.

There were two main critics which he gave us:

- The first was a concern that although this may be a step up from command line manipulation and exploration of the file, we should still provide an easily accessible way for people to opt to use commands and searching methods that they are familiar with. He asked us if there was some way to mock the way that people currently search these kinds of files on the command line. I thought it was a very helpful critic because we were so anxious to improve the processes that we didn't give much thought to what people are already used to. In response to his critic, we think that the filtering options available may not be clear enough. We will test a basic implementation on a user who is very familiar with searching these files and use her feedback to determine whether or not we are addressing the concern that James surfaced for us.
- Another critic was that James thought there should be some way to compare features side-by-side. This was in response to our showing him our feature detail window which only shows one feature at a time. At his suggestion we are going to try to implement an option to add feature windows on screen. This will enable users to pull up two features (such as genes) and be able to visually compare amounts of parents and children, the different scores, etc. This was really helpful feedback and it was really critical that we got it so early on so that we can adjust our design before we begin coding.

Modifications to Current Design 11/9/18:
When plotting the contents of different GFF3 files we realized that there are many different features as well as different sources. This may negatively affect the users ability to differentiate between different sources if they are coded by color. To alleviate this issue we will incorporate a parallel coordinates plot from sources to types that connects to the types of elements in the GFF3 file.

Design Evolution
We considered a few different visualizations for the summary charts in the initial view. Most of these ideas stemmed from our desire to show a chromosome-level view. One prototype used a radar view which showed density of feature types by chromosome. Another included pie charts for each chromosome which would give a quick glance-view of which types made up that chromosome. However, when we decided to go with a chromosome HTML select option most of those initial plans went away. We also decided to put our type and source charts side-by-side rather than being accessed by tabs. We felt that this was more conducive to exploratory

analyses because it allowed you to leverage both information sources simultaneously. Another evolution was our detail tree. Initially this was going to be a master tree which would zoom and center on a selected feature in the file. Time constraints were a big reason why this design evolved, however there were also limitations in the file because not all sub-trees connected into a master tree. For these reasons we decided to show just a feature, its parent, and its children in a tree which would reset for each feature selected.

Question Evolution

What are the most useful information summaries we can generate for the contents of GFF3 files.

There are a lot of fields in GFF3 files and that was really hard to get past for the first few weeks. We were just swamped with possible data to display. We ended up settling on type totals and source totals as the most useful initial visualization summaries.

How can we encode features by chromosome?

Initially, we wanted to incorporate some sort of chromosome-specific charts for the summary visualizations but decided that it was just too much to display visually. We decided on an HTML select to toggle between chromosomes and just treat each chromosome as a new subset to load the displays with. This became a lot more important than we initially thought.

What information can we include in the tree representation of the features?

The tree actually became a less important aspect of the project the more we thought about it. The GFF3 files store parent and children ID references for features which have parents and/or children which was an initial draw to use a tree visualization. However, we found that the amount of novel data we could display in the tree (without being redundant to what is made available through the list) was minimal. The tree is an easy way to see what kinds of features are related to a feature of interest, but it wasn't as important as we initially thought.

Which fields are most useful to enable filtering on?

Filtering was a big question. We decided that location was probably one of the most important filters because this is what made features of the same type unique in regards to each other. We also found that filtering by source and type could be useful depending on if the list is launched from the type chart or the source chart. The search bar was a useful implementation whose need became more apparent as this question evolved.

How can we keep the viewer from feeling too foreign to users who are used to classic methods of viewing the file?

This idea of a "classic" view for the file (tab-delimited), seemed like an important feature in the beginning. However, as our prototypes became increasingly functional we saw the classic feature as somewhat redundant. It could be an interesting tool if a lot of functionality was encoded within that classic view. We didn't have the time to make that very functional so we left the classic view just as a text blob rendering of the file.

New Questions

- What new hypotheses could be generated if two files could be compared side-by-side?
- Can the viewer be generalized to accommodate non-human annotations?
- What kinds of simplified chromosome-level charts or visualizations could be utilized?

Implementation

Our viewer provides both file-wide and feature-specific visualizations. The first charts shown are file-wide and show both the total number of feature types shown in the file and the total percentage of sources which contributed each type. As shown in Figure 12, the source chart is aligned with the total chart by feature type. This allows us to use similar scaling and axis. A tooltip provides numeric values for the percentages shown in the source chart as stacked bars. A legend is also provided so that the user is not dependent on the tooltip in order to determine the meaning of the source colors. These charts give the user a brief glimpse at the contents of their file.

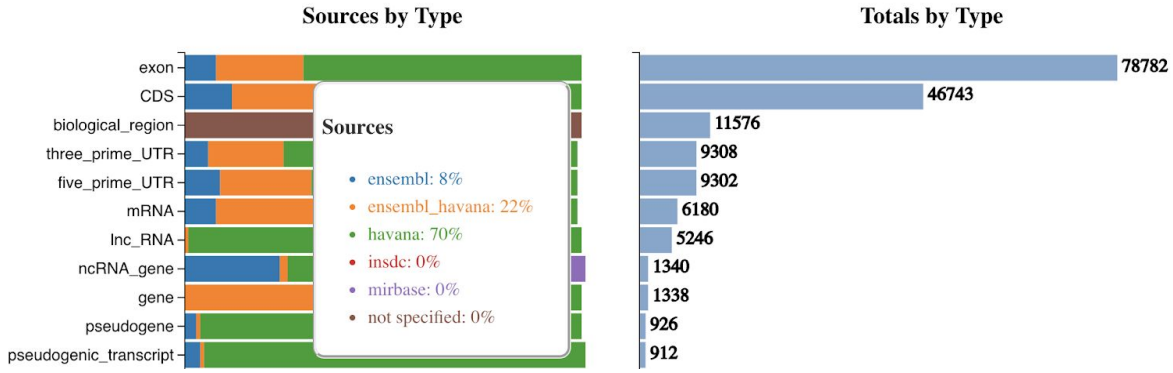

Figure 12 - Summary charts with tooltip for source percentages

Figure 13 - Source summary chart legend

Along with the summary charts, when a file is uploaded a sidebar also populates with type totals. This provides a more clear rendering of the type name. It is important because this sidebar persists once the summary charts are replaced with feature-specific visualizations. This ensures that the user always can toggle between certain feature types. It also holds the chromosome selector. Once a new chromosome is selected, the type totals on the sidebar and in each visualization only reflect features found on that chromosome. This is demonstrated in Figure 14. Once a type on the sidebar is clicked, it replaces the summary charts with the feature-specific list view.

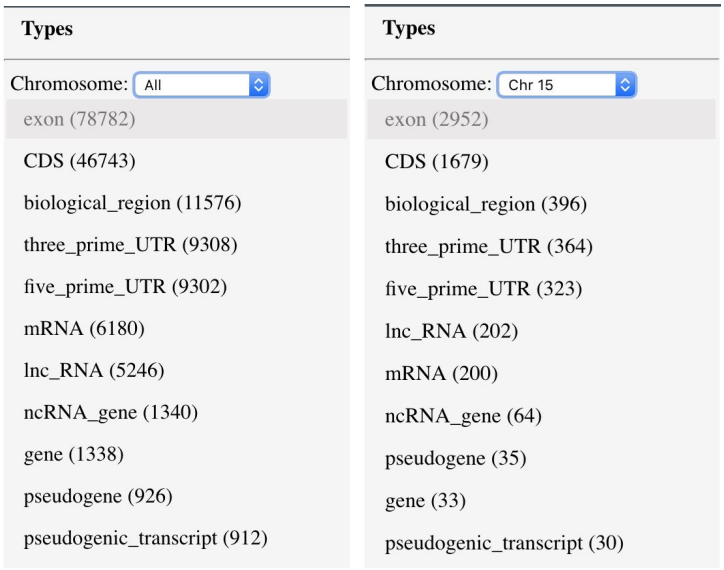

Figure 14 - Sidebar with different chromosomes selected

The feature list allows a user to scroll through individual features and has extensive sorting capabilities. As shown in Figure 15, the list has 4 table headings. These are the most useful fields to sort on. Clicking on the sorting icon next to each heading will sort the features by that field. The sorting icons change to help the user track which direction the list is currently sorted.

Search features..

Chr	Type	Source	Location
16	miRNA	mirbase	17052-17119
16	miRNA	mirbase	770183-770277
17	miRNA	mirbase	1022476-1022559
5	miRNA	mirbase	1062896-1062974
1	miRNA	mirbase	1296110-1296170
10	miRNA	mirbase	2076019-2076089
19	miRNA	mirbase	2235829-2235926
19	miRNA	mirbase	4770670-4770779
4	miRNA	mirbase	7310450-7310524
19	miRNA	mirbase	7617439-7617505
12	miRNA	mirbase	12764649-12764743
19	miRNA	mirbase	15179283-15179350
16	miRNA	mirbase	16306370-16306434
17	miRNA	mirbase	18340814-18340886
22	miRNA	mirbase	19963753-19963834

Figure 15 - Feature list sorted by ascending genomic location

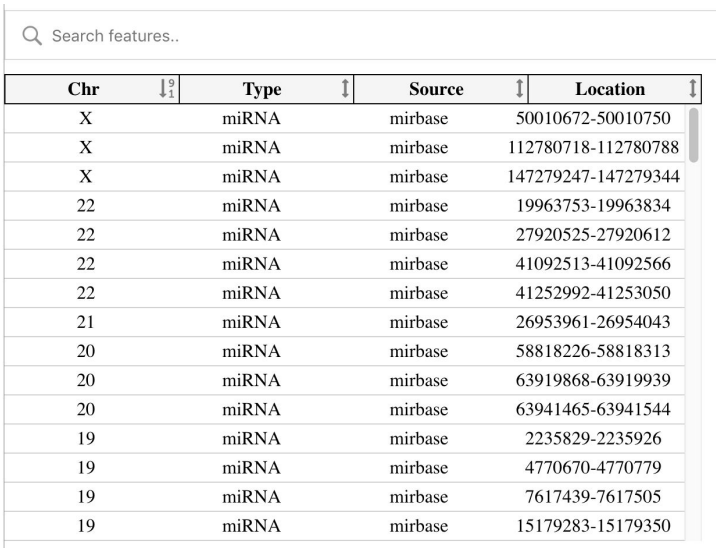

Figure 16 - Feature list sorted by descending chromosome number

The feature list also has a search bar which further enables the user to explore the features. Any text entered into the search bar will filter the list to only show features with that text appearing in one of its fields. An example of the utility this offers is shown in Figure 17 where the user wants to find features at a specific genomic location but don't want to scroll through a long list (even if it is sorted ascending or descending). This can also help a user shortcut the process of finding features for a specific source, type, or chromosome.

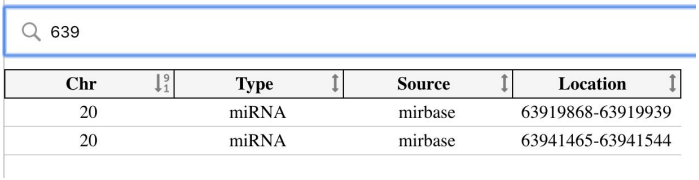

Figure 17 - Feature list search bar filtering by text entry match

Apart from the feature list, feature-level visualizations include a summary of feature details as well as a tree showing hierarchical relationships with other features in the file. For feature without children a detail display will look similar to that found in Figure 18. If a feature does have a children then it will be displayed as shown in Figure 19.

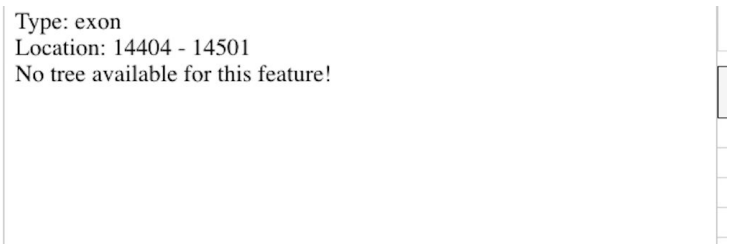

Figure 18 - Details for feature without children

Figure 19 - Example tree visualization for a feature with one child.

Evaluation

The biggest thing we learned is that GFF3 files aren't as "tree-friendly" as we originally anticipated. We thought that every feature would have a parent and at least one child but most features didn't have either. We also learned that exons are by far the most common type in a GFF3 file. This isn't super apparent from the raw file but our visualizations show it very plainly. We also saw that many features frequent certain chromosomes but not others which was interesting. I think that this tool will really help researchers who need to analyze the quality of their annotations during their workflow. It is a bit clunky with large files but gives solid visualization for the contents of the files. The tree is still very basic and can be further enriched with data from SO for the different features. The viewer would also be much more useful in exploratory analyses if it allowed two files to be uploaded and compared side-by-side. This would enable certain genomes to be directly compared.

References

1. https://software.broadinstitute.org/software/igv/GFF
2. http://docs.blast2go.com/user-manual/gene-finding/gff-viewer/
3. https://uswest.ensembl.org/Homo_sapiens/Info/Index
4. https://github.com/The-Sequence-Ontology/Specifications/blob/master/gff3.md