Abstract text mining to create an exhaustive disease-disease correlation database

Suchir Misra

2019-2020

A. Rationale

Correlating diseases based on shared molecular mechanisms, such as craniosynostosis and cancer, allows for improvement of disease therapies through drug target identification and reposition of existing drugs while also advancing the fields of disease taxonomy and etiology [1]–[6]. Building a comprehensive database of disease-disease correlations based on similar molecular mechanisms would be beneficial [1]. Current databases have reduced benefits as a lack of expression data causes rare diseases to be overlooked. Craniosynostosis, the second most common craniofacial abnormality caused by premature fusion of one or more cranial sutures, is one such disease that gets overlooked in current databases [7]. Shortcomings in studying craniosynostosis in disease-disease correlation databases illustrate the need for a new method to create such databases. Using literature searches to build databases ensures that rare diseases will be accounted for. Automation of such literature searches is necessary as if done manually, it would need months to elucidate a disease-disease correlation. Text mining is the most efficient method of extracting information from publications and text mining of abstracts is beneficial rather than text mining of full papers, as abstracts allow for a wider range of studies to be accounted for, due to limited accessibility of full text papers, and for only the most important genes to be considered, since abstracts only include the most pertinent information [8]. There are no current disease-disease databases that exclusively use text mining of abstracts to elucidate disease-disease correlations. This study will aim to build a database of disease-disease correlations built exclusively on abstract mining by collecting a list of diseases, extracting all gene names from all abstracts related to genetic papers for those diseases, and calculating the strength of the correlations by determining the significance of the gene overlap between diseases. A secondary aim of the study will be to use the database to elucidate novel disease-disease correlations for craniosynostosis.

B. Research Questions, Hypotheses, and Expected Outcomes

Research Questions

- How accurate is the newly created database in elucidating disease-disease correlations?

- What novel disease-disease correlations for craniosynostosis can be elucidated using the new database?

Hypotheses

- If a significant gene overlap ($p < 3.68*10^{-8}$ (0.05/1,357,128, where 1,357,128 is the total number of disease-disease correlations)) is found, then two diseases are significantly correlated.
- If a significant gene overlap ($p < 3.68*10^{-8}$ (0.05/1,357,128, where 1,357,128 is the total number of disease-disease correlations)) is found, craniosynostosis shares a significant disease-disease correlation to that disease.

Expected Outcomes

- Expected that the top ten disease-disease correlations with the lowest p-values have all been previously elucidated in literature, which would validate the accuracy of the database
- As craniosynostosis is a craniofacial abnormality, it is expected to be significantly correlated to another craniofacial abnormality.

C

Procedures

Materials

- Mayo Clinic Website
- Microsoft Excel Version 16.16
- iPython Version 6.4 [9]
- PubMed Database from the National Library of Medicine
- pgAdmin Version 4.15 [10]
- 2012 13-inch MacBook Pro

Procedure

Part I – Creating a Disease List

- Search "Mayo Clinic diseases and conditions list" into Google, and click on the first link.
- Copy all of the diseases into a Microsoft Excel file.
- Remove all of the duplicate diseases to create a final disease list.

- Add the word "genetics" to the end of each disease to create a list of PubMed search terms.

Part II – Collecting Abstract IDs for Each Disease

- Copy the list of search terms into a text file.
- Write a Python program using the iPython interface to collect abstract IDs for each paper for each search term.
    - Import necessary Python libraries to access PubMed.
    - Use a built-in function in the Metapub library that fetches PubMed IDs, to retrieve all IDs for each search term on the list of search terms.
        - Set the maximum value of IDs retrieved to 100,000 to ensure that all papers are being included.
    - Write the search term in the first column of an output file, and the PubMed ID in the second column of the same file.
- Copy the PubMed IDs only into a separate text file.
- To prepare for parsing of each abstract, write a Python program using the iPython interface to numerically sort the abstract IDs and remove all duplicate IDs.
    - Open the text file of PubMed IDs only, and cast each value so that they are now integers.
    - Use the sorted function in Python to numerically sort the list of IDs.
    - Use the dict.fromkeys function in Python to remove all duplicate IDs from the sorted ID list.
    - Write each distinct abstract ID to a new output file.

Part III – Extracting Gene Names from Each Abstract

- Create a list of all gene names using the HUGO Gene Nomenclature Committee (HGNC) [11].
- Create a list of genetic terms which indicate a gene mutation rather than overexpression of a gene.
- Write a Python program using the iPython interface to extract gene names from each abstract.
    - Open the text file of all of the sorted abstract IDs.

    - Create a while loop such that the following processes will repeat for each ID on the list of sorted abstract IDs:
        - Remove all punctuation from the abstracts and split them so each word is considered separately.
        - Create a new list of all gene names that occur in that abstract by matching them to the list created using HGNC.
        - Create a for loop such that for each gene in that gene list, the gene name will only be written to the output file if it is within three words of one of the terms on the genetic terms list.
        - Write the abstract ID to a "Last Processed ID" file.
- Write a Python program using the iPython interface to conduct a binary search function to ensure restartability of the program.
    - Find the first and last indices of the input list of IDs.
    - Create a while loop such that:
        - If the middle ID is equal to the last processed ID from the "Last Processed ID" file, the program to extract gene names begins to run again from that ID onwards.
        - If the middle ID is greater than the value of the last processed ID, then make the last index one less than the value of the previous middle index.
        - If the middle ID is less than the value of the last processed ID, then make the first index one greater than the value of the previous middle index.
        - The loop will keep running until the middle ID is equal to the last processed ID.
- Write lines of code that state that:
    - If starting a new list of IDs from the beginning, run the first program until a potential crash.
    - If starting a list of IDs from the middle, run the binary search function to determine a starting point for the first program.

Part IV – Database Creation

- Install PostgreSQL using Homebrew.
- Download the pgAdmin app from the PostgreSQL website.

- Start PostgreSQL using the command line.
- In the new PostgreSQL database, load the following lists:
    - The list of all diseases
    - The list associating diseases to abstract IDs
    - The list of sorted abstract IDs
    - The list associating abstract IDs to gene names
- Perform a join query so that the list associating diseases to abstract IDs is joined with the list associating abstract IDs to gene names on abstract IDs, so that a list associating diseases to genes is created.

Part V – Determining Strength of Each Disease-Disease Correlation

- Using the list associating diseases to genes, determine the number of diseases from the original disease list that have genes associated with it.
    - Calculations will be performed for only disease-disease combinations involving these diseases.
- Determine the number of genes that overlap between each disease in a disease-disease combination.
- Set up a Fisher's Exact Test input in a text file in four columns, as shown below from left to right:
    - Number of genes that overlap in both diseases
    - Number of genes that are in the first disease, but not in the second disease
    - Number of genes that are in the second disease, but not in the first disease
    - Number of genes that are in neither disease
- Repeat the above steps for each disease-disease combination.
- Write a Python program using the iPython interface such that for each disease-disease correlation, the significance of the gene overlap is calculated.
    - Import necessary Python statistics libraries and open the input file.
    - Create a for loop such that each of the processes below will be completed for each disease-disease combination:
        - Make each line of inputs its own separate string, then separate that string and make each number within that line its own integer.

        - Put the integer values into a contingency table, and calculate the p-value and odds ratio using the Fisher's Exact Test function.
        - Write the following into an output file from left to right:
            - The first disease
            - The second disease
            - The p-value
            - The odds ratio
- Perform the Bonferroni Correction to set a threshold p-value to filter down the number of significant disease-disease correlations.

Part VI – Creating a User Interface

- Create a user interface to host the data from the output table of the PostgreSQL database using the procedure as detailed in Uttariello, 2019 [12].
- Setup the backend of the interface using the command line:
    - Create a directory to store all interface-related folders.
    - Create a controllers directory within the main interface directory.
    - Install NPM, the main package manager for JavaScript, and necessary subpackages.
    - Create a JavaScript server file such that the interface will be able to connect to the PostgreSQL database.
    - Create a JavaScript main file within the controllers directory such that all data from the output table will be returned given a disease name as the input.
    - Start the backend of the interface by typing "npm start" in the command line.
- Setup the frontend of the interface using the command line:
    - Create a new directory within the main interface directory to host all files related to the front-end of the database.
    - Install the bootstrap, reactstrap, and react-csv JavaScript plugins.
    - Create a new source directory within the front-end directory.
    - Create a components directory within the source directory.
    - Create a tables directory within the components directory.
    - Create an App JavaScript file within the source directory such that the formatting of each query result is detailed.

        - Creating such a file allows the interface to update dynamically as new information is added to the database.
    - Create a Data Table JavaScript file within the tables directory such that the formatting of the resulting data table displayed in the interface is detailed.
        - There are three columns, Disease, P-value, and Odds Ratio.
- To run the interface, use the "npm start" command inside the front-end directory.

Part VII – Creating Disease Networks

- Calculate the genetic distance $\sqrt{n1 * n2} - m$ for each disease-disease combination where n1 is the total number of genes mutated in the first disease, n2 is the total number of genes mutated in the second disease, and m is the number of genes mutated in both diseases.
- Create a matrix of all of the genetic distances from all disease-disease combinations.
- Use the multidimensional scaling function on R to read the distance matrix and create a plot of all diseases based on their distances to all of the other diseases [13].
    - Each disease would be represented as a dot on the plot.
- To validate the accuracy of the genetic distance plot, create a disease network for Alzheimer's Disease, a commonly studied disease.
    - Choose the 10 diseases closest to Alzheimer's Disease on the genetic distance plot.
    - Input these diseases into a new Cytoscape file to create the network [14].
        - Select Alzheimer's Disease as the source node, and the other diseases as the target nodes.
- Repeat the above steps for craniosynostosis to create a craniosynostosis disease network to elucidate novel disease-disease associations for craniosynostosis.

Risk and Safety

The only material to be used for this study is a computer, thus this study complies with minimal safety regulations. The only risk to the study is improper temperature modulation leading to overheating or malfunctioning of the computer, which would not pose a threat to the other occupants of the lab.

Data Analysis

Once the data is obtained and the number of genes that overlap between each disease is recorded, then the significance of the overlap will be analyzed using a Fisher's Exact Test. Since there are over 1.3 million disease-disease correlations, there is a chance that there will be hundreds of thousands of significant gene sets if the threshold of significance is set at $p < 0.05$. Thus, the Bonferroni correction will be performed filter down the number of significant disease-disease correlations.

1. **Human Participants Research** – Not Applicable
2. **Vertebrate Animals Research** – Not Applicable
3. **Potentially Hazardous Biological Agents Research** – Not Applicable
4. **Hazardous Chemicals, Activities, and Devices** – Not Applicable

NO ADDENDUMS EXIST

D. Bibliography

[1] C.-C. Liu *et al.*, "DiseaseConnect: a comprehensive web server for mechanism-based disease-disease connections," *Nucleic Acids Res.*, vol. 42, no. Web Server issue, pp. W137-146, Jul. 2014, doi: 10.1093/nar/gku412.

[2] J. Yang *et al.*, "DNetDB: The human disease network database based on dysfunctional regulation mechanism," *BMC Syst. Biol.*, vol. 10, no. 1, p. 36, May 2016, doi: 10.1186/s12918-016-0280-5.

[3] A. Guilhem *et al.*, "Intra-venous bevacizumab in hereditary hemorrhagic telangiectasia (HHT): A retrospective study of 46 patients," *PloS One*, vol. 12, no. 11, p. e0188943, 2017, doi: 10.1371/journal.pone.0188943.

[4] S. Lauro, C. E. Onesti, R. Righini, and P. Marchetti, "The use of bevacizumab in non-small cell lung cancer: an update," *Anticancer Res.*, vol. 34, no. 4, pp. 1537–1545, Apr. 2014.

[5] V. A. Venkatesha *et al.*, "P7170, a novel inhibitor of mTORC1/mTORC2 and Activin receptor-like Kinase 1 (ALK1) inhibits the growth of non small cell lung cancer," *Mol. Cancer*, vol. 13, p. 259, Dec. 2014, doi: 10.1186/1476-4598-13-259.

[6] Y. Zeng *et al.*, "MicroRNA-205 targets SMAD4 in non-small cell lung cancer and promotes lung cancer cell growth in vitro and in vivo," *Oncotarget*, vol. 8, no. 19, pp. 30817–30829, May 2017, doi: 10.18632/oncotarget.10339.

[7] X. Wu and Y. Gu, "Signaling Mechanisms Underlying Genetic Pathophysiology of Craniosynostosis," *Int. J. Biol. Sci.*, vol. 15, no. 2, pp. 298–311, 2019, doi: 10.7150/ijbs.29183.

[8] D. Westergaard, H.-H. Stærfeldt, C. Tønsberg, L. J. Jensen, and S. Brunak, "A comprehensive and quantitative comparison of text-mining in 15 million full-text articles versus their corresponding abstracts," *PLoS Comput. Biol.*, vol. 14, no. 2, p. e1005962, 2018, doi: 10.1371/journal.pcbi.1005962.

[9] F. Perez and B. E. Granger, "IPython: A System for Interactive Scientific Computing," *Comput. Sci. Eng.*, vol. 9, no. 3, pp. 21–29, 2007, doi: 10.1109/MCSE.2007.53.

[10] M. Stonebraker, L. A. Rowe, and M. Hirohama, "The implementation of POSTGRES," *IEEE Trans. Knowl. Data Eng.*, vol. 2, no. 1, pp. 125–142, Mar. 1990, doi: 10.1109/69.50912.

[11] B. Braschi *et al.*, "Genenames.org: the HGNC and VGNC resources in 2019," *Nucleic Acids Res.*, vol. 47, no. D1, pp. D786–D792, Jan. 2019, doi: 10.1093/nar/gky930.

[12] J. Uttariello, "Build a CRUD Template Using React, Bootstrap, Express & Postgres." Medium, 25-Mar-2019.

[13] A. Buja *et al.*, "Data visualization with multidimensional scaling," *Journal of Computational and Graphical. Statistics*, vol. 17, no. 2, pp. 444-472, 2008, doi: 10.1198/106186008X318440

[14] P. Shannon *et al.*, "Cytoscape: A software environment for integrated models of biomolecular interaction networks, *Genome. Res.*, vol. 13, pp. 2498-2504, 2003, doi: 10.1101/gr.1239303.