Do You Know Where Your Research Is Being Used?An Exploration of scientific literature using Natural Language Processing

Theodore J. LaGrow, Jacob Bieker, Boyana Norris

University of Oregon

Author Note

Theodore J. LaGrow and Jacob Bieker are undergraduate researchers for Boyana Norris in the High-Performance Computing Laboratory at the University of Oregon.

Correspondence concerning this article should be addressed to Theodore J. LaGrow, Jacob Bieker, Boyana Norris, Computer and Information Science, University of Oregon, Eugene, OR 97403.

Contact: tlagrow@uoregon.edu, jbieker@uoregon.edu, norris@cs.uoregon.edu

Abstract

In such a complex and dynamic field as computer science, it is of interest to understand what resources are available, how much the resources are used, and for what purpose the resources are used. We demonstrate the feasibility of automatically identifying resource names on a large scale from scientific literature in arXiv’s database and show that the generated data can be used for exploration of software and topics. While scholarly literature surveys can provide some insights, large-scale computer-based approaches to identify methods and technology from primary literature is needed to enable systematic cataloguing. Further, these approaches will facilitate the monitoring of usage in a more effective method. We developed a software tool using Natural Language Processing to determine if the article relates to the technology and methods of question. We were then able to evaluate a trend of technology and methods used in each specific areas of science. As we continue to expand this software, we will also analyze the researchers’ sentiment about the technology and methods.

*Keywords*: natural language processing, scientific literature, database, computer software

Do You Know Where Your Research Is Being Used?

An Exploration of scientific literature using Natural Language Processing

With expanding databases of scientific articles, there is exceedingly greater access to publications on specific scientific topics. Hucka and Grahams (2016) suggest in their article *Software search is not a science, even among scientists,* that“When searching for ready-to-run software, the top five approaches overall are: (i) search the Web with general-purpose search engines, (ii) ask colleagues, (iii) look in the scientific literature.” These dated technology search methods can be painstaking and arduous. These laborious searches cannot cover the amount of articles a program can parse through. We were curious if there was a method to finding trends of technology usage by analyzing large data from these databases.

Recently, linguistic machine learning has been implemented to inference across large data sets (Bird et al, 2009). Scientific databases can be incorporated into large sets of collections from a given number of articles by using various methods for text extraction and filtering. Linguistic machine learning can be used to derive meaning with this type of large data. We decided to use natural language processing to explore and infer the types of technologies and methods that are being used in various disciplines of science.

**Natural Language Process Overview**

Bird et al. (2009) describe natural language processing (NLP) as the ability of a computer program to understand human speech as it is spoken. Natural language processing is a field of artificial intelligence and computational linguistics concerned with the interactions between computers and natural languages. Modern NLP is based on machine learning, especially statistical machine learning. The programing paradigm of machine learning is different from that of most prior attempts at language processing. Up to the 1980s, most NLP systems were based on complex sets of hand-written rules (Jones, 2001). Starting in the late 1980s, however, there was a revolution in NLP with the introduction of machine learning algorithms for language processing. This was due to the steady increase in computational power over time (Jones, 2001). Machine learning calls for using general learning algorithms, often grounded in statistical inference. The main idea is to automatically learn such rules through the analysis of large corpora of typical real-world examples. A corpus is a set of documents (or sometimes, individual sentences or strings) that have been hand-annotated with the correct values to be learned. The accuracy of the analysis can vary depending on the format of the data. The cleaner the data and corpus, the better the desired output.

**Method**

To obtain the data, we first parsed through arXiv.org search results for our topics of interest. arXiv.org is a major online hub where researchers pre-publish their articles as their papers get peer-reviewed. The four topics we were curious about were: galaxy evolution, Hawkes Processes, t-cell receptor genomes, and natural language processing itself. We went through the downloaded PDFs, extracting text using PDFminer (Shinyama, 2014) and Python (van Rossum, 1991). Since we were limited to a Window’s 10 desktop (specification: i7 core processor and 32GB RAM), Window’s 10 laptop (specification: i5 core processor, and 6GB RAM), and a MacBook Pro (specification: i7 core processor and 8GB RAM), we decided to only extract the first 100 articles from the topic searches because of limited computing capabilities of the computers available. Once we converted the PDFs to text, we applied filters to the text to remove non-alphanumeric characters, and removed any lines that were less than seven characters, to clean up the text documents. Once the documents were cleaned, we used the Natural Language Toolkit ("Natural Language Toolkit.", 2016) to parse the text, giving us the parts of speech of each word, a frequency distribution of n-grams containing predefined interesting words, and lists of words similar to the user-defined interesting words. N-grams take an interesting word and use it as a center point in the string of the ‘n’ given length. Table 1 comprises of the interesting words we found to optimized the output.

*Table 1: Interesting words used for n-grams*

|  |
| --- |
| **Dictionary of Interesting Words** |
| simulation, software, code, analysis, using, program, analyzed, scripted, automated, description, implements, function, modifies, operated, pipeline, helps, allows, manipulate, processed |

We decided to use n-grams of length 15 because the average length of a sentence is 6-7 words giving us roughly the sentence on either side of the interesting word. Once that was done, we traversed the collection of n-grams, only taking the noun phrases from the n-grams and counting the occurrences of each noun phrase. The counted noun phrases became the basis for the generated word clouds, which visualize the hierarchical significance of the word to the corpus of data related to the discipline being examined.

**Results**

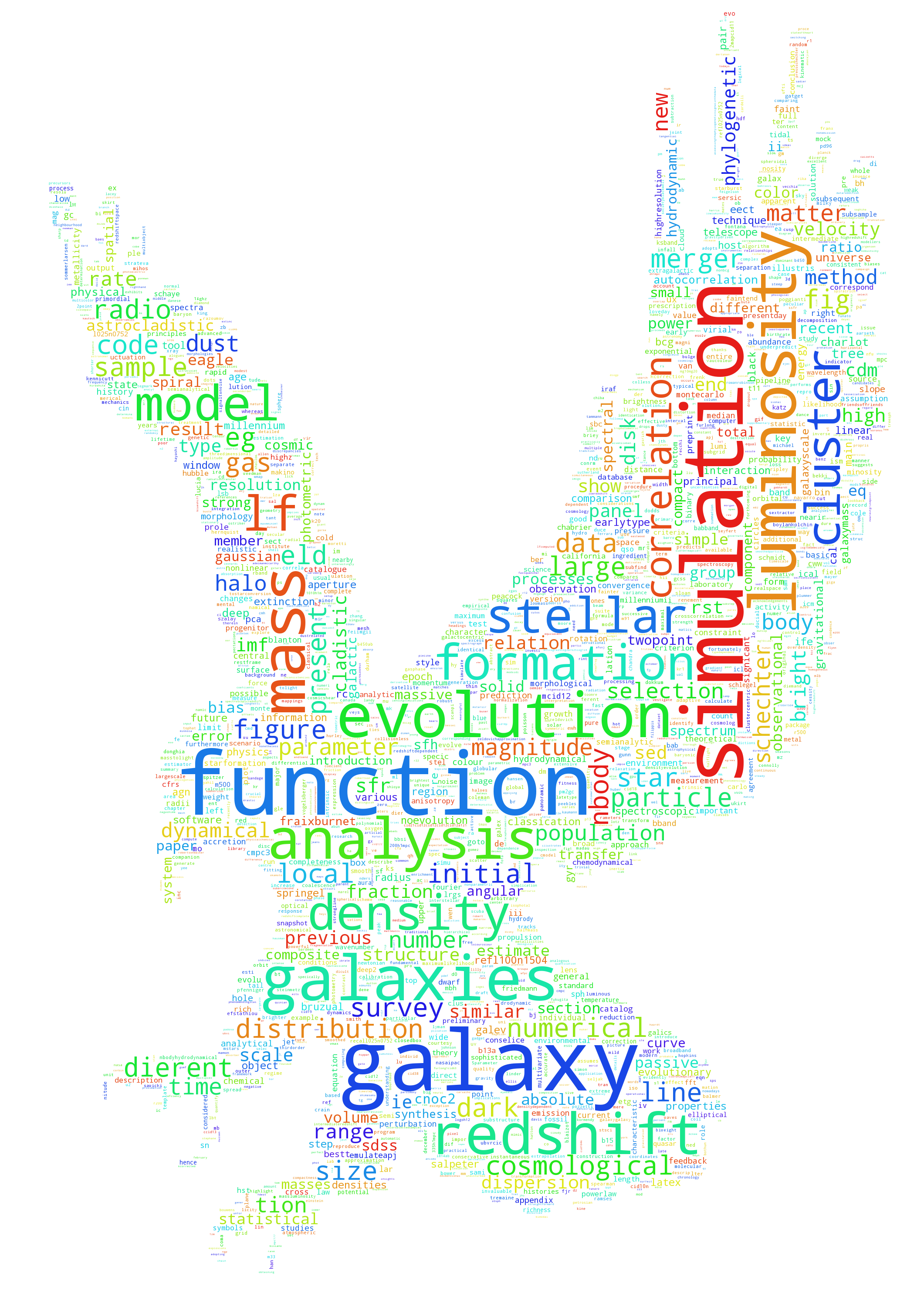
We found that each data set produced a variety of similar words. A few similar words included, function, method, and analysis. These words had relatively high frequencies compared to the more unique words related to the data sets. We suspect that because these words are in our interesting word dictionary, they typically occur close to the other interesting words in our corpus. Interesting results we found include: Gadget (a galaxy imaging technology), Velvet (an assembly program), and morphological (a method dealing with the structure of things). Both the technologies and the method extracted pertain heavily to each respected field. We did not know the technology Gadget before we searched the database. This output signifies that our method of extraction will produce additional technology not known to the user.

**Output Frequencies**

Our first target was analyzing publications on Hawkes Processes. Table 2 displays the top thirty noun frequencies as a result of our analysis. Figure 1 shows the outputted words sized by the frequency of words.

*Table 2: Top 30 words and frequencies generated with search phrase: Hawkes Process*

|  |  |
| --- | --- |
| **Word** | **Number of Occurrences** |
| Hawkes | 815 |
| Rate Function | 349 |
| Large Deviation Principle | 113 |
| Lemma | 109 |
| Exciting Function | 107 |
| Point Processes | 99 |
| Eq | 95 |
| Theorem | 90 |
| Poisson | 82 |
| Fig | 78 |
| Residual Analysis | 78 |
| Hawking | 74 |
| Ix | 70 |
| Black Hole | 67 |
| Intensity Function | 58 |
| Correlation Function | 54 |
| Conditional Intensity Function | 54 |
| Contrast Function | 51 |
| Excitement Function | 51 |
| Consider | 49 |
| Genome Analysis | 42 |
| Numerical Simulations | 44 |
| Simulation Study | 44 |
| Morphological | 42 |
| Partition Function | 42 |
| Exponential Function | 40 |
| Distribution Function | 39 |
| Cost Function | 39 |
| Kernel Function | 38 |
| Wienerhopf | 38 |
| Fourier | 37 |

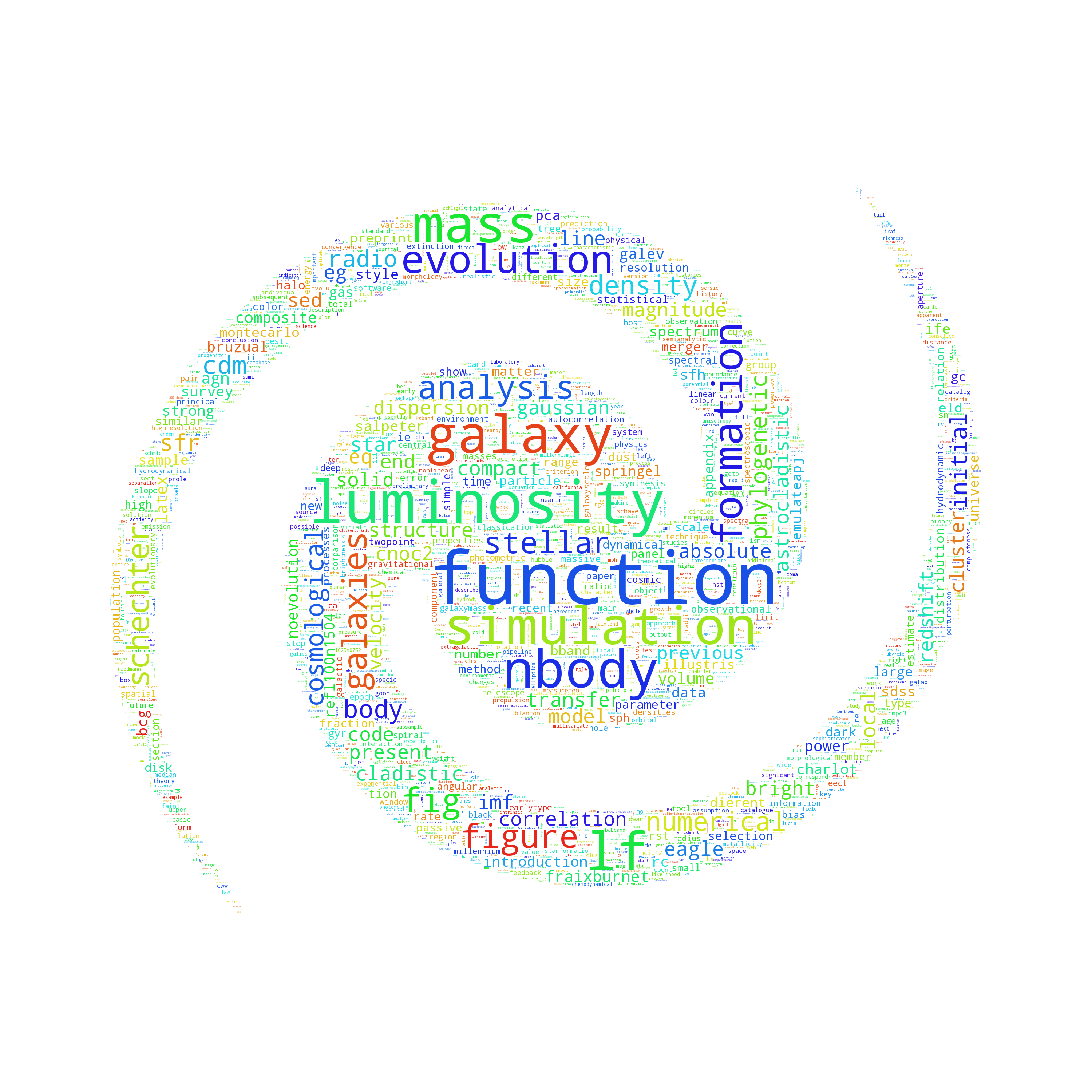


*Figure 1: Output distribution word cloud of the search phrase: Hawkes Process*

Our second target was analyzing publications on galaxy evolution. Table 3 displays the top thirty noun frequencies as a result of our analysis. Figure 2 shows the outputted words sized by the frequency of words.

*Table 3: Top 30 words and frequencies generated with search phrase: galaxy evolution*

|  |  |
| --- | --- |
| Word | Number of Occurrences |
| Luminosity Function | 332 |
| N-body | 145 |
| Fig | 128 |
| Schechter | 101 |
| Exciting Function | 107 |
| Point Processes | 99 |
| Eq | 95 |
| Galaxy Luminosity Function | 72 |
| Galaxy Evolution | 71 |
| Galaxy Formation | 70 |
| CDM | 67 |
| Phylogenetic Analysis | 65 |
| Body Simulations | 63 |
| Numerical Simulations | 60 |
| Initial Mass Function | 54 |
| Cosmological Simulations | 53 |
| Astrocladistics | 52 |
| Mass Function | 49 |
| Stellar Mass | 45 |
| Transfer | 45 |
| Eagle | 45 |
| Local Density | 39 |
| Compact Galaxies | 39 |
| Gaussian | 37 |
| Cladistic Analysis | 36 |
| Gadget-3 | 36 |
| Radio Galaxy Luminosity Function | 36 |
| Star Formation | 36 |
| Cluster Galaxies | 33 |
| Correlation Function | 33 |
| Bright End | 33 |

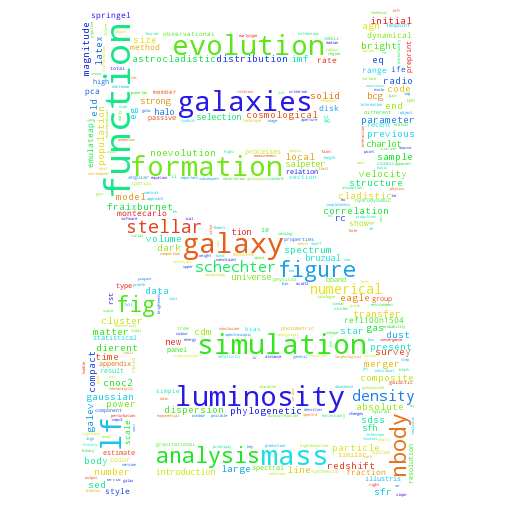


*Figure 2: Output distribution word cloud of the search phrase: galaxy evolution*

Our third target was analyzing publications on T-cell receptor genome. Table 4 displays the top thirty noun frequencies as a result of our analysis. Figure 3 shows the outputted words sized by the frequency of words.

*Table 3: Top 30 words and frequencies generated with search phrase: T-cell receptor genome*

|  |  |
| --- | --- |
| **Word** | **Number of Occurrences** |
| Monte Carlo | 244 |
| Eq | 244 |
| Fig | 119 |
| TCR | 82 |
| DNA | 71 |
| RNA | 68 |
| SNPS | 59 |
| Chipseq | 59 |
| Numerical Simulations | 58 |
| Partition Function | 54 |
| Ligand Concentration | 53 |
| Methods | 52 |
| Correlation Function | 52 |
| MC | 50 |
| Gillespie | 46 |
| RNAseq | 44 |
| Microarray Analysis | 43 |
| Maximum Likelihood | 41 |
| Bayesian | 39 |
| Simulation Study | 39 |
| Velvet | 39 |
| Data Analysis | 38 |
| SNP | 37 |
| Stochastic Simulation | 36 |
| Cluster Size | 36 |
| Covariance Function | 35 |
| Dierent Values | 30 |
| Greens | 28 |
| Phylogenetic Analysis | 27 |
| Quantitative Analysis | 27 |

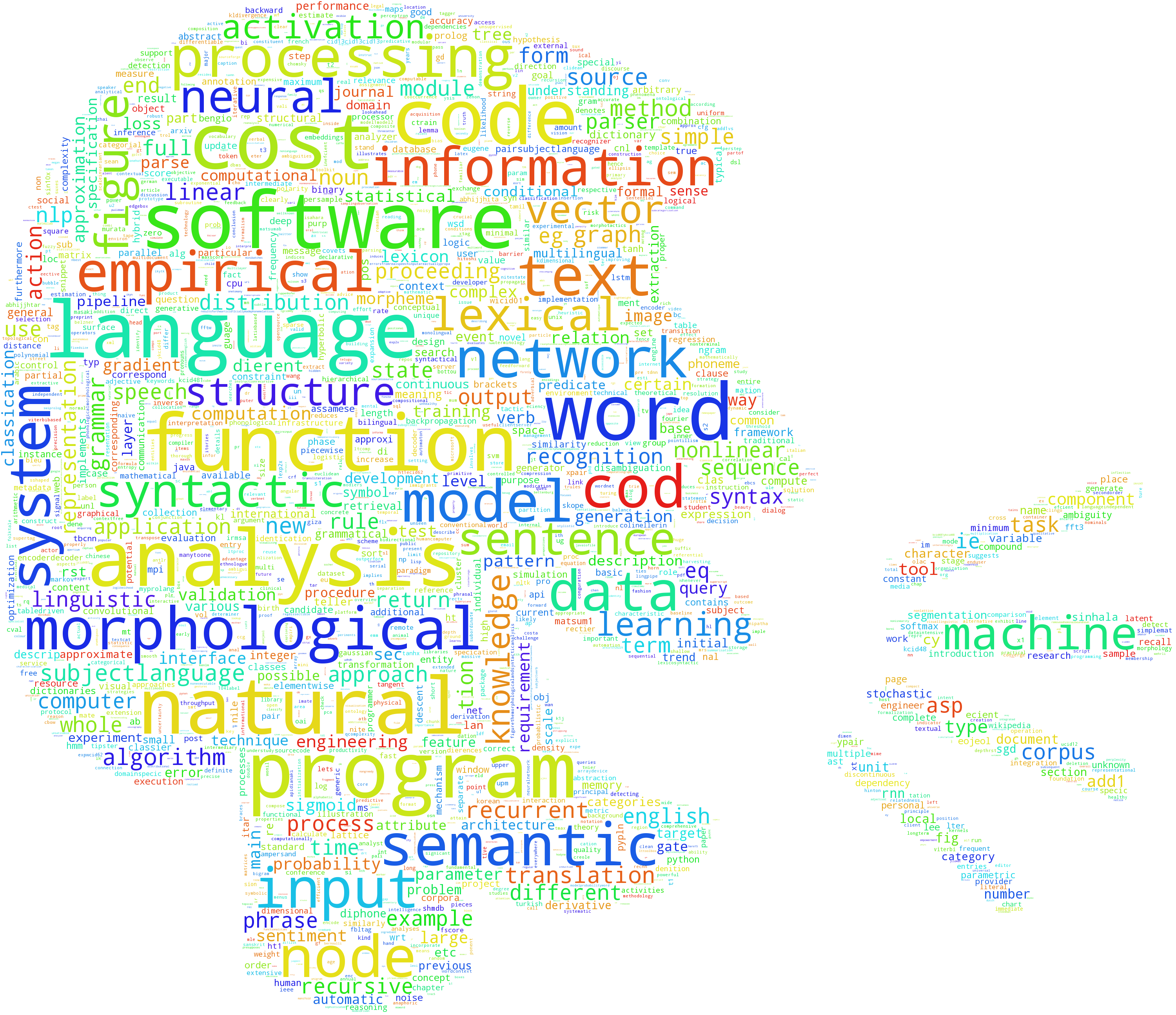


*Figure 3: Output distribution word cloud of the search phrase: T-cell receptor genome*

Our fourth target was analyzing publications on Natural Language Processing. Table 5 displays the top thirty noun frequencies as a result of our analysis. Figure 4 shows the outputted words sized by the frequency of words.

*Table 4: Top 30 words and frequencies generated with search phrase: Natural Language Processing*

|  |  |
| --- | --- |
| Word | Number of Occurrences |
| Cost Function | 260 |
| Figure | 159 |
| Morphological Analysis | 118 |
| Empirical Cost Function | 114 |
| NLP | 102 |
| Proceedings | 101 |
| ASP | 88 |
| Function F | 79 |
| Syntactic Analysis | 76 |
| English | 75 |
| Eq | 74 |
| Language | 71 |
| Function Node | 70 |
| X Language | 58 |
| Fig | 56 |
| Sec | 54 |
| Cost Function C | 52 |
| Y Subject Language | 47 |
| Sigmoid Function | 47 |
| Lexical Analysis Graph | 46 |
| Function Approximation | 44 |
| Empirical Cost Function C | 42 |
| Sentiment Analysis | 41 |
| Semantic Analysis | 41 |
| Morphological | 40 |
| Activation Function | 39 |
| Pair Subject Language Code | 37 |
| Recursive Function | 36 |
| Machine Learning | 35 |
| Teller Machine | 34 |



*Figure 4: Output distribution word cloud of the search phrase: Natural Language Processing*

**Limitations**

We only used 100 articles for each scientific topic because of the computational limit of the computers used. Each search varied in amount of PDFs, however we wanted to make sure we were able to keep around the same corpus size for each analysis. The data sets ranged to around 600,000 strings and 29,000,000 characters after being parsed with n-grams. These are not terribly large files of text, however to iterate over each string can take some time. The program took around twenty minutes to run the corpus creation where we downloaded each PDF and extracted and filtered the text, then another half hour to run our analysis program. The PDF parser program we developed is not the most efficient we could have used. Most of the time the parser worked, however, when a PDF was older than a certain data, had too many pictures, or was too short, the text would come out fussed in a single string or ASCII characters and we would have to threw that document away. In the future, we will seek more reliable means of extracting text from PDFs.

**Conclusion**

The results of our analysis demonstrate that we can evaluate trends of technology and methods in various disciplines. This information lays the groundwork for building a network of software used by various researchers to evaluate the effectiveness of National Science Foundation and other agencies’ funding of different software projects. From these initial results, we are planning on continuing improving the software to extract common methods and tools used in research in any given discipline from the literature, with the hope of connecting researchers to tools that they might not know about, or informing the development of future software packages to better address the needs of their users.

References

Bird, Steven, Edward Loper and Ewan Klein (2009), Natural Language Processing with Python. O’Reilly Media Inc.

Hucka, M., & Graham, M. J. (2016, May 8). ArXiv.org cs arXiv:1605.02265. Retrieved May 11, 2016, from https://arxiv.org/abs/1605.02265

Jones, Karen S. "Natural Language Processing: A Historical Review". Language (2001): n. pag. Print.

"Natural Language Toolkit." Natural Language Toolkit — NLTK 3.0 Documentation. N.p., 9 Apr. 2016. Web. 15 Apr. 2016. <http://www.nltk.org/>.

Noam Chomsky's Theories on Language. (n.d.). Retrieved May 13, 2016, from http://study.com/academy/lesson/noam-chomsky-on-language-theories-lesson-quiz.html

Shinyama, Yusuke. "PDFminer." PDFminer. GitHub, 24 Mar. 2014. Web. 11 Jan. 2016. <https://github.com/euske/pdfminer>.

van Rossum, Guido. "Python." Https://www.python.org/. Python Software Foundation, 20 Feb. 1991. Web. 19 Dec. 2015.

What is natural language processing (NLP)? - Definition from WhatIs.com. (n.d.). Retrieved May 13, 2016, from http://searchcontentmanagement.techtarget.com/definition/natural-language-processing-NLP