Utilizing Web Scraping and Natural Language Processing to Better Inform Pedagogical Practice

Stephanie Lunn
School of CS
Florida International University
Miami, FL 33199, USA
slunn002@fiu.edu

Jia Zhu
School of CS
Florida International University
Miami, FL 33199, USA
jzhu004@fiu.edu

Monique Ross
School of CS
Florida International University
Miami, FL 33199, USA
moross@fiu.edu

***Abstract*—This research full paper describes how web scraping and natural language processing can be utilized to answer complex questions in computer science education. In this work, we apply connectivism as the theoretical framework, and demonstrate how web scraping can be useful for extrapolating large amounts of data from publicly available web pages to pool data from a wider array of sources and to further knowledge in the field. In addition, we discuss how natural language processing can be used to reliably obtain salient information from textual data, and how it can complement qualitative analysis. To illustrate these techniques in practice, we provide a specific application in which we examine the current trends in the job market for computer science students. The information gathered in this example provides additional areas for educational consideration, such as offering students Python programming language and machine learning. Also, the job postings delineate a clear need for applicants to exhibit programming and testing skills. Although programming may be taught already, testing is widely considered a knowledge deficiency, which suggests that educators should consider placing an increased emphasis on this area to ensure their students are adequately prepared for their career endeavors, and able to transfer the knowledge taught to critically assess and debug their own programs.**

***Index Terms*—Web scraping, natural language processing, computing research, data acquisition**

I. Introduction

Computer science education (CSE) is a unique interdisciplinary field situated at the crossroads of education, psychology, and computing fields (computer science, information technology, and computer engineering) [1]. Applying diverse theoretical frameworks and empirical evidence, strong research works in the field provide useful data and information that shape pedagogical practices. Typical methods and measures include both quantitative and qualitative approaches, using data gathered from interviews, direct observation, questionnaires, standardized tests, teacher-created tests, and a reliance on existing data [2]. However, given the rapidly evolving nature of technology itself, we suggest that researchers should expand the current repertoire to include additional methods, and potentially more automatized methods for gathering and assessing information. Process automation refers to the use of computers and software to complete tasks while minimizing human intervention, and it can be beneficial in speeding up the time to completion, or handling routine items [3]. Automating tasks can go hand in hand with using the World Wide Web to connect with students and researchers, and obtaining information readily available to learn more about educational practices and preferences, as well as the dissemination of results.

Connectivism is considered a relatively new learning theory that has been suggested to be beneficial to the field of education [4]–[6]. Its emphasis on collaboration, creativity, and connectivity demonstrates that the capacity to know more is of greater value than what is presently known. Furthermore, connectivism draws attention to the benefits of non-human appliances for human learning [4]. In this work, we discuss how connectivism can provide a useful lens for researchers to transverse knowledge networks and to consider how automated approaches can be applied to gather and analyze information. The research questions guiding this work are:

1) *How can researchers extrapolate large amounts of data from publicly available web pages to create datasets?*
2) *How can automated processing techniques be used to reliably obtain salient information from qualitative data?*

To answer these questions we suggest two techniques with numerous potential applications, web scraping [7] and Natural Language Processing (NLP) [8], [9]. Although the techniques themselves are not novel, they are rarely applied to CSE research. Here, we describe them both and demonstrate the benefits of their application with a specific example, in which we examine the current trends in industry hiring of computer science (CS) students. Implementing web scraping on Indeed.com for job postings, we create a structured dataset. Then, we apply NLP techniques to identity trends including the major job titles, skills requested, salaries by city, and degree preferences.

Knowledge of such information could be useful for higher education's consideration of future course material and could also affect graduate employability. Furthermore, although prior research has considered web scraping job postings, typically literature has focused on information technology, or other specific areas like software testing, which may not be applicable to all CS students [10]–[12]. In this work, we use the search keywords "computer science" to create a broader look at the field, and it also does so through consideration across multiple cities in the United States, rather than studying job needs for a single geographical region.

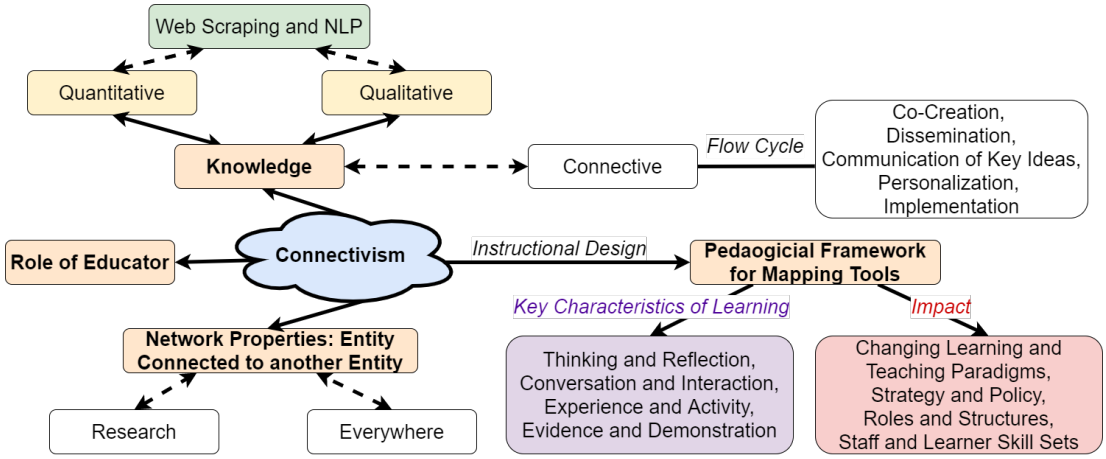

Fig. 1: A mapping of connectivism and its interrelated role with knowledge, educators, networks, and instructional design to inform pedagogy. Also shows how web scraping and NLP can be applied to enhance quantitative and qualitative research.

In this document, we will discuss connectivism, the theoretical framework to guide the work in section II. Next, we will cover the related work in the field in section III. We then provide information about our application of web scraping and NLP, and its importance in section IV. Section V includes information about how these techniques can be applied, and describes the tools and procedures implemented to extract job information from postings in "computer science." After this, we discuss the findings from the specific results of the example in section VI. Finally, we provide a discussion of our findings and conclude with suggestions for future work in section VII.

II. Theoretical Framework

Connectivism is a framework credited to Siemens and Downes, that views learning as a network phenomenon that has its roots in technology and socialization [4]–[6], with epistemological roots in distributed knowledge [13]. The foundation of the connectivist model considers the learning community as a node within a larger network. Networks arise out of two or more nodes that join to share resources, and knowledge is distributed across the network and stored digitally [14]. An individual's knowledge is predicated on a system of networks that fuel organizational knowledge, and can cyclically give back into the system. This process ensures that learners are able to update their own knowledge base to remain current through their established connections [5], [15]. Moreover, groups are able to define social networks towards common goals to promote knowledge.

Connectivism further describes key principles in a digital age [4], [16]. Apart from the emphasis on non-human appliances already mentioned, it also describes the importance of using current and accurate information as the intent for connectivist activities [5], [6]. Additionally, it stresses the necessity of filtering out extraneous and inapplicable information for learning and decision making. Accordingly, what may be the right answer at a particular moment in time, could shift based on the climate affecting decisions [4].

Although connectivism has been challenged on the grounds that knowledge is disparate from the process of learning and education itself [17], proponents have suggested that through engagement of learners in the development of their own networks, meta-cognition results in deeper understanding [18]. It has been demonstrated to be a foundation through which teaching and learning of digital technologies can be understood and managed [15], [19]–[21]. We suggest it as an effective tool via which researchers, educators, and students can benefit from utilization of novel technologies to aid in learning and applied pedagogical practice.

We describe an overview of this framework in Figure 1. Pruned to limit the scope [22], our model demonstrates how connectivism is the central facet uniting knowledge, educators, networks, and pedagogical frameworks. As it relates to our work here, we focus on gathering and analysis of quantitative and qualitative data, using web scraping and NLP, to further knowledge. This information, in turn, can be used to perpetuate knowledge development and additional inquiry as these findings are disseminated, communicated, and then further implemented by others. The relation to instructional design arises through key characteristics of learning, and their potential impact shaping items such as strategy and policy. The network itself, comprised of educators, students, administrators, and other entities, can lead to knowledge sharing through research and/or everywhere, including social relationships and the internet. Meanwhile, the role of the educator can take on many forms, based on different definitions.

According to Drelxer [23], educators may include acting as an information filer, facilitator, guide, researcher, and change agent. However, Siemens [4] describes educators as a combination of roles: knower, concierge assisting with way-finding, modeler of behaviors rather than via direct instruction, network administrator, curator of potential learning approaches, and evaluators [22]. Irrespective of the approach taken though, educators clearly play an important role in the network and in shaping education. Connectivism facilitates continual learning and promotes discussion and collaboration, to assist with decision making, understanding information, and problem solving. Together, connectivist principles guide our

endeavor to integrate new techniques to further the knowledge in computer science education so that it can ultimately enhance student learning. Rather than just accumulating knowledge, it is about using these techniques to obtain meaningful answers to specific research questions. In this work, connectivism is being used to justify the expansion of methods to include NLP and other machine learning techniques to contribute to the body of knowledge.

III. Related Work

As mentioned, we apply connectivism as the guiding framework to suggest web scraping and NLP are tools that can be used to contribute to knowledge. In this section, we will describe background information pertaining to web scraping and NLP in sections III-A and III-B, respectively.

A. Web Scraping

There is an increasing amount of information available online, connecting different entities and offering additional sources of knowledge. For researchers looking for data to improve pedagogy, input can be gathered from social media, digital textbooks, logs/forums from Massive Open Online Courses (MOOCs), and from school websites [24]. Since the resources posted online are considered public, it allows content retrieval of numerous pages and records.

Web scraping refers to the process of extracting unstructured data from the internet, that can be harvested to build large scale datasets of structured data [7], [25]. There are multiple ways to obtain data from a website, although some are more labor intensive than others. Web scraping can be conducted manually, through a hired corporation, through an application or browser extension, or through software. One of the easiest involves directly copying and pasting material from a page, however, this can be quite time consuming for larger quantities of information [7]. In addition, if a website has its own application programming interface (API), data can be retrieved directly from it. Notwithstanding, each provider may have different workflows to do so, there may be a high charge to use the API, and the policies to access the data may be unique [25]. Otherwise, the HTML and/or XML of the page can be accessed directly to obtain useful information using programming languages such as C/C++, PHP, Python, Node.js, or R [7], [25], [26].

Since different sites are built using varied frameworks, languages, and forms, it is important to consider different options to find the right choice for a particular project [7], [25]. The source itself (such as brief tweets from Twitter, university curricula, or more lengthy interview transcripts), the context (looking at performance outcomes or student reactions), the ultimate goal (contextual analysis, topic modeling, or classification), and the desired output (Excel or comma-separated values (CSV) files) all should be taken into consideration [25]. Once the data is collected, it may require additional processing and cleaning.

B. Natural Language Processing (NLP)

NLP is considered an emergent area that is concerned with bridging the gap between humans and computers, and involves using machines to process, interpret, and manipulate language [8], [9]. With teachers, educators, and researchers in mind, it can be used to help automate tasks that would otherwise require manual work [24]. NLP can be useful for rapidly analyzing electronic documents, interview transcripts, or datasets containing text-based content [27].

As one of the major tasks of NLP, text mining is a process by which useful knowledge is obtained from text that is free or unstructured [28]. Discovering and obtaining meaningful relationships may include information retrieval (which can work in tandem with web scraping to obtain information from a website, or may include document retrieval), text classification, topic identification, or event extraction. Furthermore, it is possible to use statistical-based, empirical approaches to the processing of language, rather than purely linguistic theory. However it should be noted that analysis of the syntactic and morphological factors that contribute to the linguistic aspects of text can ensure more rapid analysis [7], [8].

Multiple languages can be used for NLP tasks such as Python, Java, C/C++, R, Prolog, or MATLAB [28]–[30]. However, Python is considered one of the easiest options since it includes a number of tools, packages, and libraries that have built in corpora and resources (such as grammars and ontologies) to expedite NLP applications [31]. Although we will describe some of these further in the methods of section V, it should be noted that the Natural Language Toolkit (NLTK) is a Python library which is a particularly great asset that is well suited for research purposes [29], [31].

IV. Our Application of Web Scraping and NLP

Connectivism emphasizes expanding opportunities for learning and sharing information distributed online. In this paper, we will apply the techniques described to examine a particular application of web scraping and NLP to assess factors that may contribute to CS students' graduate employability using jobs posted on the internet. Graduate employability is typically defined by an ability to obtain a job, to maintain that position, and then to find another [32]. Employability is predicated on competence, and the assumption that graduates will possess certain attributes and requirements for future jobs.

Although schools may teach theoretical understanding and programming, the concepts taught and languages offered may not align with what is presently required by the industry. According to the definition proposed by Rademacher and Walia, a knowledge deficiency includes "any skill, ability, or knowledge of concept which a recently graduated student lacks based on expectations of industry or academia" [33]. While ultimately, academic needs of the students must drive the development of curricula, it is also necessary to ensure graduates are prepared to address practical challenges pertaining to current technologies, and to resolve knowledge deficiencies [33], [34].

Studies applying NLP in CSE are not common in current literature, based on our literature search, however, there are

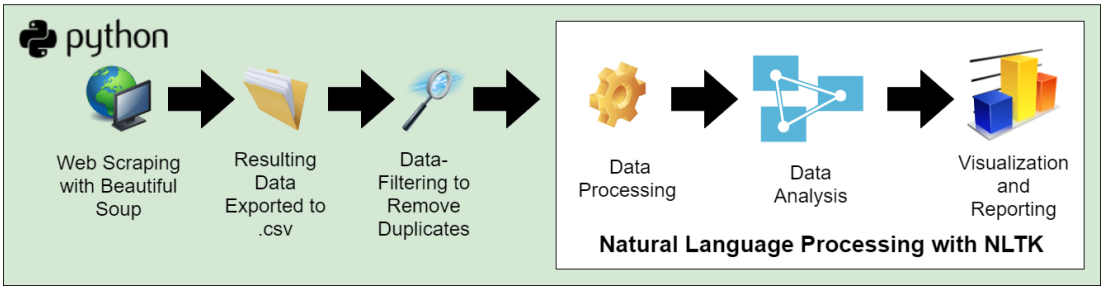

Fig. 2: Overview of process from web scraping to natural language processing using Python

some that perform trend analyses of jobs in computing fields like Information Technology [10], [11], or for more specific applications such as Big Data Software Engineering [35] or Software Testing [12]. However, such papers and postings may not be applicable for all computing students, and these are often regionally limited to a particular city or state. Thus, in our work, we consider an example in which we apply broader search keywords that may encompass the range of options for CS students, specifically examining positions pertaining to the keywords "computer science." Moreover, rather than focusing on a single city or geographic area like other studies, we scrape data from five different cities across the United States.

In addition to the broader research questions guiding this work, the application we describe also includes the following, example-specific, research questions:

1) *What are the top positions being offered in the CS industry?*
2) *What are the major skills that companies expect graduates to have and are there any topics that companies expect graduates to be familiar with?*
3) *Which cities offer the highest average annual salaries for job seekers?*
4) *Which degrees are most requested in CS job postings?*

V. Methods

An overview of the procedure to obtain data from the web, and the intermediate steps before the data can eventually be used and analyzed to obtain useful knowledge are described further in Figure 2. As demonstrated Python can be used for the entire process, and web scraping can be achieved using Python's Beautiful Soup library, then the web data must be exported to a usable format (such as a CSV file). Then, it undergoes pre-processing to filter out and remove duplicates, and then natural language processing can begin. Python's NLTK can be applied and the data can then be processed, analyzed, and eventually used for visualization and reporting. Below we describe a specific application of web scraping and NLP, to examine a large dataset collected from Indeed.com.

We discuss the chosen programming language, Python, in section V-A. Then we describe how web scraping and data filtering were performed in section V-B. In section V-C we review the NLTK. Next, we discuss how data was prepared and pre-processed in section V-D. In section V-E, we review the analysis and visualization tools further. Finally, we discuss the manual validation of the results obtained in section V-F.

A. Programming Language

Python is considered a high-level, dynamic, object-oriented programming language [31]. It is known for being quick and simple, yet effective. Widely employed by researchers and in industry, Python includes its own standard library, but also allows external toolkits and libraries to be added for additional functionalities. All web scraping and NLP were conducted in our application using Python version 3.6.7.

B. Web Scraping and Data Filtering

We scraped data from Indeed.com, a job searching website. The dataset that we created used "computer science" as the job searching keywords, across five cities in the United States ranked highly for tech talent, most jobs available, and with the highest startup investment rates: New York City (New York), San Jose (California), San Francisco (California), Washington (District of Columbia), and Seattle (Washington) [36]–[38].

In addition, we specifically applied the following libraries and packages:

- **Beautiful Soup**: A package useful for extraction of data from HTML and XML files [39], we used BeautifulSoup4 for our web scraping, version 4.6.3.
- **lxml**: A Python library that is used to process XML and HTML information [40], we used version 4.2.5.

Data was collected with the features described further in Table I. After removing any duplicates based on the JobID, our resulting dataset included n = 3,824 listings. This amounted to 770 from New York, 774 from San Francisco, 745 from San Jose, 752 from Seattle, and 783 from Washington DC.

C. Natural Language Toolkit (NLTK)

For NLP, we utilized the NLTK, version 3.3. NLTK was selected since it is well suited to linguistic tasks and comes with explicit documentation. It can be utilized for a number of linguistic processing tasks, such as tagging parts-of-speech, as well as text classification [29], [31].

D. Data Preparation and Pre-Processing

For text analysis or fitting a machine learning model, it is important to clean and process raw text into a usable form. This requires first splitting text into its component words or

Feature	Description of Feature	Example(s)
JobID	A unique ID associated with each posting	p_9d81f582fa82816c; p_9e989fd2ee77c487
City	City as search keywords and those associated with the job posting	New+York; Washington+DC; Seattle; San+Francisco; San+Jose
QueryTerms	"computer science" for all postings	computer+science
Title	The job title posted for the position	QA Analyst; Data Science Intern; Software Developer Intern; Program Manager, Data Center Systems Engineering; Junior Programmer
Company	The hiring company	–None shown for privacy purposes–
Location	Full address information listed	–None shown for privacy purposes–
Salary	Posted salary or salary range (if given)	16.19 - 20.76 an hour; 35000 a year; 51000 - 66000 a year
FullText	The complete job description	successful candidate possess following 10 years bs engineering science languages c, java, python, qt windows applications android applications preferred job type fulltime engineering 10 years preferred education bachelor's required location rochester, ny 14607 required location one location benefits offered paid time health insurance dental insurance healthcare spending reimbursement accounts hsas fsas retirement benefits accounts
Link	Full link to access the job posting	–None shown for privacy purposes–
PostDate	When the job was posted	7 days; 10 days; 30+ days

TABLE I: Features collected from Indeed.com

"tokens" using the whitespace that occurs between words. Then, it is important to remove additional barriers that could otherwise confound the results like punctuation, and/or case sensitivity. Typically, researchers will want to make all of the text lowercase. Moreover, lemmatization of words and/or stemming may be applied to further refine the text data and to ensure comparable foundations or roots.

In our application, we applied the following specific modifications [31]:

- **Regular-Expression Tokenizer**: Divides a string into substrings using regular expressions, extracting alphabetic sequences, monetary amounts, and any other consecutive sequence not separated by whitespace.
- **Brown Corpus**: We removed 150 most common English words based on Brown corpus, a million-word, 500 source, collection of text created at Brown University.
- **WordNetLemmatizer**: WordNet is a lexical database that contains semantic relationships between words. We apply the WordNet lemmatizer, using WordNet version 0.0.1b2.
- **Stopwords**: Stopwords or a stoplist are used when text mining to remove words such as determiners or prepositions that commonly occur in a language, and do not carry meaning associated with concrete concepts. This may include removal of words such as "the" or "a" to limit querying to important terms. We use stop-words version 2018.7.23. In addition to common English stopwords, we also included several domain-specific words that appear in job descriptions at a high frequency such as disclaimers to note the employer does not discriminate based on an individual's characteristics or preferences. Our custom stoplist included items such as "sexual", "orientation", "pregnancy", "religion", and "disability" to ensure the discriminating power of our application was not affected by the frequency of such words.

In addition to the text processing described above, we also cleaned and processed the salary information. It is important to note that not all the job postings included their salary. After removing HTML tags, dollar symbols or other unnecessary signs, we then separated the salary into a new feature we called "PayFrequency" which extrapolated whether pay was offered by year, month, week, day or hours. Then, depending on the PayFrequency, all values were converted into an annual salary. For example, any salary listed as monthly was multiplied by 12 to obtain the annual amount. The total number in the cleaned salary dataset was n = 417 listings. In total, this amounted to 109 from New York, 59 from San Francisco, 60 from San Jose, 56 from Seattle, and 133 from Washington DC. It should be noted that this significantly reduced dataset was only used for evaluating salary information, and the complete set was used for all other evaluations.

Furthermore, job postings were examined for degree information. Although some postings specified a Bachelor of Science (B.S.) or Bachelor of the Arts (B.A.), others simply specified a Bachelor's degree. Likewise, for Master's degrees, some specified a Master's directly, whereas others specifically requested a Master of Science (M.S.) or a Master of Business Administration (M.B.A.). For doctoral level, postings either requested a Doctorate or they may have asked for a Doctor of Philosophy (Ph.D.). Other than specifically requesting or "Associate's degree", postings also included either Associate of Science (A.S.) or an Associate of Arts (A.A.) or simply an "Assoc. degree" at this level of degree. Since a single job posting may request more than one degree, the degree information was split into fourteen categories to better capture the degree requirements: bachelor only, master only, doctorate only, associate only, associate and bachelor, associate and master, associate and doctorate, bachelor and master, bachelor and doctorate, master and doctorate, associate and bachelor and master, associate and bachelor and doctorate, bachelor and master and doctorate, and all 4 combined (bachelor and master and doctorate and associate). Among the list examined for all five cities, there were n = 2,109 mentions of degrees.

E. Data Analysis and Visualization

Data Analysis and visualization was performed using the following libraries [31]:

- **WordCloud**: Bigrams are when two words occur consecutively in a sequence, and a trigram refers to three consecutive words. We visualized our bigrams using

WordCloud, a data visualization technique that illustrates text data with the size of each word in the cloud corresponding to its frequency in the text [41]. We applied a WordCloud package for Python, version 1.6.0.

- **Pandas**: Used for data manipulation and analysis, we applied version 1.0.1 for analysis of cities and salaries.
- **Matplotlib**: Used to generate diagrams, histograms, and plots, we applied version 3.1.3.
- **NumPy**: The numerical mathematics extension of Matplotlib, we used version 1.18.1.

F. Validation

To confirm the validity of the results identified using NLP, a subset of 50 randomly-selected postings were manually inspected to confirm the data pulled accurately portrayed the description listed. Additionally, using Excel to inspect the .csv, the job titles column was examined to confirm the top titles. Moreover, the presence of the top 20 most frequent terms, bigrams, and trigrams in the job descriptions were analyzed with a find all function to ensure reliability during the automated analysis.

VI. Results

The results are described further, by section of the job listings assessed, with the information gathered from the job titles described in section VI-A and information from the job descriptions pertaining to requested skills, programming languages, and areas of knowledge described in section VI-B. Information about the salaries offered for for posted positions is in section VI-C. We later discuss what types of degrees are requested by potential employers in section VI-D.

A. Assessing Job Titles

Using descriptive statistics to group from the entire dataset, we obtained a list of the top job titles obtained using the search keywords "computer science." Then the NLTK was applied, and after tokenizing, removing punctuation, and using the WordNet Lemmatizer, the entire job title list was traversed to obtain frequencies of the bigrams and trigrams associated with the titles. The resulting counts, pertaining to the frequency these titles appeared in the dataset are shown in Table II. The most frequently offered position for job seekers was Software Engineer, followed by Data Scientist.

B. Assessing Job Descriptions

Using the NLTK on large bodies of text in the complete job descriptions, after removal of stopwords and 150 of the most common words (based on the Brown corpus), 89.82% content remained. We identified that among the skills requested, testing and programming were among the most important (occurring 2,031 and 2,008 times, respectively), and Python was the most requested programming language (occurring 1,496 times), followed by Java (occurring 1,253 times). In addition to collecting bigram frequencies, we utilized a WordCloud to illustrate the most frequent bigrams in our text (replacing spaces with an underscore), as demonstrated in Figure 3. Interestingly, "Machine Learning" was the top bigram present in the job descriptions (occurring 1,617 times), highlighting a growing emphasis on this area in computing.

Job Title	Term Frequencies
Software Engineer	446
Data Scientist	117
Software Developer	80
Data Analyst	68
Software Development Engineer	52
Business Analyst	32
Machine Learning Engineer	24
Java Developer	21
Full Stack Developer	16

TABLE II: Frequency of top CS job titles

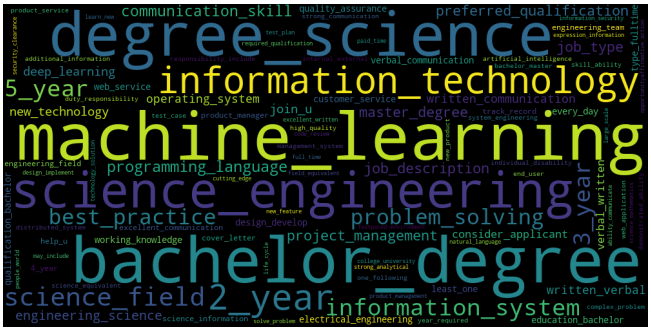

Fig. 3: WordCloud of most frequent bigrams

C. Salary

As previously described, salary information was not available for all postings, and accordingly was only analyzed for 417 for which it did exist. After salary was pre-processed to provide an annual value for all, as previously described, we grouped the average salary for each city in United States Dollars (USD). A comparison of each average, by city, is illustrated in Figure 4. As shown, San Francisco had the highest average salary at $122,647.71, and New York had the lowest average at $79,852.75. Among the cities sampled, the annual salaries ranged from $28,600 to $250,000, with a median salary of $88,400.00 dollars.

D. Degree Breakdown

We used term frequency to examine the job descriptions for the types of degrees employers were looking for. Since it is unknown if these mentions were merely the minimum required, or preferred degrees for a particular listing, we grouped by

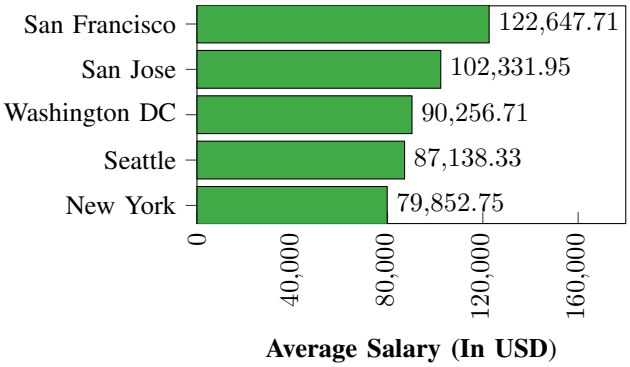

Fig. 4: Average salary by city

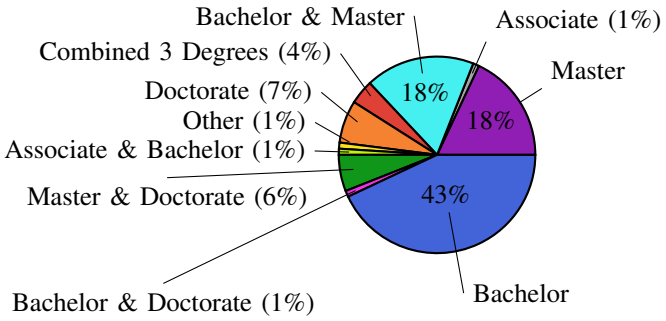

Fig. 5: Degrees requested for CS positions

how the degrees were mentioned (either in singularity or with other degrees). The degree breakdown is shown in Figure 5.

There are fourteen different categories for combinations of the degree requirements presented in the job postings. In the figure, any pairings of degrees (from the list described in section V-D) which occurred at less than 1% were grouped into a category called "Other." The "Other" category included combinations such as Associate & Master, all four degrees, etc. It should be noted that "Combined 3 Degrees" is a term which refers to a job description requesting a Bachelor's level, Master's level, and Doctoral level degree, which occurred in only 4.3% of the postings. The most requested attainment level was Bachelor's degree only, which occurred in 43.3% of the postings. Comparatively, the postings that mentioned only a Master's degree or only a Doctorate degree occurred 18.4% and 7.3% of the time, respectively. An associate's degree was requested by itself only 1.1% of the time. The remainder of the postings requested more than one degree, or were willing to accept multiple levels of attainment. For the most part, these were presented as the minimum degree requirements or the preferred degree requirements in the job postings.

From the pie chart we can see that the majority of postings with degree requirements, did request at least a Bachelor's degree. However, there were a significant percentage of postings that did request a graduate level degree as well. Thus, although a Bachelor's may be the minimum expected, more than half of the postings with degree information may prefer a higher level degree for CS related jobs.

VII. Discussion and Conclusions

Web scraping and NLP can be useful tools to extract and analyze non-trivial knowledge from free text, such as job descriptions. Whether the aim is quantitative counts of term frequency, or qualitative examination of content, these approaches are beneficial to research and to using online connections to enhance the overall knowledge, as described by connectivism. We discuss our application, looking at industry trends for postings in computer science in section VII-A. Then we discuss how web scraping and NLP are useful, in general, for computer science education in section VII-B. We close with conclusions in section VII-C.

A. Our Application of Computer Science Jobs

In terms of our specific application, web scraping data from Indeed.com revealed several important insights that could be useful for educators and administrators. Although it is not the purpose of higher education to serve industry, considering what skills could enhance graduate employability is something that would be beneficial to both sides. As emphasized in connectivism, expanding the knowledge in the field is ultimately beneficial for informing instructional design and affecting pedagogical practice. While ultimately, curricula are meant to be solely focused on catering to students' academic needs, being informed about what industry expects and giving consideration to best practices can enhance students' success in their field [34]. Accordingly, creating effective computing curricula not only requires teaching fundamentals and theories, but also resolving knowledge deficiencies, and offering practical applications and recent technological advancements that could be important to the discipline [11], [33].

Programming and testing had a high occurrence in the postings. Although both may be taught already, their importance should not be neglected. Previous work in both academia and industry have demonstrated that programming and software testing are knowledge deficiencies for students and newly hired software developers [33], [42], [43]. Accordingly, these may be areas that require additional exploration, and for which universities may want to consider placing additional emphasis or adjusting their current teaching. Based on prior work, students' system testing skills are particularly poor, and it has been demonstrated that students may be unable to properly use test coverage tools [42]. Furthermore, qualitative interviews with new graduates has mentioned that they would prefer less emphasis on theory, and increased focus on practical applications and writing tests. Moreover, students would prefer for testing to be taught earlier in their education, and have commented that it would be beneficial to include testing during programmatic development to adjust the mindset so that when coding, consideration is always given to this area as well [44].

In addition to the skills described, and as evidenced by the most frequent bigram in postings, machine learning is presently a topic growing in demand, and perhaps could or should be included as an elective course to further prepare students for opportunities in the field. Alternatively as has been demonstrated previously, applications of machine learning could be folded into other courses to enhance student learning, or incorporated as a theme to unify topics in artificial intelligence and ground it in CS [45]. Furthermore, given the ease and power of Python, and that it is the programming language mentioned most frequently in job descriptions, schools not offering it already may want to consider its inclusion. Both adjustments may help students to develop their abilities in areas relevant to developing their careers.

There were limitations to the example we demonstrated as well. First, computer science is a dynamic field, and industry requirements are constantly changing in terms of the skills, abilities, and knowledge expected from hirees. It should be noted that the data collected was obtained in January 2020, which represents only a slice of potential shift in postings that may occur over a more lengthy timeframe. Furthermore, the data collected was obtained using the keywords "computer

science." Going forward it would be valuable to consider either a longitudinal or a cross-sectional analysis of postings, as well as to explore additional keywords to create a wider scope for study. In addition, for the degree information we implemented a manual approach using a pre-defined term dictionary. However, more advanced techniques can be employed such as concept mapping or modeling job descriptions.

B. Web Scraping and NLP for Computer Science Education

With an increasing need to obtain quality data for CSE research, web scraping presents a real opportunity to gather large quantities of unstructured information rapidly. It is accurate, and can be easy to implement. Furthermore, it allows acquiring unstructured information from the web, and can put it in a usable format for further analysis. Web scraping could be used for collecting admissions information from different college websites, or for assessment of standardized test materials.

Web scraping can be a useful way to obtain information to create novel datasets, but it has its limitations as well. Although setting up the program to extract the data may be a one-time investment, actually pulling large quantities of information can take time, depending on the processing power of computer and/or the network connection. Based on the size of the dataset desired, this must be accounted for. In addition, once the data is collected, some fields may be incomplete, or data may be missing. As such, it is important to pre-process the data appropriately.

In the realm of CSE and research, NLP can be used to process text and linguistic information in meaningful ways including understanding spoken language dialogue, web scraping to build data sets, data mining, entity recognition, information retrieval, information extraction, and the assessment of qualitative data [24], [31]. NLP is a tremendous asset for automating the process of extrapolating concepts, keywords, and relevant information from large quantities of text. It could also be used for summarizing qualitative articles and interviews, as well as student feedback. Furthermore, concept mapping and dynamic modeling may provide organization, structure, and representation as well as additional information recovery. In addition, processing language could be used to examine the corpora of dialogue between tutors and students, and to assess classroom dialogue between teachers and students [24].

NLP-based technology has many important applications, from its use as an education tool to its use as an educator [24], [46]. MOOCs draw on connectivist principles to facilitate access to software and systems to promote learning and sharing. However, MOOCs are environments where the student to teacher ratio can be widely disproportionate, and utilizing rapid analysis can assist in pruning through lengthy forums and posts. NLP could also be used to help draw instructors attention to students that may require assistance [24]. Additionally, it could be applied to create curricula or materials to evaluate students [47], [48], or for scoring students [49], [50]. Moreover, intelligent dialogue-based tutoring systems could be used to further knowledge for students that may require additional support. Although it is unlikely computers can replace humans, and indeed students score higher when working with a human tutor than a computer tutor [51], it could provide a way to increase access for students and to offer additional resources to complement their classroom education.

NLP also has its own caveats. Although a computer may be able to follow explicit commands and rules, it is unable to extrapolate anything except what it is programmed to do. For example, when parsing through content, it may be harder to convey sarcasm or emotions, obscuring the intended meaning if analyzing dialogue between teachers and students. Like with any program, there is also the possibility of an inherent programmer bias, and algorithmic accountability must be considered. It is always important to think about what dictionaries are applied to text, and what information is relevant to ensure empirical methods are applied for analysis.

Although web scraping and NLP do not need to entirely replace other methods for processing and evaluation, they can be beneficial for answering novel questions/problems, and can complement other techniques. By introducing reliable and reproducible methods for data analysis and model building, and utilizing stepwise procedures that adhere to scientific principles, NLP can be applied to validate qualitative data as well. Using systematic methodologies could provide empirical results that demonstrate rigorous experimental practice for evaluating such data, expanding the knowledge in the field, which in turn could be used to aid in educational development. We hope that going forward, CSE researchers will consider applying these techniques to their own work either as a primary method, or as a secondary means of confirmation.

C. Conclusion

In this work, we have demonstrated techniques that could be utilized for numerous applications to further knowledge in CSE. Additionally, we provided a specific example of an application for these methods. The findings presented illustrate that there are several important areas to consider addressing in curricula revisions to bolster graduate employability.

To summarize, programming and testing are considered widely important skills for obtaining a job in CS. Moreover, knowledge of Python is often requested, and machine learning is a knowledge area that students may need to be familiar with. Furthermore, it should be noted that the methods suggested in this example could be built upon to collect data over a longer timeframe, or automated for further data analysis using increasingly sophisticated modeling techniques.

In general, we show that the application of web scraping and NLP are useful in obtaining and analyzing pertinent information from internet sources. Individually or in combination, they can expedite manual tasks, and can be used with other techniques for additional validation. Since new information is constantly being generated [6], finding new acquisition methods can only serve to benefit education. Considering the principles of connectivism, we propose that going forward, computer science educators should contemplate utilizing the tools described to further their own work, to improve transfer of knowledge and to inform pedagogical practice.

REFERENCES

[1] S. Fincher and M. Petre, *Computer science education research*. CRC Press, 2004.

[2] J. J. Randolph, G. Julnes, E. Sutinen, and S. Lehman, "A methodological review of computer science education research," *Journal of Information Technology Education: Research*, vol. 7, no. 1, pp. 135–162, 2008.

[3] A. M. Christie, *Software process automation: the technology and its adoption*. Springer Science & Business Media, 2012.

[4] G. Siemens, "Connectivism: Learning as network-creation," *ASTD Learning News*, vol. 10, no. 1, pp. 1–28, 2005.

[5] G. Siemens, "Connectivism: Learning theory or pastime of the self-amused," 2006.

[6] G. Siemens, "Connectivism," *Foundations of Learning and Instructional Design Technology*, 2017.

[7] S. d. S. Sirisuriya, "A comparative study on web scraping," *8th International Research Conference, KDU*, p. 135–140, November 2015.

[8] D. Jurasky and J. H. Martin, "Speech and language processing: An introduction to natural language processing," *Computational Linguistics and Speech Recognition. Prentice Hall, New Jersey*, 2000.

[9] K. M. Alhawiti, "Natural language processing and its use in education," *Computer Science Department, Faculty of Computers and Information technology, Tabuk University, Tabuk, Saudi Arabia*, 2014.

[10] R. B. Mbah, M. Rege, and B. Misra, "Discovering job market trends with text analytics," in *2017 International Conference on Information Technology (ICIT)*. IEEE, 2017, pp. 137–142.

[11] M. A. Mardis, J. Ma, F. R. Jones, C. R. Ambavarapu, H. M. Kelleher, L. I. Spears, and C. R. McClure, "Assessing alignment between information technology educational opportunities, professional requirements, and industry demands," *Education and Information Technologies*, vol. 23, no. 4, pp. 1547–1584, 2018.

[12] R. Florea and V. Stray, "Software tester, we want to hire you! an analysis of the demand for soft skills," in *International Conference on Agile Software Development*. Springer, 2018, pp. 54–67.

[13] S. Downes, "Learning networks and connective knowledge. instructional technology forum: Paper 92," 2006.

[14] R. Kop, "Web 2.0 technologies: Disruptive or liberating for adult education," in *Adult Education Research Conference*, 2008, pp. 5–7.

[15] D. C. Kropf, "Connectivism: 21st century's new learning theory." *European Journal of Open, Distance and E-learning*, vol. 16, no. 2, pp. 13–24, 2013.

[16] S. Al-Shehri, "Connectivism: A new pathway for theorising and promoting mobile language learning," *International Journal of Innovation and Leadership on the Teaching of Humanities*, vol. 1, no. 2, pp. 10–31, 2011.

[17] B. Kerr, "Msg. 1, the invisibility problem. online connectivism conference: University of manitoba," 2007.

[18] S. Downes, "Msg 1, re: What connectivism is. online connectivism conference: University of manitoba," 2007.

[19] F. Bell *et al.*, "Connectivism: a network theory for teaching and learning in a connected world," *Educational Developments, The Magazine of the Staff and Educational Development Association*, vol. 10, no. 3, 2009.

[20] B. Duke, G. Harper, and M. Johnston, "Connectivism as a digital age learning theory," *The International HETL Review*, vol. 2013, no. Special Issue, pp. 4–13, 2013.

[21] J. Utecht and D. Keller, "Becoming relevant again: Applying connectivism learning theory to today's classrooms." *Critical Questions in Education*, vol. 10, no. 2, pp. 107–119, 2019.

[22] A. T. Bates, "Teaching in a digital age: Guidelines for designing teaching and learning," 2018.

[23] W. Drexler, "The networked student model for construction of personal learning environments: Balancing teacher control and student autonomy," *Australasian journal of educational technology*, vol. 26, no. 3, 2010.

[24] D. Litman, "Natural language processing for enhancing teaching and learning," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[25] S. Munzert, C. Rubba, P. Meißner, and D. Nyhuis, *Automated data collection with R: A practical guide to web scraping and text mining*. John Wiley & Sons, 2014.

[26] J. Ward, *Instant PHP web scraping*. Packt Publishing Ltd, 2013.

[27] P. Jackson and I. Moulinier, *Natural language processing for online applications: Text retrieval, extraction and categorization*. John Benjamins Publishing, 2007, vol. 5.

[28] A. Kao and S. R. Poteet, *Natural language processing and text mining*. Springer Science & Business Media, 2007.

[29] E. Loper and S. Bird, "NLTK: the natural language toolkit," *Proceedings of the ACL-02 Workshop on Effective tools and methodologies for teaching natural language processing and computational linguistics*, 2002.

[30] J. Silge and D. Robinson, *Text mining with R: A tidy approach*. O'Reilly Media, Inc., 2017.

[31] S. Bird, E. Klein, and E. Loper, *Natural language processing with Python: analyzing text with the natural language toolkit*. O'Reilly Media, Inc., 2009.

[32] F. Suleman, "The employability skills of higher education graduates: insights into conceptual frameworks and methodological options," *Higher Education*, vol. 76, no. 2, pp. 263–278, 2018.

[33] A. Radermacher and G. Walia, "Gaps between industry expectations and the abilities of graduates," in *Proceeding of the 44th ACM technical symposium on Computer science education*, 2013, pp. 525–530.

[34] A. J. Hurst, A. Carbone, M. G. Eley, A. E. Ellis, D. L. Hagan, S. J. Markham, J. I. Sheard, J. E. Tuovinen, J. Lynch, and F. E. Collins, "Teaching ICT-the ICT-Ed project-the report on learning outcomes and curriculm development in major university disciplines in information and communication technology," 2001.

[35] F. Gurcan and N. E. Cagiltay, "Big data software engineering: Analysis of knowledge domains and skill sets using LDA-based topic modeling," *IEEE Access*, vol. 7, pp. 82 541–82 552, 2019.

[36] K. Cai, "Vancouver, Portland leap in ranking of best cities for tech jobs," Jul 2019. [Online]. Available: https://www.forbes.com/sites/kenrickcai/2019/07/15/best-cities-for-tech-jobs-vancouver-portland-leap-rankings/

[37] P. Adler, R. Florida, K. King, and C. Mellander, "The city and high-tech startups: The spatial organization of schumpeterian entrepreneurship," *Cities*, vol. 87, pp. 121–130, 2019.

[38] N. Kolakowski, "Top 25 cities for tech job postings," December 2019. [Online]. Available: https://insights.dice.com/2019/12/20/top-25-cities-tech-job-postings/

[39] L. Richardson, "Beautiful soup," Jan 2020. [Online]. Available: https://www.crummy.com/software/BeautifulSoup/

[40] S. Behnel, M. Faassen, and I. Bicking, "lxml: Xml and html with python," 2005.

[41] "word_ cloud," Feb 2020. [Online]. Available: https://github.com/amueller/word_cloud/

[42] A. Radermacher, "Evaluating the gap between the skills and abilities of senior undergraduate computer science students and the expectations of industry," Ph.D. dissertation, North Dakota State University, 2012.

[43] V. Garousi, G. Giray, E. Tüzün, C. Catal, and M. Felderer, "Aligning software engineering education with industrial needs: a meta-analysis," *Journal of Systems and Software*, vol. 156, pp. 65–83, 2019.

[44] M. Karlson and F. Olsson, "Investigating the newly graduated student-sexperience after university," 2019.

[45] S. A. Wallace, I. Russell, and Z. Markov, "Integrating games and machine learning in the undergraduate computer science classroom," in *Proceedings of the 3rd international conference on Game development in computer science education*, 2008, pp. 56–60.

[46] J. Burstein, J. Sabatini, and J. Shore, "Natural language processing for educational applications," in *The Oxford Handbook of Computational Linguistics 2nd edition*, 2014.

[47] E. Miltsakaki and A. Troutt, "Real time web text classification and analysis of reading difficulty," in *Proceedings of the third workshop on innovative use of NLP for building educational applications*, 2008, pp. 89–97.

[48] S. E. Petersen and M. Ostendorf, "A machine learning approach to reading level assessment," *Computer speech & language*, vol. 23, no. 1, pp. 89–106, 2009.

[49] E. Miltsakaki and K. Kukich, "Evaluation of text coherence for electronic essay scoring systems," *Natural Language Engineering*, vol. 10, no. 1, pp. 25–55, 2004.

[50] A. Loukina, K. Zechner, L. Chen, and M. Heilman, "Feature selection for automated speech scoring," in *Proceedings of the Tenth Workshop on Innovative Use of NLP for Building Educational Applications*, 2015, pp. 12–19.

[51] K. VanLehn, "The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems," *Educational Psychologist*, vol. 46, no. 4, pp. 197–221, 2011.