Skill Finder: Automated Job-Resume Matching System

Thimma Reddy Kalva
Utah State University

Follow this and additional works at: https://digitalcommons.usu.edu/gradreports

Part of the Computer Sciences Commons

Recommended Citation
https://digitalcommons.usu.edu/gradreports/343
SKILL FINDER: AUTOMATED JOB-RESUME MATCHING SYSTEM

by

Thimma Reddy Kalva

A Plan B report submitted in partial fulfillment of the degree requirements for the degree of

MASTER OF SCIENCE

in

Computer Science

Approved:

_____________________
Dr. Nicholas Flann, PhD
Major Professor

_____________________
Dr. Vladimir A. Kulyukin, PhD
Committee Member

_____________________
Dr. Xiaojun Qi, PhD
Committee Member

UTAH STATE UNIVERSITY
Logan, Utah

2013
ABSTRACT

SKILL FINDER: AUTOMATED JOB-RESUME MATCHING SYSTEM

by

Thimma Reddy Kalva, Master of Science
Utah State University, 2013

Major Professor: Dr. Nicholas Flann, PhD
Department: Computer Science

Skill Finder is a tool which ranks the student skills from the resumes to the job requirements from Employers, Department and Faculty looking for student interns, full-time employees and also Research, Graduate and Teaching Assistants. Skill Finder hosts student resumes, Academic history and contact information. Authorized users from the Department can post jobs, view job applicants and the Skill finder automatically sorts the students resumes based on the degree of match with the job. Skill finder also sends e-mail alerts to students about the jobs posted and keeps track of employer’s history by maintaining the complete history of jobs posted from the employers.

(56 Pages)
ACKNOWLEDGEMENTS

Foremost, I would like to extend my sincere gratitude to my advisor Dr. Nicholas Flann for his support, guidance and valuable suggestions. I am very grateful for his insightful discussions and encouragement which helped me resolve several design issues. Without his guidance and help this report would not have been possible.

I am grateful to my committee members Dr. Vladimir A.Kulyukin and Dr. Xiaojun Qi for their valuable guidance and support.

I especially thank my family, Bhaskar Reddy Kalva, Sunanda Kalva, Anitha Kalva and Harish Reddy Gajulapalle and my friends for their immense support and care. I would not have made it this far without them.

Thimma Reddy Kalva
# CONTENTS

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ABSTRACT</td>
<td>iii</td>
</tr>
<tr>
<td></td>
<td>ACKNOWLEDGEMENTS</td>
<td>iv</td>
</tr>
<tr>
<td>1</td>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>REQUIREMENT ANALYSIS</td>
<td>2</td>
</tr>
<tr>
<td>2.1</td>
<td>Use-Cases</td>
<td>2</td>
</tr>
<tr>
<td>2.2</td>
<td>Structural Analysis</td>
<td>7</td>
</tr>
<tr>
<td>2.3</td>
<td>Functional Requirements</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>ARCHITECTURAL DESIGN</td>
<td>10</td>
</tr>
<tr>
<td>3.1</td>
<td>User Interface Design</td>
<td>15</td>
</tr>
<tr>
<td>3.2</td>
<td>Database Layer</td>
<td>15</td>
</tr>
<tr>
<td>4</td>
<td>SKILL FINDING ALGORITHM</td>
<td>17</td>
</tr>
<tr>
<td>4.1</td>
<td>Training a model</td>
<td>18</td>
</tr>
<tr>
<td>4.2</td>
<td>Identifying the entity skill using the trained model</td>
<td>20</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Extract skills by using Named Entity Recognition training model</td>
<td>21</td>
</tr>
<tr>
<td>4.2.2</td>
<td>Finding more skills by applying rules on results from step 1</td>
<td>25</td>
</tr>
<tr>
<td>5</td>
<td>SOFTWARE TESTING</td>
<td>29</td>
</tr>
<tr>
<td>5.1</td>
<td>Unit Testing</td>
<td>29</td>
</tr>
<tr>
<td>5.2</td>
<td>Integration Testing</td>
<td>29</td>
</tr>
<tr>
<td>6</td>
<td>CONCLUSION</td>
<td>55</td>
</tr>
<tr>
<td>7</td>
<td>REFERENCES</td>
<td>56</td>
</tr>
</tbody>
</table>
LIST OF FIGURES

Figure 1: Actors in Skill Finder........................................................................................................3
Figure 2. Use-case diagram describing the goals of the user Student.................................................4
Figure 3. Use case diagram describing the goals of the user Faculty...............................................5
Figure 4. Use Case diagram describing the goals of the user System itself.....................................6
Figure 5. Class diagram describing the Student entity.................................................................7
Figure 6. Class diagram describing the Faculty entity.................................................................8
Figure 7. 3-Tier client/server architecture.....................................................................................10
Fig 8. MVC Architecture.............................................................................................................12
Fig 9. Student User fills the profile................................................................................................13
Fig 10. URL, Servlet mapping........................................................................................................13
Fig 11. Invoking the model class.....................................................................................................14
Fig 12. Fetch data from Database....................................................................................................14
Fig 13. Dispatch the response to the View......................................................................................14
Fig 14. User Interface navigation....................................................................................................15
Fig 15. Enhanced Entity Relationship model of Skill Finder database schema............................16
Fig 16. Training data Job Snippets Download...............................................................................19
Fig 17. Training data downloader....................................................................................................19
Fig 18. Sample Training data with skills marked with spans........................................................20
Fig 19. Creating the training model to identify skills......................................................................20
Fig 20. Sentence detection and Tokenization.................................................................................23
Fig 21. Find skills using the trained model.....................................................................................23
Fig 22. Skills required identified by Skill Finding Algorithm.........................................................27
Fig 23. Sample of Skills extracted from student resume...............................................................28
Fig 24. List of Applicants with required skills.................................................................................30
CHAPTER I
INTRODUCTION

SKILL FINDER:

Selecting a candidate for a job is a multi-phase process. First, the job will be published. Secondly, an initial screening will be done based on rigid qualifications which are considered required like degree and major. Third, the selection will be based on qualifications like professional knowledge and professional skills and then the interview process. In this process an automatic skill matching algorithm can help greatly by reducing the labor intensity of manual selection of resumes. By using a statistical classifier trained on identifying skills required from job description and skills possessed from the resumes a matching algorithm is developed for ranking the students.

Skill Finder is web-application developed using J2EE technologies running with apache-tomcat server. It uses JAVA MVC framework to manage the business logic and handle the client requests to the server. The database interface is developed using java JDBC technologies to access the MYSQL database. The user interface uses JSP’s and javascript to provide a dynamic web environment. Users can access the application over internet and perform the actions which they are authorized to do like post jobs, apply to jobs and receive email alerts.
CHAPTER II
REQUIREMENT ANALYSIS

This chapter documents the functional requirements of the Skill Finder tool using Unified Modeling Language (UML) diagrams use-case and class diagrams. Use-case diagrams visualize, specify and document the behavior of the system. Use-Case diagrams in Section 2.1 provide a high level overview of the functionality the Skill Finder tool should have. Class diagrams are the building blocks of Object-Oriented programming [1]. The Class diagrams in Section 2.2 describes the systems classes, their attributes, operations and the relationship among them. Class diagrams provide the following important insights to the developer in order to solidify the design of the system. They help in conceptualization, specification and implementation.

2.1. Use-Cases:

Use-Cases capture the functional i.e. the behavioral requirements of a system. Use-Cases also tell the interactions between various actors in a system. Actors can be a user or the system itself. Actor has a goal in using the system. A goal can be anything the actor wants to achieve by interacting with the system. Use-Cases capture all the different goals the various actors have in using the system. Use-Cases are typically found in the requirement specifications. UML use-case diagrams serve as a visual table of contents to written use-cases [2].

Key actors identified in the Skill Finder tool include Students who wants to apply for the jobs, Faculty or authorized users from the department who posts the jobs, and the Skill Finder system administrator. The users and their interactions are depicted in Figure 1.
Students will have goals like creating their profile, view job listings and apply for the jobs.

Faculty will have goals like publish a job, view the applicants, and get a list of short-listed students.
Administrator will typically maintain the system and troubleshoot the problems.

Figure 2. Use-case diagram describing the goals of the user Student.

The use-case diagram in Figure 2. captures the goals of the user Student. The Student user creates the profile by filling the form with his personal information, academic history consisting of his major of study etc. and uploads a resume and cover letter. The student user checks the job listing available in the Skill Finder and applies for a job by submitting his resume and cover letter from the profile or by uploading a new resume and cover letter. The student user also receives email alerts as when a new job is posted in the Skill Finder tool.
Figure 3. Use case diagram describing the goals of the user Faculty

The use-case diagram in Figure 3. describes the goals of the user Faculty. Faculty user creates the job by filling a form with details like job title, job summary with skills required and employer details etc.. The skill finder tool sends an email alert to the Student users as soon as the job is submitted in the Skill Finder tool. The Faculty user checks the list of student applicants and gets a list of shortlisted students by running the skill finder algorithm.
Figure 4. Use Case diagram describing the goals of the user System itself.
2.2 Structural Analysis

Class diagrams are used to illustrate relationships between classes of an object-oriented system in a Graphical way. Class diagrams are used to refine the use cases. A class diagram describes the kinds of objects that populate the system and the relationships that exists between the classes and objects. Classes will have features like attributes and methods. The UML class diagrams in this section describe the key object classes in the Skill Finder tool.

Figure 5. Class diagram describing the Student entity

From the functional requirements I was able to observe that there are three types of users Student, Faculty and System administrator. Student is a type of user group and has account_details with properties like anumber, first_name, last_name, email_id etc.. Student also has academic history maintained in the class student_current_program with properties like current_program, major, expected_grad_date etc... Student also has a class student_resume which maintains the student resume details and the
The student class has an association with the jobs_applied class which maintains the jobs applied by the student.

Figure 6. Class diagram describing the Faculty entity

Figure 6 describes the class Faculty and its associated classes. Faculty class is a type of class user_group and has an account_details class. Like the student_account_details class faculty account_details class has properties like first_name, last_name, anumber, email_id etc. As faculty creates jobs, the faculty class has association with jobs_info class. The jobs_info class maintains all the job requests from the department as well as the requests from employers looking for students. The jobs_info class has properties like job_description, employer, whether the job is a partTime_fullTime, job_location etc.
2.3 Functional Requirements

The application's functionality with respect to user groups:

Students:

- Host Student Resumes and Academic history.
- Apply to jobs
- Receive email alerts

Department:

- Look for RA/TA and intern/Full-Time positions
- Get a list of shortlisted candidates matching the job description.
- Keeps a history of each employer's requests that can be looked up.
CHAPTER 3
ARCHITECTURAL DESIGN

The Skill Finder application is developed using the 3-tier architecture the client layer, application layer and the database layer. The client a web browser only displays the GUI and data. The middle tier plays an intermediary role by running application programs and storing business rules that are used to access the data from the database server. The application layer accepts the requests from the client, processes the request and send queries to the database server. The data response from the database server is further processed and filtered by the application layer before presenting it to the client.

Figure 7. 3-Tier client/server architecture

The 3-tier architecture offers advantages like scalability as the application servers can be deployed on many machines and the database no longer requires a connection from every client. The middle tier improves data integrity by ensuring that only valid data is allowed to be updated in the
database. The 3-tier architecture offers improved security since the client doesn’t have direct access to the client. The 3-tier architecture is implemented using Model-View-Controller design pattern. MVC helps to decouple data access and business logic from the manner which it is displayed to the user [3].

Model: The model represents data and the rules that govern access to and updates of this data. In a web application, a model often serves as a software representation of a real world process.

View: The view renders the contents of a model. It specifies the way the data should be presented. The view updates its presentation as and when the data of a model changes. This is achieved by using the push model where the view registers itself with the model for change notifications or by using a pull model where in the view calls the model as and when it needs the latest data.

Controller: The controller translates the user’s requests into actions that the model will perform. The user requests are sent to the controller as GET and POST HTTP requests. Based on the request the controller presents various results as a web page.
1. The *client* makes a request for the .html page.

2. The *container* retrieves the .html page

3. The *container* returns the page to the browser where the user fills the form.

Fig 8. MVC Architecture
4. The browser sends the request data to the container.

5. The container finds the correct servlet based on the URL, and passes the request to the servlet.

```
87  <servlet>
88     <servlet-name>updateStudentProfile</servlet-name>
89     <servlet-class>com.pupil.UpdStudentProfile</servlet-class>
90   </servlet>
91  </servlet-mapping>
92  <servlet-mapping>
93     <servlet-name>updateStudentProfile</servlet-name>
94     <url-pattern>/updateProfile.do</url-pattern>
   </servlet-mapping>
```

Fig 10. URL, Servlet mapping
6. The servlet calls the model for the business logic.

```java
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException
// Auto-generated method stub
HttpSession session = request.getSession();

// invoke the model
GetStudentProfileFromDb studProfDb = new GetStudentProfileFromDb();

studProfDb.getStudentAccountDetails(request, response, (String)session.getAttribute("email"));
studProfDb.getStudentContactDetails(request, response, (String)session.getAttribute("email"));
studProfDb.getStudCurrentProgram(request, response, (String)session.getAttribute("email"));
studProfDb.getStudentResumeDetails(request, response, (String)session.getAttribute("email"));
```

Fig 11. Invoking the model class

7. The model class returns an answer, which the servlet class adds to the request object.

```java
public void getStudentAccountDetails(HttpServletRequest request, HttpServletResponse response, String emailId, Connection con){
    String query = "SELECT * FROM account_details WHERE EMAIL_ID=?";
    StudentAccountDetails stAcDet = new StudentAccountDetails();
    try {
        PreparedStatement stmt = (PreparedStatement) con.prepareStatement(query);
        stmt.setString(1, emailId);
        ResultSet rs = stmt.executeQuery();
        while(rs.next()){
            stAcDet.setAccountNumber(rs.getString(1));
            stAcDet.setGroupId(rs.getString(2));
            stAcDet.setFirstName(rs.getString(3));
            stAcDet.setLastName(rs.getString(4));
            stAcDet.setEmailId(rs.getString(5));
            stAcDet.setAccountNumber(rs.getString(6));
            request.setAttribute("StudentAccountDetails", stAcDet);
        }
    } catch (SQLException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}
```

Fig 12. Fetch data from Database

8. The servlet forwards the request to the JSP.

```java
// forward the request to the view
RequestDispatcher view = request.getRequestDispatcher("updateStudentProfile.jsp");
view.forward(request, response);
```

Fig 13. Dispatch the response to the View

9. The JSP gets the answer from the request object.

10. The JSP generates a page for the container.

11. The container returns the page to the user.
3.1 User Interface Design

The User Interface package contains classes and html forms. These forms enable the student user to perform the following tasks.

- login securely
- visit profile
- update contact information
- update academic history
- update resume and cover letter
- view jobs
- apply jobs

![User Interface navigation diagram]

3.2 Database Layer

After analyzing the requirements a conceptual schema is designed which helped in detailing the data requirements. High level user queries and operations were identified during the phase. Next the database is designed using the MYSQL database. A database schema is developed transforming the high level data model into implementation data model.
Fig 15. Enhanced Entity Relationship model of Skill Finder database schema
CHAPTER 4

SKILL FINDING ALGORITHM

The objective of this algorithm is to rank the student resumes based on skills with respect to a job. Student resumes can be ranked by comparing the skills from the resume to the skills required in the job description. From the observations I found that skills are proper nouns (named entities). The process of identifying nouns is called Named Entity Recognition (NER) [4]. One approach for finding named entities is to use a combination of lists and regular expressions. In this approach we basically need to codify the observations and patterns as rules and then apply these rules on the text. This approach is difficult to maintain because of the reasons like maintaining the lists is labor intensive, many proper nouns are also valid in other rules i.e dealing with ambiguity is hard and also it is difficult to model dependencies between nouns across a document using rules based on regular expressions.

Other approach that is easy to extend and doesn’t require creating large lists to be maintained is to use a statistical classifier to identify named entities [4]. Typically a classifier looks at each word in a sentence and decides if a word is start of a named entity or if it is a continuation of a named entity or not part of a named entity at all. By combining these predictions, a classifier identifies a sequence of words that makes up a named entity.

Classifiers use different approaches for identifying nouns. One approach is to use the tagging approach or regular expressions to identify the text that contains a name of any type and secondly distinguish the different types of nouns. Another approach is to simultaneously distinguish between different types of nouns by predicting entity type along with noun start or continuation. Another approach to use different classifiers for identifying different noun types.

Regardless of the classification approach, a classifier needs to be trained on a collection of human trained text to learn how to identify names. The advantages of this approach are lists can be incorporated
as approaches as one source of information, it is easy to model a context within a sentence and in a document, the classifier can be retrained to add new features. The main disadvantage of such approach is the need for human annotated data. The training data should contain at least 15000 sentences to create a model which performs well.

4.1 Training a model:

There are Name Entity Recognition api’s like apache OpenNLP [8], Stanford Name Entity Recognizer [9] which can detect named entities. Apache openNLP framework is chosen because of great api, community support and free and open source software. This api’s have pre-trained models which can detect named entities like location, time, person, organization, money, percent, data. To make a Named Entity recognizer detect a new entity like Skill a model needs to be trained. The biggest difficulty in training a model is to create training data sufficient enough for statistical modeling. To create the training data I downloaded more than 3000 jobs and 80 resumes from the website indeed using the web service api [10].

The indeed web service gives an xml response which consists of a snippet about the job and not the complete job description. But, the xml response has the url for the complete job description. The code in the below figure is used to download the job snippets and the detailed job url’s and the data is written to a file.
for(int i=0;i<numberOfJobs+1;){
    String link="http://guide.indeed.com/r/Sagorika-Roychowdhury/0d76f9b94e6e958a/pdf";
    try {
        url = new URL("http://api.indeed.com/ads/apisearch?publisher=3662876754714213&q=software%20n
        &start=24&limit=1000&fromage=&filter=&latlong=1&co=us&chnl=&" +
        "userid=1.2.3.4&useragent=Mozilla/5.0%20Firefox%20v%202");
        url = new URL(link);
        is = url.openStream(); // throws an IOException
        br = new BufferedReader(new InputStreamReader(is));
        while ((line = br.readLine()) != null) {
            bw.append(line);
        }
    //break;
    } catch (MalformedURLException mue) {
        mue.printStackTrace();
    } catch (IOException ioe) {
        ioe.printStackTrace();
    }
    finally {
        if (is != null) is.close();
    } catch (IOException ioe) {
        // nothing to see here
    }
}

Fig 16. Training data Job Snippets Download

The file with job snippets is parsed by writing the python script below and the url’s for the
detailed jobs are extracted. Using the urls the training data is downloaded.

```python
from bs4 import BeautifulSoup
import re
import sys
import requests

def parseLog(file):
    file = sys.argv[1]
    handler = open(file).read()
    soup = BeautifulSoup(handler)
    indeed_jobs_description_3000_spaces.txt, '
    for tag in soup.find_all(re.compile('url')):
        r = requests.get(tag.get_text().strip())
        soups = BeautifulSoup(r.content)
        soups.prettify().encode('utf-8')
        f.write(soups.prettify().encode('utf-8'))
        for tags in soups.find_all('span'):
            str1="summary"
            str2="tags"
            tags.get('class')
            if str2 and str1 in str2:
                for string in tags.strings:
                    f.write(string.encode('utf-8'))
                    f.write('test2')
                    f.write('tags.string
            f.close()
```

Fig 17. Training data downloader
Apache openNLP has a command line tool as well as a training api which can be used to train the model. The jobs data (more than 15000 sentences) is converted to openNLP name finder training format by bootstrapping from an initial handmade list. Which is one sentence per line, tokenized with spans which mark the skills.

```
Adaptable to a business environment with fast changing requirements and priorities .
<START:skill> Javascript <END> / <START:skill> GWT <END> <START:skill> web <END>
<START:skill> programming <END> highly desirable
```

Fig 18. Sample Training data with skills marked with spans

Once the training data is generated, command line tool of apache openNLP is used to train the skill name finder.

```
thimma@thimma-ThinkPad-T400:-$ ./opennlp TokenNameFinder .../.../Fall_2013/myprojects/tests/nov-1/ner-sent-skill.bin
```

Fig 19. Creating the training model to identify skills

4.2 Identifying the entity skill using the trained model

In order to rank resumes of students with respect to a job, skills of a student are extracted and skills required for a job are extracted from the job description. Then students having most number of required skills for a job are ranked higher.

Skills are identified in two steps

1. Using the Named Entity Recognition training model.
2. Finding more skills by applying rules on results from step 1.
4.2.1 Extract skills by using Named Entity Recognition training model:

When a job is posted by the faculty or when a resume in student’s profile is updated the NER algorithm gets triggered and extracts skills from the job description and student resume respectively. A department user typically gets an email from an employer about a job available in their organization. Department user copies the job description from the email and pastes it in the job summary field and saves it to the database clicking the submit button. Below is a job summary from a job posted in website Glassdoor.com. Skills are highlighted as Bold manually for reference.

**Input:**

**Job Summary:**

Want to work at a start-up where you can really make a difference?

Glassdoor, the world’s fastest growing career community and 2013 Webby Winner for Best Employment Website, is looking for a talented Sr. Java Software Engineer to help take our product to the next level. This is your chance to share your work with a worldwide community of over 20M members and make your mark on the #1 employment app on Facebook and we’re just getting started.

**Responsibilities**

As a Senior Java **Software Engineer**, you will play a central role in the design and development of the Glassdoor product. The ideal candidate will have a passion for development and a strong background in building dynamic, high-volume **web sites** with **Java** using an **Agile** development methodology.

As a fast-growing startup, we look for self-starters who thrive in a fast-paced, **agile** environment which means wearing many hats, being able to change direction quickly, and showing an eagerness to learn new technologies as the need arises. In this role you work with a small, collaborative team of engineers, product managers, and designers so excellent **interpersonal** and **communication skills** are also a must.

And most importantly we look for people that can prioritize, multi-task, and deliver because it’s a lot more fun to get things done.

**Skills and Requirements**

5+ years of software development experience on large scale, high-volume sites
Experience with faster, lighter? Java tools including refactoring IDE’s, Spring, Hibernate, Lucene and other open source technologies

Understanding of presentation layer technologies such as HTML, CSS, templating frameworks (Sitemesh, Freemarker, Velocity, Tiles), Javascript, AJAX and Javascript libraries (YUI, Dojo, GWT)

Bonus: experience working on information retrieval or hard core search technologies

Bonus: experience building mobile web applications or products for international audiences, including localization

Extra Bonus Points: you?re fun to be around!

Why Glassdoor?

We believe that work is one of the most important parts of our lives, so we also believe in a winning culture and great benefits:

Competitive salary and stock options

Great culture & team spirit

Take a break when you need it, unlimited PTO

100% company paid medical/dental/vision/life insurance (90% dependent)

Fully stocked break rooms with unlimited drinks, snacks and daily bbq

Pet-friendly office

Onsite yoga and meditation room

Paid volunteer days

Mac or P, you choose!

Sunny Sausalito offices just a block from the water

2012 & 2013 Winner Best Places to Work in the North Bay

Team outings (Ball games, hikes, paddle-boarding etc.)

Relocation assistance provided
On submit, the skill finding algorithm is triggered.

- Job Summary is split into each sentence and is tokenized before feeding to the skill finding model. Sentence detection and tokenization are done using apache openNLP framework.

```java
//Load the sentence detection model
SentenceDetectorME sentenceDetector = new SentenceDetectorME(model);
//Load the tokenization model
Tokenizer tokenizer = new TokenizerME(tokenMmodel);

HashSet<String> skills = new HashSet<String>();
//String sent=request.getParameter("job_description");
jobDesc = mapParameters.get("job_summary")[0];
if(jobDesc.length()!=0){
    //sentence detection
    String sentences[]=sentenceDetector.sentDetect(jobDesc);
    SFParseFindSkill parseFindSkills = new SFParseFindSkill();
    for (int si = 0; si < sentences.length; si++){
        //tokenization
        Span[] tokenSpans = tokenizer.tokenizePos(sentences[si]);
        String[] tokens = Span.spansToStrings(tokenSpans, sentences[si]);
        //Find Skills in each sentence
        Span[] names = nameFinder.find(tokens);
        for (int ni = 0; ni < names.length; ni++) {
            Span startSpan = tokenSpans[names[ni]].getStart();
            int nameStart = startSpan.getStart();
            Span endSpan = tokenSpans[names[ni]].getEnd() - 1;
            int nameEnd = endSpan.getEnd();
            String name = sentences[si].substring(nameStart, nameEnd);
            if(!skills.contains(name.toLowerCase())){
                skills.add(name.toLowerCase());
            }
        }
    }
}
```

Fig 20. Sentence detection and Tokenization

- Once the data is setup, text is fed to skill finding routine.

```java
//load the skill model
SkillModelIn = new FileInputStream("/home/thimma/Fall_2013/myprojects/tests/nov-1/en-ner-sent-skill.bin");
Span[] tokenSpans = tokenizer.tokenizePos(sentences[si]);
String[] tokens = Span.spansToStrings(tokenSpans, sentences[si]);
//Find Skills in each sentence
Span[] names = nameFinder.find(tokens);
for (int ni = 0; ni < names.length; ni++) {
    Span startSpan = tokenSpans[names[ni]].getStart();
    int nameStart = startSpan.getStart();
    Span endSpan = tokenSpans[names[ni]].getEnd() - 1;
    int nameEnd = endSpan.getEnd();
    String name = sentences[si].substring(nameStart, nameEnd);
    if(!skills.contains(name.toLowerCase())){
        skills.add(name.toLowerCase());
    }
}
```

Fig 21. Find skills using the trained model
The output of this routine is text with skills identified and enclosed in tags <START:skill> <END>.

Output:

Job Description

Glassdoor, the world’s fastest growing career community and 2013 Webby Winner for Best Employment Website, is looking for a talented Sr. <START:skill> Java <END> <START:skill> Software Engineer <END> to help take our product to the next level. This is your chance to share your work with a worldwide community of over 20M members and make your mark on the #1 employment app on Facebook and we’re just getting started.

Responsibilities

As a Senior <START:skill> Java <END> Software Engineer, you will play a central role in the design and development of the Glassdoor product. The ideal candidate will have a passion for development and a strong background in building dynamic, high-volume <START:skill> web <END> sites with <START:skill> Java <END> using an Agile development methodology.

As a fast-growing startup, we look for self-starters who thrive in a fast-paced, agile environment which means wearing many hats, being able to change direction quickly, and showing an eagerness to learn new technologies as the need arises. In this role you work with a small, collaborative team of engineers, product managers, and designers so excellent interpersonal and <START:skill> communication skills <END> are also a must. And most importantly we look for people that can prioritize, multi-task, and deliver because it’s a lot more fun to get things done.

Skills and Requirements

5+ years of <START:skill> software development <END> experience on large scale, high-volume sites
Experience with ?faster, lighter? <START:skill> Java <END> tools including refactoring IDE?s, Spring, Hibernate, <START:skill> Lucene <END> and other <START:skill> open source <END> technologies

Understanding of presentation layer technologies such as HTML, CSS, templating frameworks (Sitemesh, Freemarker, Velocity, Tiles), Javascript, <START:skill> AJAX <END> and <START:skill> Javascript <END> libraries (YUI, Dojo, GWT)

Bonus: experience working on information retrieval or hard core search technologies

Bonus: experience building mobile <START:skill> web applications <END> or products for international audiences, including localization

4.2.2 Finding more skills by applying rules on results from step 1:

The output of the skill finding routine has skills identified but not all. In order to find the unidentified skills rules are applied. From the observations I found that if a sentence from the output of skill finding routine has a skill identified in it and is having comma separated noun phrases then the comma separated noun phrases are also skills. For example in the below sentence python is a skill which is identified already, from the rule as python is surrounded by comma separated noun phrases we can determine the other noun phrases SOLR, MongoDB,Cassandra,Hadoop also as skills.

_in addition, technologies such as SOLR, MongoDB, Cassandra, Hadoop, and <START:skill> Python <END> will be incorporated as well._

The second observation is that when a skill is followed by a Preposition Para phrase which is followed by a Noun Phrase, then the Noun Phrase is observed as a skill. For example in the below sentence open source is a skill which is identified already in step 1, which is followed by a Preposition phrase such as and Noun Phrases SOLR,MongoDB,Cassandra,Hadoop. From this rule these Noun
Phrases can be determined as skills. Noun Phrases and Preposition Phrases are determined by using Stanford NLP Parser.

In addition, <START:skill> open source <END> technologies such as SOLR, MongoDB, Cassandra, Hadoop will be incorporated as well.

Skills from job description are stored in jobs_skills table and skills from student resumes are stored in resume_skills table. When faculty user checks for list of applicants for a job, applicants will be listed with students with most number of required skills for the job. A Cartesian product on job_skills and resume_skills produces the result of students with required skills for a job. Students with most number of skills are list on the top.
Fig 22. Skills required identified by Skill Finding Algorithm from job description

| 75 c#   |
| 75 c++  |
| 75 cassandra |
| 75 communication skills |
| 75 creation |
| 75 data |
| 75 data structures |
| 75 hadoop |
| 75 mongodb |
| 75 multimedia |
| 75 news |
| 75 open source |
| 75 problem solving |
| 75 professionals |
| 75 relational databases |
| 75 reporting |
| 75 search |
| 75 software engineer |
| 75 software engineering |
| 75 solr |
| 75 sql |
| A01234567 | hibernate |
| A01234567 | hql        |
| A01234567 | html       |
| A01234567 | integration |
| A01234567 | j2ee       |
| A01234567 | java       |
| A01234567 | javascript |
| A01234567 | Jdbc       |
| A01234567 | Jquery     |
| A01234567 | Jsp        |
| A01234567 | Mhv        |
| A01234567 | Ms access  |
| A01234567 | Mumps vista database system |
| A01234567 | Mvc databases |
| A01234567 | MyHealthcVet project |
| A01234567 | Operating systems |
| A01234567 | perforce    |
| A01234567 | Pl/sql      |

Fig 23. Sample of Skills extracted from student resume
CHAPTER 5
SOFTWARE TESTING

Software testing is a process of validating and verifying the quality of a product to provide stakeholders with information about the benefits and risks at implementation of the software product [11]. To test the quality and usability of Skill Finder, we performed unit testing and integration testing.

Sections 5.1 and 5.2 explain unit and integration testing.

5.1 Unit Testing:

Unit testing takes the smallest piece of testable software in the application, isolates it, and determines if it behaves as expected. Each unit is tested separately before being integrated into modules. A large percentage of defects are identified during unit testing [12].

Unit tests are written from programmer’s perspective. They ensure that a particular method of a class successfully performs a set of operations. Unit tests drives the design. In Skill Finder unit tests are performed on every function of the class throughout the development.

5.2 Integration Testing:

Integration testing is a logical extension of unit testing. In integration testing, two individual units already tested are combined into a component and tested. The idea is to test combinations of pieces and eventually expand the process to test all the modules with those of other groups. Eventually all the modules making up a process are tested together [12]. Integration testing is performed in three ways: the top-down, bottom-up, and umbrella approaches.

For Skill Finder we followed bottom-up approach, i.e the lowest level units were tested and integrated first. Below modules were tested and all the bugs encountered were resolved during testing.

1. Account creation for student and faculty user
2. Student Profile Creation
3. Job Creation

User from Department like Faculty logs in to Skill Finder and navigates to Create Job page by clicking on Create Job link in the Navigation Pane. This takes the user to Job Creation Page. Once the user has copied the job summary and filled the form, on submit the job is saved and an email alert will be sent to the Student users.

Fig 24. Job Creation

4. Email alerts to students on Job Posting
5. List jobs by filtration criteria.

Filters can be used while looking through the jobs. Jobs can be filtered by Company, Date Range during which the jobs are posted and also Jobs which are either open or closed.
6. **Apply Job**

A student user can navigate to view jobs page by clicking on view jobs link in the navigation pane and can apply for a job by clicking on apply job. The student user receives an email confirmation after applying for the job.

7. **View Applicants & Rank Student resumes**

A faculty user can view the applicants for a job by selecting the job from the list and clicking ‘view applicants’. The result of the skill finder algorithm for a job is described below. The Details of the job and the resumes of the applicants are also attached below. Skill finder algorithm sorts the applicants with the applicant having most number of required skills of the job on top and then the applicant having second most number of required skills and so on. A Cartesian product and an equi join on job\_skills having skills required for a job and resume\_skills having skills possessed by students produces the result in the Fig. 29 below. On hovering the mouse on View Profile link as shown in Fig 28, one can see the required skills the student is having.
### Fig 28. List of Applicants

<table>
<thead>
<tr>
<th>Name</th>
<th>Email</th>
<th>View Profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ho Chun Edison Wong</td>
<td><a href="mailto:hochunew@gmail.com">hochunew@gmail.com</a></td>
<td>View Profile</td>
</tr>
<tr>
<td>andrew miller</td>
<td><a href="mailto:adnrew@gmail.com">adnrew@gmail.com</a></td>
<td>View Profile</td>
</tr>
<tr>
<td>joshua thames</td>
<td><a href="mailto:joshuat@gmail.com">joshuat@gmail.com</a></td>
<td>View Profile</td>
</tr>
<tr>
<td>avneep dhanju</td>
<td><a href="mailto:anveepd@gmail.com">anveepd@gmail.com</a></td>
<td>View Profile</td>
</tr>
<tr>
<td>chao zhang</td>
<td><a href="mailto:chaoz@gmail.com">chaoz@gmail.com</a></td>
<td>View Profile</td>
</tr>
<tr>
<td>michael peterson</td>
<td><a href="mailto:michaelp@gmail.com">michaelp@gmail.com</a></td>
<td>View Profile</td>
</tr>
<tr>
<td>tianna bircher</td>
<td><a href="mailto:tiannab@gmail.com">tiannab@gmail.com</a></td>
<td>View Profile</td>
</tr>
</tbody>
</table>

- c/c++, linux, problem solving, python, software engineer, testing

### Fig 29. Result comparing the applicant’s skills

<table>
<thead>
<tr>
<th>SKILL</th>
<th>Student 1</th>
<th>Student 2</th>
<th>Student 3</th>
<th>Student 4</th>
<th>Student 5</th>
<th>Student 6</th>
<th>Student 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>c/c++</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>communication skills</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>consistent</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>customer</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>data structures</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>debugging</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>early</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>implementation</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>linux</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>low-latency</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>make</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>operating</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>performance</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>problem solving</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>programming</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>python</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>scripting</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>software engineer</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>software engineering</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>support</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>testing</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
</tbody>
</table>
The above table shows in detail the result of the skill finder algorithm with student 4 being the student with most number of required skills and student 7 the least. All the required skills which a student is having are colored in green and marked Y, a skill which is required for the job and the student is not having is marked N. Below is the job and applicants resumes used for testing.

Fig 30. A Sample of Skills required and Skills Possessed
Fig 30. Shows the skills required (center table) for a job stored in jobs_skills and skills possessed (tables with inbound arrows) by students stored in resume_skills table. A Cartesian product with an equijon on job_skills and resume_skills produces the result of students with required skills for a job. Students with most number of skills are list on the top.

Job Description:

**Business Card Title / Posting Title**
Senior Platform Software Engineer - FlashRay Product

**Job Summary:**
Join a small, rapidly growing, and dynamic project team to develop a new solid state storage architecture - FlashRay. The FlashRay product architecture is founded on rich scale-out and efficiency capabilities to maximize the benefits of All Flash Arrays (AFA). The new product line will combine consistent low-latency performance, high availability, and integrated data protection with enterprise storage efficiency features such as inline deduplication and compression.

FlashRay is one of the most strategic projects within the company. In this project, we will be combining cutting edge technologies with innovative data handling in a creative development environment that can best be described as a startup within a larger organization. This role is a truly once in a lifetime opportunity to deliver innovations and make a large impact.

For this position, we are looking for a software engineer who will be designing and implementing platform software & software upgrade infrastructure to support our next-generation flash based storage.

**Job Requirements**

Requirements

- Strong knowledge and experience in designing and implementing platform software & software upgrades.
- Excellent problem solving skills.
- Strong in Linux kernel fundamentals, system and kernel programming.
- Understanding of platform software, device drivers, Intel architecture, OS fundamentals and computer architecture is required.
- Thorough understanding of data structures and standard programming practices
- Solid C/C++ development and debugging skills in Linux operating environments
- Exposure to python or any other scripting language along with shell programming.
- Possess excellent written and verbal communication skills
- Should be a good team player.

Responsibilities

- Drive and deliver innovations for the next generation FlashRay product with focus on platform software & software upgrade infrastructure.
- Design, implement and debug platform software subsystems.
- Make changes to Linux kernel and user space drivers as and when required.
- Participate in all phases of a product development, from product definition and architecture and design, through implementation, debugging, testing and early customer support
- Work collaboratively with other engineers, and be willing to take on any engineering tasks that contribute to the progress towards the goals of the team and the company.

**Education and Experience:**
Fig 31: A sample Job Description for Test

Applicants:

The applicant resumes are in the same order as the result of a skill finder algorithm.

Student Resumes:

Student 4

WORK EXPERIENCE

Software Engineer – Firmware

Finisar - Sunnyvale, CA - July 2012 to Present

• Developed a HTTPS network to encrypt optical transceivers for clients with source control system (GIT, SVN):
  o Set up server and client-server request-response handling with XML-RPC, Apache, PHP and MySQL
  o Encrypted clients’ modules by AES algorithm
• Added more than 30 firmware regressions test cases programmed in PERL, Python and C++
• Debugged device drivers for the ARM processor inside the transceivers with I2C serial bus protocol using JTAG in C
• Automated experiment by controlling equipment (e.g. Oscilloscope) through GPIB, USB and Serial Ports in C++
• Led a team of 3 to deliver Product Verification Tests on optical transceivers through building DLLs in Visual C++
• Multi-threaded the testing procedures on different optical channels inside the module to ensure independency

• Constructed Object-Oriented API for the company’s IDE powered by IronPython Engine in C#.NET

• Voluntarily introduced and set up GIT under JIRA platform for the whole software group to replace SVN

• Improved GUI efficiency through better use of data structure on handling XML and JSON configuration files

• Built Python and Perl scripts for extracting, sorting and analyzing large amount of laboratory data

• Analyzed experiment test results and eliminated 40% of unnecessary test cases using statistical model

• Integrated Agile development framework with Scrum method to traditional Waterfall model in my team

• Developed web pages and Android app displaying CFP Multi-Source Agreement documents

Data Analyst Intern

comScore - Reston, VA - May 2012 to July 2012

• Captured online customers’ behaviors data on Telecommunication sector

• Generated SQL codes to extract and analyze data through various statistical models

Database Administrating Intern

Census and Statistics Department - Hong Kong, Hong Kong Island - June 2011 to August 2011

• Designed SQL programs for data analysis of 2011 Hong Kong Population Census

• Customized user interfaces with FoxPro to promote data entry efficiency by 50%

• Streamlined the existing quality assurance procedures to optimize the team productivity by 300%

• Maintained the computer system and provided technical support for over 80 staff

Data Analyst Intern

CIVIL AND ENVIRONMENTAL DEVELOPMENT DEPARTMENT - Hong Kong, Hong Kong Island - July 2010 to August 2010

• Wrote VBA programs in Excel for data analysis of correlation between rainfall and landslide
• Carried out spatial analyses of rainfall and landslides using GIS tool

EDUCATION

MS in System Engineering
University of Pennsylvania - Philadelphia, PA
2010 to 2012

BS in Electrical Engineering
University of Michigan - Ann Arbor, MI
2008 to 2010

SKILLS

• Language: C/C++, C#, Java, MySQL, PHP, PERL, Python, VBA, Bash, Shell Scripting, Assembly, Verilog
• IDE: Visual Studio, Eclipse, IAR Embedded Workbench, Cadence, LabView, Matlab
• Web Technology: Apache, HTML, XML, MapReduce, Hadoop, Internet protocol suite (HTTP, TCP/IP)
• Operation Systems: Mac
OS X, Unix, Linux, Android and MS Windows
• Source Control & Project Management: GIT, Subversion, JIRA, Agile, SCRUM

ADDITIONAL INFORMATION

• Fluent in English, Cantonese and Mandarin

• Proven to be dependable, motivated and possess excellent problem solving skills by employers’ recommendation

• Acquired strong leadership, communication and presentation skills from leading several students’ organizations
Student 1

Recent graduate of Florida State University in the Computer Science Program, who's academics credentials are reinforced by an internship with the Florida State University Computer Science Systems group. Strong grasp on OO programming, software engineering, debugging, testing, GUI interfaces, with emphasis in the C++ programming language. All skills currently being used and sharpened on start up work for an autonomous day trading program.

Ability to quickly learn new programming languages and technologies as shown by my technological summary list.

Experienced team leader with the ability to clearly communicating project goals and deadlines and then following through, shown by my extensive academic project list.

WORK EXPERIENCE

Chief Program Architect and C++ programmer

Applied Sciences and Technology LLC - September 2012 to Present

Fall 2012 to Present

• Grant Work for autonomous day trading program
• Chief Program Architect and C++ programmer

Systems Group Member

Florida State University Computer Science Department - September 2012 to November 2012

Intern, Help Desk and Windows and Linux systems diagnosis and repair

Cashier and food preparation

Dairy Queen - June 2007 to September 2007

Internship

Motorola Inc - 2006 to December 2006
December 2006

• Event Support LACCR MOTOCONNECTION

• Database coordination of reservations from countries of Latin America and Caribbean using MS Access

Summer Intern, Mobile Devices iDEN (Nextel)

Motorola Inc - May 2006 to August 2006

Data compilation using Brio Query, Microsoft Excel and Microsoft Access

• Help Desk Ticket system

EDUCATION

BS in Computer Science

Florida State University - Tallahassee, FL

2008 to 2012

Western High School - Davie, FL

2008

SKILLS

Technology Summary Programming/Languages: C++, Java, C, C#, Python, PHP, Java Script, makefiles, UML

Editors/IDEs: Notepad++, Putty, vim, Eclipse, Visual Studio Databases: MYSQL, SQLite, MS Access

Office


Other

Applications: SSH, Apache2, Samba, iptables

ADDITIONAL INFORMATION

Hardware Knowledge: Current HW Trends, Custom ROMs for routers, Atrix, Galaxy SIII

Student 2:

WORK EXPERIENCE
Teaching Assistant for Introduction to C++ Programming
Penn State - Harrisburg, PA - March 2011 to May 2011
Duties involve supervising over 40 students during C++ lab sessions and solve problems they encounter, grading homework assignments, quizzes and exams.

Teaching Assistant for Introduction to Programming Languages (C++)
Penn State - Harrisburg, PA - March 2010 to May 2010
Supervised over 40 students during C++ lab sessions and solved problems they encountered while programming. Graded papers and programming assignments.

Math/Computer Science Tutor
Learning Center Penn State - Harrisburg, PA - September 2009 to March 2010
Fall 2010, Spring 2011)
Worked with students in a one-to-one learning situation, in a drop-in clinic and in small groups.
Spent an average of 10 hours per week working as a Math, Computer Science and Statistics tutor.

Teaching Assistant for Computer Organization & Architecture
Penn State - Harrisburg, PA - September 2009 to November 2009
Graded all the homework assignments which involved theory questions as well as logic design questions using DIGLOG and assembly programming questions using MARIE.

EDUCATION
M.S. in Computer Science
Pennsylvania State University - Harrisburg, PA
January 2009 to January 2011
B.Tech. in Computer Science and Engineering
Guru Nanak Dev Engineering College - Ludhiana, Punjab
January 2004 to January 2008

SKILLS
1. C/C++, Java - Proficient in Object Oriented Concepts. Have written long codes in C/C++ and Java as
C#

3. Perl - Regular Expressions, DBI scripting and extracting information from Oracle database. CGI
scripting
and dynamic webpages. 4. Python, Javascript, HTML, XML - Web Development with Python using the
SimpleHTTPServer module. 5. Unix/Linux commands, Shell and Kornshell scripting - maintained a
Solaris
lab at school as a graduate assistant so quite conversant in most unix commands. 6. Matlab -
Manipulating
images and using the functionality of the Image Segmentation Toolbox for various algorithms and
techniques
of image segmentation.

ADDITIONAL INFORMATION

PROJECTS/RESEARCH

1. Submitted abstract for paper titled 'In vivo tissue characterization of coronary lipid plaques:
comparison of
Optical Coherence Tomography and Near-Infrared Spectroscopy' with Brajeshwar Maini, Larisa
Buyantseva
and Greg W. Stone in the Transcatheter Cardiovascular Therapeutics 2011. The paper studies the pre-
stenting
and post-stenting plaque presence in the proximal, mid and distal regions of LipiScanTM images. We
used
the fuzzy-c means algorithm with automated initial centroids using the bump-hunting method for
segmenting
the images into two classes, one with the higher intensity value representing plaque and the other with
smaller
intensity value representing the blood vessel. The program was written in Matlab and used various functions of the Image Segmentation Toolbox.

2. Writing a paper on mid-level approach for honeypots. Experiments performed with a novel fingerprinting mechanism for the POP server reveal a better emulation and a semi-automated fingerprinting of the service when used with a low-level honeypot. The fingerprinting program interacted with the remote POP3 server via tcpflow and developed a finite state machine based on the user inputs and server responses. This finite state machine was then used by the emulation program to mimic the behavior of the fingerprinted POP3 server with a low-interaction honeypot, namely honeyd. This semi-automated approach has paved way for further research towards full automation of fingerprinting-emulation mechanism of a service. The technologies used were C/C++, tcpflow, Python and Linux kernel subsystems.

3. Using Artificial Neural Networks to recognize handwritten digits. Implemented the Feed-Forward Backpropagation Network and Self Organizing Maps (or Kohonen Networks) and compared the results on two different data sets. MNIST dataset was used to test the Feed-Forward Backpropagation Network whereas, Semeion Handwritten Digit Dataset was used to test both Neural Networks.

4. 'Slap First', a distributed card game, implemented in a client/server architecture using C# and Microsoft .Net Remoting. The project involved the game being played on the server between clients (implemented with GUIs) connected to the server, the two communicating with each other through encrypted XML messages.
Technologies used C#, .Net Remoting, XML.

5. Parallelized an Ant Based Algorithm for finding Degree Constrained Minimum Spanning Trees over Nvidia's GPU's using C for CUDA.

Student 3:

Full time job of Software Development & Programming

WORK EXPERIENCE

C Programmer

Yalnix Operating System Project - January 2013 to Present

Implemented semaphores and monitor to realize process synchronization

• Used paging technique to reduce memory fragmentation

Object Oriented Programming Project - Peer2Peer Chat Application

Java Programmer

RICE UNIVERSITY - November 2012 to December 2012

11/2012 - 12/2012

• Designed public API with a 3 person team

• Used Java RMI proficiently, binding and looking up "stub" to implement peer connection

• Enhanced knowledge of design pattern to skillful level (observer, factory, strategy and visitor)

• Developing Environment: Java 7/Eclipse Juno

iOS Developer

Jailbreak Piggy Project - July 2011 to January 2012

Refined concept definition document for programmers, chose style of game, designed game flow diagrams and tracked project progress

• Amended game formula and data, revised probability distribution curve to improve balance

• Used cocos2D and box2D game frame, completed user-defined collision detection of rigid
body by realizing interfaces

• Developing Environment: XCode 4

4S Order Management System - January 2011 to March 2011

XINGZHI SOFTWARE Co., Ltd 01/2011 - 03/2011

• Conducted market research for existing user friendly interfaces

• Designed user interface through HTML, JSP technique, used MySQL database as back-end

• Tracked daily progress of group work, recorded and modified system bugs

• Used read-write lock to solve problems of synchronous data

SKILLS:

Linux Data Structure & Algorithm Design Pattern C/C++ Java Python

JavaScript HTML/CSS SQL

EDUCATION

Master in Computer Science

RICE UNIVERSITY - Houston, TX

2013

Bachelor in Computer Engineering

XI’AN UNIVERSITY OF SCIENCE AND TECHNOLOGY

June 2011

SKILLS

Linux, Design Patterns(Strategy, Factory, Visitor, etc.), C/C++, Python, Java, JavaScript, JSP, HTML/CSS, SQL

Student 5:

WORK EXPERIENCE

C++ Consultant/Intern

GEO Semiconductor - Orlando, FL - August 2012 to Present
Job Title: C++ Consultant/Intern

• Wrote low level SIMD algorithms using C++/assembly for Realta IC

• Designed regression suite for image library algorithms

• Implemented configuration management procedures (previously no CM)

• Determined optimal machine learning algorithms for pedestrian detection

• Implemented HOG w/SVM classifier for pedestrian detection

EDUCATION

Computer Engineering

University of Central Florida

December 2013

SKILLS

• Software Design - Project lead in phased incremental SDLC; image library programmer, tester, and configuration manager in XP SDLC; relational database management systems; OO design, threads, locks; embedded programming; machine learning, AdaBoost, cascade classifier • Hardware Design - Implementing

microelectronics in OP-Amps, embedded systems, computer/system architecture, memory management, computer communication networks • Languages - C/C++ (proficient), Java (prior experience), MIPS, IA-32, TI assembly (prior experience), Bash, MySQL • Image Processing - Edge detection, face detection/ recognition, pedestrian detection, HOG with SVM classifier, object tracking; SIMD algorithms

LINKS

http://www.github.com/jthames

ADDITIONAL INFORMATION

Technical Experience:
• **Software Design** - Project lead in phased incremental SDLC; image library programmer, tester, and configuration manager in XP SDLC; relational database management systems; OO design, threads, locks; embedded programming; machine learning, AdaBoost, cascade classifier

• **Hardware Design** - Implementing microelectronics in OP-Amps, embedded systems, computer/system architecture, memory management, computer communication networks

• **Languages** - C/C++ (proficient), Java (prior experience), MIPS, IA-32, TI assembly (prior experience), Bash, MySQL

• **Image Processing** - Edge detection, face detection/recognition, pedestrian detection, HOG with SVM classifier, object tracking; SIMD algorithms

**Personal Qualifications:**

• Proven ability to lead projects of variable scope and size

• Proven ability to take instruction and maintain a supportive role

• Proven ability to maintain system view and accomplish objectives

• Proven ability to gather info on, plan for, and implement complex algorithms

• Able to see vertically and horizontally to make informed decisions

**Tools:**

• Linux, Microsoft Visual Studio 2012, Eclipse, Code Composer Studio, Git, MultiSim, Xilinx FPGA ISE Design Tool, Diligent EXPORT

**Development and Testing Equipment:**

• Oscilloscopes, DMM, BASYS Development Board, Arbitrary/Function Generators, Soldering Iron, Breadboard
Student 6:

Obtain an internship position for a company which will help me to broaden my knowledge and skills as an Electronics Engineer.

WORK EXPERIENCE

Systems Engineer - Intern

Northrop Grumman Corporation - Clearfield, UT - June 2012 to Present

Systems Engineering, Integration, and Test

• Designed and implemented software to analyze atmospheric data for modeling purposes
• Conducted State-of-Technology survey for a project anticipating the design of a new avionics system
• Automated analysis processes using C, Fortran, Python, and VBA scripting languages
• Prepared PowerPoint presentations and briefed U.S. Air Force Customers
• Performed post-flight analysis and generated briefings and reports
• Obtained and currently maintain Department of Defense security clearance

EDUCATION

B.S. in Electronics Engineering

Weber State University - Ogden, UT

2014

A.S. in General Studies

Weber State University - Ogden, UT

2011

SKILLS

Programming (C, Python, Fortran, MATLAB, Mathematica, VHDL), Micro-Controllers (SL8051, PIC18, ARM2835), FPGAs (XILINX), Circuit Board Design and Construction (Eagle CAD), Operating Systems (Windows and Linux), Electrical troubleshooting and repair, Microsoft Office (Window, Excel, Access, PowerPoint, Project)
AWARDS

Research Grant
April 2013
I successfully wrote a research grant proposal which was funded by the Ralph Nye Charitable foundation for
the design and construction of a data logging compute for studying the lower stratosphere.

Phi Kappa Phi National Honor Society
November 2011
Nominated for the Phi Kappa Phi National Honor Society for academic excellence.

High Honors Scholarship
May 2008
Awarded the Weber State University High Honors Scholarship for academic excellence.

High Honors Sophomore Scholarship
September 2010
Awarded the Weber State High Honors Scholarship for academic excellence.

Kennecott Scholarship
September 2010
Awarded the Kennecott Scholarship.

Keith Wilcox Engineering Scholarship
December 2010
Awarded the Keith Wilcox Engineering Scholarship.

August Ahlf Scholarship
September 2010
Awarded the August Ahlf Scholarship.

College of Applied Science Scholarship
September 2011
Awarded the Weber State University College of Applied Science Scholarship.

College of Applied Science Scholarship

September 2012

Awarded the Weber State University College of Applied Science Scholarship.

Micron Scholarship

September 2012

Awarded the Micron Scholarship for engineering.

Abrelia S Hinckley Scholarship

September 2013

Awarded the Abrelia S Hinckley Scholarship for engineering.

CERTIFICATIONS

Amateur Radio License

July 2013 to Present

Hold a General level Amateur Radio License; Call Sign: KG7EJB

GROUPS

American Institute of Aeronautics and Astronautics

November 2011 to Present

Elected Secretary of the Weber State University Chapter of the AIAA.

Institute of Electrical and Electronics Engineers

November 2011 to Present

National Member of IEEE.

High Altitude Reconnaissance Balloon for Outreach and Research

September 2011 to Present

MSA Team Leader - Weber State Flight Team

* Design and implement electronic systems for data acquisition in the upper atmosphere.

* Develop software for embedded microprocessors using C programming language
* Design and build PCBs using Xilinx and pSpice simulation software

* Conduct system calibration and testing to ensure components are adequately hardened for operations in near space environments

* Wrote a successful grant proposal and was awarded an Undergraduate Research Grant funded by the Ralph Nye Charitable Foundation

PUBLICATIONS

A multi-sensor array to study flight dynamics, atmospheric pollution and gas composition in Earth's atmosphere.


March 25, 2013

Wrote an abstract and presented a poster at both the Ninth Annual Undergraduate Research Symposium and at the Nineteenth Annual Fellowship Symposium (Utah Space Grant Consortium - 6 May 2013) for the design and construction phase of an atmospheric data logging computer.

ADDITIONAL INFORMATION

Obtained in 2012 and Currently maintain Department of Defense Security Clearances.
Student 7:

WORK EXPERIENCE

Teaching Assistant

Western Washington University - Bellingham, WA - April 2013 to June 2013

Assisted with the labs for two C++ programming classes by tutoring the students and grading assignments.

Theatre shop assistant

Western Washington University - Bellingham, WA - March 2010 to August 2011

Built, painted, and installed the sets for WWU's theatre productions.

EDUCATION

Bachelor of Science in Science and Technology

Western Washington University - Bellingham, WA

June 2013

SKILLS

Technical Skills Languages: Proficient in C++, Java, Python Experience with C, Scheme, PHP, Perl, SQL,

ADA, XML OS: Windows (2000, XP, Vista, 7,8), Linux Software: Emacs, Eclipse, AdaGide, DrRacket, IDLE,

SQLite, Codeblocks, SSH, CVS Relevant Courses • Data Structures • Analysis of Algorithms I and II • Object Oriented Programming • Principles of Concurrent Programming • Web Scripting • Mobile Device Programming

(Andorid) • Database Systems • Linear Optimization

ADDITIONAL INFORMATION

Extracurricular Activities

• Member of the National Society of Collegiate Scholars.
• Captain and first board of Kamiak High School Chess Club for two years.

• Volunteered as a summer school teacher's aide at Cedar Valley Elementary School for eight summers in a row.

Student 8:

A web developer seeking to apply his experience and skills in JavaScript, PHP, MySQL, HTML, and CSS, to enhance the bottom line of a progressive company that appreciates hard work, integrity, creativity and intelligence.

WORK EXPERIENCE

Programmer

BYU Print and Mail Center - Provo, UT - 2011 to 2012

Worked with a team to develop, improve and maintain various web applications. Primarily worked with PHP on the back-end of the applications such as storefronts and worked with our custom PHP library. Later helped convert much of the application into Yii, a PHP framework and did some minor work with C++ creating Windows applications. Overall I felt I contributed much and enjoyed my experience working there.

Web Developer

Johanna's Kitchen - Sandy, UT - 2011 to 2011

Developed websites for James Marshall, the owner of Johanna's Kitchen, including the restaurant's official web site and an American trails website. Spent much of the time researching the various trails and awards in addition to creating and organizing the many static html pages.

Various other projects as an independent web developer and programmer at Freelancer.com, consistently receiving highest ratings (http://www.freelancer.com/users/860961.html)
EDUCATION

Canyons Technical Education Center - Sandy, UT

2010 to 2011

Riverton High School

2007 to 2011

SKILLS

Web Developer (PHP, MySQL, HTML, CSS, JavaScript, jQuery, etc.)

ADDITIONAL INFORMATION

QUALIFICATIONS:

• CIW Web Design Specialist

• CIW Associate Design Specialist

• CIW Site Development Associate

• Knowledge and experience in JavaScript and the jQuery Framework

• Proficient in PHP coding and utilizing MySQL and Oracle Databases

• Familiar with Java

• Familiar with Database Design Principles

• Pick up new languages and frameworks quickly and always growing skill set

• Proficient with both HTML and CSS and Website Creation

• CTEC Outstanding Webmaster of the 3rd Quarter Award

• CTEC Outstanding Webmaster of the Year Award

• USOE CTE certificate - Business Web Page Design, Web Development A and B
CHAPTER 6

CONCLUSION

The Skill Finder tool works efficiently in matching the student resumes to the jobs posted. It allows to maintain a complete history of job requirements from external employers and department. It hosts student personal information, academic history and student resumes. It successfully sends email alerts to students on a job posting. This system is scalable and flexible to extend further for adding new functionalities.
REFERENCES

[4]. Taming Text by GRANT S. INGERSOLL, THOMAS S. MORTON, ANDREW L. FARRIS
[12]. http://searchcio-midmarket.techtarget.com/sDefinition/0,,sid183_gci836031,00.html