P1 – Digital CV

Software, Aalborg Universitet København

David Doctor Heyde Rasmussen, Hans Erik Heje,
Mikkel Kaa & Ming Hui Sun
Group 5

December 18, 2020

STUDENT REPORT
AALBORG UNIVERSITY

AALBORG UNIVERSITET
STUDENTERRAPPORT

Software
Aalborg Universitet
http://www.aau.dk

Title:
P1-Project

Theme:
Digital CV

Project Period:
Efterårssemestret 2020

Project Group:
SW 1 Gruppe 005 Klynge 1

Participant(s):
David Doctor Heyde Rasmussen
Hans Erik Heje
Mikkel Kaa
Ming Hui Sun

Supervisor(s):
Johannes Bjerva

Page Numbers:
54 pages

Date of Completion:
18. december 2020

Abstract:

This project is going to focus on the theme "Digital CV". The process of applying for a job can be time-consuming and strenuous. Especially for the unemployed, their potential is wasted by a seemingly unfair process. The project objective is therefore to find a software solution that can produce a Digital CV that will improve the US and UK unemployed chances of producing a large quantity of CVs and getting past the ATS scanner and hiring personnel. During the project an experiment was conducted, to evaluate if the Digital CV passed the hiring personnel. The results of the experiment were that the Digital CV had moderate success at passing the hiring personnel. However, due to a lack of experimentation and results from the other aspects of the objective, the project´s objective was not fully met.

Preface

This rapport is apart of the first semester project worth 10 ects points for the bachelor of software engineering. In connection with the P1 project, and it is about a process from finding a problem to finding a solution based on software. The report is aimed at the supervisor and the internal censor and for use of inspiration with the same topic. The report has been prepared from 08/10/2020 - 18/12/2020 with guidance from Johannes Bjerva and has been made in 2 months and 10 days.

David Doctor Heyde Rasmussen
<ddhr20@student.aau.dk>

Hans Erik Heje
<hheje20@student.aau.dk>

Mikkel Kaa
<mkaa20@student.aau.dk>

Ming Hui Sun
<msun20@student.aau.dk>

Contents

1. Introduction

Every country around the world rely on its citizens being productive, this usually translates to a job. It is therefore costly when unemployment happens and when those that might have much to contribute, never get the chance. This is partly due to the difficulty of both applying and obtaining a job, specifically for the unemployed. To ratify this, the overarching objective of this project is to aid The United States and The United Kingdom unemployed in obtaining a job using a CV.

To fulfill this objective, the process of obtaining a job, along with the content and structure of the CV will need to be discussed. Furthermore, the unique aspects of applying as unemployed will need to be outlined so to better aid them in obtaining a job. Using what has been discussed, it will then be possible to outline many of the crucial factors that determine the chances of the unemployed obtaining a job.

Analyzing these factors, the most relevant factors to the unemployed will be selected. It will then be theorized how the selected factors could be optimized to increase the chances of the unemployed obtaining a job.
Utilizing the theoretical optimizations, it will be attempted to implement these optimizations into a product. The creation of the product will be documented and any decisions taken examined and discussed. The product will be tested in practice and the results from the test document will be analyzed. The effectiveness of the product will then be evaluated, to answer if the project's following problem statement was solved: "How can we make a software solution, that enables the British and American unemployed people, to create a large number of CV applications. Such that they can get a higher chance of getting past the ATS scanner and hiring personal, than if they were to individually create each application while not expending more effort?" Thereafter, any further development of the product including ethical concerns will be examined, discussed, and reflected upon.

2. Background

2.1. What is Unemployment and how does it Materialize

Every country is built on its population producing something of value. In a utopia, all citizens of each country would produce value enough to cover themselves and a little more for the structures of their country. However, in reality, the value of each citizen's production value fluctuates. Some citizens are hyper-active and produce immense value for their country. While in every corner of the world, members who produce little to nothing are also found. In the modern economic infrastructure, the citizens known as the "unemployed" make up a portion of those without value. However, they have the potential to produce and support their respective countries. But what exactly is unemployment and how does it materialize?

Unemployment occurs when a person who is available and actively searching for employment is unable to find work. [1] At first glance, it is difficult to tell what types of people unemployment encompasses. Although we can see that unemployment requires two factors, for the person to be available for work and be actively searching for work. Firstly, being available means not to be preoccupied part-time or full-time with another occupation. Therefore unemployment does not for example include children, retired, full-time students, part-time workers, disabled, or those on maternity leave. Secondly, to be actively searching entails that the person has actively looked for work in the prior four weeks. [2] So, those individuals who are jobless but not actively searching for work are not unemployed. As such, unemployment has been defined, but how does it materialize?

Unemployment is found in every economy and society, but how does it materialize? Unemployment could materialize for many different reasons, to understand these reasons it is useful to divide unemployment into four different types. Thereafter discuss what each type encompasses and how the unemployed could be affected.

1. Four types of unemployment:
2. Structural unemployment
3. Cyclical unemployment
4. Frictional unemployment
5. Seasonal unemployment[3]

Structural unemployment occurs if there is a mismatch between offered and demanded skills. This could be a lack of demand for workers of a certain skill set, or an excess supply of a job with a lack of workers with the matching skill set. The unemployed, it often required to learn new skill sets or further educate themselves to gain a job. Cyclical unemployment arises if there is a downturn in the economy and no jobs are available. This is the biggest cause of unemployment and can have significant consequences on unemployment globally. [4] It can develop when there is a reduction in the demand for a firm's products or services and the firm, therefore, has no need for high production, cutting back on their workforce. Frictional unemployment refers to workers who are in transition between jobs. It is not entirely a bad thing as often it is caused by a worker finding a job more suitable for their skills. This also involves those workers who recently left or were fired and are actively searching for a job. Seasonal unemployment occurs when the demand for workers varies throughout the year. This type of unemployment often refers to climate dependant economic sectors, such as agriculture or tourism. It is fairly predictable in most cases and often requires workers to find another occupation for the rest of the year.

As closure, four potential ways for unemployment to materialize have been discussed. Unemployment can potentially materialize from almost any economical, psychological, cultural, seasonal, or institutional reason. A more economical method would conclude that unemployment could materialize "from both the demand side, or employer, and the supply side, or the worker." [5] Now, it has been discussed how unemployment can materialize through the four types of unemployment. [1]

2.1.1. The Effects of Unemployment on Country and Individual

Unemployment is not a consistent state, as discussed in 2.1, unemployment can materialize due to a variety of conditions. However, the discussion of what unemployment is and how it materializes leaves out the individuals and country affected. Consequently, it is meaningful to further discuss unemployment and how it affects both country and the individual. However, to make any conclusions on the effects of unemployment, it must refer to statistical measurements of unemployment, country, and individual. Therefore, the methods used to measure unemployment, country, and an individual will also be discussed. So, how does unemployment affect both country and individual?

To gain an understanding of unemployment´s effect on both country and individual it is vital to first define what the country and individual encompasses. As both country and the individual are loose and imprecise terms it is helpful to narrow the discussion down to a couple of countries and a group of relevant individuals. Unemployment differs wildly from country to country, so to choose two countries similar enough to make definitive findings on unemployment is productive. Furthermore, it is valuable to choose countries that are relatively stable and reliable in their unemployment statistics so research and findings are constructive. The two countries that are optimal as candidates are The United States of America and the United Kingdom. The US and UK are both relatively stable and have reliable statistics within comparably similar socio-economic structured countries. [6] Additionally, "effected" is quite a sweeping term, so the discussion can be furthermore narrowed down to how unemployment affects the economics of the US and the UK. Similarly to discussing all countries, discussing the effect on all individuals in the USA and UK would be too general and speculative. Therefore, it will be most productive to discuss those most directly affected by unemployment, the unemployed. As unemployment can have almost any imaginable effect on the unemployed, it is helpful to focus exclusively on how unemployment directly affects the unemployed´s chances of obtaining a job. Thus, the discussion will be narrowed down to; how does unemployment affects the US and UK economies and their unemployed prospects of obtaining a job?

Firstly, how does unemployment affect the US and UK economies? The first obstacle to answering the question posed is to find how to best represent unemployment and the economies of the two countries as a statistic. For unemployment, the clear best statistical representation is the percentage of unemployed, represented by the unemployment rate of a country. How to represent the economy of a country is more difficult, economists generally use GDP and inflation as an overall measurement of an economy. Although other statistics are also used to represent the economy of a country, GDP and inflation are standard and are relatively reliable statistics. [7] Inflation has no real direct statistic representing represented for a whole country, therefore wage inflation will be used as the most representative statistic. Similarly, to measure GDP over time, the GDP growth rate will be used as the best representation of GDP. It is now possible to compare the unemployment rate with the US and UK economies, using wage inflation and GDP growth rate as a representation of the economy. Depending on the correlation between the unemployment rate and the GDP growth rate and wage inflation in a country, it is possible to see the effects of unemployment on the US and UK economies. Discussing the unemployment rate and growth rate of the GDP of the US than in the UK. By inserting the unemployment rate and growth rate into the same graph a correlation might be seen:

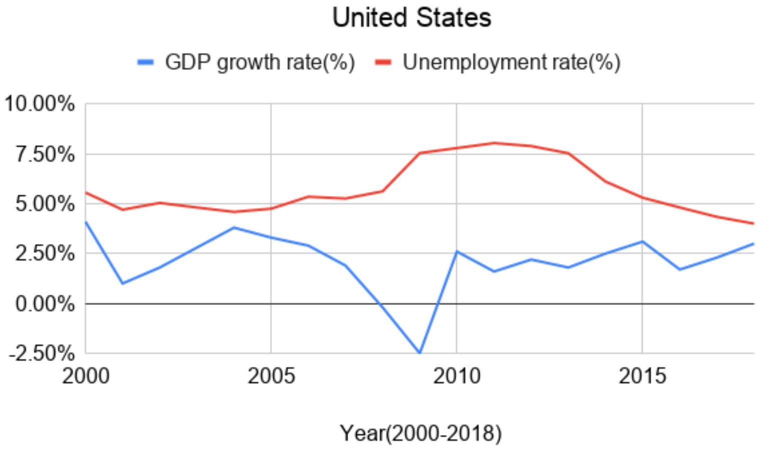


Figure 1.: Unemployment and Growth rate[8][9]

Here we see a somewhat clear correlation between the growth rate of the GDP growth rate and the unemployment rate of the US economy. As seen, there is an inverse correlation between the two, meaning that when one decreases the other increases and vice versa. If we look at the effect of the unemployment rate on the GDP growth rate in the UK it should mirror this conclusion.

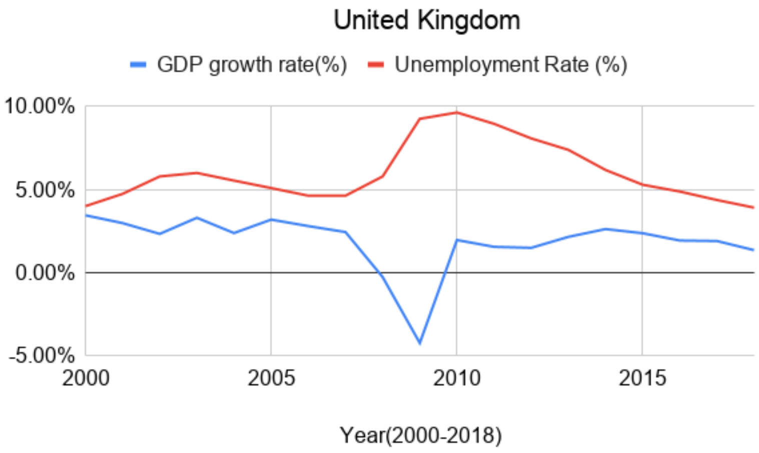


Figure 2.: Unemployment and Growth rate[10][11]

As shown, a clear inverse correlation between the growth rate of the GDP and unemployment rate can again be concluded in the UK. This was expected and supported by the economic theory; Okun´s law. Okun´s law states that the unemployment rate and GDP of a country have an inverse correlation. Therefore meaning that the unemployment rate would also have an inverse effect on the growth rate of the GDP. [12] This correlation goes both ways, meaning that increasing the GDP growth rate of a country would decrease unemployment, and increasing the unemployment rate would decrease the GDP growth rate.

Now examining the correlation between the unemployment rate and wage inflation, we attempt to see if a correlation exists between the unemployment rate and wage inflation in an economy. According to Keynesian Macroeconomic theory, there is a direct correlation: The general economic trend is that when unemployment is high the supply of labor exceeds the number of jobs available. So, when unemployment is high; more workers are available than jobs to fill. Therefore, employers have little incentive to raise wages to attract workers main stagnant or decrease, and wage inflation will not occur. However, when unemployment is low the demand for labor exceeds the number of jobs available. Therefore, employers now have a strong incentive to pay higher wages to attract workers, and wages will increase, and wage inflation will occur.[13] This is all further supported by the Keynesian economic theory, the Philips curve. The Philips curve supports the inverse correlation between the unemployment rate and inflation rate. It states that "A Philips curve illustrates a tradeoff between the unemployment rate and the inflation rate; if one is higher, the other must be lower." [14] Therefore, a conclusion that being made that an inverse correlation exists between unemployment and inflation.

Secondly, how does unemployment affect the US and UK unemployed prospects of obtaining a job? To better discuss the effect of unemployment let us divide unemployment

into two-time frames, short-term and long-term unemployment. Short-term unemployment is defined as any unemployment period from 0 to 27 weeks, while long-term unemployment defined is as unemployment 27 weeks or more. [15] If a can be found between the prospects of obtaining a job and unemployment, a conclusion can then be made on the effect of unemployment on the unemployed prospects of obtaining a job.
Previously, the unemployment rate was used to conclude a correlation between unemployment and the economy of the US and the UK. However, it is problematic to make a concrete correlation between the unemployment rate and the prospects of obtaining a job for the unemployed. This is because, as already discussed, there exists a clear correlation between the unemployment rate and the economy of a country. Therefore, it is difficult to isolate the unemployment rate as an independent factor in an experiment, as the unemployment rate is dependent on the economy of a country and vice versa.

A correlation between the unemployment rate and the US and UK unemployed prospects of obtaining a job would consequently be meaningless. As such, any conclusions made on the effect of unemployment would have to come from the status of being unemployed, not the unemployment rate. So, is there a correlation between being unemployed and the prospects of obtaining a job in the US and UK?
Studies show a clear drop in the prospects of obtaining a job for the unemployed the longer they are unemployed. This can be seen in the following figure, where the Federal Reserve Bank of New York conducted a study comparing the probability of finding a job and the unemployment duration in months. The blue line representing when controlling for characteristics such as age, education, ethnicity, and gender bias, while the red represents when not taking into account characteristics. Simply put, the study tries to isolate unemployment as a factor, making the blue line or the full controls group more reliable:

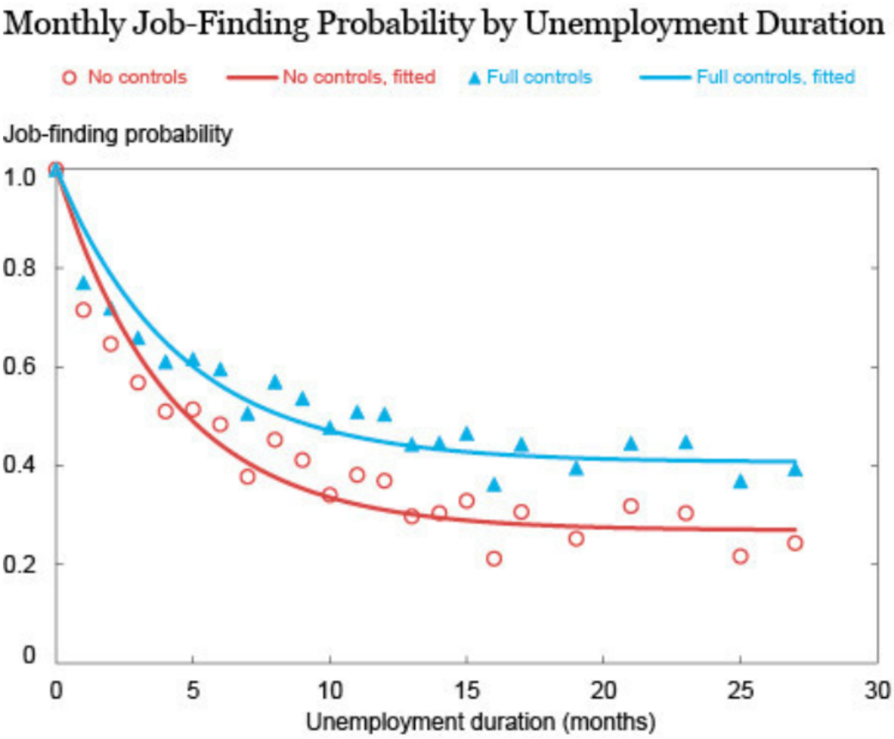


Figure 3.: Unemployment and Growth rate [16]

From the figure, it can be seen that in the first eight months of being unemployed the job-finding rate falls by roughly 50% for the controlled group and even lower for the group with no controls. From this, we can conclude that a correlation between being unemployed and the prospects of obtaining a job exists and this correlation is affected by time. We can similarly conclude that short-term unemployed have a rapidly decreasing probability of obtaining a job, while long-term unemployed have a stable but lowest probability of obtaining a job. However, it is productive to ask why this sharp decline in obtaining a job for the unemployed exists.

While many reasons could exist, a major factor is hiring personnel discrimination against the unemployed. A study where researchers sent out the same CV only varying the length of time being unemployed found that "long-term unemployed workers can be up to 45 percent less likely to receive interview invitations than newly unemployed or currently employed people who look just like them".[16] This indicates that hiring personnel discrimination becomes a major factor the longer the duration of unemployment. However, this discrimination is not always unwarranted.

A 2018 report by the National Bureau of Economic Research on discrimination of unemployed about time unemployed found that hiring personnel do not discriminate due to the status of being unemployed:[17] "We show that such instances are rare when firms discriminate in anticipation of an ultimately unsuccessful application. Discrimination in callbacks is thus largely a response to dynamic selection,". Dynamic selection referring to a finding of the study that "low ability workers are more likely to be long term unemployed and duration contains information about a worker's ability". Therefore, hiring personnel discrimination is based on the likelihood of unemployed being less qualified, meaning that the unemployment status is not what sanctions discrimination, it is a tendency for unemployed to be less qualified. In conclusion, unemployment has a definitive negative effect on the prospects of the US and UK unemployed prospects of obtaining a job. These negative effects include that the probability of obtaining a job rapidly decreases for the short-term unemployed and is consistently low for the long-term unemployed. A negative effect that is hard to deal with is the warranted discrimination of hiring personnel against all unemployed, intensifying with the duration of unemployment.

In overall conclusion, the original narrowed question posed was: How does unemployment affect the US and UK economies and their unemployed prospects of obtaining a job? It is possible to now answer that in terms of the unemployment effect on the US and Uk economy, unemployment has an inverse effect on the economies. This indicates that it is in the best interest of every country to decrease the number of unemployed as this would increase economic productivity and that a. Thereafter, it can be answered that unemployment decreases the unemployed probability of obtaining a job. This decrease happens rapidly roughly the first six months and it is in the best interests of any country to urgently find employment for the unemployed.

2.2. Obtaining a Job and The Process of Applying For a Job

As discussed, it can become increasingly more difficult for the unemployed to get a job as time goes on. This overlooks the discussion of the actual process of obtaining a job both for the unemployed and any other individual. To discuss the process of getting a job, it is productive to first discuss the method by which jobs are obtained. Thereafter, the process of applying for a job and the time frame of the process of obtaining a job. So, how does one obtain a job and what does the process of obtaining a job look like?

Firstly, how do people obtain jobs? Lou Adler, CEO of "Performance-based hiring learning systems", tried answering just this; he conducted an online survey on LinkedIn and asked how people obtained their current job. He divided people based on job status before obtaining their current job, dividing people as variations of active or passive job searchers. Active job searchers, as discussed in the 2.1, are people that have applied for a job in the prior four weeks. He further divided active job searchers into previously unemployed and employed active searchers. Passive job searchers are therefore anybody who has not applied to a job in the prior four weeks. He further divided passive job searchers into tiptoers, those who passively search for other jobs without actively searching, and employed who were passively offered a job. The results of the survey are as followed:

The results are outlined in fig. 4.

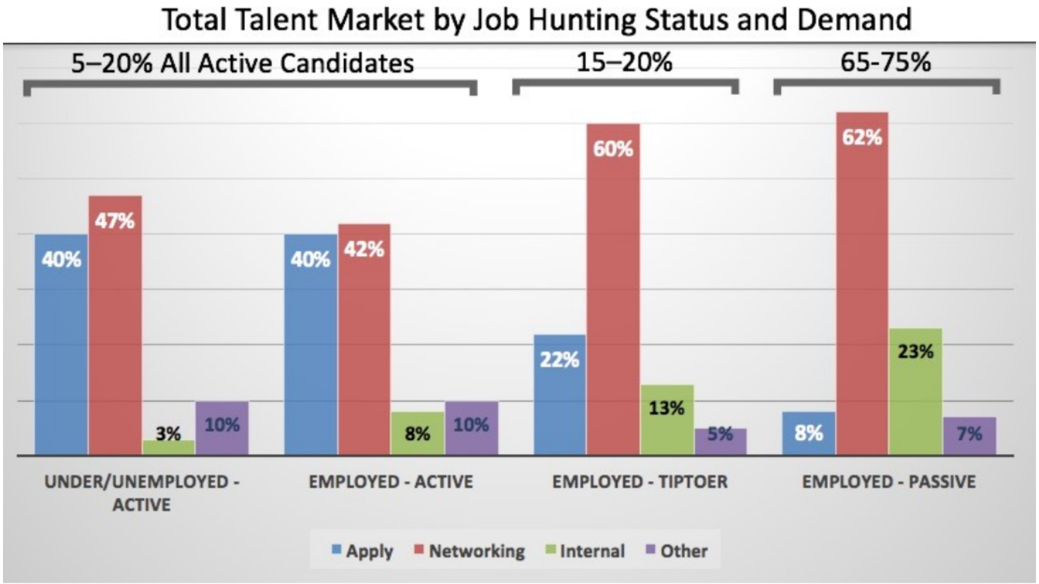


Figure 4.: How people get jobs 2015 and 2016 [18]

Here we see four categories that people could be hired from.

The two most relevant categories are; "Apply" which would most likely be an application through CV and "Networking" which would be hiring through acquaintances or media. From the result, we can discern that networking is the most likely factor in obtaining a job, followed by applying in most categories. However, we see a sizable difference between the effectiveness of applying between active and passive job searchers. It is clear that applying has less importance for passive job searchers while applying has nearly the same effectivity as networking for active job searchers. This means that an effective CV process could be as beneficial for active job searchers as good networking. But what does the process of applying to a job look like for active searchers?

The process of obtaining a job has five main stages. These five stages are most relevant to those who apply to a firm through CV, the most likely process of application. Although every application process is different, most firms/organizations contain these five stages. The five stages are usually chronological and all have an obstacle that if passed lead to the next stage, ending with a job offer:

1. Applying through CV to a job position.
2. ATS scanner scans your CV [19].
3. Hiring personnel view your CV.
4. Job interview with firm/organization.

5. Job offer.[20]

Each stage merits a discussion of the process it presents. The first stage "Applying through CV to job position" is when an applicant sends a CV to a firm/organization. Thereafter the CV is scrutinized by the second and third stages and is either rejected or accepted to further stages. The second stage is the ATS(Applicant Tracking System) scanner and it scans an applicant's CV for relevant keywords based on the job position. The third stage is the hiring personnel who will review the applicant's CV and decide which are qualified for a job interview. The fourth stage is when the applicant is invited to a job interview, possibly more than one interview, the job/organization decides if the applicant is compatible. In the fifth and final stage, if all previous stages are passed, the applicant is given a job offer.

But, how long can it take to get a job and how many CV´s does it take? It is surprisingly difficult to obtain a job by applying. A survey done by TalentWorks found that on average, when you send out an application, there is an 8.3 percent probability, that you will be invited to a job interview. Furthermore, on average, it takes around 10-15 interviews, before one gets a job offer. However, this greatly varies depending on the person, country, and economy. However, we can conclude that on average it will take us roughly 150 applications before an applicant gets a minimum of one job offer.[21]

This is a large amount of CV´s and can take a large amount of time to write a CV, apply to jobs, and go to job interviews. Another factor is the time that hiring personnel takes before responding to a CV application. For most jobs, it takes around three days before the applicant gets a response, however, for a less demanded career it can take anywhere from 10 to 30 days.[21] On average across all jobs, one can expect to hear back from employers within a week 41 percent of the time and within a couple of weeks 85% of the time. This means that the job process can take much longer than anticipated in most cases.

A study conducted by recruitment agency Randstad US surveyed 2000 Americans and found that on average it took five months to obtain a job.[22] Therefore for the average applicant, the process of applying for a job can take a lot of effort and time. This study was not exclusively conducted on the unemployed but a mix of active searchers. The time it takes to write a CV is another aspect. The time can vary, based on the potential quality of the CV and applicant writing.

Sources also report contrasting time, time spent completing a CV varying from a couple of hours to several weeks. While empirical evidence is sparse, taking a rough estimate can more accurately represent the time it takes to get a job. An average of eight hours to write a single good CV can be found, this adds additional time for any applicant applying for a job. Many applicants only write one CV and use it to apply to all jobs, there CV writing process taking eight hours.[23] However, those that choose to optimize their CV for each job they apply to will have to spend even more time and effort.

In conclusion, the process of obtaining a job can be strenuous and time-consuming.[24] As discussed it can take even longer for the unemployed to obtain a job, especially the long-term unemployed. Therefore while five months is a better lower bounds estimate, it can easily take more than five months for the unemployed.

2.2.1. Optimize Interview Rate

There are many factors to consider, if one wishes to optimize ones chances of getting an interview, one of the more empirical proven ways is considering what time and day one sends their application. To get the highest chances, you have to apply between early Tuesday morning and Thursday before noon using the employers local time. Monday is even better, increasing your chances by 46 percent in regard to the average. If one should apply on another day, the most important factor is that it's done before 10AM, since the interview chances drops below 5 percent for the majority of late evening applications.[25] Perhaps one of the most influential condition of wether you get an interview, is how fast you are at applying: Based on 30000 data points from the company Speedrecruiters, you need to apply within the first 14 days to have a practical chance of getting an interview. it is such that 50 percent of the people who got an interview for the job applied within the first week, and 75 percent of those who got an interview applied within the first 14 days, whilst the chances of getting an interview thereafter dwindles exponentially.[26] Other factors that one can optimize in a CV that influence the interview rate are as following:

1. Being older, but no older than 35, increases the interview rate by 25 percent.
2. Having more than one degree, increases ones chances by 22 percent.
3. Adding industry buzzwords increases your chances by 29 percent. e.g. If you are a software developer, then add buzzwords such as machine learning, artificial intelligence etc.
4. Demonstrating earlier job results using numbers increases chances by 40 percent. e.g. "Increased profits by 20 percent from Q3 to Q4"
5. Listing achievements, where you weren't in charge, but only a helping hand decreases your chances by 50 percent e.g. "Helped management organize financial reports" instead of "Organized financial reports"
6. Using leadership affiliated buzzwords increases your chances by 51 percent
7. Not using personal pronouns in the employment section increases your chances by 55 percent.
8. Including a key skills section and buzzwords of the key skills increases your chances by 59 percent

9. Start ones sentences with distinct action verbs, increases ones chances by 140 percent. e.g. Do "Developed a mainframe architecture that dramatically increased efficiency" instead of "After surveying people, the mainframe architecture that increases efficiency was developed by me."[27]

2.2.2. ATS Scanner and Hiring Personnel

As discussed the process of applying for a job has five stages. Understanding the five stages in the process of applying to a job is crucial for those applying, as The second and third stage, the ATS scanner and hiring personnel, both treat the first stage, that of applying with your CV. This makes the making of the CV the most significant step in obtaining a job. Furthermore, most applications are rejected by the ATS scanner and hiring personnel, making it the most crucial stages to pass. Therefore, discussing the ATS scanner and hiring personnel working in more detail can give an applicant an idea of how to pass them. So, how does the ATS scanner and hiring personnel screen an applicants CV?

Firstly, the ATS scanner is a keyword scanner that helps to hire personnel screen applicants and narrow the potential candidate pool. The ATS scanner is widely used and "70%-80% of large companies worldwide utilize an ATS, while smaller businesses utilize it about 50% of the time".[28] Furthermore, according to Columbia University, 75 percent of applicants are phased out because their CV does not pass the ATS Scanner test. This is a staggering amount and therefore optimizing one's CV to pass the ATS scanner is vital to obtaining a job. In Denmark, most of the biggest companies are using ATS scanners to lessen consumed time.[29]
So, how does the ATS scanner screen an applicant's CV, and how does an applicant optimize their CV for the ATS scanner? For an applicant to optimize their CV for the ATS scanner, the way the ATS scans an applicant's CV must be discussed. Many versions of ATS scanners also exist, with the most widely used being the iCIMS, Bullhorn, Taleo, and Greenhouse. While no two are the same, they all scan for the same underlying elements in an applicants CV:

1. Scan education
2. Scan work experience
3. Scan relevant keywords throughout the CV

However, how does the ATS scanner scan for these elements in a CV? All ATS scanners have slightly different methods, so, therefore, it is productive to use Taleo, the most widespread ATS scanner as a template example. The Taleo ATS scanner works by having pre-defined keywords chosen by the firm/organization, that it then searches for in the applicants CV´s.[30] The most constructive way to discuss the working of Taleo's ATS scanner is to have an example scenario.

A scenario could be a shipping company, that is searching for a candidate with a bachelor's, that also has work experience with programming and has leader characteristics, they will have "bachelor", "programmer" and "leader" as pre-defined keywords. Taleo will then search through every applicant's CV´s for the keywords "bachelor", "software" "leader".[31] Often Taleo will categorize a CV based on the headers established by the applicant in the CV. The shipping company will most likely give instructions to Taleo to only search for the keywords "bachelor" under the header "education". Therefore eliminating any random occurrence of the word out of the context of education.

A firm/organization can also set the importance of certain pre-defined keywords through Taleo. Some pre-defined keywords may be awarded a system of points for how often they appear in an applicant's CV, or some words may be required by Taleo. Meaning that if certain pre-defined keywords do not appear in the applicant's CV, then the CV is rejected.

Taleo along with all modern ATS scanners can recognize synonyms and therefore a small deviation from the pre-defined keywords may not be fully penalized. If the shipping company´s ATS scanner scans a CV and finds the word "director", it may award the points equal to or a fraction of "leader". If an applicant's CV contains all the required keywords, then the points are added and ranked among the other CV´s. Hiring personnel tends to consider those CV´s with the highest ranking, so it is preferable to rank as highly as possible.

The ATS scanner can seem as a nuisance for applicants, as the firm/organization selects the predefined keywords and the applicant has to guess which keywords to include in their CV. Furthermore, a study by career arc found that " 62% majority of employers who use applicant screening tools admit that some qualified candidates are likely being automatically filtered out of the vetting process by mistake".[32] This points to the ATS scanner is potentially a flawed system, screening qualified applicants and allowing less-qualified applicants through. As discussed, most applicants only write one CV, partly because it can be extremely time-consuming if an applicant wishes to create a CV optimized for the ATS scanner for every job they apply to. However, the ATS scanner's purpose is to screen those that are not qualified, in this it is predictable and can be systematically be overcome. A CV that contains the right keywords and structure should be both in the interest of the firm/organization and applicant, to pass and rank highly on the ATS scanners list.

Secondly, as discussed, after all, CV´s have been screened by the ATS scanner, the hiring personnel views that CV´s not rejected by the ATS scanner. The method by which they scrutinize the applicant's CV´s varies, much like the ATS scanner. Understanding the method by which the hiring personnel view applicant's CV´s can aid applicants in passing the hiring personnel stage, thereafter being invited to a job interview. Unlike the ATS scanner, which is predictable in its systematic scanning. Hiring personnel inserts a human element and therefore an unpredictability. However, some general trends can still be made about how hiring personnel screen applicants CV.

According to the majority of studies, hiring personnel spend 5 to 9 seconds (depending

on the study) considering an applicant's CV, if the CV does not appeal in this small time frame, the CV will be rejected. This means that the first impression is vital to the success of passing the hiring personnel and warranting a job interview. James Reed is the author of the book "The 7 Second CV" and has written three other books on the job interview process.[33] According to James Reed, most hiring personnel will only spend 7 seconds on each CV, and that a CV must be able to stand out in those first 7 seconds. [34]
James Reed book puts much emphasis on the following concepts:

1. Accessibility to content
2. Content relevance
3. Wording of content

[33] It is vital to discuss each concept and what does it entail. Accessibility to content highlights the need for a good first impression. The structure and content of a CV must invite further study, it must catch the hiring personnel's interest. The CV must also be of a length in which the hiring personnel does not reject the CV based on assumptions. If the CV is too short the hiring personnel may regard the applicant as under-qualified. Furthermore, if the CV is too long the hiring personnel may regard the CV as incomprehensible and disorganized.

According to a survey by TalentWorks that analyzed 6000 job applications the optimal length for a CV is 475-600 words.[35] These should be enough words to convey the necessary content to the hiring personnel effectively. However, arguably number of pages is what leaves a stronger impression on the hiring personnel. CareerAddict outlines the optimal number of A4 pages in most cases as being based on applicant:[36] "One-page CVs are best for school leavers, recent graduates, and generally early job seekers who have little to no professional experience at all."[36] While two pages are "preferred by 91% of recruiters", CareerAddict claims many companies "reject CVs that are less than two pages because they simply don't provide enough evidence about the applicant's skills and qualifications".[36] Three pages are generally used only by executives or academics that have a long list of qualifications. Therefore, the most accessible length of a CV for most applicants is two pages with 475-600 total words.

The next concept is content relevance and it can be difficult as it pertains to each applicant's CV differently. As the first impression is everything to passing the hiring personnel, the content must be immediately conveying a sense of qualification. Therefore, meaningless content and content that can not stand alone apart from other content must be discarded to optimize a CV for passing the hiring personnel. The wording of content pertains to the method with which the CV addresses the applicant. Implementing the keywords to optimize for the ATS scanner will fill the same objective with the hiring personnel. As the CV will look more relevant to the job position for the hiring personnel.

2.3. Required and Situational Content of a CV

As discussed, the content of a CV is crucial to passing both the hiring personnel and ATS scanner. However, what is a CV in actuality, and what is required content for a CV to contain. Furthermore, as each application has different qualifications, each applicant's CV must-have content situationally purposed for their CV. Therefore, what is a CV and what is the required and situational content of a CV?

A CV is "a short account of one's career of qualifications prepared typically by an applicant for a position".[37] When applying for a position within a firm, some aspects of a CV are required. These requirements can come from the position or firm that the CV is intended for or from the definition of the CV itself. Putting aside the firms or positions requirements, all CVs must include the five following requirements to be defined as a CV:

- 1. Contact information
- 2. CV objective
- 3. Relevant skills
- 4. Work experience
- 5. Education[38]

The substance of each requirement varies from applicant to applicant. However, every CV must include these five requirements to be effective. A further explanation of each requirement is due: Contact information is required as the firm at the bare minimum must have some way to contact the person so he or she can be accepted for the position. CV objective is required as it specifies what and who the CV is intended and without it, the CV can fill no purpose. Relevant skills are required as without them you have no relation to the CV objective. Work experience and education are required as without it one has no qualifications.

As a CV is an account of one's qualifications it is also possible to leave education and work experience empty if one has none. However, it is then hardly an effective CV.[37] Along with the requirements of a CV, some aspects are situational. These vary and most likely the firm or position to which one is applying will lay out these aspects, or it is apparent from the position itself. If not specified it is difficult to know what to include. Too little and one will under-qualified, too much and the CV will be too long.[39], excesses and unorganized. Therefore, it is essential to distinguish one's CV with the right quantity and quality of situational content. Let's examine some of the more typical situational content that could be included in a CV:

- 1. Professional association

- 2. Volunteer experience
- 3. Languages
- 4. Additional training courses
- 5. Publication
- 6. Awards/Honors
- 7. References[40]

The effectiveness of a CV can drastically change due to use of situational content. Therefore it is necessary to further explain each situational content:
A professional association is any trade union, learned societies, regulatory universities, and other inter-professional societies. Many associations have certain prestigious and hold their member to a certain standard of quality. Therefore a professional association can improve a CV if relevant.[41][42] Volunteer experience is any volunteer work relevant to the position. Languages are any spoken or written language relevant to the position. As most firms in our interconnected market interact with some multilingualism, languages can easily increase the quality of a CV. Additional training courses are any extra courses relevant to the position. Publications are any reports, books, or other published materials that could show qualifications for the given position. Awards and honors are any university or professional awards or honors given that show qualification for the given position. References are very situational, as putting references in a CV may make you seem unsure of yourself and in need of validation from others to show qualifications. However, if a CV has the right references it can assure employers of your qualifications and experience.

2.4. Structure of The CV

As discussed, the structure of a CV is crucial to an applicant obtaining a job. Writing an effective CV isn't just about including the right content. It's also about how you present that information. The structure is especially vital to leaving a good first impression upon the hiring personnel. As such, the structure of an applicant's CV is essential to passing the hiring personnel and obtaining a job. Therefore, what are the possible structures of an applicant's CV, and does an optimal structure exist for obtaining a job?

Firstly, a CV could be structured in many different ways. To discuss if an optimal structure can be found, it is productive to first discuss the most common CV structures. The three most popular CV structures are the Chronological structure, Functional structure, and Combination structure. [43] Each structure has strengths and weaknesses, and an applicant should choose a structure that is best suited to their qualifications. The Chronological is the most popular structure and to gain an understanding of the standard CV structure, it is possible to outline the Chronological structure.

1. Contact information
2. CV objective
3. Work experience
4. Education
5. Relevant skills

However, just because the Chronological structure is the most popular does not mean it is the most optimal. A study done by LADDERS tracked the eyes of hiring personnel when reading an applicant's CV. [44] They found that hiring personnel spent an average of 7.4 seconds on each CV and therefore first impression was essential to passing the hiring personnel. The study found that the most successful CV´s had; "Clear, simple layouts with clearly marked section and title headers. Recruiters in our study spent more time focusing on job titles than on any other element". [44] Furthermore, the worst-performing CVs had a; "Cluttered look and feel, characterized by long sentences, multiple columns, and very little white space". [44] The study concluded with a list of recommendations for applicants to pass the hiring personnel. Summarized, the study concluded that:

1. The best structure had a simple layout and font
2. Hiring personnel scanned for headers and these should therefore be clearly stated
3. Short declarative statements are easier to process for hiring personnel
4. The optimal length for a CV was two pages
5. Any keywords meant to pass the ATS scanner should appear in context

As such, the exact structure or position of content is not significant. Only that the layout is clear and accessible to hiring personnel and that the content is relevant. The content should be short and declarative, as any non-relevant information will not be read. With only 7 seconds, it is hard to leave a good first impression on the hiring personnel. Making it essential for applicants to correctly structure their CV.

3. Analysis

3.1. Target Group

If the process of obtaining a job by CV could be optimized to increase the likelihood of obtaining a job, it would benefit most if not all applicants. As outlined in the 2.2 five stages exist to obtaining a job through a CV. If the process of obtaining a job by CV is to be optimized, one of these stages must be optimized. However, optimizing one stage may have a negative effect on another stage. Therefore, a process optimized for the majority of applicants may fail to optimize anything like the majority of applicants is an extensive target group. As such, it would be an advantage to choose a target group that can benefit from certain optimized stages.

It also stands to reason, that choosing a target group that benefits most from obtaining a job would be an advantage. Using what has been discussed in the background, it is possible to choose a target group that would benefit greatly from an optimization of the process of obtaining a job.

As outlined in 2.2, four types of job searchers exist; unemployed searchers, active searchers, employed tiptoers and employed passive. Taking the four types as the four possible target groups, it is possible to discern which type would benefit most from being the target group. As shown in 4 the unemployed searchers and active searchers have a significantly higher likelihood of obtaining a job using a CV. Therefore, they would benefit more from an optimized process of obtaining a job than the employed job searchers. As such, the unemployed searchers and active searchers are the two most relevant target groups for an optimized process.

When analyzing which of these two target groups would benefit more, it is essential to define what would constitute "benefit more". Two factors have the potential to benefit, the individual's chances of obtaining a job and the countries economy in which the individual resides. Benefiting more would mean either a higher chance of obtaining a job or a positive effect on a countries economy.

As discussed in 2.1.1, unemployment can have severe negative effects on both economy and the unemployed individual. Therefore optimizing the process for the unemployed has already established positive effects on both the economy and the individual. While active searchers could potentially benefit just as much as unemployed searchers: A lack of sources supporting the benefits of optimizing the process for specifically active searchers indicates an insignificant benefit.

While an abundance of sources outline the benefits of optimizing the process for unemployed searchers exists, as seen in Background 2.1.1. As such, the unemployed searchers would benefit more from an optimized process than the active searchers. Therefore, by comparing the four types of job searchers, it can be concluded that the unemployed searchers would be the most justifiable target group for an optimized process of obtaining a job by CV. Consequently, any optimizations to the process of obtaining a job will be considered with hindsight to the unemployed.

Furthermore, CV´s can be written in any fathomable language and any optimization to one CV language might not translate the same in too others. Therefore, it is important to select a language so that any optimizations to the process of obtaining a job would not have to include all languages. As English is the most spoken language it makes sense to focus any optimizations to one or more English speaking countries. The United States and United Kingdom are the most influential countries with English as the first language. Therefore, the target group of the unemployed will have focus on the US and UK. As such the target group will be: The US and UK unemployed.

3.2. Factors to Optimize

To optimize the process of obtaining a job by CV for the unemployed, and the unemployed applicant must have a higher chance of obtaining a job by CV. However, to optimize the process, it is essential to know what factors have an influence on the likelihood of an applicant obtaining a job. As these factors are what dictates the unemployed chances, any increased chances of obtaining a job would have to come from optimizing one of the factors. But what exactly are the factors that determine the chances of an unemployed obtaining a job? Using what has been discussed in section 2 on page 2, it is possible to identify each factor and analyze how it could be optimized.

As outlined in 2.2 the process of obtaining a job has five stages. If an unemployed applicant were to obtain a job they would have to pass through these five stages. Different factors will determine the unemployed applicant's success in obtaining a job. It is vital to determine these factors as this will indicate what could potentially increase unemployment applicants' chances of obtaining a job. Therefore, leading to what an optimized process of obtaining a job by CV could look like for the unemployed. Using what has been discussed in the background and further sources, it is possible to analyze which factors will determine if an unemployed applicant obtains a job.

Many factors influence unemployed applicants' chances of obtaining a job. That does not necessarily mean that all factors must be optimized to obtain a job. However, optimizing the process of obtaining a job will most likely mean optimizing one or several factors. The most influential factors necessary for obtaining a job are:

- Unemployed applicant´s qualifications
- Time spent unemployed
- Passing the ATS scanner

- Passing the hiring personnel
- Passing the job interview
- Quantity of CV´s
- Quality of CV´s
- Economy of a country

It is essential to understand that factors will overlap and in some cases, and that two factors may be optimized by the same solution. To understand how one or more of these factors could be optimized, each factor requires an explanation of its correlation with obtaining a job:

An unemployed applicant´s qualifications are essential and as the five stages of obtaining a job purpose are to choose the most qualified applicant, the applicant with relevant qualifications has a natural advantage. However, relevant qualifications can be difficult for many unemployed applicants to obtain. This is due to it becoming harder to obtain a job as an unemployed as time on, as discussed in 2.2. With relevant qualifications it becomes easier to obtain a job, leading to more relevant qualifications, as such success leads to success, this is known as the "job life cycle".[45] Therefore, a process that increases the chances of obtaining a job will increase the magnitude of qualifications for the unemployed applicant. However, the job life cycle can also work in reverse. Meaning that few or no qualifications may lead to not obtaining a job and therefore no new relevant qualifications. This may lead to a longer period of unemployment, which is the second factor; time spent unemployed.

As seen in section 3, the likelihood of obtaining a job declines drastically during short-term unemployment, that is the first 27 weeks of unemployment. Therefore, the time spent unemployed is a factor that has more influence on unemployed applicant's chances of obtaining a job the longer they are unemployed. As such, the unemployed applicant's qualifications can be optimized through other factors, as it entails obtaining a job. While the time spent unemployed can be optimized by decreasing the time needed to obtain a job.

As discussed in Background 2.2.2, to obtain a job an applicant is required to pass the ATS scanner and hiring personnel. Therefore, passing the ATS scanner and hiring personnel is a very non-ignorable factor in an unemployed applicant for obtaining a job. Firstly, as outlined in Background 2.2.2, the ATS scanner scans for three underlying elements in an applicant's CV. The ATS scanner scans by searching for pre-defined keywords relevant to the job Therefore, if a process could incorporate these keywords into a CV, it would increase the likelihood of passing the ATS scanner. Secondly, the hiring personnel will glance at an applicant's CV, spending an average of 7 seconds on each CV. An applicant's CV must be able to leave a good first impression and pass the "7-second test". As outlined in Background 2.2.2, three concepts largely determine the chances of passing the hiring personnel. As such, if a process could integrate these three concepts into an unemployed applicant's CV, it would increase the likelihood of passing the hiring personnel.

Passing the job interview is an important factor for the unemployed to obtain a job. Unlike the relative predictability of the ATS scanner and hiring personnel due to a shared objective of screening applicants CV. The job interview is very unpredictable, possible because it attempts to screen the applicant directly, not through the CV that the applicant applied with. It also factors in the human element, as the interviewer could make choices based on bias or mood. Trg claims that "there is little relationship between candidates' performance in interviews and subsequent on-the-job performance".[46] The job interview is also the final stage in obtaining a job before the job offer, so the hiring personnel will seek any specialized skills that are unique to that job. As such, the lack of a system other than practicing the large pool of commonly asked interview questions makes passing the job interview factor hard to optimize.[47] An unemployed applicant will also naturally improve at conducting job interviews the more job interviews the applicant participates in. So, optimizing this factor over time can be done by focusing on factors that will net more job interviews and subsequently more job offers.

The quality and quantity of CV applications an unemployed individual sends is an important factor in obtaining a job. Quality and quantity are very loose terms but are meant to be so, as many overlapping factors and circumstantial aspects affect the quality and quantity of CV. Quality is the likelihood that a CV will net an unemployed applicant a job. Therefore, a quality CV is not necessarily the most time-consuming CV, it is the CV most likely to pass all five stages of obtaining a job. Optimizing for a high-quality CV would means an increased chance of each individual CV obtaining a job for the unemployed applicant. On the other hand, quantity is the amount of CV´s an unemployed applicant will have the potential to apply with. However, the quality and quantity of CV are inversely correlated. As such, an increase in quality will lead to a decrease in the quantity of CV´s sent. Therefore, any optimization to quality or quantity will have to be weighed carefully.

As outlined in 2.1.1, unemployment has an inverse correlation with the economy of a country. Therefore, the state of the economy will influence the likelihood of an unemployed obtaining a job. As such, in a robust economy, the chance of obtaining a job will increase, while in a fragile economy the chances of obtaining a job will decrease. It is improbable that this factor can in any way be directly optimized. However, an optimized process of obtaining a job could lead to less unemployment in an economy over time.

Consequently, there are many potential factors to optimize. However, it is possible to optimize those factors most relevant in increasing the chances of the unemployed obtaining a job. Although open to question, the following factors seem most relevant to optimize for the unemployed:

- Passing the ATS scanner
- Passing the hiring personnel
- Quantity of CV´s

Passing the ATS scanner and hiring personnel is essential for any applicant, but even more so for the unemployed. As discussed in 2.2.2, the majority of applicants are screened during the ATS and hiring personnel phase. As such, to increase the likelihood of obtaining a job for the unemployed, the ATS scanner and hiring personnel are the greatest obstacles. The Quantity of CV´s is essential for the unemployed, even more essential then the Quality of CV´s. It stand to reason that it would be productive to focus on either the Quantity or Quality of CV´s. However, as discussed in 2.1.1, hiring personnel discriminate strongly against unemployed, especially long-term unemployed, and are 45% less likely to be invited tp a job interview then non-unemployed despite the same qualifications. Furthermore, as discussed in 2.1.1 unemployed tend to have less qualifications then the non-unemployed applicant therefore making it difficult for many unemployed to compete through quality. Another argument for focusing on quality would be that it would guarantee to always pass the ATS scanner and hiring personnel. However, as discussed in 2.2.2, the ATS scanner is susceptible to screening out qualified applicants by mistake and the hiring personnel´s "7-second test" is unpredictable and as such will not always choose the most qualified applicant.

These three optimizations will also overlap other factors, such as Quantity of CV´s hopefully optimizing the Time Spent Unemployed. It is important to mention, that optimizing these three factors, does not mean neglect to other potential factors. As such, if it is relevant, other optimizations to factors should also be incorporated. Therefore, the most relevant factors to optimize for the unemployed are the ATS scanner, hiring personnel and quantity of CV´s.

3.3. Why a Software Solution

The most relevant factors to optimize are outlined in 3.2. The method by which the optimizations can occur is not clear. It is also important to mention that any possible optimizations would have to be based on the applicants qualifications, as the applicant has to be able to defend themselves in the job interview. any solution to the potential optimizations needs to be built not as a "system cheater", but as a fair representation of the applicants best aspects. As such, it is vital to describe the method by which the factors have the potential to be optimized.

Both the process of passing the ATS scanner and hiring personnel can be optimized systematically. Furthermore, a solution most be able to produce a large quantity of CV´s. As such, all three factors invite a software solution. A software solutions excels at repetitive and iterative tasks. This points to the fact that, if a optimal system of passing the ATS scanner, hiring personnel and produce a large quantity of CV´s can be found. Then a software solution can replicate that optimal system faster then any human. Therefore also allowing a large quantity of "optimal" CV´s to be produced.

A software solution would therefore need to produce these optimizations through a system. A structure by which software generally produces through a system is through a software program. Consequently, a software program would be useful in making any optimizations into a solution.

3.3.1. Streamlining

As discussed in both 2.2.2 and 2.4, the length of a CV can play a key role in passing the hiring personnel. The length of a CV will also affect the quantity of CV´s produced, therefore it is productive to determine the optimal length of a CV and to stick to that length so to not waste an applicants time. As outlined in 2.2.2 the optimal length of a CV for most applicants was found to be two pages with 475-600 total words. This optimal length would most likely be the same in the case of the unemployed applicants. Therefore, it would be beneficial to aim for this optimal length to maximize the unemployed chances of obtaining a job.

Furthermore, in 2.2.2, it was determined that content relevance was essential and that non-relevant information would harm the applicants chances of obtaining a job. Many applicants only write one CV for all the jobs they apply to, leading to less relevant content, and in-correct keyword usage. As such a method with could make an unemployed applicant´s CV the optimal length and sort through the content of the CV to cut any non-relevant information could be very beneficial. This can be solved by utilizing a software program that systematically streamlines an unemployed applicants CV.

Streamlining an unemployed applicants CV using a software program, would contribute to optimizing passing the hiring personnel, passing the ATS scanner and producing a larger quantity of CV´s. It would increase the unemployed applicants chances of passing the hiring personnel as it makes the length of the CV more accessible and the content more relevant. Since it removes any non-relevant content, it also removes the keywords that would not be relevant to the job. Streamlining would also allow a larger quantity of CV´s produced, as the software program would systematically optimize the CV allowing the unemployed applicant to not waste time.

3.3.2. Structuring and Producing a CV

For a CV to be effective it must both be sufficiently organize its content and be accessible to the hiring personnel when viewed. The structure of a CV is vital to passing the hiring personnel. With only 7 seconds to make a good impression, the structure of a CV must convey competence. Furthermore, the method by which the hiring personnel views the applicants CV is also important. It is therefore essential to analyze both the structure and production of a CV.

As discussed in 2.4, several elements go into making a good structure. In essence, to optimize passing the hiring personnel, the structure should incorporate the recommendations of the "Eye Tracking Study. [44] The structure of the CV should be simple and accessible, with short declarative statements that are relevant. Furthermore, the CV should be optimally be two pages, as already mentioned in 3.3.1. No one structure is optimal, however any optimization would have to take notice of the recommendations outlined in 2.4, to optimize passing the hiring personnel. A potential optimization using a software program could be to have a fixed structure that follows all of the recommendations that would then be filled by the unemployed applicants content.

The production of a CV is meaningful as it refers to the type of file the CV will be sent as. It is essential that a CV, once structured correctly is accessible and practical for both the hiring personnel and ATS scanner. The two most popular methods of production are the PDF file and Word document.

If the job specifies a preference for one form or the other, the applicant should follow the advice. However if not specified in most cases the PDF file is superior. [48] This is because a wide range of programs can be used to open and view PDF files and the file will look the same no matter which computer is used to open it. The PDF file also prevents hiring personnel from making changes to the CV. The only advantage of the Word document being that it tends to takes up less space in a computer's storage than the average PDF file. Therefore the optimal production of a CV is in most cases the PDF file. A potential optimization to produce the CV as a PDF file could be using LaTeX to automate the process.

3.4. Keywords

As discussed in 2.2.2, "70%-80% of large companies worldwide utilize an ATS, while smaller businesses utilize it about 50% of the time". This is a significant percentage of firms/organizations and therefore it is essential to incorporate a method into the software program to optimize passing the ATS scanner. Using what has been discussed in section 2 on page 2, it is possible to find a method which could be implemented to increase the US and UK unemployed chances of obtaining a job.

As outlined in 2.2.2, the ATS scanner scans for three elements in an applicants CV. The ATS scanner will scan for education, work experience and for certain pre-defined keywords throughout the CV. As discussed in 2.2.2, most ATS scanners will award points as a way of screening applicants. Most firms/organizations will have a threshold of points, if an applicant does not meet that threshold of points, the CV will will be rejected and will not proceed to the hiring personnel. In other words, it is the minimum to pass the

ATS scanner, however, the more relevant keywords a CV contains for each position, the more effective the CV process will become at passing the ATS scanner. The more relevant keywords a CV has the more points the ATS scanner will give and the higher ranked the CV will be listed when viewed by hiring personnel. To optimize passing the ATS Scanner, the software program should at the minimum pass the ATS scanner and if possible score highly on the ATS scanners ranking. However, what would a potential optimization look like?

The first elements the ATS Scanner scans for are most likely the work experience and education. This is so the firm/organizations can screen those applicants without the qualifications for the job. It is vital to note that the firm decides what the ATS scanner will find to be relevant work experience and education. However, in most cases, the job description has a section listing the minimum qualifications for the position, this gives a clear minimum of what qualifications the applicant must have. Not much can potentially be optimized by the software program, as either the applicant has or does not have the qualifications. However, the ATS scanner will differentiate work experience and education from the other sections by the header. Therefore, a potential optimization could be to have the headers clearly marked as work experience and education. This would also optimize passing the hiring personnel, as it makes the sections of the CV more structured and accessible.

When the ATS Scanner scans the entire CV for keywords, it is important to note that the keywords are pre-defined. The most effective method of passing the ATS scanner would be to directly implement the pre-defined keywords into the unemployed applicants CV. However, the pre-defined keywords can differ from job to job and firm to firm, making it hard to guess the pre-defined keywords. Therefore, the only real hint the applicant will have for which keywords should be implemented into the CV are the job position and job description. It would be extremely difficult for any software program to make precise guesses as to which keywords to implement. Therefore, it would likely still largely fall on the unemployed applicants shoulders to guess the keywords. However, even though the software program might not be able to guess the exact keywords, the process of implementing the keywords can be optimized. As such, instead of the unemployed applicant having to write a new CV with the correct keywords every time they apply. The software program might be able to systematically implement the keywords selected by the applicant into their CV.

3.5. Narrowing the Analysis

To summarize, we have narrowed down the key aspects in our analysis to the following:

- The target group is the US and UK unemployed.
- The solution will optimize the factors of; passing the ATS scanner, passing the hiring personnel, producing a large quantity of CV´s

- The solution has to be a software program.
- The solution must
- The solution has to be a PDF file.
- The solution must incorporate Keywords.

4. Problem Statement

How can we make a software solution, that enables the British and American unemployed people, to create a large number of CV applications. Such that they can get a higher chance of getting past the ATS scanner and hiring personal, than if they were to individually create each application while not expending more effort?

5. Product Development

5.1. Planning Phase

Before building the program, we brainstormed and created a Minimal Viable Product (MVP). This MVP fig. 5 is a general high level schematic outlining what our program is supposed to do.

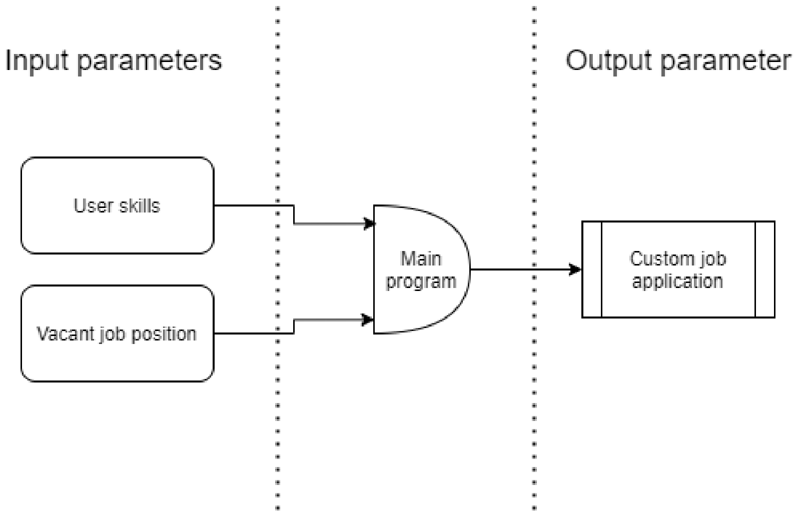


Figure 5.: MVP

Here we discussed what were the most essential parts, and came to the conclusion that the following things were needed: The users skills/abilities and the information from the vacant job position. These things should be transformed into a custom application.

After further consideration, we had encountered a problem: That generating a custom CV from nothing but skills/abilities would create a very childlike, maybe even unreadable, application. This could be solved with a lot of data, and therefore a lot of time calibrating the process of automatically creating sentences. This may also result in a lower quality application, if this calibration doesn't happen. A CV, that might and might not even get through the "ATS/keyword scanner" as outlined in the analysis.

We therefore sought to instead create a filter, where the high quality sentences would be somewhat guaranteed and maintained. This filter is only supposed to filter out all the unnecessary parts of a longer quality CV, into an application that is ready to be sent. In other words, we decided to instead concentrate on creating a tool that aids in sending out applications, instead of a CV generation program.

We then changed the MVP schematic to the following:

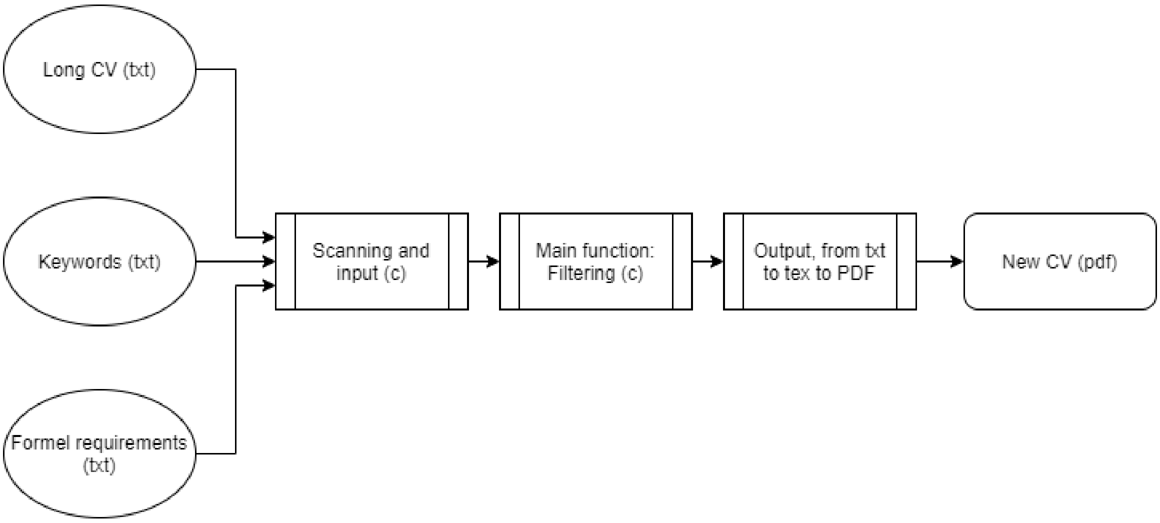

Long CV (txt)
Keywords (txt)
Formel requirements (txt)
Scanning and input (c)
Main function: Filtering (c)
Output, from txt to tex to PDF
New CV (pdf)

Figure 6.: Revised MVP

This program is supposed to take in the keywords/job application, some formal requirements for the structure of the new CV, and the originally long CV. The long CV should contain as many pages as possible, of all abilities and skills one has. It should also include prior work experience, and anything else relevant on a CV. In this way, the program chooses the most important sentences, work experience and skills to be included, and from the structural requirements creates a new CV.

From here we decided to organize our MVP into a UML diagram, to make it more apparent what the essential functions were supposed to do, and which functions that were essential:

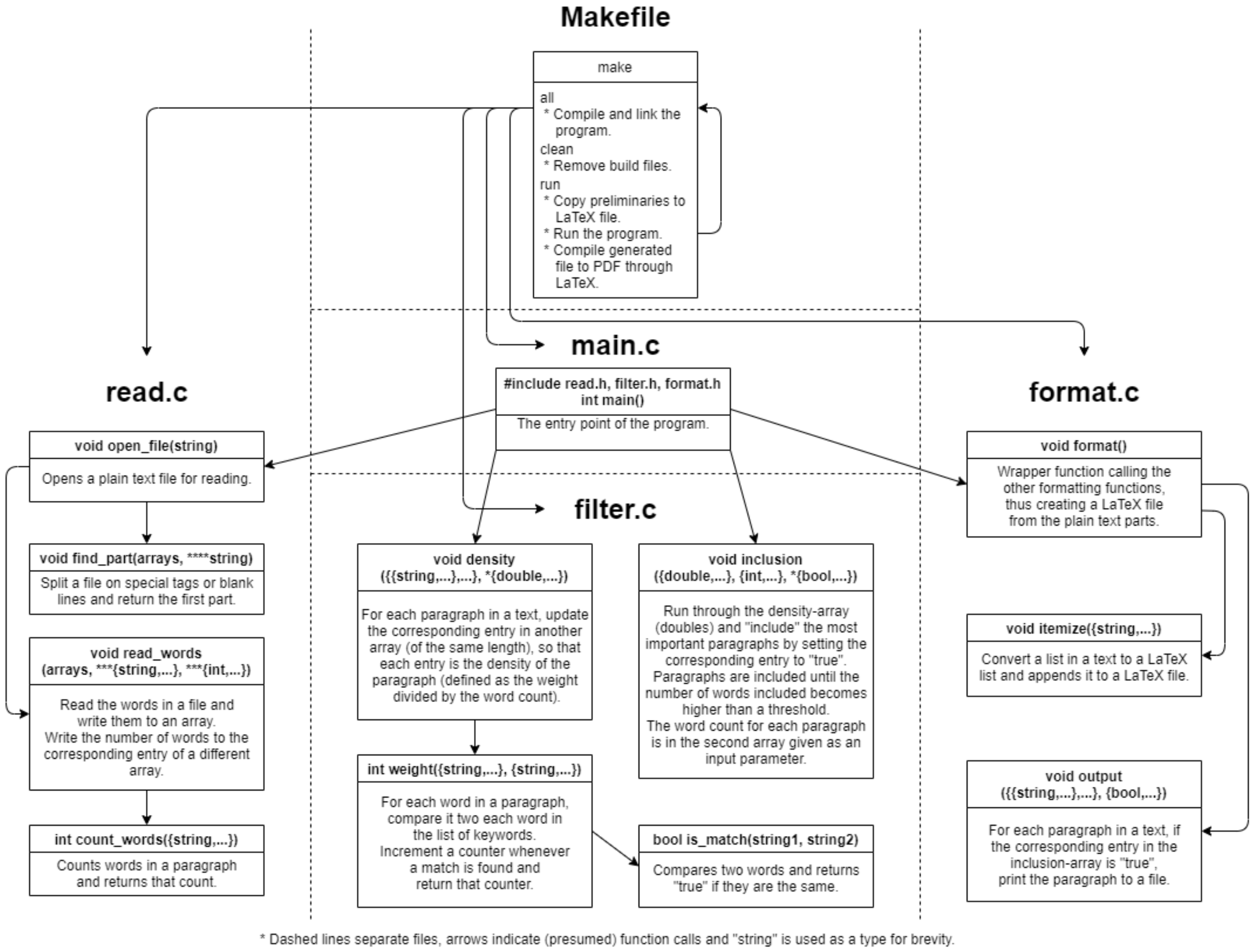

Makefile
make
all
* Compile and link the program.
clean
* Remove build files.
run
* Copy preliminaries to LaTeX file.
* Run the program.
* Compile generated file to PDF through LaTeX.
main.c
#include read.h, filter.h, format.h
int main()
The entry point of the program.
read.c
void open_file(string)
Opens a plain text file for reading.
void find_part(arrays, ****string)
Split a file on special tags or blank lines and return the first part.
void read_words (arrays, ***{string,...}, ***{int,...})
Read the words in a file and write them to an array. Write the number of words to the corresponding entry of a different array.
int count_words({string,...})
Counts words in a paragraph and returns that count.
filter.c
void density ({{string,...},...}, *{double,...})
For each paragraph in a text, update the corresponding entry in another array (of the same length), so that each entry is the density of the paragraph (defined as the weight divided by the word count).
int weight({string,...}, {string,...})
For each word in a paragraph, compare it two each word in the list of keywords. Increment a counter whenever a match is found and return that counter.
void inclusion ({double,...}, {int,...}, *{bool,...})
Run through the density-array (doubles) and "include" the most important paragraphs by setting the corresponding entry to "true". Paragraphs are included until the number of words included becomes higher than a threshold. The word count for each paragraph is in the second array given as an input parameter.
bool is_match(string1, string2)
Compares two words and returns "true" if they are the same.
format.c
void format()
Wrapper function calling the other formatting functions, thus creating a LaTeX file from the plain text parts.
void itemize({string,...})
Convert a list in a text to a LaTeX list and appends it to a LaTeX file.
void output ({{string,...},...}, {bool,...})
For each paragraph in a text, if the corresponding entry in the inclusion-array is "true", print the paragraph to a file.
* Dashed lines separate files, arrows indicate (presumed) function calls and "string" is used as a type for brevity.

Figure 7.: UML

5.2. TKOP-Model

A good way to describe the production could be the TKOP-model also known as technique, knowledge, organization, and product. It gives an easy impression of what the project group had performed. First is the technique, it is essential to create a product for our project to find a solution. one must use several techniques including C-programming, LaTeX, and GitHub techniques, researching on the web, etc. Those techniques are a good way to establish a project solution. C-programming can help us to produce a CV using the techniques of commands, functions, etc. LaTeX can give a monotonous structured format all the text is always the same.

These functions take one to have a fair amount of knowledge to make a LaTeX text properly. First of all knowledge about a CV is important, and how to use the knowledge is also important. Therefore it's best to start research about programming online that we don't know about, and afterward, the gained information will be used to implement our code and see if it's working. If not, then it's best to research more about the exact problem. This is the same way the waterfall model is being used. We don't exactly use it directly but it is a way to be finished with the code.

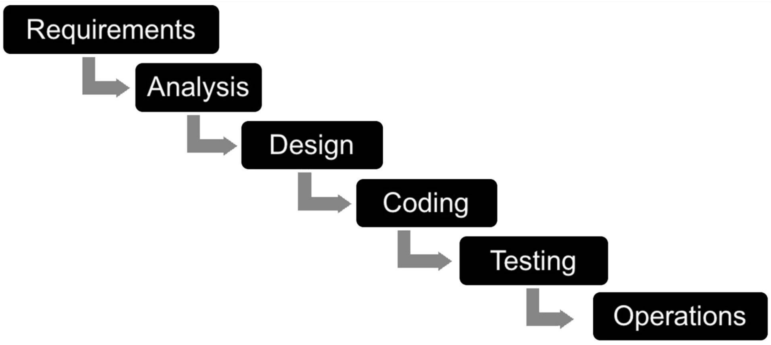


Figure 8.: Waterfall model [49]

It has the elements that have been implemented in our coding. First, we have the requirements for the program, so we can have a program that is working properly, next, we will design the structure of the program just like from the UML diagram in figure fig. 7 on the preceding page
so we know what should have in the program. Implementation comes next, so we could test the program later, and if we have other suggestions for improvement we can write them down for further development.

In the process of coding the product, we choose to use a vertical work distribution, as many of the product components can be made parallel to each other. We divided it into several functions, so the group members can either do it alone or in pairs for each function to work with. Each small group has to know future updates about the function development, so the program can be put together later on. To get an overview of what we have done, there is a temporary division of labor for the schedule, so the group can see who is finished and what can be finished.

At last, our product is a program from C that will help to generate a CV. This way the CV can be made multiple times, that means the use-value will be very high for the applicant will have a greater chance of getting a job interview, rather than an ATS scanner will decline a CV, even though the text is very detailed with a lot of good points. The C-program itself will not make the CV but it is an extension of a complex way to load from other text editors other than LaTeX, then compile it to LaTeX, so the tex file will make a pdf for the person.

If the whole process should be looked at different levels then there are five of them:

1. Our program should be researched in four different aspects
   a) Knowledge to find the rights methods
   b) Techniques to develop the product
   c) A strategy to give every single person a piece of the project, so all have an issue to work with.
   d) Knowledge about the product itself, and summarize the parts from all the issues to make a final product.
2. Knowledge about the product of creating a CV.
3. After gaining the knowledge, we can start coding for our software solution with help of programming techniques
4. Our product has a long code with more function than the group itself, Organize for each individual in the group is the key to get the project done faster.
5. After making the product comes the testing where it can be done with help of experts in the CV. This process will afterward lead to testing with the applicants if it is good enough for our criteria. All the coding parts will be added together in a readable way, and summarize all of it into a pdf file from a makefile.

5.3. Read.c

Read.c functions as the de facto file for creating the necessary data structures to contain the input files content. At first, the files are read one char at a time into each their char array that gets extended when it runs out of space, replaces the End Of File marker with a char Null and stops reading from the file.

5.3.1. Datastructures in Use

To overview the data structures the content of the input files gets organized. The long CV gets split into

- A Nullterminatet char array, for general information.
- A triple pointer char containing list items, accompanied by an int array for several items in each list and an int for the number of lists.
- Another triple pointer char containing the free text area, also accompanied by an int array and an int.
- An int counting the number of words used outside of the free text.

Alongside the contents of the keywords file organized as

- A double char pointer alongside an int.

The data structures described here are returned from the Read.c code section through pointers, so while in the Read.c area, they all have another level of a pointer than described. A visual representation of the variable containing the free text area of the long CV, the quadruple pointer is for containing the address to the triple pointer Main.c initialized. The structure then follows the triple pointer pointing to an array of double pointers each pointing to an array of pointers, wherefrom each of these then points to char arrays.

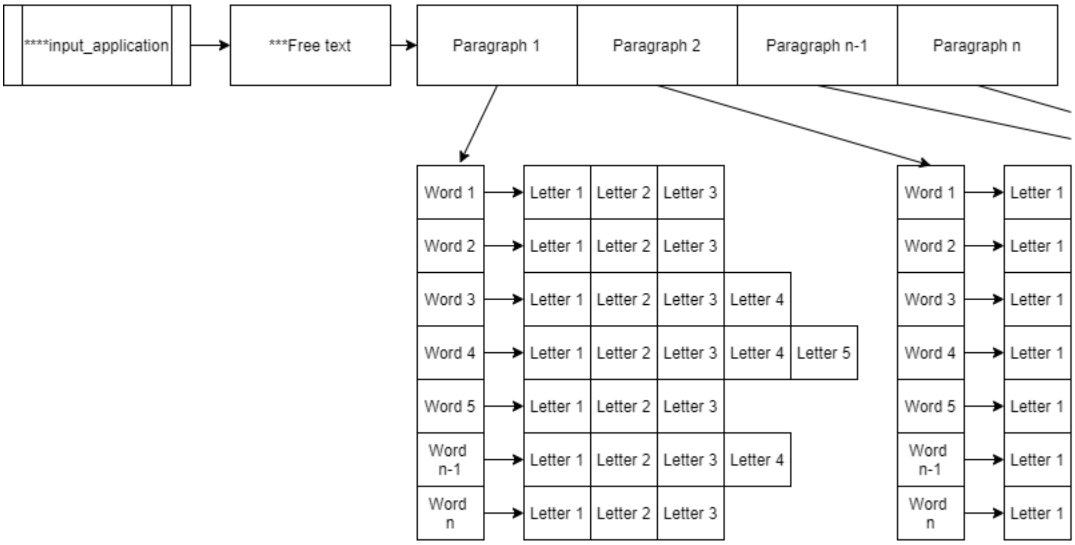


Figure 9.: Data structure for the free text area

As seen the data structure contains arrays of pointers, and as the size of the arrays isn't known at the time of compiling we used dynamic memory allocation. Witch further means the number of elements in the array can't be determined solely by the array itself, therefore the number of sections is kept track of by an int and the number of words in each section in an int array. In the last level of an array, the char arrays containing words are all Nullterminated and work as strings, so the length can be determined at a later point in time.

The triple pointer for the list items is structured as an array for each category of the list and points to arrays of items to be listed, each item is a line of text.
The free text is divided into sections, each section containing individual words.
The keywords are structured as an array of individual words.

5.3.2. Data deviding

In dividing the string containing the raw text from the long CV input file into the appropriate data structure, the first step is to make a dynamically expanding array containing char pointers, and setting the values to the locations in the raw text string where there is a "#", and only if it is preceded by a linebreak to avoid false positives. Thereafter the first area is assumed to be the general info and is copied to a new location as a separate string. From here the rest of each line that started with a "#" is copied into a string to determine if it's the free text area or a listing area, then treating the area accordingly. For listing areas, an element is added to the array containing all the listings, the element containing the address of an array containing strings, each string being copied from the raw test separating at line breaks. The number of lines in each listing and the number of listings is dynamically increased as needed. For the free text area, when found, is then scanned through to find a double linebreak, which is considered a section, the section has then copied a word at a time into the triple pointer data structure meant for the free text area. The section amount and word in that section amount are set to dynamically increase as needed. What constitutes a separator of words for our code is the character the spacebar produces and linebreaks.

After all, areas that were separated by tagged lines have been handled, the temporary double-pointer containing the tag locations "#" is then freed up, and so is the long raw text string of long CV, hence the reason for copying everything into the new data structures.

After this the keywords are handled in mutch the same way, temporary raw text string then divided into words and stored in the appropriate data structure. In all places where an array is dynamically extended the number of elements contained in the array is kept track of by local variables, that's mostly returned by pointer alongside the array to keep track of.

5.4. Filter.c

The filter file is the next step in the process to creating a new CV, after read.c has been executed. The filter.c file has the following structure and interaction with the main.c file:

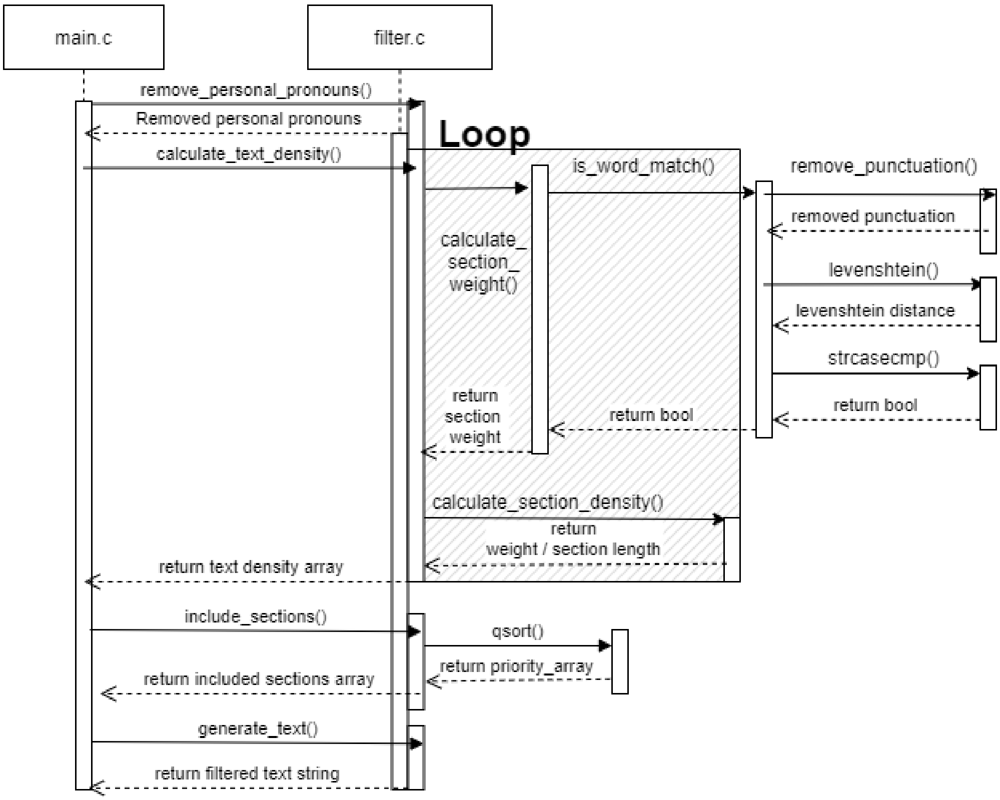


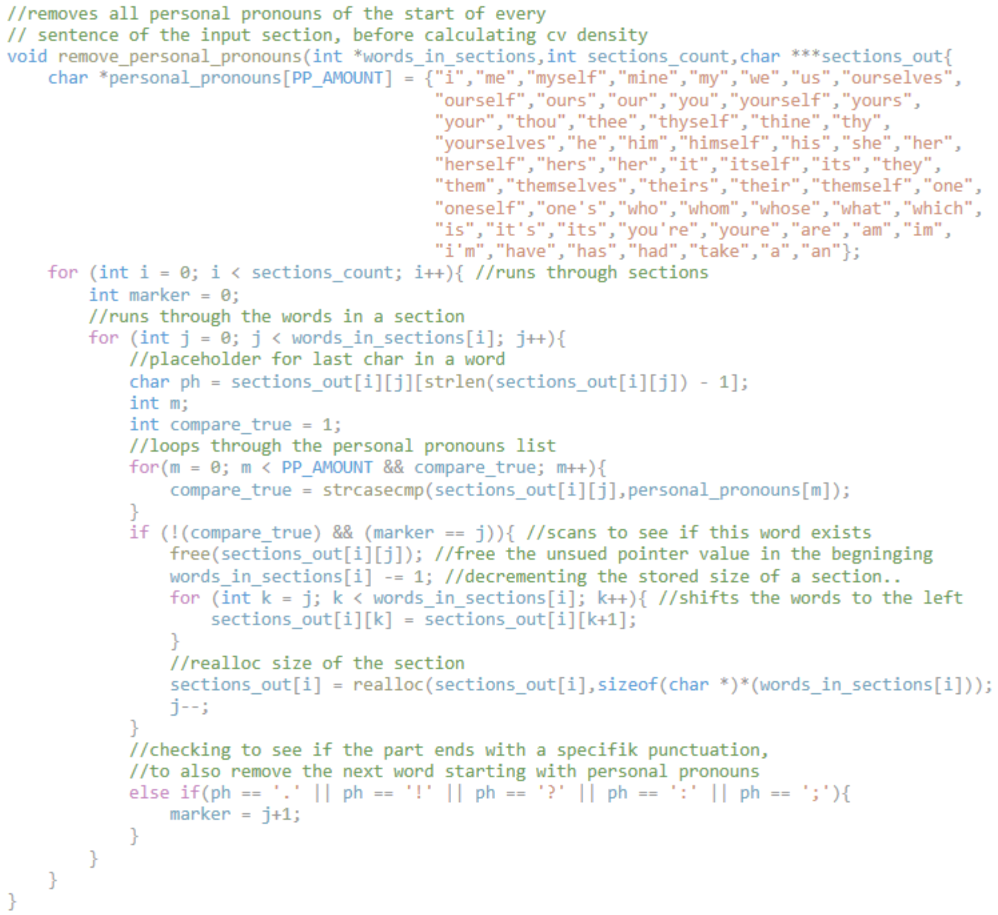
Figure 10.: Filter Sequence Diagram

The end result from the filter file will then later on be used in the format.c file.

5.4.1. Removing Personal Pronouns

After receiving the "free text" part, from the read.c file, we need to remove any personal pronouns. We do this, since personal pronouns are quite unnecessary, since they don't add any substance to the sentences. This is important to do since hiring personals, as we talked about in the background section, spend a fraction of a minute to look through the application. Having more unnecessary words will just distract the hiring personals from understanding why they should hire you. As talked about in the background section, not using personal pronouns increases your chances of getting an interview by 55 percent.

To remove all personal pronouns before filtering the rest of the text, we made the following function:



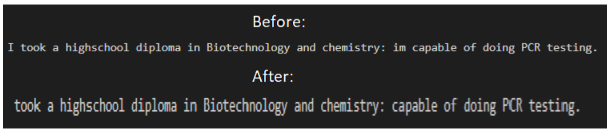
```c
//removes all personal pronouns of the start of every
// sentence of the input section, before calculating cv density
void remove_personal_pronouns(int *words_in_sections,int sections_count,char ***sections_out{
    char *personal_pronouns[PP_AMOUNT] = {"i","me","myself","mine","my","we","us","ourselves",
                                          "ourself","ours","our","you","yourself","yours",
                                          "your","thou","thee","thyself","thine","thy",
                                          "yourselves","he","him","himself","his","she","her",
                                          "herself","hers","her","it","itself","its","they",
                                          "them","themselves","theirs","their","themself","one",
                                          "oneself","one's","who","whom","whose","what","which",
                                          "is","it's","its","you're","youre","are","am","im",
                                          "i'm","have","has","had","take","a","an"};
    for (int i = 0; i < sections_count; i++){ //runs through sections
        int marker = 0;
        //runs through the words in a section
        for (int j = 0; j < words_in_sections[i]; j++){
            //placeholder for last char in a word
            char ph = sections_out[i][j][strlen(sections_out[i][j]) - 1];
            int m;
            int compare_true = 1;
            //loops through the personal pronouns list
            for(m = 0; m < PP_AMOUNT && compare_true; m++){
                compare_true = strcasecmp(sections_out[i][j],personal_pronouns[m]);
            }
            if (!(compare_true) && (marker == j)){ //scans to see if this word exists
                free(sections_out[i][j]); //free the unsued pointer value in the begninging
                words_in_sections[i] -= 1; //decrementing the stored size of a section..
                for (int k = j; k < words_in_sections[i]; k++){ //shifts the words to the left
                    sections_out[i][k] = sections_out[i][k+1];
                }
                //realloc size of the section
                sections_out[i] = realloc(sections_out[i],sizeof(char *)*(words_in_sections[i]));
                j--;
            }
            //checking to see if the part ends with a specifik punctuation,
            //to also remove the next word starting with personal pronouns
            else if(ph == '.' || ph == '!' || ph == '?' || ph == ':' || ph == ';'){
                marker = j+1;
            }
        }
    }
}
```

Figure 11.: Remove Personal Pronouns Function

It's a pretty simple function. To start it reads a certain section. Then it removes all personal pronouns at the start of each sentence in that section. It loops through it, until there is no longer any personal pronouns left.

In order to identity which words are personal pronouns, we have hardcoded a list of personal pronouns[50] (including some words that are also unnecessary), that will be 'case insensitive compared'. Each time a word is identified as a personal pronoun, a loop swaps all the words 1 to the left in the array, where the last word in that array will then be freed. There are also some hardcoded punctuation symbols, to indicate what is defined as the "start of a sentence".

this results in the following:



```
Before:
I took a highschool diploma in Biotechnology and chemistry: im capable of doing PCR testing.
After:
took a highschool diploma in Biotechnology and chemistry: capable of doing PCR testing.
```

Figure 12.: Removing Personal Pronouns Example

As we can see, the filtered result is much cleaner and easier to read, without losing any information. There is only problem left with it. Since the words are just shifted 1 to the left in the array, the sentence no longer begins with a capitalized letter. This will later be fixed in a function in format.c, where such things will be formatted.

5.4.2. Word Matching

Word matching is one of the most interesting and most fundamental functions. It is the function that allows for keyword matching to exist, so that the application may get through the ATS/keyword scan. How it works is it checks if input word 1, matches with input of word 2. Where input 1 is the unfiltered original application text and input 2 is the keywords for the job posting. If it is a match, then one might have an idea, that this section where the matched keyword is included may be important. It does this using the function strcmp(n,r), which returns 0 if the string (or in our case, word) is identical.

One quickly stumbles upon the following problems trying to match keywords.

- 1. Since letters are ASCII defined, when comparing "Bathtime" with "bathtime", it won't return it as a match, even though it clearly is.
- 2. What if the word is the last word in a sentence? Comparing "Bathtime." with "Bathtime" should return a match, but it won't.
- 3. What if one has made simple spelling mistakes, such as spelling "Twillight" with 1 'l' instead of the required 2 'l's.

In order to overcome the first problem, one could easily make a loop, to loop through each letter, and check for both the capitalized and the small letters, and return a match. This is especially easy, since the small letters always has a 32 higher ASCII value from their capitalized counterpart. But after reading through the string library we found a function that does just the same comparison while ignoring capitalized letters: strcasecmp(n,r).

The second problem has a quite a simple solution as well. Using the ASCII value of different kinds of punctuation symbols, we can easily remove these symbols by inserting a null terminator. instead, thus making the string shorter. This is done, on a temporarily saved word, such that this temporarily saved word can be used in the string compare function. This is done since we don't want to change the punctuation of the final output.

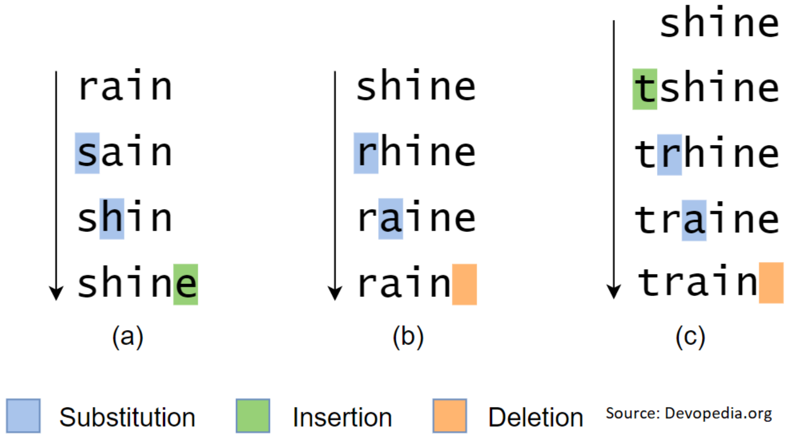
The third problem, is the trickiest one of the bunch. To solve this we decided on using an algorithm to find the levenshtein distance. We found a translation of this algorithm in C and imported it[51]. The way this algorithm works, is by finding the distance between words. The larger the distance, the greater the difference between the words. The algorithm determine distance as the amount of operations needed to change string A to string B. The algorithm uses 3 different types of operations:

- 1. Insertion
- 2. Deletion
- 3. Substitution

to give an example of how these 3 operations can be applied:

- Turn "Cat" into "Fat" (Substitution, 'C' with 'F')
- Turn "Fart" into "Far" (Deletion, removing the 'T')
- Turn "Sittin" into "Sitting" (Insertion, adding a 'G')

as an example, the levenshtein distance between rain -> shine is 3, shine -> train is 4:



Source: Devopedia.org

Figure 13.: example of levenshtein distance, [52]

The levenshtein distance can be defined using the following naive recursive definition where |a| and |b| is the distance of a and b respectively:

$$\operatorname{lev}(a,b) = \begin{cases} |a| & \text{if } |b| = 0, \\ |b| & \text{if } |a| = 0, \\ \operatorname{lev}(\operatorname{tail}(a), \operatorname{tail}(b)) & \text{if } a[0] = b[0] \\ 1 + \min \begin{cases} \operatorname{lev}(\operatorname{tail}(a), b) \\ \operatorname{lev}(a, \operatorname{tail}(b)) \\ \operatorname{lev}(\operatorname{tail}(a), \operatorname{tail}(b)) \end{cases} & \text{otherwise.} \end{cases}$$

Figure 14.: Naive Recursion, [53]

The "tail" part is the entire word minus the first letter. The base steps go as following: It returns the length of a as the distance, if the length of word b is zero, and vice versa. if the first letter of a and b equal each other, then it recursively runs the algorithm with the tail part of a and b. Otherwise, it finds the minimum value (assuming the operations are weighted differently) of either operation 1 (Deletion), 2 (Insertion) or 3 (Substitution) and adds 1 to the result.

Using the levenshtein distance quickly raises a problem though, since the distance between cat and hat is smaller than the distance between artificial and artificially. It's obvious that this isn't a spelling mistake, but 2 distinct words. Therefore we ran some tests, to see how long words need to be, before something like this doesn't happen to often. The problem quickly becomes, that the longer the requirement the better the accuracy. But also the fewer words we can use the algorithm on, making the algorithm redundant. We came to the conclusion based on our tests, that having words with greater than 4 letters, would be the best compromise between length and accuracy.

Now the problem is, what distance is accepted as 2 words being the same? Here the problem also becomes, that grammatical endings are still the same words, just conjugated. Based on a list of grammatical endings, the longest English ending adds a suffix of 3 letters[54]. Thus, the max levenshtein distance should be under 4 letters. Retesting using these values, we find that in nearly all arbitrary cases, the levenshtein algorithm matches the right words. One problem may arise from the fact, that words can have different meanings in different contexts. Unfortunately taking account for semantics is out of the scope of this program.

After using the algorithm to find the levenshtein distance, one could argue that the remove punctuation function is redundant. But this is not the case. To the contrary, removing any unnecessary parts of a string before finding the levenshtein distance, makes the algorithm more precise. This is the case, since we can make sure the allowed distance is smaller than what it had to be, if we hadn't removed punctuation.

The overall structure of this functions ends up looking like the following:

The overall structure can be described as this:

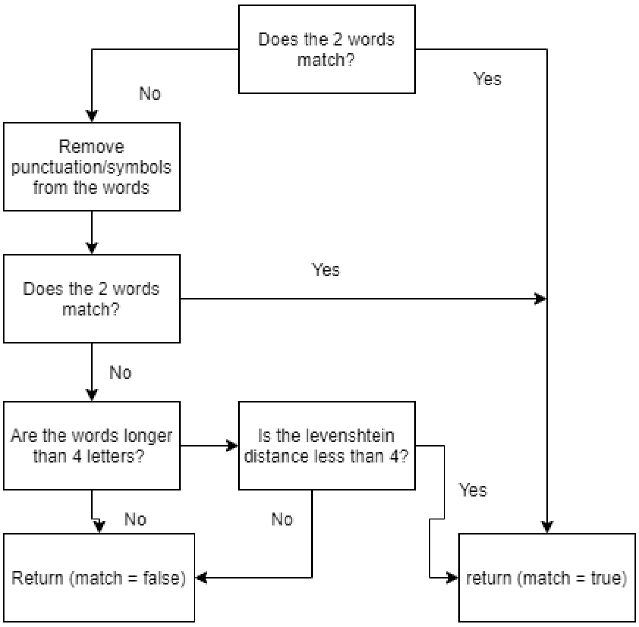
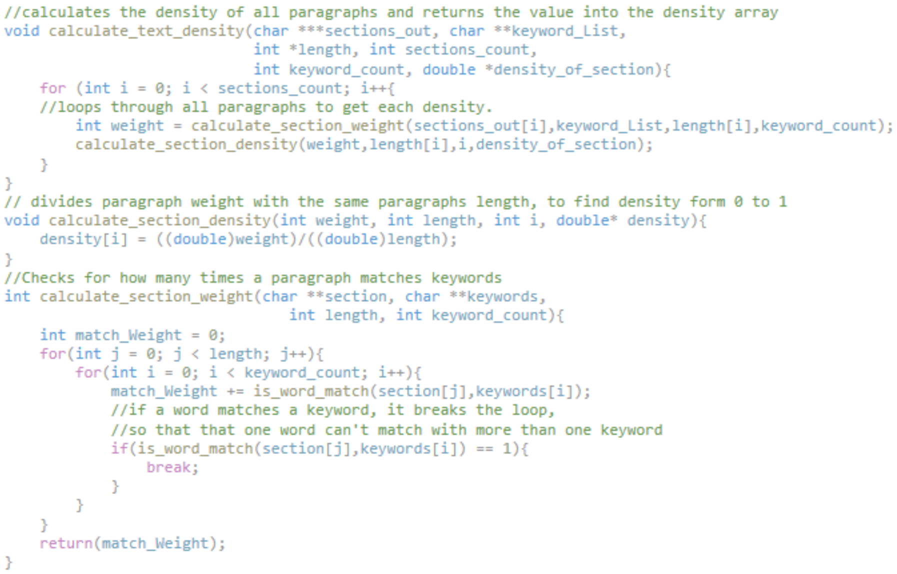

Does the 2 words match?
No
Yes
Remove punctuation/symbols from the words
Does the 2 words match?
Yes
No
Are the words longer than 4 letters?
Is the levenshtein distance less than 4?
Yes
No
No
Return (match = false)
return (match = true)

Figure 15.: Flow diagram of word matching

The structure is constructed as a chain of if statements, to avoid unneccessary calculations, so that the run time is optimized as much as possible.

5.4.3. Section Density



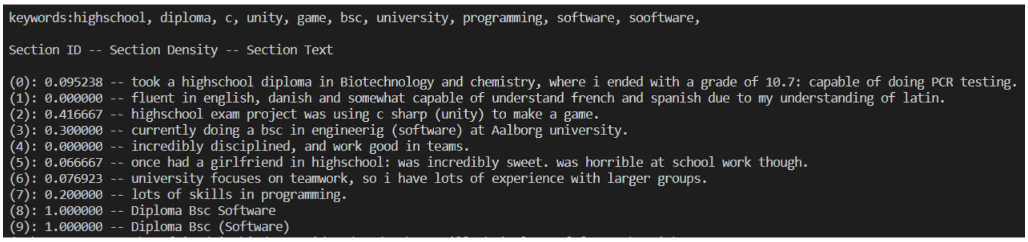
```c
//calculates the density of all paragraphs and returns the value into the density array
void calculate_text_density(char ***sections_out, char **keyword_List,
                            int *length, int sections_count,
                            int keyword_count, double *density_of_section){
    for (int i = 0; i < sections_count; i++{
    //loops through all paragraphs to get each density.
        int weight = calculate_section_weight(sections_out[i],keyword_List,length[i],keyword_count);
        calculate_section_density(weight,length[i],i,density_of_section);
    }
}
// divides paragraph weight with the same paragraphs length, to find density form 0 to 1
void calculate_section_density(int weight, int length, int i, double* density){
    density[i] = ((double)weight)/((double)length);
}
//Checks for how many times a paragraph matches keywords
int calculate_section_weight(char **section, char **keywords,
                                int length, int keyword_count){
    int match_Weight = 0;
    for(int j = 0; j < length; j++){
        for(int i = 0; i < keyword_count; i++){
            match_Weight += is_word_match(section[j],keywords[i]);
            //if a word matches a keyword, it breaks the loop,
            //so that that one word can't match with more than one keyword
            if(is_word_match(section[j],keywords[i]) == 1){
                break;
            }
        }
    }
    return(match_Weight);
}
```

Figure 16.: Density and weight function

Section density is based on the same concepts as in physics: Weight(mass)/Volume = Density. In this case, weight is how many times in a section different words match with a keyword, and volume is how many words there are in a section. The boundaries for the density function is as follows: Density goes from zero to one. Where zero is that there is no matches in a section, and one, is that every single word matches. Though a problem quickly arises from doing it this way: A word may match with multiple keywords.

- Section: "physic expert", keywords: "Physics, physical".

Since the levenshtein distance is only 3 between physic and physical, and 1 between physic and physics, the word physic will match with them both, thus potentially creating a section with a density higher than the allowed 1. To fix this issue, there was made a break condition for the weight loop, if a word from the section matched with a keyword. The break function makes it such that no word could match with more than 1 keyword. Once the sections density has been calculated, the calculated density will be saved chronologically in a calloc'ed array of double values. This will be used to asses which sections are the most "important", from the definition that the more keyword matches, the greater the importance.
Printing the result from this, we can see:

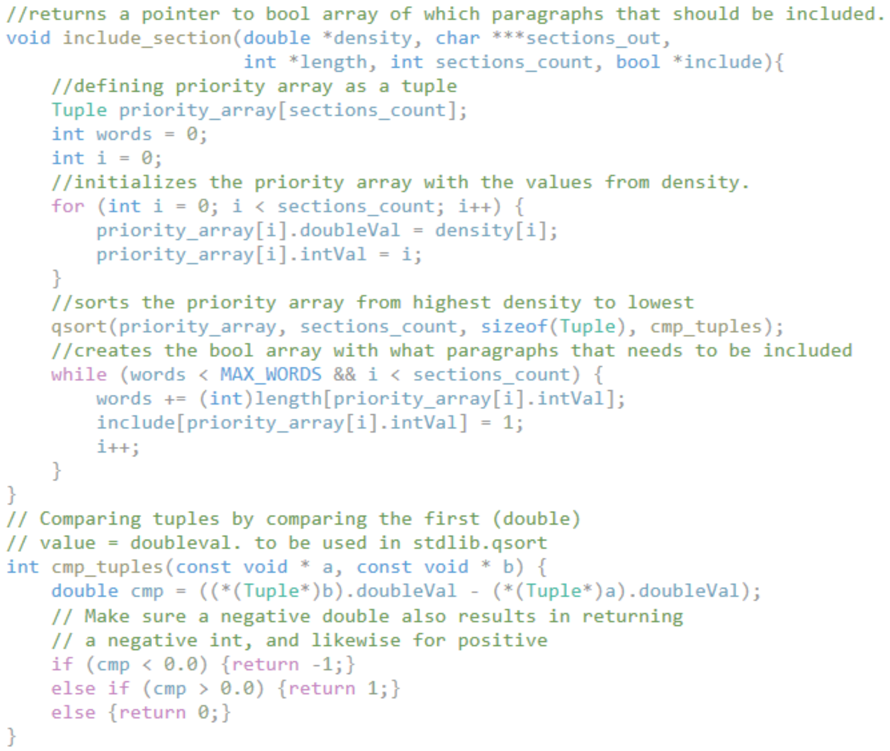


```
keywords:highschool, diploma, c, unity, game, bsc, university, programming, software, sooftware,

Section ID -- Section Density -- Section Text

(0): 0.095238 -- took a highschool diploma in Biotechnology and chemistry, where i ended with a grade of 10.7: capable of doing PCR testing.
(1): 0.000000 -- fluent in english, danish and somewhat capable of understand french and spanish due to my understanding of latin.
(2): 0.416667 -- highschool exam project was using c sharp (unity) to make a game.
(3): 0.300000 -- currently doing a bsc in engineerig (software) at Aalborg university.
(4): 0.000000 -- incredibly disciplined, and work good in teams.
(5): 0.066667 -- once had a girlfriend in highschool: was incredibly sweet. was horrible at school work though.
(6): 0.076923 -- university focuses on teamwork, so i have lots of experience with larger groups.
(7): 0.200000 -- lots of skills in programming.
(8): 1.000000 -- Diploma Bsc Software
(9): 1.000000 -- Diploma Bsc (Software)
```

Figure 17.: Density Array Printed

Thus it can easily be seen, which sections may be best suited for the job opening. In this case, an example of a job opening specializing in software.
To demonstrate that the functions work, it can be seen that both line 8 and 9 gets printed with a result of exactly 1. Even though one of the words match with 2, and the other, though misspelled, also matches with 2.

5.4.4. Included Sections and Generation



```c
//returns a pointer to bool array of which paragraphs that should be included.
void include_section(double *density, char ***sections_out,
                     int *length, int sections_count, bool *include){
    //defining priority array as a tuple
    Tuple priority_array[sections_count];
    int words = 0;
    int i = 0;
    //initializes the priority array with the values from density.
    for (int i = 0; i < sections_count; i++) {
        priority_array[i].doubleVal = density[i];
        priority_array[i].intVal = i;
    }
    //sorts the priority array from highest density to lowest
    qsort(priority_array, sections_count, sizeof(Tuple), cmp_tuples);
    //creates the bool array with what paragraphs that needs to be included
    while (words < MAX_WORDS && i < sections_count) {
        words += (int)length[priority_array[i].intVal];
        include[priority_array[i].intVal] = 1;
        i++;
    }
}
// Comparing tuples by comparing the first (double)
// value = doubleval. to be used in stdlib.qsort
int cmp_tuples(const void * a, const void * b) {
    double cmp = ((*(Tuple*)b).doubleVal - (*(Tuple*)a).doubleVal);
    // Make sure a negative double also results in returning
    // a negative int, and likewise for positive
    if (cmp < 0.0) {return -1;}
    else if (cmp > 0.0) {return 1;}
    else {return 0;}
}
```

Figure 18.: Inclusion Function

The process of finding the included sections can be described in three steps:

- 1. Create a tuple struct holding the density value, and the section ID.

- 2. Q-Sort the struct after density value.
- 3. save what section ID's should be included in the final output, based on the density, in a bool array.

First a struct is created. This is to ensure, that after sorting after highest density, the original order is still maintained. This is important, since it can make it unreadable, if all the sections are mumbo jumbo'ed around. After that, a Q-sort algorithm is used from the standard library. This algorithm makes sure, that the highest density comes first, by sorting by densities. Q-sort is based on quick sort, which is a type of divide and conquer algorithm. Which as the name suggests, divides the list (array) into atomic parts, and conquers the parts, such that the new list is sorted.

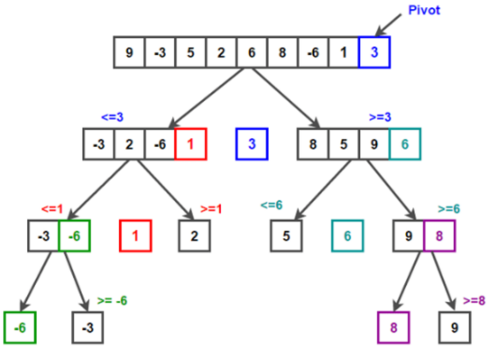


Figure 19.: Quick Sort [55]

The way it works, is that it chooses a pivot point, traditionally randomly, the left most or the right most element. In more modern versions of quick sort, the pivot point is chosen as the median. The rest of the list is then divided up into 2 lists. The elements of the 2 lists are chosen by wether they're are smaller or larger than the pivot point. A new pivot point is selected, and this process continues until each list only contains one element. Then the elements can be recombined in the new order, and thus the list (array) us sorted.
Next, a bool array is created and calloc'ed. The values for the bool array are determined following this flowchart:

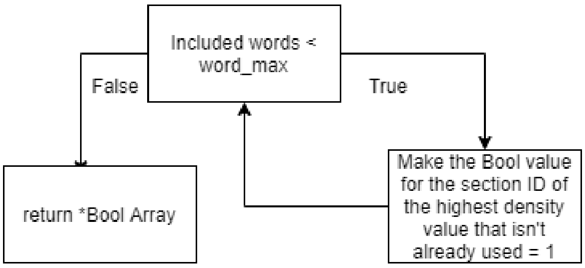
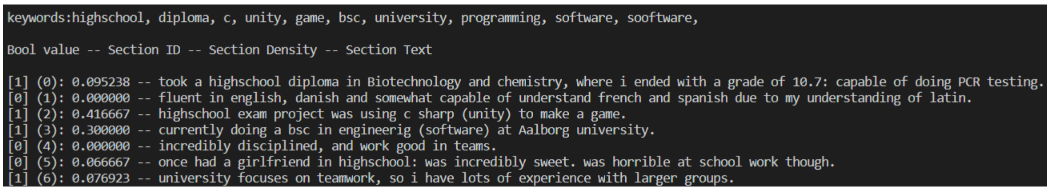

Included words < word_max
False
True
return *Bool Array
Make the Bool value for the section ID of the highest density value that isn't already used = 1

Figure 20.: Flow Diagram of the Bool Array

In order to determine what sections should be included, a while loop is run, until a certain amount of words are included. The amount of allowed words in the output is restricted to around 650, as that is the range, which grants the most amount of interviews. That is, according to the research as described in the background section.
Here is an example, of it working:



```
keywords:highschool, diploma, c, unity, game, bsc, university, programming, software, sooftware,

Bool value -- Section ID -- Section Density -- Section Text

[1] (0): 0.095238 -- took a highschool diploma in Biotechnology and chemistry, where i ended with a grade of 10.7: capable of doing PCR testing.
[0] (1): 0.000000 -- fluent in english, danish and somewhat capable of understand french and spanish due to my understanding of latin.
[1] (2): 0.416667 -- highschool exam project was using c sharp (unity) to make a game.
[1] (3): 0.300000 -- currently doing a bsc in engineerig (software) at Aalborg university.
[0] (4): 0.000000 -- incredibly disciplined, and work good in teams.
[0] (5): 0.066667 -- once had a girlfriend in highschool: was incredibly sweet. was horrible at school work though.
[1] (6): 0.076923 -- university focuses on teamwork, so i have lots of experience with larger groups.
```

Figure 21.: Example of the Bool Array

The generation of the cv is quite simple in theory. All it does, is include all sections that the bool array determines should be included, and saves it into a long string. Starting from the lowest section ID to the highest ID.
The process of getting here is a little more complex, since it requires a lot of dynamic arrays, and inserting a lot of Null terminators, such that it doesn't print null pointers (I.e "(NULL)").

5.5. Format.c

After the CV has been filtered, our next plan is to print out the text to a LaTeX format, and afterward compile it to a pdf file. This program will check the format and edited it accordingly to how a CV should be visualized. There are several step before entering the main function for format.c, and it's done accordingly to this diagram.

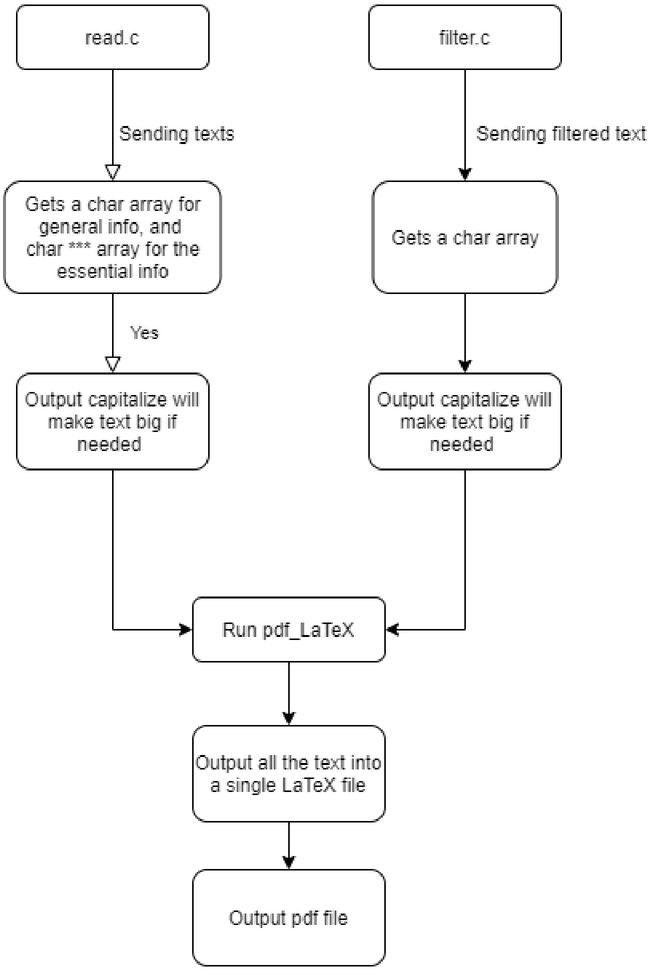

read.c
filter.c
Sending texts
Sending filtered text
Gets a char array for general info, and char *** array for the essential info
Gets a char array
Yes
Output capitalize will make text big if needed
Output capitalize will make text big if needed
Run pdf_LaTeX
Output all the text into a single LaTeX file
Output pdf file

Figure 22.: Flow diagram for coming to format.c

5.5.1. Output Capitalized

The first function will capitalize any letter that has a dot, question mark or an exclamation mark. The function has one variable, a char array to check after characters if these need to be capitalized, it can therefore be useful for all the contents in the CV - the general information, education, work experience or free text. There are two integer markers to indicate the index of an array, and right now, it's outside of the index, so the markers will not have accessed to the first index in the array, since it doesn't need to be for finding text, as long as file is being read. To see the length of a string, a function called strlen will know how long is a string.

A string can be a sentence, and every time for the first letter will always be capitalized, and it can be done so with toupper. This is a function that are included from the c-type header. After a sentence is finished with a dot, question mark or an exclamation mark, the letter is always capitalized after those markings. For doing line breaks in LaTeX it only needs two backslashes, a par or a newline command. In order to these above properly for a full text, a for loop needed to check the length of the char array, it will do so by -1 with it, because it has to end at an exclamation point before the null terminator.

The integer 'i' should always go up by 1 if it's still lower than the length, and that means until the text is finally done it will continue to capitalize letters. The capitalized letter will only capitalize with two scenarios, the letter will be either after a space and if there is a line break. There is also a chance if the text turns up to have a dot, question mark or an exclamation mark like described before. This will let the marker index 1 and 2 have their value changed by +1, since the program will afterward look for a space.

5.5.2. LaTeX General Contents

The most crucial part is how to output the entire text from for example an ordinary txt file to LaTeX format. output general contents has a char array that will take the whole section from general contents and added it in the CV, also, there are a file handling variable as well to output the text to a specific file.
The function output capitalized is included to this function, if the text need to capitalized. Finally, the general text will begin at the center with a LaTeX command, follow up by a minipage command, and at last make the textwidth smaller, so the text will be moved more at the left side, and the title is auto generated.

General contents from read.c will not be sorted after keywords or anything similar since name, address, e-mail etc. is important to let the hiring personals know and therefore it should not be filtered from filter.c. Output of the text will be a string from the char array and end it with a minipage and hfill - the unused lines will be used by this command in LaTeX. After hfill has been executed, the output of a picture begin a minipage and then print out a picture with includegraphics, and it is been programmed to be right beside the general info.

5.5.3. Handling Essential Contents and Free Text

Essential contents has a char array with three-pointers to check the sentences, the words and the letters to get an opportunity for adding educations and work experiences. These will therefore have an n-amount of itemizes and n number of items in an itemize, and item and itemize is a form for putting bullet points. The program will start printing out the self-selected subtitle and n number of itemizes. Texts under a hashtag are being loaded from read.c and it will afterward be sent to a LaTeX CV.

To check the amount of itemizes for the essential contents it will use a for loop to check each value, and after the 'i' is bigger than amount of itemizes the loop will stop - the amount of itemizes is coming from read.c. After having checked hashtags and the text as well, and hashtags are for indicate the sections, this way the array essential contents variable will therefore have an 2-dimension array. Same goes for the number of items to output because of there can be n amount of jobs to writes, therefore, the program is using a for loop to run down after how many jobs that has been transferred from read.c. The arrays has also 2 dimensions, first for checking the amount of itemizes [i] and for the number of items [j] in the indexes. when there is no need to make more items and itemizes the for loops will stop and end the functions.

As for the free text and this is where the most common for background information if this relevant for the specific job. This only function that are being filtered by filter.c. It is fairly simple, the text that are being received is a char array and will therefore be treated as a string just like before for the general information. All information is going to the CV file at the end of the function. It will have a static title so it is not going to be edited.

5.5.4. Run The Whole LaTeX Document

When all of the functions above is created this function will then call all other function and run it add all the text into a file, and build the document into LaTeX and later on a pdf file. First of the function will create or overwrite a file if possible. There can be various kinds of reason why a file can't be created. If the file can't be opened then it will be the end of the program with "EXIT FAILURE", and it will go out with a message "Cannot open file". As long as the file is created or overwritten the rest of the program will run.

There are included a preamble and this will enhance the LaTeX format readability for the user and hiring personal when reading it. Most of the time a picture af oneself is required for most of the jobs, therefore it is possible to manually insert a photo to the CV.

5.6. Main.c

As the program is heading to the final phase, all other c programs will be added to a large file, this is going to become the main.c, and it will have read.c, filter.c and format.c included for printing out the CV pdf file.

The program will start of with running the start read. In this particular function it will load out the whole file, similarly speaking it will split a complete text up to four separate if possible. A section is the biggest part of a text, therefore it will start with checking how many sections, then the sentences, words and letters at last. This function will also do the job to load out all the necessary bullet points, keywords and word counter.

In the filter.c it has four function included in the main file. At first it has to check which sentence is best. A sentence length can be as long as it is, there in conclusion we have used calloc as a way to make a dynamic memory space for both "density of section" and "included section". It can be tiring to have a lot of "i" in a CV, therefore remove personal pronouns is also included to remove those pronouns, and will check a whole section for every time there is at least one pronoun.

calculate text density will calculate each string from the free text (background information) to see if a keyword is matching any words in a sentence. When this is done the include section function is check with help of boolean values. These values is consisting only with 1 and 0, 1 is true while 0 is false in programming language. Afterwards these selected text will be generated by generate text, and that is the last part of the filter function before heading to format.c.

For creating a CV LaTeX file, it is going to run run all the function for general information, work experience information, education information and the free text (other background information). After the function has printed out the CV pdf file, it will come to a close with freeing all the variables of those were mallac, realloc and calloc, so all of them don't continue to allocate more memory. All of those were will also apply to the "the ending" function.
Below the whole program is explained in a simple graphic method about what the main.c file contains in which order to execute the numerous aforementioned functions. It follows a generally chronologically imperative flow:

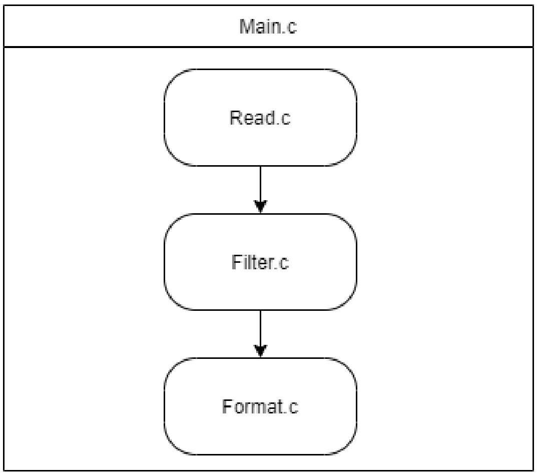


Figure 23.: Flow Diagram of main.c

5.7. Makefile

In order to compile the entire program, a makefile has been made. It links the header and .c files together, to create object files. Which are then compiled into one file named "generate cv". This newly compiled file, when executed, can include the user picture, the user made keywords.txt and the user made long cv.txt, and from there create a pdf file. This means, that the user mealy has to execute one file, for the program to work.

6. Experiments

6.1. Method

In order to test if our product helps solve our product statement, it is possible to set up two experiments that quantitatively evaluates the effectiveness of our program. The first experiment to test if the digital CV would have a higher success rate at passing the hiring personnel, and one for the ATS Scanner.

Firstly, to test for the hiring personnel, it would be productive to compare the success of the digital CV with a CV written by human hands. In order to do this, we have devised the following experiment:

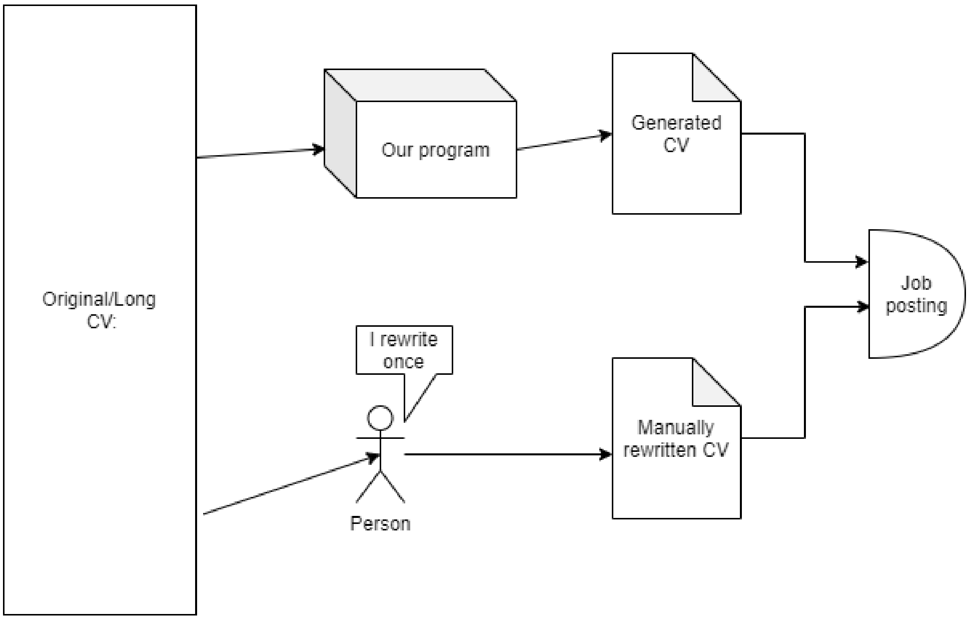


Figure 24.: Experiment Method

In the experiment we will simulate the same person in both CV´s. We do this to ensure that there is no extra information in the human rewritten CV that our program couldn't have created, such as extra qualifications. In this way, we ensure the test is as fair as possible.

Thereafter the danish job center will be asked, to answer which of the two CVs is most suited for the job posting. How this works is as such:

The human will shorten down and create a CV based on the long CV. This CV is is gonna be compared to n number of job postings. Our program is gonna filter down the long CV, and create a job application, for each job posting. To ensure, that some job postings don't just have lower requirements, we will be comparing the new CV applications from both the program and the person to the same job postings each time. In total, we will be comparing n number of applications.

In this way, we can assume that both the human who had rewritten one CV and the program, which has rewritten n number of CVs will take about the same amount of time, to create n CVs. Therefore we can somewhat ensure that the results are somewhat fair, in relation to the amount of effort each set of n CV's requires. The structure of both CV's should also remain the same in the testing, such that one CV isn't prioritized because of better formatting. This is important since the test is primarily testing the CV's contents compared to the job opening.

This experiment will compare the results of how many times one gets a "job interview" using one of the two methods: Comparing the amount of "job interviews" (based on what the job center thinks is most likely to earn an interview) from the qualitatively created CV from the human, to the quantitatively created CV from the program to each other.

6.2. Results

After we were finished with the program, we decided to print out one CV that has been made by a human, and 15 almost identical CV to each other the only differences is the background information is matching to each keyword there is in the given job posting, therefore the text has been custom-tailored to the job posting.

We went to the job center to ask a job counselor about which of those 2 CV's were most likely to grant one an interview.

She had the following comments:

- 1. She thought the overall structure of the human-made CV was better since it was less robotic.
- 2. She thought that the contents of the custom made CV, were much more likely to grant one an interview.
- 3. She commented on, how the biggest utility of the generated CV, was that it made it a lot easier to include appropriate skills. Since most people have lots of skills but are bad at selecting which to include.
- 4. She remarked, that since it was more robotic, it may not catch the hiring personals eye if they are hiring personalities instead of skills.

Overall, it can be concluded, that the digital CV was comparable to the CV written by human hands. The digital CV was a moderate success in that it did marginally better in the eyes of the job counselor. However, the digital CV was still lacking in some aspects and could be improved.

6.3. Resume Scanner Experiment

As a second experiment, we decided to test if our product was able to pass through a resume scanner. We did this, through a website called resumeworded.com [56]. This website is capable of telling if the CV would pass an ATS scanner and if the CV has other errors, such as structural errors. The final score for the human written CV was 8, and so was the average score of the computer written CVs. That is a very low score seeing as the scale goes from 1-100. In conclusion, it could be said that our product doesn't work properly, but it is just as likely, that the original long CV is badly written. This is due to the fact, that both CVs get around the same score. On the flip side, if they both get the same score, then it's not worse to use the custom-built CV. In other words: Our program effectivities the process, without losing quality.

7. Discussion

Even though the program works much to the intent, it still leaves a multitude of things to be desired: If two sections has the same content, but formulated in different ways, both sections will be included. This makes the finished product seem unprofessional at best. As an example, with the keyword "English", and the sentences:

- 1. "Fluent in English"
- 2. "Mastery level of English grammar"

Both mean approximately the same, but only one sentence should be included. For this, one could envision a system, where only the section that matches "English" with the highest density, should be included.

Another problem stems from the fact, that the output can seem disjoint, cold and robotic. This could be fixed, if the filter was capable of understand basic semantics, and filling in conjunctions that make the output more cohesive.

The structure, depending on which industry, may not always be the best suited structure for the job position. If one was to develop further, this will surely have to be more dynamic.

Then there is the problem, that most specialized high level job openings require both a CV and a separate job application. This program will be effectively useless in the cause to aid the job applicant in this process. But this program is much more of a proof of concept, then a ready to go fix all solution. Though most people who are educated enough to apply for such job positions are probably able to write it them selves without much trouble. In the end, the experiment showed that this solution is much more applicable, to those who have a hard time narrowing what skills and abilities should be included. To the people who are generally more desperate and "just need a job", as a lot of unemployed are. In this instance, the product we have designed would be viable.

There is also the problem that the program is restricted to linux, which is not ideal, seeing as a larger majority of people use Windows. Even if you have linux, and follow the steps outlined in the readme file, then it's still a hazel to actually make it work, if one doesn't have a technical background. If the program had a GUI instead, or run on a website such that no extra programs needed to be installed, then it might be viable for deployment. Until these things have been fixed, then the program is only viable as a niche solution, as it's to underdeveloped to catch mainstream success.

8. Conclusion

This project's overarching objective was to aid The United States and The United Kingdom unemployed in obtaining a job by means of CV. The target group of the unemployed was chosen due to the findings that unemployment had a negative effect on the economy of the US and the UK. Furthermore, it was found that unemployment drastically reduces the likelihood of obtaining a job as time goes on. Therefore, to aid the unemployed in obtaining a job, the process behind applying and obtaining a job by CV was discussed. It was determined that there were five stages to obtaining a job and that the two most paramount stages were the ATS scanner and hiring personnel. Therefore, the three elements that the ATS scanner scans for in a CV and the three concepts that would pass the hiring personnel´s "7-second test".

Using our previous findings, the factors which determined the likelihood of an unemployed obtaining a job were found. These factors were then analyzed and it was found that the most productive method of helping the unemployed was to optimize some of these factors. The factors which optimized would benefit the unemployed most were found to be; sending a large quantity of CVs, passing the ATS scanner and passing the hiring personnel.

To optimize sending a large quantity of CVs, it was determined that using a software solution would be ideal. To optimize passing the ATS scanner, keywords would be incorporated into the unemployed applicant's CV, the CV headers would be correctly named and the CV would be produced as a PDF by LaTeX. To optimize passing the hiring personnel, the CV would be aptly structured and streamlined. A software solution would further assist in systematically incorporating the optimizations into a CV, to pass the ATS scanner and hiring personnel.

These optimizations became a C program where a user, likely an unemployed applicant, would write a "Long CV" consisting of all their qualifications. The user would thereafter use keywords to streamline the Long CV into a relevant CV. The users would also write their work experience and education and this would be structured efficiently and lastly be converted into a PDF, ready to be used in the unemployed's application process. Simply put, this project objective became a program that automatically generates a digital CV optimized for the US and UK unemployed.

This project's digital CV was thereafter tested by a job counseling expert, which should simulate the hiring personnel. A CV produced by the program was compared to a CV written entirely by human hands. The feedback for the program passing the hiring personnel was good, but still lacking. This was because the program's free text lacked a smooth transition. Though, the content of the CV was a drastic improvement. Consequently, the project's problem statement was not fully met. The program's CV had the same success rate at passing the hiring personnel as a CV written by human hands. However, the lack of experimentation with the ATS scanner and the quantity of CVs produced leaves an indecisive result. Further experimentation would have to be done to conclude the effectiveness of the project's digital CV.

9. Reflection

There may be some ethical concerns surrounding the use of automation in regards to job applications. Will it make us less honest? Will it create an unfair advantage?
As outlined in the software section of the analysis: Aiding one's ability to be productive, is not unethical, as long as it's not lying. Just as having the ability to use spell checking, didn't make us more unethical as a species, but to the contrary: More productive. It won't give an unfair advantage, as it's a means to be more productive, not a means to create a CV of greater quality. If one's skills are bad, the program won't help much. In fact, we would argue that it levels out the playing field, since those who may need our program, are those who have the least time, but maybe just as qualified. Again, the program is a tool, not a magic solution.

One might say, that creating unethical means of getting an interview encourages unethical means of filtering people out. But the encouragement to use unethical means of filtering people out already exists: It's called profit and cost management. By the very nature of capitalism, the companies are encouraged to be unethical. If one wishes to advance, we need not only successful companies, but also successful people. Adding unto this, we need not the few to be successful, we need the society at large to be successful. This program encourages not unethical behavior, but more so equality of opportunity. Those qualified people that are bad at applying, may still end up qualifying for the same position as those whose specialty is the skill in applications.

References

[1] BRIAN BARNIER, ed. *What is unemployment.* English. Investopedia. Apr. 3, 2020. URL: https://www.careeraddict.com/best-cv-formats (visited on 11/25/2020).

[2] U.S. BUREAU OF LABOR STATISTICS, ed. *How the Government Measures Unemployment.* English. U.S. BUREAU OF LABOR STATISTICS. Oct. 8, 2015. URL: https://www.bls.gov/cps/cps_htgm.htm#concepts (visited on 11/25/2020).

[3] Sherif O S, ed. *4 Types Of Unemployment In Economics.* English. USA American. Nov. 20, 2020. URL: https://usabyt.com/4-types-of-unemployment-in-economics/ (visited on 11/25/2020).

[4] Jodi Beggs, ed. *Understanding the 4 Basic Types of Unemployment.* English. Thought Co. July 14, 2020. URL: https://www.thoughtco.com/types-of-unemployment-in-economics-1148113 (visited on 11/25/2020).

[5] Nitisha, ed. *Classical Theory of Employment.* English. Economics Discussion. Nov. 25, 2020. URL: https://www.economicsdiscussion.net/employment-theories/classical-theory-of-employment-with-diagram/3981 (visited on 11/25/2020).

[6] Foreign Policy News, ed. *United but different Differences and similarities between the US and the UK.* English. Foreign Policy News. 2017. URL: https://foreignpolicynews.org/2017/08/04/united-different-differences-similarities-us-uk/ (visited on 11/27/2020).

[7] Mike Moffat, ed. *A Beginner's Guide to Economic Indicators.* English. ThoughtCo. Feb. 19, 2018. URL: https://www.thoughtco.com/beginners-guide-to-economic-indicators-1145901 (visited on 12/08/2020).

[8] Macrotrends, ed. *US Unemployment Rate 1991-2020.* English. Macrotrends. 2020. URL: https://www.macrotrends.net/countries/USA/united-states/unemployment-rate (visited on 11/27/2020).

[9] Macrotrends, ed. *U.S. GDP 1947-2020.* English. Macrotrends. 2020. URL: https://www.macrotrends.net/countries/USA/united-states/gdp-gross-domestic-product (visited on 11/27/2020).

[10] Macrotrends, ed. *U.K. Unemployment Rate 1991-2020.* English. Macrotrends. 2020. URL: https://www.macrotrends.net/countries/GBR/united-kingdom/unemployment-rate (visited on 11/27/2020).

[11] Macrotrends, ed. *U.K. GDP Growth Rate 1961-2020.* English. Macrotrends. 2020. URL: https://www.macrotrends.net/countries/GBR/united-kingdom/gdp-growth-rate (visited on 11/27/2020).

[12] WILL KENTON, ed. *Okun Law.* English. Investopedia. Mar. 6, 2020. URL: https://www.investopedia.com/terms/o/okunslaw.asp (visited on 11/30/2020).

[13] ELVIS PICARDO, ed. *How Inflation and Unemployment Are Related*. English. Investopedia. Apr. 19, 2019. URL: https://www.investopedia.com/articles/markets/081515/how-inflation-and-unemployment-are-related.asp (visited on 11/30/2020).

[14] BCcampus, ed. *25.3 THE PHILLIPS CURVE*. English. BCcampus. Nov. 28, 2020. URL: https://opentextbc.ca/principlesofeconomics/chapter/25-3-the-phillips-curve/ (visited on 11/30/2020).

[15] CFI, ed. *Unemployment*. English. 2020. URL: https://corporatefinanceinstitute.com/resources/knowledge/economics/unemployment/ (visited on 12/08/2020).

[16] Gregor Jarosch and Laura Pilossoph, eds. *The Reluctance of Firms to Interview the Long-Term Unemployed*. English. Federal Reserve Bank of New York. Aug. 3, 2016. URL: https://libertystreeteconomics.newyorkfed.org/2016/08/the-reluctance-of-firms-to-interview-the-long-term-unemployed.html#.V6haDrgrKUl (visited on 11/30/2020).

[17] Gregor Jarosch and Laura Pilossoph, eds. *Statistical Discrimination and Duration Dependence in the Job Finding Rate**. English. NATIONAL BUREAU OF ECONOMIC RESEARCH. Jan. 2018. URL: https://www.nber.org/system/files/working_papers/w24200/w24200.pdf (visited on 11/30/2020).

[18] Lou Adler. *New Survey Reveals 85 procent of All Jobs are Filled Via Networking*. English. linkedin. Feb. 29, 2016. URL: https://www.linkedin.com/pulse/new-survey-reveals-85-all-jobs-filled-via-networking-lou-adler/ (visited on 10/17/2020).

[19] indeed, ed. *Get Your Resume Seen With ATS Keywords*. English. July 7, 2020. URL: https://www.indeed.com/career-advice/resumes-cover-letters/ats-resume-keywords (visited on 11/13/2020).

[20] ALISON DOYLE, ed. *A Step-by-Step Guide to the Job Application Process*. English. The Balance Careers. Nov. 28, 2019. URL: https://www.thebalancecareers.com/job-application-process-2061600 (visited on 11/30/2020).

[21] Kushal Chakrabarti. *How Long Does It Take to Get a Job in America Today? 84.3 Days for HR*. English. TalentWorks. Sept. 22, 2017. URL: https://talent.works/2017/09/22/how-long-does-it-take-to-get-a-job-60-days-if-youre-in-hr-or-sales/ (visited on 10/17/2020).

[22] Tyler Schmall. *It Takes 5 Months of Searching to Land a Job, Study Finds*. English. SWNS digital. Oct. 16, 2018. URL: https://www.swnsdigital.com/2018/10/it-takes-5-months-of-searching-to-land-a-job-study-finds/ (visited on 10/17/2020).

[23] The Guardien Jobs, ed. *Should you use one generic CV?* English. Aug. 17, 2017. URL: https://jobs.theguardian.com/article/should-you-use-one-generic-cv-/ (visited on 12/08/2020).

[24] GRAEME JORDAN, ed. *HOW MUCH TIME SHOULD I SPEND WRITING MY CV?* English. Social Hire. 2020. URL: https://social-hire.com/blog/candidate/how-much-time-should-i-spend-writing-my-cv (visited on 12/07/2020).

[25] Leslie Stevens-Huffman. *Best Times and Days to Submit Your Résumé and Job Application*. English. insights dice. Oct. 30, 2019. URL: https://insights.dice.com/2019/10/30/best-times-days-submit-resume/ (visited on 10/17/2020).

[26] Jobfinder, ed. *Her er det bedste tidspunkt at sende din ansøgning*. Danish. Jan. 8, 2018. URL: https://www.jobfinder.dk/artikel/her-bedste-tidspunkt-at-sende-din-ansoegning/220987 (visited on 10/18/2020).

[27] Kushal Chakrabarti. *The Science of The Job Search, Part I: 13 Data-Backed Ways To Win*. English. Talent works. Jan. 8, 2018. URL: talent.works/2018/01/08/the-science-of-the-job-search-part-i-13-data-backed-ways-to-win/ (visited on 10/18/2020).

[28] Social Hire, ed. *Applicant Tracking System (ATS) Preparation and Information*. English. 2020. URL: https://www.jobtestprep.com/how-to-beat-applicant-tracking-system (visited on 12/08/2020).

[29] indeed, ed. *ATS -Applicant Tracking Systems – endeligt farvel til mennesket i rekrutteringsprocessen?* Danish. July 9, 2019. URL: https://www.linkedin.com/pulse/ats-applicant-tracking-systems-endeligt-farvel-til-i-allan-nielsen (visited on 12/16/2020).

[30] Job Test Prep, ed. *Beat Oracle's Taleo Applicant Tracking System (ATS)*. English. 2020. URL: https://www.jobtestprep.com/taleo-applicant-tracking-system (visited on 12/08/2020).

[31] Dave Rietsema, ed. *How Does an Applicant Tracking System Work?* English. Applicant Tracking Systems. 2020. URL: https://www.applicanttrackingsystems.net/how-applicant-tracking-systems-work/ (visited on 12/08/2020).

[32] Tallulah David, ed. *23 Surprising Stats on Candidate Experience – Infographic*. English. CareerARC. May 14, 2016. URL: https://www.careerarc.com/blog/2016/06/candidate-experience-study-infographic/ (visited on 12/08/2020).

[33] Rebecca Koenig. *The Perks of Professional Organizations. Industry societies help workers learn, network and gain credibility*. English. USnews. Sept. 5, 2018. URL: https://www.glassdoor.co.uk/blog/7-second-cv/ (visited on 10/19/2020).

[34] Emily Moore, ed. *Your CV Has 7 Seconds to Get a Recruiter's Attention — Here's How to Make It Count*. English. Glassdoor. Mar. 15, 2019. URL: https://www.glassdoor.co.uk/blog/7-second-cv/ (visited on 11/14/2020).

[35] Meredith Lepore, ed. *How many words should a resume be?* English. Ladders. Oct. 18, 2020. URL: https://www.theladders.com/career-advice/this-is-the-exact-number-of-words-your-resume-should-be (visited on 12/08/2020).

[36] Chris Leitch, ed. *Size Matters: The Ideal Length of Your CV*. English. CareerAddict. Nov. 16, 2017. URL: https://www.careeraddict.com/cv-length (visited on 12/08/2020).

[37] Alison Doyle. *The Difference Between a Resume and a Curriculum Vitae*. English. The Balance Careers. July 30, 2020. URL: https://www.thebalancecareers.com/cv-vs-resume-2058495 (visited on 10/18/2020).

[38] Andrei Kurtuy. *How to Write a CV (Curriculum Vitae) in 2020 [31+ Examples]*. English. The Balance Careers. June 3, 2020. URL: https://novoresume.com/career-blog/how-to-write-a-cv (visited on 10/18/2020).

[39] Kushal Chakrabarti. *The Definitive Guide to the Science of the Job Search — 2018 In Review*. English. TalentWorks. Dec. 28, 2018. URL: https://talent.works/?utm_source=blog (visited on 10/17/2020).

[40] Michael Tomaszewski. *What to Include in a CV: 6 Must-Have Sections to Put on a CV in 2020*. English. Zety. Sept. 9, 2020. URL: https://zety.com/blog/what-to-include-in-a-cv (visited on 10/17/2020).

[41] Doug Levin. *Professional Associations and Organizations. List of Professional Associations and Organizations by Industry*. English. Jobstars. June 30, 2016. URL: https://jobstars.com/professional-associations-organizations/ (visited on 10/18/2020).

[42] Rebecca Koenig. *The Perks of Professional Organizations. Industry societies help workers learn, network and gain credibility*. English. USnews. Sept. 5, 2018. URL: https://money.usnews.com/careers/company-culture/articles/2018-09-05/the-perks-of-professional-organizations (visited on 10/19/2020).

[43] indeed, ed. *Resume Format Guide: Tips and Examples of the Best Formats*. English. Nov. 12, 2020. URL: https://www.indeed.com/career-advice/resumes-cover-letters/resume-format-guide-with-examples (visited on 11/13/2020).

[44] Ladders, ed. *Eye Tracking Study*. English. 2018. URL: https://www.theladders.com/static/images/basicSite/pdfs/TheLadders-EyeTracking-StudyC2.pdf (visited on 12/08/2020).

[45] Helen Tupper, ed. *How to navigate the four phases of your job life cycle*. English. xeim. June 25, 2018. URL: https://www.marketingweek.com/helen-tupper-manage-your-job-life-cycle/ (visited on 12/08/2020).

[46] Rick Yvanovich, ed. *You Probably Already Know Most Job Interviews Are Unreliable. But Why?* English. trg. Sept. 7, 2020. URL: https://blog.trginternational.com/most-job-interviews-are-unreliable (visited on 12/08/2020).

[47] indeed, ed. *125 Common Interview Questions and Answers (With Tips)*. English. Dec. 4, 2020. URL: https://www.indeed.com/career-advice/interviewing/top-interview-questions-and-answers (visited on 12/08/2020).

[48] resume, ed. *PDF vs. Word: What's the Best Way to Submit Your Resume?* English. Mar. 6, 2020. URL: https://www.resume.com/career-advice/resumes/resume-pdf-or-word/ (visited on 11/13/2020).

[49] IT-universitet, ed. *Hvordan kommer man fra vandfald til agil UX?* Danish. June 22, 2017. URL: https://www.itu.dk/om-itu/presse/nyheder/2017/hvordan-kommer-man-fra-vandfald-til-agil-ux (visited on 12/03/2020).

[50] Anonymous, ed. *personal pronouns*. English. Wikipedia. Nov. 11, 2020. URL: https://en.wikipedia.org/wiki/English_personal_pronouns (visited on 11/11/2020).

[51] N/A, ed. *Levenshtein distance algorithm*. English. Wikipedia. Nov. 11, 2020. URL: https://rosettacode.org/wiki/Levenshtein_distance#C (visited on 11/11/2020).

[52] arvinpdmn, ed. *Levenshtein Distance*. English. Apr. 9, 2019. URL: https://devopedia.org/levenshtein-distance#:~:text=The%20Wagner%2DFischer%20Algorithm%20is,coined%20by%20Wagner%20and%20Fischer (visited on 12/07/2020).

[53] Wikipedia, ed. *Levenshtein Distance Definition - illustration*. English. Nov. 15, 2020. URL: https://en.wikipedia.org/wiki/Levenshtein_distance (visited on 12/07/2020).

[54] Institute of education science, ed. *English endings*. English. Wikipedia. 2020. URL: https://ies.ed.gov/ncee/edlabs/regions/southeast/foundations/resources/thirdgrade/rec3/3.3_Act_10_Building_Words_With_Grammatical_Endings.pdf (visited on 11/11/2020).

[55] deepai.org, ed. *Q-sort - illustration*. English. 2020. URL: https://deepai.org/machine-learning-glossary-and-terms/quicksort-algorithm (visited on 12/08/2020).

[56] resumeworded, ed. *ATS scanner website*. English. 2020. URL: https://resumeworded.com/resume-scanner (visited on 12/18/2020).

A. appendix

Arbejdsdagen

§1. Hver arbejdsdag startes med scrum-statusmøde, en dagsorden og valg af referent.

§2. Projektmål og -struktur revideres hver arbejdsdag.

§3. Alle deltager i alle samtaler om projektrelevante emner.

§4. Små opgaver kan uddelegeres til enkelte medlemmer, men alle skal have det fulde overblik.

§5. Visuelle hjælpemidler benyttes i videst mulige omfang – eksempelvis benyttes whiteboardet ved alle gruppemøder og -diskussioner.

Faglige forventninger

§1. Alle medlemmer skal vide, hvad gruppen laver.

§2. Faglige diskussioner holdes indenfor emnet.

§3. Visuelle forklaringer benyttes om muligt.

§4. Alle skal deltage nogenlunde ligeligt i såvel programmering som rapportskrivning.

§5. Aftalt hjemmearbejde skal overholdes.

§6. Produktet skal gennemgås og afleveres gennemarbejdet.

§7. Kode skal kommenteres.

§8. I udgangspunktet forventes timer brugt svarende til normeringen (ca. 20 timer om ugen for P1), og denne forventning kan om nødvendigt øges nær deadline.

Sociale forventninger

§1. Gruppens primære kontakt foregår gennem Messenger.

§2. Alle skal tjekke Messenger (mindst) dagligt.

§3. Gruppen mødes på campus alle hverdage, med mindre andet er aftalt.

§4. Alle møder på det aftalte tidspunkt, og der gives besked, hvis man er mere end et kvarter forsinket.

§5. Alle lytter til og respekterer hinandens person og meninger.

§6. Konflikter og uenighed løses ved demokratisk afstemning i gruppen.

Fravær

§1. Alle skal i udgangspunktet møde til alle forelæsninger og møder.

§2. Bliver man syg, eller har man anden god grund til fravær, gives der besked i rimelig tid.

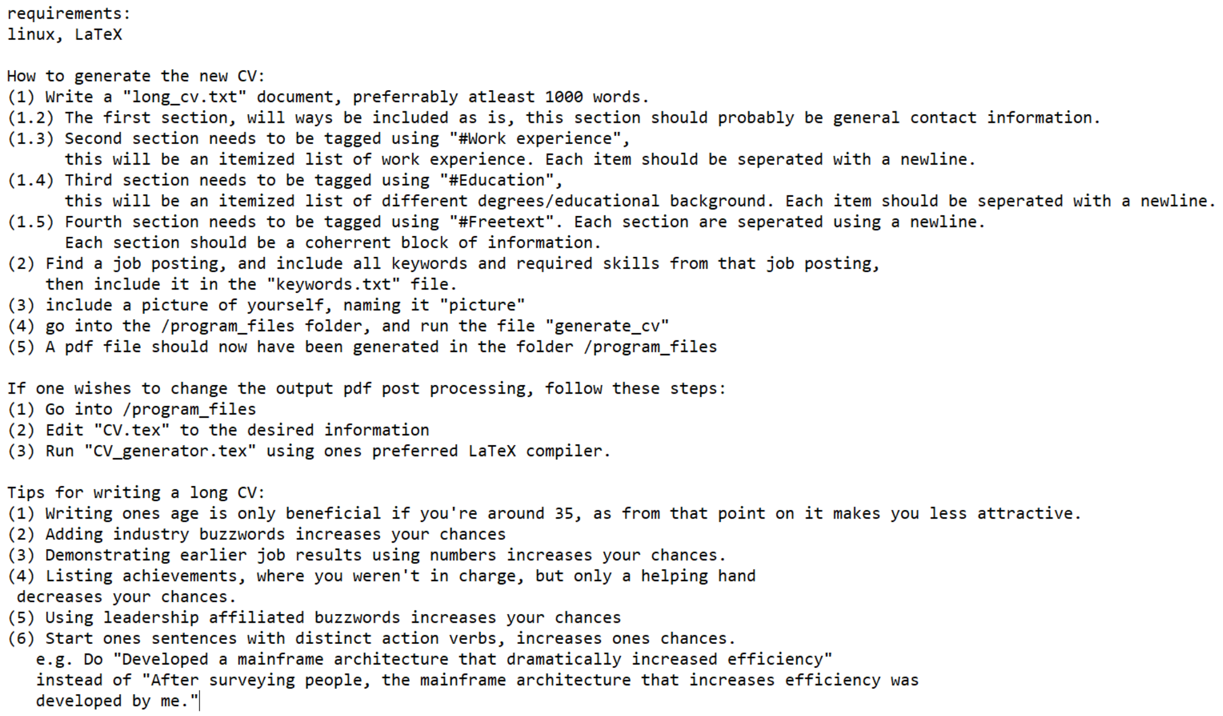
Konsekvenser

§1. Gruppen kan når som helst stemme om, hvorvidt et gruppemedlem skal tildeles en advarsel.

§2. Uddeling af advarsel kræver almindeligt flertal blandt gruppens øvrige medlemmer.

§3. Er et gruppemedlem tildelt en advarsel, kan gruppen efter yderligere overtrædelser når som helst stemme om, hvorvidt gruppemedlemmet skal ekskluderes.

§4. Ekskludering kræver enstemmighed blandt gruppens øvrige medlemmer.



```
requirements:
linux, LaTeX

How to generate the new CV:
(1) Write a "long_cv.txt" document, preferrably atleast 1000 words.
(1.2) The first section, will ways be included as is, this section should probably be general contact information.
(1.3) Second section needs to be tagged using "#Work experience",
      this will be an itemized list of work experience. Each item should be seperated with a newline.
(1.4) Third section needs to be tagged using "#Education",
      this will be an itemized list of different degrees/educational background. Each item should be seperated with a newline.
(1.5) Fourth section needs to be tagged using "#Freetext". Each section are seperated using a newline.
      Each section should be a coherent block of information.
(2) Find a job posting, and include all keywords and required skills from that job posting,
    then include it in the "keywords.txt" file.
(3) include a picture of yourself, naming it "picture"
(4) go into the /program_files folder, and run the file "generate_cv"
(5) A pdf file should now have been generated in the folder /program_files

If one wishes to change the output pdf post processing, follow these steps:
(1) Go into /program_files
(2) Edit "CV.tex" to the desired information
(3) Run "CV_generator.tex" using ones preferred LaTeX compiler.

Tips for writing a long CV:
(1) Writing ones age is only beneficial if you're around 35, as from that point on it makes you less attractive.
(2) Adding industry buzzwords increases your chances
(3) Demonstrating earlier job results using numbers increases your chances.
(4) Listing achievements, where you weren't in charge, but only a helping hand
 decreases your chances.
(5) Using leadership affiliated buzzwords increases your chances
(6) Start ones sentences with distinct action verbs, increases ones chances.
   e.g. Do "Developed a mainframe architecture that dramatically increased efficiency"
   instead of "After surveying people, the mainframe architecture that increases efficiency was
   developed by me."
```

Figure 25.: Read Me file - The tips are based on the background section "Optimize Interview rate"

A.1. Link to program code

Link til github program: https://github.com/Hans-EH/Semester-1-cv-filter