# Job post scanner

Pablo Lizcano Sánchez-Cruzado

## Introduction

Understanding if a certain position is what we are looking for and if we are a good fit for the company is the first step to get a new job. If the outcome of this first analysis is positive, i.e. we have found a good position for which we are a good match, we can proceed with the second step. The second second is to write a cover letter where we highlight the experience and skills which are relevant to that position. Instead of writing a cover letter, we sometimes have to answer a series of questions to send our application. In any case, during this stage and the ones that follow it, one must keep in mind what both parties (the company and the applicant) are looking for.

Finding a new job can be exhausting: reading job posts, filtering the ones we are really interested in, writing cover letters or answering questions on the application phase, to re-read the descriptions if we are called for an interview and enlist our doubts, sitting a test or doing an exercise to check our hard skills… and that's why I thought it could be a good project to automate a part of the process. **This project is a proof of concept for a job post scanner which extracts and organises all the information contained in a job description.** For this POC, I will focus on jobs related to data science (data scientist, data analyst, machine learning engineer, NLP practitioners…), but I guess it would be easily generalizable to other technical positions.

## Relevant information about a job

There are some questions I try to answer while reading a job post:

1. About the company:
    a. Company name?
    b. Sector (Fintech, proptech…)?
    c. Product or service offered (consulting services, predicting analysis software…)?
    d. How many customers and users (if they are different) do they have?
    e. How many employees?
    f. What is its position in the market? How much are growing?
2. About the position:
    a. General information:
        i. Kind of job (IT developer, manager)? Job title (data scientist, data engineer…)?
        ii. Job location (country or town)?
        iii. Team composition: size, multidisciplinar or homogeneous?
    b. Requirements (must and nice to have):
        i. Required work experience?
        ii. Required education?

        iii. Required technologies?
            1. Programming languages (Python, R), libraries and frameworks (TensorFlow, PyTorch)
            2. Disciplines (Statistics, Machine Learning, Deep Learning, Databases, specific domain knowledge)
            3. Cloud computing platform knowledge (AWS, Google Cloud)
            4. Other technologies (Docker, Git, Tableau)
        iv. Required soft skills?
        v. Required languages?
    c. What is offered
        i. Type of employment (full-time, part-time…)?
        ii. Remote working and flexibility?
        iii. Salary?
        iv. Holidays?
        v. Benefits?

Once I have obtained that information, I am fully able to answer some questions which determine if I am a good fit for that company, and vice versa:

1. Does the company sector interest me ? Do I like that kind of company? (1)
2. Can I move to the job location? Would I be happy to work with that team? (2.a)
3. Do I meet the requirements or can I meet them in little time after starting working there? Am I willing/interested in the tasks I will have to carry out in this position? Are there tasks or technologies I would like to learn in the near future which are not included in the job description? (2.b)
4. Does their offer meet my expectations? (2.c)

Rarely do we find such a detailed description so we will have to ask some questions during the hiring process.

## Linkedin job offers and company pages

Job offers on Linkedin always have the same format. On top, we have the job title (2.a.i), followed by the name of the company (1.a) and the town (2.a.ii).

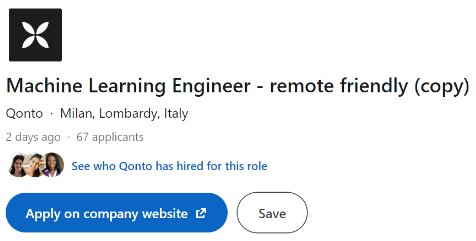
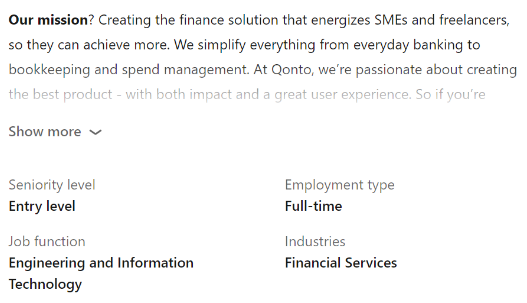

Machine Learning Engineer - remote friendly (copy)
Qonto · Milan, Lombardy, Italy
2 days ago · 67 applicants
See who Qonto has hired for this role

Figure 1: Job header (job page)

On the section below, we can find the job description, the seniority level (2.b.i), the employment type (2.c.i), the job function (2.a.i) and the industry (1.b).



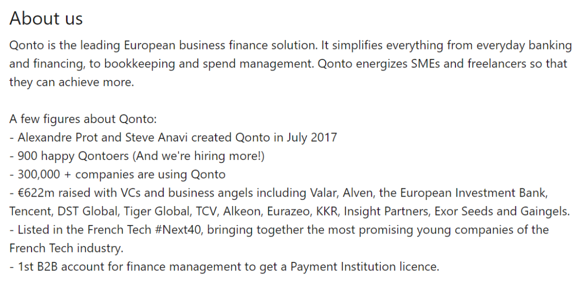
**Our mission**? Creating the finance solution that energizes SMEs and freelancers, so they can achieve more. We simplify everything from everyday banking to bookkeeping and spend management. At Qonto, we're passionate about creating the best product - with both impact and a great user experience. So if you're

Show more

| | |
|---|---|
| Seniority level<br>Entry level | Employment type<br>Full-time |
| Job function<br>Engineering and Information Technology | Industries<br>Financial Services |

Figure 2: Job description and some details (job page)

If we want to know more about the company, we can click on the logo in order to access the company page. In this company page, we can find a general description, which hopefully may help us answer the questions 1.c, 1.d and 1.f.



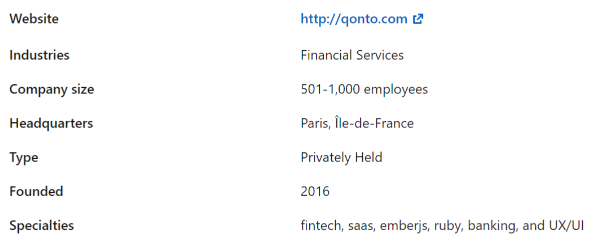
About us

Qonto is the leading European business finance solution. It simplifies everything from everyday banking and financing, to bookkeeping and spend management. Qonto energizes SMEs and freelancers so that they can achieve more.

A few figures about Qonto:
- Alexandre Prot and Steve Anavi created Qonto in July 2017
- 900 happy Qontoers (And we're hiring more!)
- 300,000 + companies are using Qonto
- €622m raised with VCs and business angels including Valar, Alven, the European Investment Bank, Tencent, DST Global, Tiger Global, TCV, Alkeon, Eurazeo, KKR, Insight Partners, Exor Seeds and Gaingels.
- Listed in the French Tech #Next40, bringing together the most promising young companies of the French Tech industry.
- 1st B2B account for finance management to get a Payment Institution licence.

Figure 3: Company description (company page)

In the company page we can also find other relevant information such as the company size (1.e) and the specialties (related to 1.c).



| | |
|---|---|
| Website | http://qonto.com |
| Industries | Financial Services |
| Company size | 501-1,000 employees |
| Headquarters | Paris, Île-de-France |
| Type | Privately Held |
| Founded | 2016 |
| Specialties | fintech, saas, emberjs, ruby, banking, and UX/UI |

Figure 4: Company details (company page)

There are some questions which haven't been matched with a specific section of these pages. We may find the answer in the job description (job page).

## Extracting relevant information

The goal of this project is to understand if it is possible to extract and organise the information contained in a Linkedin job offer. Since I don't have data to train an intelligent model which does it automatically and my time to generate this data is limited, I will analyse the resources which are already up to use. The solution may consist of a combination of these resources.

As we could see before, some information is organised in a structured way on Linkedin job offers and company pages.

## Tools

The programming language Python will be used in this POC along with a variety of libraries.

- Selenium is used to open the webpages in a navigator and find all the elements mentioned above: job title, job location, job description...
- BeautifulSoup allows making the text contained in the job description plained, without HTML tags, while keeping the structure (line breaks, bullets…).
- HuggingFace pretrained models related to question answering and token classification (part-of-speech and entity recognition tasks).
- Spacy NLP pipelines also perform part-of-speech and entity recognition tasks.
- Regular expressions identify patterns of characters in a text.

## Methodology

There are 9 questions we can directly answer by extracting the right fields from the job page and the company page. **Selenium** is enough to accomplish this task.

| Information | Field |
|---|---|
| 1.a | Company name (job post) |
| 1.b | Industries (job post) |
| Sometimes 1.c | Specialties (company page) |
| 1.e | Company size (company page) |
| 2.a.i | Job title and function (job post) |
| 2.a.ii, sometimes also 2.c.ii | Job location (job post) |
| 2.b.i | Seniority level (job post) |
| 2.c.i | Employment type (job post) |

After answering these nine (or seven, in the worst case) questions, twelve (or ten) questions remain unanswered:

- We may be able to find the answers to the rest of the questions about the company (1.c, 1.d and 1.f) in the company description (company page).
- The other questions about the position (2.x.x) may be answered with the information contained in the job description (job post).

Before using NLP models, the text in the descriptions should be plain, without HTML tags. The raw descriptions are also extracted with **Selenium**, and their format is cleaned with the **BeautifulSoup** library: only characters and punctuation marks are allowed. In order to automatically find the answers in these texts, we will use pre-trained **NLP models** (Spacy, HuggingFace). **Regular expressions** can be of use if the previous solution fails to achieve a certain level of performance.

## Question answering with HuggingFace models

In this section, I will try to get the answers for the remaining questions with the Question Answering models in the HuggingFace library. On 1st November, the most popular model is deepset/roberta-base-squad2.

This pre-trained model takes as inputs a question and a context. In our case, the context is the job post or the company page, and the question must be a complete interrogative sentence. For example, the question 1.c can be “What is the company’s product?”, “What has the company created?”, or “What does the company do?”.

The model returns the answer, which consists of a substring of the description string, and a score. For the previous example, the answers are, respectively, 'Qonto is the leading European business finance solution' (score: 0.068), '900 happy Qontoers' (score: 0.004), 'simplifies everything from everyday banking and financing, to bookkeeping and spend management' (score:  0.14). In this case, the model has found sensible answers for the first and the last question. The scores are coherent: the question for which the model has returned a wrong answer has the lowest score.

These are the results for the post analysed:

- The model has output acceptable answers for some interrogative sentences. These interrogative sentences rephrase questions 1.c (product), 1.d (number of customers or users), 2.b.v (required languages), 2.c.iv (holidays) and 2.c.v (benefits) had scores over 0.05. There were other interrogative sentences whose corresponding output was acceptable too, even if the score was under 0.05: 1.f (company growth and position in the market), 2.b.iii (required technologies), 2.b.iv (required soft skills).
- The model has output wrong answers for some other interrogative sentences. Most of them had scores under 0.05: 2.a.iii (team composition), 2.b.iii (required technologies), 2.c.ii (remote or in-office position), 2.c.iii (salary). The model output a wrong answer for an interrogative sentence rephrasing question 2.b.iii (programming languages) with score over 0.05.
- The answer to one question couldn’t be found in this text: 2.b.i (required education). The output of the model had a score under 0.05 for this question.

We may use the threshold of 0.05 for this use case: if the score is over 0.05, the output is trustworthy. Otherwise, it is doubtful.

## Named entity recognition with Spacy

Spacy pipelines perform named entity recognition with multiple categories: organisation, money, product, date, time, percent, language, etc. I have checked if it was possible to identify programming languages (Python), libraries (Pytorch), disciplines (Machine Learning, Data Scientist), or even the salary in the text.

I have chosen the pipeline "en_core_web_lg". The NER module has found multiple organisations which were cited in the text: Apple, WeWork… The problem is that it doesn’t have specific categories for disciplines or programming languages. The libraries Pandas and

Tensorflow are classified as products. Pytorch, Data Science or Machine Learning are considered to be an organisation.

It is possible to retrain a part of the pipeline to include programming languages. There is a tutorial on this task. Using regular expressions seems faster in this case.

The NER module can be very useful to identify money quantities, cardinal and ordinal numbers, time quantities which may answer the questions 2.d (number of customers), 2.f (position in the market), 2.a.iii (team size), 2.c.iii (salary) and 2.c.iv (holidays). It also identifies human languages, which is useful to answer the question 2.b.v.

## Regular expressions

By using regular expressions we can easily select sentences or lines which contain

- proper nouns referring to specific software (programming languages, libraries, disciplines, platforms…);
- disciplines (Statistics, Machine Learning, Deep Learning, Databases);
- soft skills;
- words referring to a certain level of instruction (degree, certificate);
- typical words like salary, insurance, benefits, stock options, flexible or flexibility, remote.

Those lines probably contain the information needed to answer the questions 2.b.ii (education), 2.b.iii (technologies), 2.b.iv (soft skills), 2.c.ii (remote working and flexibility), 2.c.iii (salary), 2.c.iv (holidays) and 2.c.v (benefits).

# Conclusions

The **HuggingFace pre-trained model does not have a performance high enough** to be used alone for this specific task: it has been trained for more general purposes. Generating data to fine tune the model may be very expensive. It would be necessary

In the meanwhile, we can take advantage of simpler approaches to help the user find information faster. A **web app** could be built **to help the user organise the information in a job offer**. A chatbot could ask those questions whose answer may only be found in the job or company description, and the **lines which probably contain the answer could be automatically highlighted**. These lines are determined with the Spacy module for **named entity recognition** and/or **regular expressions** (you can find the summary below). If the **HuggingFace model output has a high score**, it can be used to highlight potentially important lines. In this POC we have seen that the threshold of 0.05 may divide trustful and doubtful results. The experiment should be repeated on many other job posts, with a defined process, before establishing a “definitive” threshold. After gathering the data, **the app** can help the user write their **cover letter** by providing them **with a preliminary version**. At some point, users will have **generated enough data to fine tune the HF model**.

1. About the company:
    a. Company name? Simple Field
    b. Sector (Fintech, proptech…)? SF

   c. Product or service offered (consulting services, predicting analysis software…)? SF
   d. How many customers and users (if they are different) do they have? NER
   e. How many employees? SF
   f. What is its position in the market? How much are growing? NER
2. About the position:
   a. General information:
      i. Kind of job (IT developer, manager)? Job title (data scientist, data engineer…)? SF
      ii. Job location (country or town)? SF
      iii. Team composition: size, multidisciplinar or homogeneous? NER
   b. Requirements (must and nice to have):
      i. Required work experience? SF
      ii. Required education? RE
      iii. Required technologies? RE
      iv. Required soft skills? RE
      v. Required languages? NER
   c. What is offered
      i. Type of employment (full-time, part-time…)? SF
      ii. Remote working and flexibility? SF, RE
      iii. Salary? NER, RE
      iv. Holidays? NER, RE
      v. Benefits? RE

If it is necessary to include **job posts in different languages**, we would have to add a step where the text is translated. I recommend the **DeepL API**, which works quite well for Western European languages (Portuguese, Spanish, French, Italian, English, German). It handles technical terms from most domains very well.

## Next steps

Before starting hiring someone to build that web app, or studying the market to understand how many people would be interested in such a web app, there are **some other studies** which can be carried out at low cost to estimate the possible performance of a NLP model for this task. We can automatically **scrape Linkedin to obtain the following fields** from a high number of job posts related to Data Science:

I. Simple fields, which allow answering the questions 1.a, 1.b, 1.c, 1.e, 2.a.i, etc
II. Job and company descriptions
III. Features obtained with regular expressions from the job and company descriptions (optional: I would only do it for required technologies and benefits, which can be easily identified).

These are some projects which can be done with such a dataset:

- Measuring the **performance of the pre-trained HuggingFace Question Answering** model when extracting the fields in I and III from the job and company descriptions (II).

- Measuring the **performance of a pre-trained HuggingFace zero-shot classifier** for categorical fields in I (1.b, company sector, or 2.c.i, required work experience).
- **Fine tune those models** and **measure their performance** again on unseen data.
- **Fine tune the Spacy NER** module to identify the features in III, related to required technologies and company benefits. The question for some answers may be splitted in different parts of the text (probably the technologies or the benefits). The Question and Answer model outputs a single span.

The performance obtained for those simple fields may be a good indicator about the maximum performance which can be achieved for information which is only contained in the job and company descriptions.

Error analysis is highly recommended. The information found in those simple fields can contradict the information extracted from the job and company descriptions if the recruiter has filled them incorrectly.