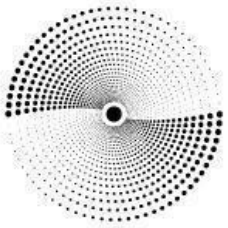
Personalized Recommendation Systems Architecture

Version 0.2

Oct 9, 2022

UNIVERSALDOT FOUNDATION



UNIVERSAL.

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation

UNIVERSAL.

Revision History

Name	Revision comment	Date
Tunahan SARI	Initial draft	09/10/2022
Igor Stojanov	Architecture and improvements	11/10/2022

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation

Introduction

This document proposes an initial design regarding the recommender systems to be used in the UDOT. The initial setup is created based on the requirements listed in SRS.v1.0. Thus, the task and the main problem behind the machine learning task are mainly taken from SRS.v1.0.

Problem Statement

In the requirements file, under the section Data Model, mandatory attributes to consider for both parties, the profile and task, are given. Thus, the matching algorithm should view all of the must-have attributes.

The given attributes for profile are:

- Interests
- Reputation
- Portfolio (To be decided)
- Availability
- Profile History

The given attributes for a task are (* = not mandatory):

- Title
- Requierements
- Budget
- Status (Created, InProgress, Completed, Accepted, Expired)
- Deadline
- Keywords
- Organization*
- Location*

Location*Thus, a pool of tasks will be available through the attributes of tasks and each new task will be added to this pool, and the search will be done based on the attributes of a profile.

On the technical side, based on the requirements file, there are a few must-haves and constraints regarding the machine learning system to be proposed.

- Security, The machine learning model should be depployed on secure servers and all communication happens via SSH an SSL.
- Usability, the errors raised by the model should be easily understandable in natural language.
- Speed, the application should be able to process information relatively fast and reliable. Moreover, upcoming data should be processed and added to the system as fast as possible.
- Optimized Networking, feature selection must be on point so the model will not process unnecessary data.
- Updatebility, the deployment of the application should be automated and should be straightforward.

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation

- Modifiability, the algorithm should adapt to new upcoming scenarios quickly; the selected design pattern must be suitable for such a task.
- Installability, the model should be available for any request, indicating that installment must be reliable on the back-end side.
- Deployment, Integration with TensorFlow Serving must be available.

Proposed Solution

Use of Approximate Nearest Neighbors with Vector Embeddings

Nearest neighbors are already well known for their quickness and scalability in vector similarity searching. Thus, it is expected this method to yield efficient results if the input attributes of a task can be reasonably transformed into vector embeddings. As a result, there are two problems to tackle in this context, the input pre-processing and providing the right nearest neighbor environment.

Nearest Neighbor Model Selection

There are various libraries for nearest-neighbor models. The most known one is from scikit-learn, most probably the most straightforward implementation of initial research on nearest neighbor models. Since this is only meant for academic usage, scikit-learn cannot satisfy more than half of the given must-have requirements. The scikit-learn models are not capable of adapting new incoming features, receiving new data, or yielding a return value in a reasonable time as the number of points or dimensions grows. Thus, another concept of nearest-neighbor, capable of working at an industry level, must be chosen. Provided by TensorFlow, Semantic Textual Similarity Task is represented with Annoy, an Approximate Nearest Neighbour model presented by Spotify for their song search tasks. Annoy can be seen under the list of ANN-Benchmarks, which lists the performance of all the industry-level nearest neighbor models. Even though annoy is performing at the industry-level, namely for Spotify, the requirement of building the dataset when a new point gets involved contradicts with the requirement of scalability of UDOT. Moreover, it is observed that Annoy is a mid-tier ANN algorithm, as shown under the list of ANN-Benchmarks. According to the list ScaNN, fais and hnswlib are the best choices in UDOT's case since the searching environment can be saved, scaled and expanded without building a whole searching ground from scratch, and they are fast enough for many industry-level tasks. ScaNN has the upper hand in getting integrated with Tensorflow serving through custom docker images. In this study, ScaNN is proposed for the searching environment since:

Only one way of searching will be implemented, satisfying the security requirements.

- As google_research states, ScaNN is robust towards miss data fittings, satisfying Usability.
- As seen in the ANN-Benchmarks, ScaNN is fastest in several different tests and in top 5 within the rest, satisfying Speed.
- The whole model does not require to be built again unlike many of industry level ann models and new information can be just added directly to the search environment, satisfying Updatebility.
- The model can be tuned or merged into other models, satisfying Modifiability.

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation

- The model can be created and served through TF serving, satisfying Installability.

Preprocessing

There are multiple steps proposed in the preprocessing system after retrieving the intended data. Step by step, they are:

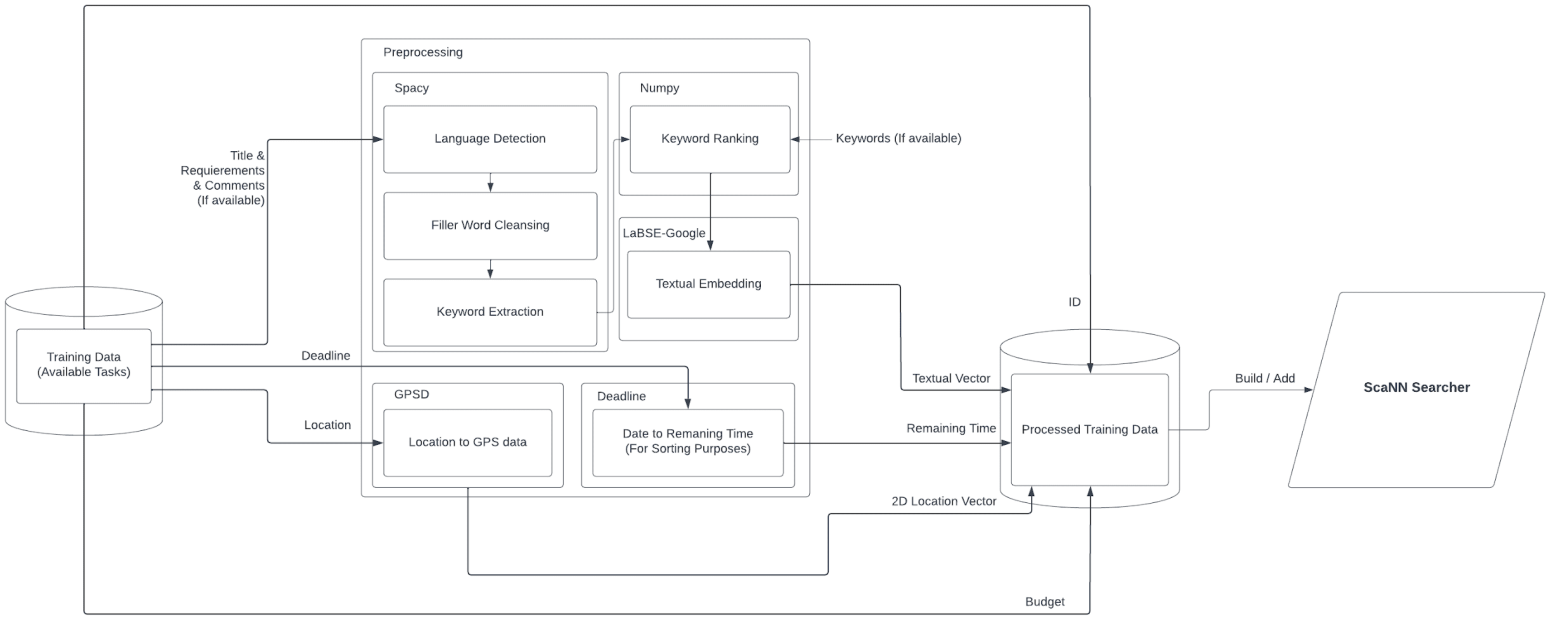
Taking the full text and applying advanced Spacy methods which are;

- Language Detection?: We control the language from the front-end app. Thus it should not be necessary to detect the language since we can provide it as locale string.
- Filler Word Cleansing: Due to the fact that there are so many filler words in the natural languages that we are using, such as "I," "will," "want," "for," "with," the similarity between texts increases, resulting the intended text to appear harder since the similarity scores will be closer than it should be through these filler words that everyone uses in their daily life. Thus, it is proposed here to filter them out to make the differences appear better in keywords and vectors.
- Keyword extraction: Simply extracting keywords and removing words we could not filter out with the filler word cleansing method. It is proposed to extract the key words of a given task.

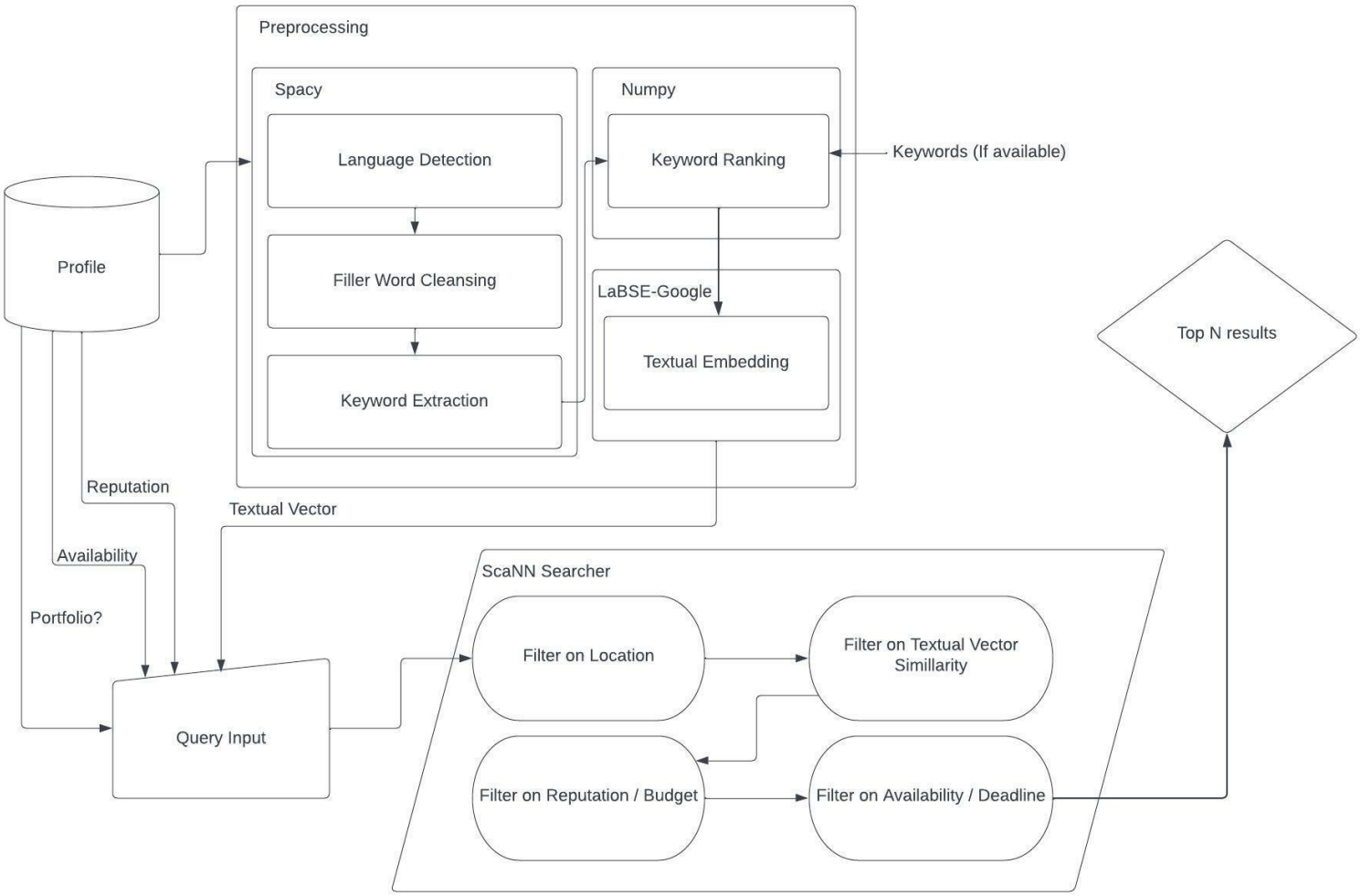
Ranking the importance of keywords: Not every keyword will have the same value in a given context. Thus, they should be considered rightfully to their corresponding value. It is proposed to consider different values of different key words rightfully.

Text to Vector (Sentence Embedding): Since the given task mainly searches for similarities in vectors in a shared n-dimension space, the input must be in the correct form. To do that, BERT-based textual-vector embedding systems are proposed. Based on the given benchmarks and usabilities, LaBSE by google is proposed for its multi-language support while yielding a compelling benchmark score. Thus, an embedder is proposed, and for this task, LaBSE is proposed.

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation



KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation



KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation

UNIVERSAL.

References:

[1] https://github.com/google-research/google-research/tree/master/scann

[2] https://github.com/google-research/google-research/blob/master/scann/tf_serving/README.md

[3] https://ai.googleblog.com/2020/08/language-agnostic-bert-sentence.html

KENNEDYPLEIN 200, 5611 ZT, EINDHOVEN, THE NETHERLANDS
Phone: +31 (0)64 3535 901 | Email: info@universaldot.foundation | https://universaldot.foundation