OPEN ACCESS
International Journal of Semantic Computing
Vol. 17, No. 2 (2023) 1–17

DOI: 10.1142/S1793351X23610019

World Scientific
www.worldscientific.com

A Scalable Recommendation System Approach for a Companies — Seniors Matching

Kévin Cédric Guyard
Information Science Institute, GSEM/CUI, University of Geneva
Route de Drize 7, Carouge 1227, Switzerland
kevin.guyard@unige.ch

Michel Deriaz
Haute Ecole de Gestion (HEG) Genève, HES-SO
Rue de la Tambourine, Carouge 1227, Switzerland
michel.deriaz@hesge.ch

Published 29 March 2023

Recommendation systems are becoming more and more present in our daily lives, whether it is for suggesting items to buy, movies to watch or music to listen. They can be used in a large number of contexts. In this paper, we propose the use of a recommendation system in the context of a recruitment platform. The use of the recommendation system allows to obtain precise profile recommendations based on the competences of a candidate to meet the stated requirements and to avoid companies to have to perform a very time-consuming manual sorting of the candidates. Thus, this paper presents the context in which we propose this recommendation system, the data preprocessing, the general approach based on a hybrid content-based filtering (CBF) and similarity index (SI) system, as well as the means implemented to reduce the computational cost of such a system with the increasing evolution of the platform.

Keywords: Artificial intelligence; machine learning; recruitment; hybrid recommendation system; content-based filtering; similarity index.

1. Introduction

This paper is proposing a recommendation system in the frame of the platform "WisdomOfAge". It is a web platform currently under creation which is aiming to connect senior workers who will be retired or are already retired with companies. On the one hand, seniors fill their resume to describe their skills, and on the other hand, companies can create mission offers. This platform has dual goals. For seniors, this will allow them to stay active and keep a social link. For companies, this will allow them to acquire internally missing skills without hiring or contracting consulting

companies which are often expensive. The recommendation system will be used to generate a list of seniors according to the company's needs when a company is creating a new mission offer on the website.

This frame leads to several specific problems for the recommendation system.

The recommendation system should be operational and able to propose recommendations starting from the first mission offer on the website (with the condition that there is at least one senior registered on the platform corresponding to the need of the company). The system should be scalable and able to self-adjust to the number of users registered on the website.

Another difficulty is the fact that seniors have often a career which has evolved during their professional life. An experience done at the beginning of the career is less relevant than an experience done at the end of the career. The recommendation system should be able to detect if experience/skill is relevant to provide suitable recommendations to companies.

The platform will be available in different languages. Companies and seniors will be matched only if they share the same language but the recommendation system must be able to deal with text from different languages.

2. State of the Art

In their paper [1], Agarwal and Dr. Senthilkumar present a resume-job recommendation system based on similarity index (SI). They have used cosine similarity to measure the similarity between a job offer and a resume to rank resume using the similarity score.

Mentec *et al.* propose in their publication [2] a conversational recommendation system to recommend resumes regarding a job offer. Their approach seems interesting since their system is able to discuss with the recruiter to refine the recommendation according to the requirements of the recruiter. However, their system needs a dataset to be tuned whereas WisdomOfAge does not have any data for now. Moreover, the platform does not give the possibility to discuss with companies to refine recommendation but it could be an interesting system if this feature could be developed.

In [3], Mishra and Rathi propose a survey of recommendation system models used in important recruitment platform like LinkedIn.

In [4], Al-Otaibi and Ykhlef present a technical survey in the field of job recommendation system.

In [5], Roy *et al.* propose an approach based on a classifier preceding a hybrid system using a content-based filtering (CBF; with cosine similarity) and a SI (using kNN).

In [6], Zisopoulos *et al.* present CBF systems and then discuss of their advantages and their drawback. They also present concrete examples currently in place.

In [7], Amato *et al.* propose a comparison of rule based, support vector machine classifier and Latent Dirichlet Allocation (LDA) to provides a classification of job offers.

As we can observe in the literature, authors suggest several ways to propose recommendations in the field of the recruitment.

On the one hand, some authors suggest the use of multiclass classifier model to achieve a classification of the candidates' profile. Thus, recruiters can filter candidates on the category attributed by the model. However, it does not allow them to research for a particular profile. Indeed, all candidates who belong to the same category are considered as "equal", there is no notion of ranking. Then, the recruiter should make his own ranking on the first filtered set.

On the other hand, some of them propose systems which provide a ranking of the candidates based on the similarity between their profiles and the job offer.

In this paper, we suggest a recommendation system based on the similarity of the mentors' profile and the mission offers. Moreover, the system also uses feedback let by precedent companies to increase its recommendation accuracy.

3. Data

To provide recommendations, the system has access to the following:

- Companies' profiles: activity sector, description.
- Mentors' profiles: education (dates, degree, faculty, description), experiences (dates, position, description), skills.
- Mission offers: title, category, description, keyword(s), discipline(s).
- Feedback: rating (0–10), description.

4. Preprocessing

In the first place, it is important to note that it is crucial to preprocess the text. Indeed, this step is important to avoid a decrease of performances in most of cases for every machine learning system. In his paper [8], Kadhim proposed the following steps: tokenization, stop words removal, stemming, text document representation and feature extraction (see Fig. 1). This preprocessing pipeline can be found in numerous text applications and has been already validated.

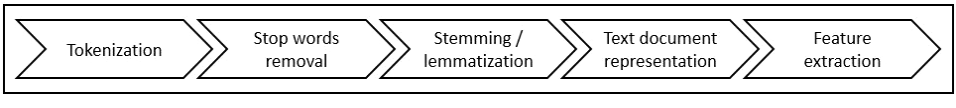

Fig. 1. Preprocessing pipeline.

Tokenization is a way of separating a piece of text into smaller units called tokens. Here, tokens can be either words, characters or sub-words. Hence, tokenization can be broadly classified into three types — word, character and sub-word (n-gram characters) tokenization [9].

The goal of removing stop words is to minimize the impact of common and worthless words. The list of stop words is depending on the language (e.g. in English,

stop words would be "the", "a", ...). Although in some cases, it could be interesting to remove rare words, they will not be removed in our context since rare words could outline rare skills.

Stemming step is a replacement of each token by its stem (it is the token in which prefix and/or suffix is removed). Nevertheless, there are some discussions to use lemmatization instead of stemming. Lemmatization is taking the morphological form of the word into account, based on a dictionary. The authors Khyani *et al.* suggest in their paper [10], that lemmatization tends to better keep the meaning of the words but is more complex to be implemented.

It is also important to note that, depending on the language, stemming and lemmatization can produce interpretation errors inherent to the principle on which is based each concept. Thus, [11] provides examples of these errors. We note that in Telugu language, stemming generates the same stem for words "robe" and "I don't share". In Greek language, lemmatization can generate the same lemma for words "imperfective" and "perfective".

It would be interesting to study each language case to determinate for which languages lemmatization gives the better performance and for which stemming performs better.

With text document representation, a numerical vector from a text is generated. A key element of working with common machine learning algorithm is that vectors should have the same length regardless of the text size of the document. Our system is using the text document representation called bag of words (BoW). For each document, a vector of the size of the vocabulary present on our whole corpus (here all mission offers, companies' profiles and seniors' profiles) is generated. Each box represents the number of occurrences of a token in the document.

There are also other text document representations such as GloVe [12], FastText [13] and BERT [14]. However, these representations are not available in all languages while WisdomOfAge is aiming to expand to several European languages. It will still be appropriate to study these possibilities to check if their implementation is or is not a possible option and if our system would benefit of using them.

The last step called feature extraction aims to show up the important features. Indeed, BoW only counts the number of occurrences of each token in a document. However, if a company or a senior searches for/owns a particular skill, the mission offer/senior profile will contain particular words which are not used frequently in the corpus. Thus, the idea is to weight the BoW using higher weight for rare words in the corpus. The weight will be computed using the inverse document frequency (IDF). It will allow our system to detect rare or particular skills and extract them from the mass of the pool of mission offers/senior profiles.

Note that text document representation using BoW in addition with feature extraction using IDF is famously called term frequency–inverse document frequency (TF–IDF).

5. General Approach

The recommendation system proposed in this paper is a hybrid recommendation system. It is constituted of recommendation subsystems whose outputs are assembled. Indeed, each recommendation system has strengths and drawbacks. Using several subsystems decreases the drawbacks of the general system to get more elaborate recommendations. There are several techniques to assemble the results of each subsystem. This point will be discussed in Sec. 5.4.

5.1. *Content-based filtering*

CBF recommendation systems are very popular in the field of recommendation system. Indeed, they allow accurate recommendations. However, they suffer from cold-start problem. This point will be discussed in Sec. 5.4. CBF can be divided into two categories: user-based filtering (UBF) and item-based filtering (IBF).

The first category, UBF, is based on the idea that if two users are similar, they have similar preferences. Thus, systems based on this principle try to find similarity between users. This similarity can be established using one or several metrics on the features described by users. The system then provides a recommendation list from the items liked by similar users.

The second category, IBF, bases its recommendation concept on the fact that if a user likes an item, he is willing to like similar items. IBF systems try to find similarities between items (through features describing items) to propose the more similar items of the items liked by the user.

In our context, users are companies posting mission offers. The features which describe these users are the elements of the companies profiles and the mission offer. On the other side, items are the seniors registered on the platform, with their features extracted from their profiles.

From the concept of IBF, we can conclude it is not adequate to reach our goals for the following reasons: the first one is that if a company enjoyed working with a senior and has a similar need later on, there is a very high probability that the company will not use the platform to ask for a recommendation list of seniors. Indeed, the company already owns the list of recommendation for its first mission with contact details and will not pay a second time for the same information. If the company has a different need, there is a low probability that the senior previously selected for the first mission will fulfil requirements for the second mission. Moreover, senior profiles include only technical features. Thus, it is not possible for now that our system understands the personality part of the senior that the company has enjoyed. So, without addition of these non-technical features, IBF concept is not interesting in our context.

On the other hand, UBF systems seem to be more suitable to our needs. Indeed, if a company has enjoyed working with a senior, there is a high probability that this senior is a good choice for a similar company with a similar need. In this way, the choice to focus on the concept of UBF is fully logical. To do this, we compute the

similarity metrics to find the more similar user to the company which requests a recommendation for a mission offer and weight the similarity regarding the feedback level given by the other user at the end of their missions with this user.

5.2. *Similarity index*

The proposed SI recommendation system compares senior profiles with the mission offer of the company. It provides a list of seniors with the highest similarity to the mission offer. To achieve this, each element of the senior profile is compared one by one to the mission offer (each skill, each education, each experience).

For the skill part, the similarity is weighted by the level assigned to the skill by the senior. However, this level is not treated in an absolute way but in a relative way. Indeed, evaluation of skill can differ from one person to another. Thus, the weight credited to the skill is relative to the weights that the senior has credited to these other skills. The idea is to detect more important skills of the senior.

For the education and experience part, each similarity is weighted by a weight which is computed depending on the duration of the education/experience and on the time from when the experience ended.

5.3. *Latent Dirichlet allocation*

As stated in Sec. 1, the recommendation system should be scalable. Moreover, it should be able to work in a classic server without particular calculation power. Up to here, the two proposed subsystems can works properly with a limited number of persons registered on the platform. However, with the expected growth of the platform, these solutions could be too greedy in calculation resources to work within acceptable time frame.

Thus, to keep the time to return the recommendations reasonable, companies, seniors and mission offers will be daily clustered using LDA which is a soft clustering method, that is, elements are not part of one cluster but belong to all clusters with a specific percentage.

When a company posts a new mission offer, the percentage of subscription to each cluster is computed. Then, the two subsystems do not work with all the data of the platform but only those from the clusters which are more representative of the mission offer (in other words, the clusters for which the percentage of subscription is the highest).

5.4. *Hybrid recommendation system*

On the one hand, CBF is more likely to produce accurate recommendation. Indeed, it does not base its recommendations only on objectives and mentors' description but also on the feedback given by companies. This principle allows CBF to capture more information from the phenomena. However, CBF suffers from cold start. It is not able to give recommendations without feedback at the beginning and is not able to

recommend a new mentor who does not have feedback. On the other hand, SI is less accurate but will not suffers from the cold start.

Thus, we suggest to combine both CBF and SI in a hybrid recommendation system to overcome the drawbacks of both systems. We propose a weighted sum to combine the output of each subsystem. In other words, the output of each subsystem is weighted and summed to obtain a unique score to rank mentors. The overview of the complete system is proposed in Fig. 2.

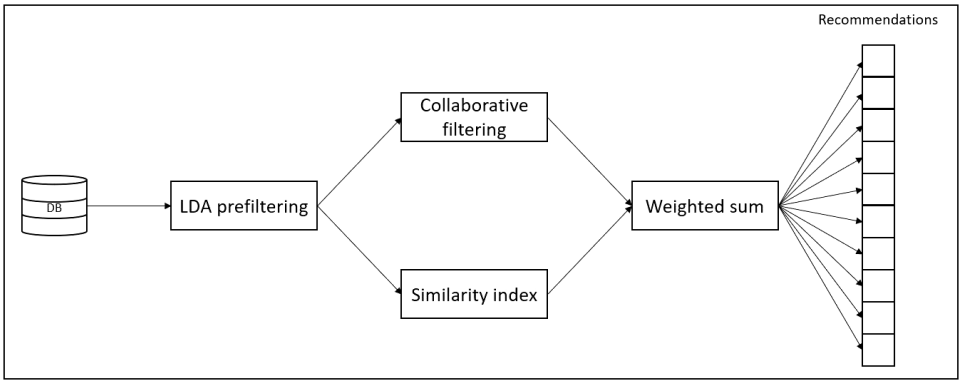

Fig. 2. Recommendation system synoptic.

The use of a weighted sum and not an equal weight sum is important to control the weight given to each subsystem along the life cycle of the platform. Indeed, at the beginning, CBF will not be able to propose a lot of accurate recommendations since there will be only few feedback. But with the growth of the platform, CBF will become capable of giving complete recommendations list without the help of the SI. In this way, new mentors will never be recommended. Thus, decreasing the weight of the CBF over time will overcome this problem.

6. Similarity Measure

The two subsystems measure similarity between two vectors which represent text information. To measure this similarity, the most known solution is the cosine similarity. However, some authors suggest noticeable improvements with the use of more elaborate methods. For example, Albitar *et al.* suggest in their paper [15], the use of SemIDF or SemTFIDF for measuring the similarity between two vectors which represent text in the context of a classification task. It allows them to achieve better performances than with simple cosine similarity. Thus, it will be necessary to assess these methods to decide if they could lead to an improvement of our system performances.

After similarity evaluation, the score is weighted according to the element. Indeed, it can be relevant to apply different weights for the different elements of

comparison. As part of CBF, it corresponds to assessing the importance of the description of the company, its sector of activity and the description of the mission offer. For the SI, it corresponds to assessing the importance of skills, education and experiences.

7. Implementation

7.1. *Text preprocessing*

We define four vectorizer functions $vec_{objective}(o)$, $vec_{company}(o)$, $vec_{education}(e)$ and $vec_{experience}(e)$ where o is an objective object and e is an education or experience object.

These functions transform raw data text of the object into a fixed size numerical vector. The text taken in account by each function is as follows:

- $vec_{objective}$: The title and the description of the objective.
- $vec_{company}$: The description of the company which has posted the objective.
- $vec_{education}$: The description of the education.
- $vec_{experience}$: The position and the description of the experience.

As described in Sec. 4, functions start by remove stop words, lemmatize and tokenize the text(s). Tokenization is down by n-grams on a range which is a hyper-parameter. Then, they vectorize the preprocessed text on the principle of TF–IDF to produce a vector of the size of the entire vocabulary of the whole corpus (i.e. the number of different tokens).

We have slightly modified the behavior of the TF–IDF. Indeed, in our case, when we fit the TF–IDF to compute the IDF vector, we consider it as a document of a corpus:

- All the text of all education and all experiences of one mentor.
- All the text of an objective and the associated company.

The choice to fit like this and not to consider each element (education, experience, objective, associated companies) as document comes from the fact that we should have consistency in the IDF vector. The following example provides better understanding: a mentor m_1 has a particular word w_1 which is linked to its domain of expertise. He has experience of 20 years. A second mentor m_2 has another particular word w_2 which is linked to its domain of expertise. He has 10 identical experiences of two years (in 10 different companies). No other mentors have these words w_1 and w_2 in their profiles. If we consider each experience as a document when computing the IDF, the weight according to w_1 will be higher than the weight according to w_2. This is not a wanted behavior because each mentor has the same experience duration to the position. So, considering all the education and all the experiences of a mentor as a document of the corpus when computing, the IDF makes the system capable of extracting features regarding mentors view.

7.2. *Education and experiences weighting*

First, we define a function which represents the decrease of a knowledge with the time. This function is obtained by seven points and a quadratic interpolation (Fig. 3).

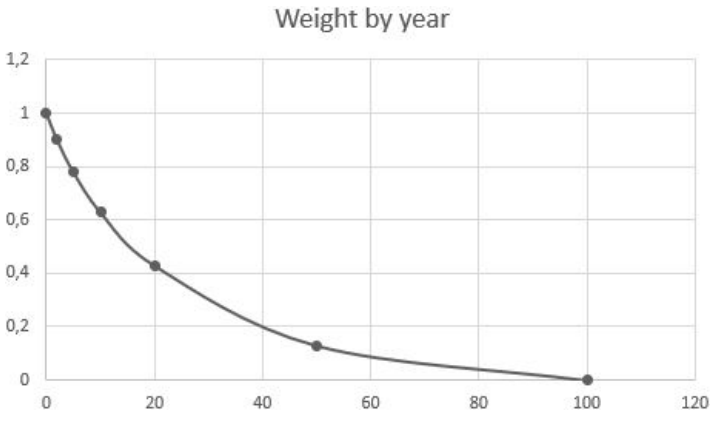

Fig. 3. Knowledge decrease with time.

Then, we define $weight(e)$ the weighting function which returns a weight for a given education/experience e. This function integrates the knowledge decrease function on the duration of the education/experience.

For example, if we take an experience e_1 which started five years ago and ends this year (five-year duration) and an experience e_2 which started 15 years ago and ended 10 years ago (five-year duration), $weight(e_1)$ will provide a weight of 4.4 years when (e_2) will provide a weight of 2.8 years.

7.3. *Content-based filtering*

To understand the implementation of the score computed by the CBF for a mentor, let us define the following:

- O is the set of objectives o solved by the mentor and for which the associated companies have left a feedback.
- $rat(o)$ is the rating left by the company to the mentor on an objective o resolution.
- x is the objective for which a company ask recommendations.
- $cs(v_1, v_2)$ is the cosine similarity between two numerical vectors v_1 and v_2.

We first define the similarity between the objective x and an objective o_i by

$$\begin{aligned} sim(x, o_i) = {} & W_{objective} * cs(vec_{objective}(x), vec_{objective}(o_i)) \\ & + W_{company} * cs(vec_{company}(x), vec_{company}(o_i)), \end{aligned} \tag{1}$$

where $W_{objective}$ and $W_{company}$ are hyper-parameters which control the importance of objective text (title and description) and company text (description) in the similarity measure between two objectives.

Then, we define the following five scores:

$$S_1(x, O) = \frac{\sum_{o \in O} \exp(sim(x, o) * W_{mean\text{-}rat\text{-}sim}) * rat(o)}{\sum_{o \in O} \exp(sim(x, o) * W_{mean\text{-}rat\text{-}sim})}, \tag{2}$$

which capture the mean rating of the mentor on objectives that he solved weighted by the similarity to objective x.

$$S_2(x, O) = \log_{10}\left(\frac{\sum_{o \in O} \exp(sim(x, o) * W_{mean\text{-}sim})}{size(O)}\right), \tag{3}$$

which captures the mean similarity of objectives solved by the mentor with objective x.

$$S_3(x, O) = \exp\left(\frac{\sum_{o \in O} rat(o) * W_{mean\text{-}rat}}{size(O)}\right), \tag{4}$$

which captures the mean rating of the mentor on objectives that he solved.

$$S_4(x, O) = \exp(\max_{o \in O}(sim(x, o)) * rat(\arg\max_{o \in O} sim(x, o)) * W_{max\text{-}rat\text{-}sim}), \tag{5}$$

which captures the rating of the mentor on the objective the more similar to objective x that he solved weighted by the similarity.

$$S_5(x, O) = \exp(\max_{o \in O} sim(x, o) * W_{max\text{-}sim}), \tag{6}$$

which captures the similarity of the more similar objective that he solved.

$W_{mean\text{-}rat\text{-}sim}$, $W_{mean\text{-}sim}$, $W_{mean\text{-}rat}$, $W_{max\text{-}rat\text{-}sim}$ and $W_{max\text{-}sim}$ are hyper-parameters.

We finish by define the score of the mentor by

$$S_{CBF}(x, O) = \prod_{i=1}^{5} S_i(x, O). \tag{7}$$

Regarding the previous formula, it is obvious that the score of a mentor belongs to the range $[0; +\infty[$. If a mentor has not solved any objective yet, the score given by the CBF is zero. After scoring each mentor, the CBF normalizes the mentors scores by dividing each score by the maximum score to obtain scores belonging to the range $[0; 1]$.

7.4. *Similarity index*

Before exploring the implementation of the score given to a mentor by the SI, let us define the following:

- EDU is the set of education e of a mentor.
- EXP is the set of experiences e of a mentor.

- x is the objective for which a company ask recommendations.
- $cs(v_1, v_2)$ is the cosine similarity between two numerical vectors v_1 and v_2.

We first define four weighted similarities:

$$sim_{education\text{-}objective}(x, e) = cs(vec_{education}(e), vec_{objective}(x)) * weight(e), \quad (8)$$

which corresponds to the similarity between the text of an education e of the mentor and the text of objective x weighted regarding the duration and seniority of education e.

$$sim_{education\text{-}company}(x, e) = cs(vec_{education}(e), vec_{company}(x)) * weight(e), \quad (9)$$

which corresponds to the similarity between the text of an education e of the mentor and the text of the company associated to objective x weighted regarding the duration and seniority of education e.

$$sim_{experience\text{-}objective}(x, e) = cs(vec_{experience}(e), vec_{objective}(x)) * weight(e), \quad (10)$$

which corresponds to the similarity between the text of an experience e of the mentor and the text of objective x weighted regarding the duration and seniority of experience e.

$$sim_{experience\text{-}company}(x, e) = cs(vec_{experience}(e), vec_{company}(x)) * weight(e), \quad (11)$$

which corresponds to the similarity between the text of an experience e of the mentor and the text of the company associated to objective x weighted regarding the duration and seniority of experience e.

Now, we define six scores

$$\begin{aligned} S_{education\text{-}objective}(x, EDU) = {} & W_{max} * \max_{e \in EDU}(sim_{education\text{-}objective}(x, e)) \\ & + W_{mean} * \frac{\sum_{e \in EDU} sim_{education\text{-}objective}(x, e)}{size(EDU)}, \end{aligned} \quad (12)$$

which capture the suitability between the text of the objective x and a mentor's set of education EDU.

$$\begin{aligned} S_{education\text{-}company}(x, EDU) = {} & W_{max} * \max_{e \in EDU}(sim_{education\text{-}company}(x, e)) \\ & + W_{mean} * \frac{\sum_{e \in EDU} sim_{education\text{-}company}(x, e)}{size(EDU)}, \end{aligned} \quad (13)$$

which captures the suitability between the text of the company associated to the objective x and a mentor's set of education EDU.

$$\begin{aligned} S_{experience\text{-}objective}(x, EXP) = {} & W_{max} * \max_{e \in EXP}(sim_{experience\text{-}objective}(x, e)) \\ & + W_{mean} * \frac{\sum_{e \in EXP} sim_{experience\text{-}objective}(x, e)}{size(EXP)}, \end{aligned} \quad (14)$$

which captures the suitability between the text of the objective x and a mentor's set of experiences EXP.

$$S_{experience\text{-}company}(x, EXP) = W_{max} * \max_{e \in EXP}(sim_{experience\text{-}company}(x, e)) + W_{mean} * \frac{\sum_{e \in EXP} sim_{experience\text{-}company}(x, e)}{size(EXP)}, \quad (15)$$

which captures the suitability between the text of the company associated to the objective x and a mentor's set of experiences EXP.

$$S_{objective}(x, EDU, EXP) = W_{education} * S_{education\text{-}objective}(x, EDU) + W_{experience} * S_{experience\text{-}objective}(x, EXP), \quad (16)$$

which captures the suitability between the text of the objective x and a mentor's set of education EDU and experiences EXP.

$$S_{company}(x, EDU, EXP) = W_{education} * S_{education\text{-}company}(x, EDU) + W_{experience} * S_{experience\text{-}company}(x, EXP), \quad (17)$$

which captures the suitability between the text of the company associated to the objective x and a mentor's set of education EDU and experiences EXP.

We finish by define the score of the mentor by

$$S_{SI}(x, EDU, EXP) = W_{objective} * S_{objective}(x, EDU, EXP) + W_{company} * S_{company}(x, EDU, EXP). \quad (18)$$

W_{max}, W_{mean}, $W_{education}$, $W_{experience}$, $W_{objective}$ and $W_{company}$ are hyper-parameters.

Regarding the previous formula, it is obvious that the score of a mentor belongs to the range $[0; +\infty[$. After scoring each mentor, the CBF normalizes the mentors scores by dividing each score by the maximum score to obtain scores belonging to the range $[0; 1]$.

7.5. *Hybrid recommendation system*

To obtain the final score of a mentor, the output of both subsystems is assembled by a weighted sum:

$$S_{mentor}(x, O, EDU, EXP) = W_{CBF} * S_{CBF}(x, O) + W_{SI} * S_{SI}(x, EDU, EXP), \quad (19)$$

where

- x is the objective for which a company ask recommendations.
- O is the set of objectives o solved by the mentor and for which the associated companies have left a feedback.
- EDU is the set of education e of a mentor.
- EXP is the set of experiences e of a mentor.
- W_{CBF} and W_{SI} are hyper-parameters.

After assembling the score for each mentor, the N best scores are selected and the recommendation system provides the mentors' profile to the company, which has requested recommendations for its objective x.

8. System Evaluation

8.1. *Labeled data*

One problem is that we are facing lack of labeled data to evaluate the model. Indeed, as the platform is in development, we do not have past data of its activity. To collect labeled data, the platform will add a feature to allow companies to label data in exchange for a discount on the website. Concretely, when a company accepts to participate, the user will be asked to create an objective linked to its activity sector. After this step, a long list of mentor's profiles will be exposed and the company will label each mentor regarding its skills to achieve the objective (label could be "relevant" or "irrelevant").

8.2. *Evaluation metrics*

There exist a large range of metrics to evaluate the classification results of recommendation systems. These metrics can be classified into two categories: Decision support metrics and ranking-based metrics. In the first one, the ranking of the item is not taken into consideration. In the second, errors are more penalized if they are in the top of the recommendations. Since the platform will propose the mentors' profiles in a vertical list, companies will probably start by taking a look on the first mentor, then the second, etc. In this way, using a ranking-based metric is necessary. The Mean Average Precision (MAP) is a good choice regarding our context.

First, we define the Average Precision (AP) to evaluate a recommendations list for a single objective:

$$AP@N(o) = \frac{1}{min(m(o), N)} \sum_{k=1}^{min(m(o),N)} P(k) * rel(k), \tag{20}$$

where

- o is the given objective for which we request recommendations.
- N is the number of recommended mentors by the recommendation system.
- $m(o)$ is the number of relevant mentors in the dataset for the given objective o.
- $P(k)$ is the number of relevant mentors in the k first recommended mentors divided by k (known as *precision@k*).
- $rel(k)$ is an indicator function. It is equal to one if the kth recommended mentor is relevant, zero otherwise.

Note that we have slightly changed the AP formula. Indeed, the business plan of the platform imposes to give N recommendations for each request as long as there are at

least N relevant mentors registered on the platform regarding the request. Thus, we have changed the N in the initial formula by $min(m(o), N)$ to ensure that the metric does not penalize a request for which there is not enough relevant mentors to achieve a complete recommendations list of size N.

We can now define the MAP to evaluate the recommendations list for all our test objectives:

$$MAP@N(O) = \frac{1}{size(O)} \sum_{o \in O} AP@N(o), \tag{21}$$

where

- O are the given objectives o for which we request recommendations (one objective).

In addition to the MAP which evaluates the classification results of our system, we will also introduce another metric: *coverage*. The *coverage* is the total number of recommended mentors for all our test objectives divided by the total number of mentors available in the platform. This metric is important to ensure that our recommendation system will not always recommend the same mentor (even if these mentors are relevant).

Since we will have a small dataset, it will be impossible at the beginning to have a complete *coverage* since some mentors could be irrelevant for all our test objectives. Thus, we will focus to maximize in first place the MAP while keeping a look on the *coverage* to ensure that a little improvement on the MAP will not induce an important decrease of the *coverage*.

9. Results

At the time of the redaction of this paper, the platform is not yet on the market. We have enrolled some mentors and early adopter companies on the platform to evaluate the recommendation system. However, no official matching was operated. Thus, we do not have feedback and rating from companies on mentors. In this way, the results in this section only provide an overview of the performances of the sub-recommendation system SI.

There are 20 mentors in several domains:

- Web development
- Machine learning engineering
- Data science
- Mechanical engineering
- Electronics engineering
- Physics science
- Chemical engineering
- Agri-food engineering

- Fabrication and process engineering

We have asked our early adopter companies to create an objective test in these domains and for each mentor, evaluate if he is relevant or not regarding his capacities to help the company to achieve its objective and overcome its challenges. We have collected a total of 11 objectives tests. Table 1 summarizes the dataset of test collected.

Table 1. Dataset of test.

	Mentor id									
	1	2	3	4	5	6	7	8	9	10
Electronics engineer										
Embedded developer										
Process engineer		✓	✓							
Machine learning engineer							✓	✓	✓	
FPGA engineer										
Building engineer										✓
Machine learning specialist							✓	✓	✓	
Data scientist							✓	✓	✓	
Project manager		✓	✓			✓			✓	✓
Java back-end developer					✓					
PHP front-end developer				✓	✓					

	Mentor id									
	11	12	13	14	15	16	17	18	19	20
Electronics engineer	✓	✓	✓	✓						
Embedded developer			✓	✓						
Process engineer			✓							
Machine learning engineer										
FPGA engineer			✓							
Building engineer										
Machine learning specialist										
Data scientist										
Project manager	✓	✓		✓				✓		✓
Java back-end developer										✓
PHP front-end developer			✓							

After tuning of the SI subsystem and the preprocessing range of n-grams by a random search, the system achieves an *MAP@5* of 43% and a *coverage* of 93%. The *MAP@5* may seem low but it was predictable for several reasons. The first one is the fact that we have a small dataset to test the system. The second is that we ask the recommendation system to provide, for each objective, a list of five mentors. When there are less than five mentors relevant for an objective, the *MAP* is not evaluated on the full list of recommendation but only on the m first recommendations (where m is the number of relevant mentors for an objective). Most of objectives test evaluate less than five mentors as relevant. In this way, we observe a decrease of the *MAP*. To finish, we expect better recommendation from the CBF than from the SI but yet

CBF is not able to provide recommendation until the opening of the platform and the collecting of feedback after objectives achievement.

10. Conclusion

In this paper, we have proposed a general approach for a scalable recommendation system in a particular context. This context raises specific issues involving technical choices. To overcome these constraints, we suggest a hybrid recommendation system based on a CBF and a SI. After the implementation of the system for English language, we provide primary results about recommendations performances. Some others solutions remain to be tested to evaluate if it increase the precision of recommendations: vectorization of raw text data with transformers, similarity of two numerical vector with more sophisticated measures.

Acknowledgments

This work was co-funded by the State Secretariat for Education, Research and Innovation of the Swiss federal government and the European Union, in the frame of the EU AAL project WisdomOfAge (AAL-2020-7-83).

References

[1] A. Agarwal and Dr. Senthilkumar, Resume recommendation system using cosine similarity, *Int. Res. J. Mod. Eng. Technol. Sci.* **4** (2022) 158–162.
[2] F. Mentec, Z. Miklós, S. Hervieu and T. Roger, Conversational recommendations for job recruiters, in *Knowledge-Aware and Conversational Recommender Systems*, 2021, https://hal.inria.fr/hal-03537355/.
[3] R. Mishra and S. Rathi, Efficient and scalable job recommender system using collaborative filtering, in *ICDSMLA 2019*, Lecture Notes in Electrical Engineering, Vol. 601 (Springer, 2020), pp. 842–856.
[4] S. T. Al-Otaibi and M. Ykhlef, A survey of job recommender systems, *Int. J. Phys. Sci.* **7**(29) (2012) 5127–5142.
[5] P. K. Roy, S. S. Chowdhary and R. Bhatia, A machine learning approach for automation of resume recommendation system, *Procedia Comput. Sci.* **167** (2020) 2318–2327.
[6] C. Zisopoulos, S. Karagiannidis, G. Demirtsoglou and S. Antaris, Content-based recommendation systems (2008), https://www.researchgate.net/publication/236895069_Content-Based_Recommendation_Systems.
[7] F. Amato, R. Boselli, M. Cesarini, F. Mercorio, M. Mezzanzanica, V. Moscato, F. Persia and A. Picariello, Challenge: Processing web texts for classifying job offers, in *Proc. 2015 IEEE 9th Int. Conf. Semantic Computing*, 2015, pp. 460–463.
[8] A. I. Kadhim, An evaluation of preprocessing techniques for text classification, *Int. J. Comput. Sci. Inf. Secur.* **16**(6) (2018) 22–32.
[9] A. Pai, What is tokenization in NLP? Here's all you need to know (2022), https://www.analyticsvidhya.com/blog/2020/05/what-is-tokenization-nlp/.
[10] D. Khyani, B. Siddhartha, N. Niveditha and B. Divya, An interpretation of lemmatization and stemming in natural language processing, *J. Univ. Shanghai Sci. Technol.* **22** (2021) 350–357.

[11] BiText, What is the difference between stemming and lemmatization? (2021), https://blog.bitext.com/what-is-the-difference-between-stemming-and-lemmatization/.
[12] J. Pennington, R. Socher and C. D. Manning, GloVe: Global vectors for word representation, in *Proc. 2014 Conf. Empirical Methods in Natural Language Processing*, 2014, pp. 1532–1543.
[13] P. Bojanowski, E. Grave, A. Joulin and T. Mikolov, Enriching word vectors with subword information, *Trans. Assoc. Comput. Linguist.* **5** (2017) 135–146.
[14] J. Devlin, M.-W. Chang, K. Lee and K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, arXiv:1810.04805.
[15] S. Albitar, S. Fournier and B. Espinasse, An effective TF/IDF-based text-to-text semantic similarity measure for text classification, in *Int. Conf. Web Information Systems Engineering*, 2014, pp. 105–114.