A personal fashion adviser application with a social component

Correia, Margarida
Instituto Superior Técnico
Lisbon, Portugal

ABSTRACT
We created an application called InsideFashion that provides a fashion recommender system with a combination algorithm to be able to retrieve outfits. In our recommender system, the user can receive suggestions of clothing items depending on the temperature and weather of a selected location, occasion and emotional status and combines the items regarding the style previously selected by the user. For that matter, the person needs to have their virtual closet available which could be done in our application by adding the items providing their photo and their description.

We also have a social network in which the users can follow other users in order to get inspiration from them. We developed our aplication incrementally and iteratively performing a user study in order to connect the emotional status with clothing pieces. We also performed an heuristic evaluation in order to evaluate our application violations and develop a solution for those errors, and usability tests that allowed us to validate our application with a real user environment. To evaluate and validate our style combination algorithm, we performed a study in which we provided a set of outfits for the user to be able to distinguish between the available styles in which we conclude that 67.16 percent of the users were able to distinguish between styles.

Keywords
Recommender systems, Fashion, Collaborative filtering systems, Content-based systems, Knowledge-based systems, Hybrid systems

ACM Classification Keywords
Algorithms, Design

INTRODUCTION
With the growth of the Internet and to increase sales, e-commerce providers started implementing recommender system (RS). These systems also allowed providers to better understand their costumers' behaviour. Providers have to be careful when integrating a RS because if it does not provide good sug- gestions, user satisfaction decreases and the user loses trust in the system and stops caring about the recommendations provided, which leads to decreasing sales. If the system provides good suggestions, the user will be more satisfied, will rely more on the system, providing ratings that help the RS, will buy more items and will be more committed to the platform increasing user loyalty to it. There are many examples of e-commerce platforms that use RS, such as Amazon that uses collaborative filtering to provide suggestions to its users and is able to scale to millions of users and items. We cannot say that recommendation systems are specific for a certain industry or domain because they are somehow relevant to all of the domains. In our project, we aim to implement and apply a recommender system to the fashion domain.

Fashion is an important industry not only from an economic perspective but also from a cultural one. Economically, it represents an important sector in the European Union (EU) where more than six million people work. This industry tends to grow faster than the EU economy in general as noticed in the 2010 crisis. Fashion is a market where art, innovation and creativity meet and produces high-valued products, it is a way to express ourselves. This is a 295.6 billion pounds market and it includes textiles, clothing and footwear industries. There are 172 755 textile and clothing industries employing more than 1.62 million people in Europe alone. [11] [10]

Figure 1: Comic figure [6]

Both men and women struggle with the process of choosing outfits on a daily basis, as shown in figure 1, making a recommender system a support tool for their daily life.

There are two types of people, as we can see in the comic: the ones that have a lot of items in their wardrobe and the ones that have few items and each group struggle with different challenges. The task of choosing an outfit or buy one item in a store that matches their clothing can be hard. To people who have many items it is also struggling because they are unable to filter all the available items. Since they have a large amount of items in their disposal,

usually they are not able to create an outfit when they have a specific event to attend or when they want to achieve different styles because they feel overwhelmed. With our RS, we intend to allow people to store all their clothing without needing to remember all of them whilst the system provides suggestions regarding their needs.

This RS will also be a great help for this group of people that are not into innovation in what concerns their clothing. It will help them creating different outfits and also buy different items to be able to combine them with already existing items in order to transform their look and achieve different styles. We will also have the possibility to follow other users being able to get some inspiration.

Everybody has their good and bad days and we want to create a RS that is able to provide suggestions based on the person's mood. It will not only consider the occasion, style, weather and temperature of the day but also his/her emotional status providing personalised suggestions.

The main goal is to create a recommender system in the fashion area that suggests outfits and takes into account personalisation. We want to suggest specific outfits for each user and consider their unique tastes in the recommendation. We aim to achieve personalisation by taking into account the following aspects: weather and temperature of a certain location, user profile, user emotional status, style wanted by the user and occasion.

BACKGROUND WORK

To create a RS, we have to decide which technique we want to apply. There are many techniques that we can apply. After weighing pros and cons and analysing which are the main goals of the RS, one can decide which technique to use.

Collaborative filtering is the most popular and used. The comparison between the collaborative and the content-based is usually made. One of the major differences is the fact that the Collaborative filtering (CF) does not depend on the domain – content of the items – to make its predictions whilst the Content based (CB) does. Relying on the domain, as the CB does, represents a challenge because the RS will be able to provide suggestions if there is sufficient metadata about it – limited content analysis. However, if it has a rich description of the items, it is able to provide suggestions even of items that no user has yet gave feedback. This is something that the CF is not able to do, because it relies on the user's neighbours ratings and, if there is no item's ratings yet available, it cannot suggest it – data sparsity. Since CB bases their suggestions only on the content of the items that the user liked in the past, when his/her preferences change, it is able to adjust the suggestions in a short amount of time but it can overspecialise its suggestions – overspecialisation.

To perform good suggestions, CF has to analyse the user's profile because the recommendations depend on it and on the user's neighbours profile. If it does not yet have the user profile available, meaning that the user did not rate any item yet, it will not be able to provide serendipitious suggestions – serendipity. Depending only on the user's and neighbours' profile, it makes almost impossible for the system to distinguish between two items that are similar but have different names – synonymy.

The knowledge-based systems are different from the previous ones because their key to provide suggestions is the knowledge model in which it relies. Since it needs to formalize the problem with the help of domain experts and create a model about it, in the beginning, the system will be able to provide better suggestions comparing with the CF and CB. But, if the learning agents are not well implemented, the quality of recommendations can be worse. Of course that, if a user changes his/her preferences, the model will adjust itself providing adequate suggestions. Regarding the hybrid, it combines different techniques bringing the best of both worlds, overcoming the limitations of one technique with the benefits of the other and optimizing the system performance. The difference between the different types of hybrid techniques is the way it implements their sub-techniques (implements the techniques it relies on). For example, it can run the algorithms separately and, in the end, combine their results.

After analysing a certain number of fashion systems, we are able to conclude that almost all of them have a knowledge base that we talked about previously to be able to store the items and perform queries. But there was also a system that used a content-based technique that we also discussed previously. The systems that have better results are the ones that apply learning algorithms to be able to combine pieces.

Some characteristics that we took into account in our recommendation system are present her: occasion, style, temperature and weather. But the emotional status is only considered in the Moody Closet. Even though we also ask the user in the annotation phase in which emotional status she/he will use a certain piece, Moody Closet uses a different approach.

ARCHITECTURE

Inside Fashion is a distributed application with a client-server architecture. The client will be the users mobile application developed as a web-application, being available to IOS and Android operating systems, using Javascript as main language. The application server is deployed with Node.js using the express module, where it can also be found our database in MongoDB.

It is a RESTful system that will provide RESTful web services such as the login, register, recover password, annotation, recommender task, among others. All the data that is exchanged between the server and the application will be represented by JSON objects. The information flow is as presented in figure 2.

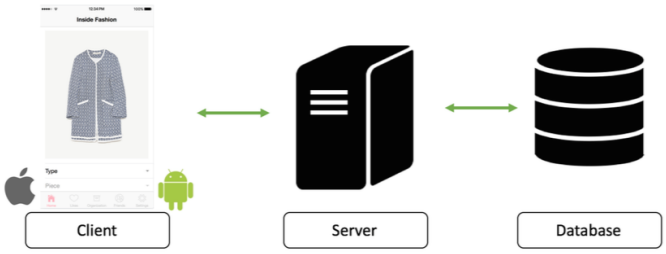

Figure 2: Information flow

InsideFashion has a model-view-controller architecture, figure 3, meaning that has three different layers because it separates the different types of information as needed in an interactive application.

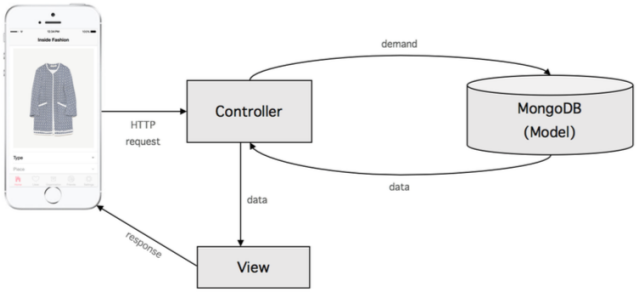

Figure 3: Inside Fashion architecture

• **Controller:** it receives the requests, will execute them and transmit the changes to the correspond ing view. If needed, it will communicate with our Mongo database, that is the model layer. This layer is composed by all the application services such as, for example, the recommender system.

• **Model:** it will process data, manage logic and respond to the controller. Data includes all the information about the users, the pieces and the outfits.

• **View:** based on the changes made previously, it will generate a new view to the user.

IMPLEMENTATION

To be able to consider so many features – temperature, weather of a certain location, style, occasion and user emotional status - in our application, there is a need to have a knowledge base in which the items are well described in order to retrieve good suggestions.

We will apply queries to our knowledge base and have a regression model as seen in other RSs explored previously but will have our own combination algorithm that considers the rules that we determined to combine previous queries results.

We aim to develop a system that is personalised that is why we also complemented with a explicit feedback tool to be able to build a user profile that will adapt whenever a user provides feedback.

In order to provide good suggestions, there is a need to well characterise each piece of clothing. That is why, before suggestions are retrieved to the user, he/she will have to take photos of his/her pieces of clothing because each one needs to be characterised. The major part of the characterisation will be done by the user and other automatically, as seen in figure 4.

• **By the user:** The user will specify the type of piece, material, pattern, which occasions, temperatures and seasons will he/she wear it.

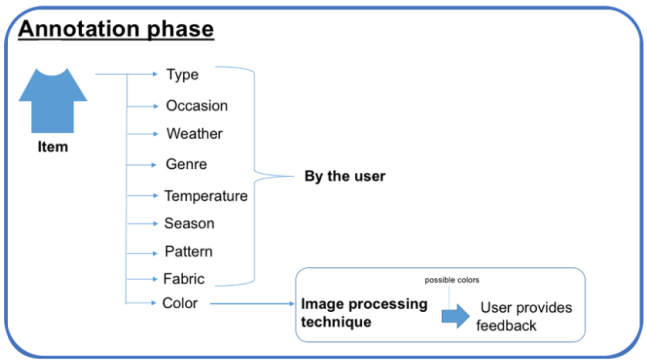

Figure 4: Annotation phase

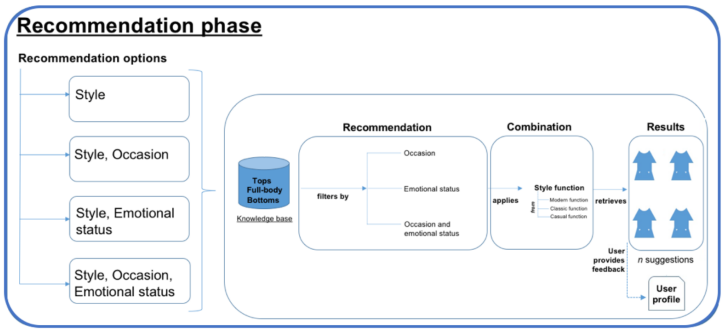

Figure 5: Recommendation phase

• **By the system:** The colour of the piece will be inferred using image-processing techniques. Basically, the system analysis the image and will display the computed colours in a pop-up. The user will validate and choose the one corresponding to the main colour if it does not have a pattern or the background colour if it does have a pattern. The user validation is necessary to guarantee that the background colour given those two possible situations: patterned clothes or not.

We developed our algorithm in two phases as shown in figure 5. Assuming that we build a knowledge base to each user because it will need to characterise each item according to many features. We divide our general algorithm in:

• **Recommendation itself**: given the fact that it is a knowledge base, we will perform queries accord- ing to the options selected by the user in the recommendation task. Meaning that our recommen- dations are the results of the queries;

• **Apply our combination algorithm** to the pieces retrieved by our previous algorithm: the pieces that satisfy those requirements will be inputed to our combination algorithm. Because we consider three different styles and because it is the combination that ultimately defines the style of an outfit, we have three combinations algorithms that receive those pieces retrieved previously and analyse if they match for the selected style and if they do, they will be retrieved as an outfit.

Recommendation phase

After we have our knowledge base built, we can perform certain queries to it regarding the temperature, weather, occasion and emotional status.

One of the features to take into account was the **weather and temperature of a certain location**. We get this information whenever a user specifies a location and the day. The user is able to specify a certain location because we provide a map in which he/she can select a certain position corresponding to the specific location. The system stores the longitude and latitude. A calendar is also provided for the user to specify a certain date that is stored by the system.

We use the DarkSky API to obtain all the forecast information. With the longitude, latitude and date we are able to make a request to the API to obtain the temperature information and the forecast icons. With the retrieved forecast icon we map it into its corresponding forecast. If it does not have yet forecast information available about that day, the system will only consider the season to make the suggestion. In the end of this feature, we are able to retrieve all the clothes that satisfy those values.

Occasion was other feature that integrated our system. The user specifies in which occasions he/she will wear that piece so we will retrieve pieces that the user would wear in the occasion selected. Regarding the **emotional status**, we had to analyse this situation more deeply. Given the fact that this is an unexplored field, we could not find any work that showed a direct connection between the emotional status and the clothes we wear. With that in mind, we performed a study in which the main goal was to understand the connection between these two fields. After analysing 110 answers, in which 61 were women and 49 men, we concluded that people wear different clothes in different moods, happy and sad. With 73 that agreed that consciously and uncon- sciously we wear different clothes according to our emotions, we analysed that people that are sad use less patterns and darker colors and when they are happy they use more patterned and colored clothes. With this survey we were able to obtain the answer to a question that makes a big difference in the way we recommend according to emotional status, that was: is it possible to modify user emotional status through clothing? 89 percent said that it was not possible.

Regarding the analysis of the obtained results, we recommend darker and without pattern clothes when the user is sad and lighter and patterned clothes when the user is happy. Meaning that the item characteristics needed to take into account were pattern and colour. F or that, we used other model of color: HSL - hue, saturation and lightness. HSL has a field named lightness that varies between 0 and 1. Lighter colors are closer to the value 1 and darker are closer to the value 0.

In a recommendation considering the weather, temperature, occasion and emotional status, the first three queries are performed as specified before and then it is performed a query regarding the emotional status. Basically, it will order the recommendation queries results obtained by the lightness value, it will then suggest the n items that are lighter or darker depending on the emotional status selected.

Combination phase

After having the results from previous queries, our combination algorithm, that is based in the style, will be performed.

We decided that we would have three general styles: casual, classic and modern. We decided to focus on just three styles because there is a wide variety of styles available which characteristics are hard to define. With those three styles – classic, casual and modern - we can more easily know its distinguishing characteristics. For that, we did a research on each style analysing and searching outfits of each one to assure that we were able to extract the main rules. That it not just a type of piece that gives the style, it is the combination that ultimately does that. With that in mind, we decided that our combination algorithm would work in two different phases: annotation phase and recommendation phase.

In the annotation phase, in which the user describes a clothing piece, our system will analyse its type, pattern and fabric as shown in figure 4. With those characteristics, it is able to evaluate according to each style if it has those previous requirements to belong to that style. For example, blazers are a type of piece of clothing that is usually used in classic and modern styles meaning that this piece would belong to the classic and modern database. That means that one piece can address one or more different styles and if it

addresses it has an entry in each style database.

So we built a linear regression model to each style that predicts if a certain piece meets the characteristics needed to belong to that style. The variables used to predict if it belongs to a style are the following item's features, as specified before: type, pattern and fabric. Our regression models only handle a item not a combination of items.

The other step on our combination happens in the recommendation itself. Because we have three different styles, we have three different ways to combine pieces meaning that we have three different combination algorithms.

On the classic style, it is usual the black and white combination. That is why, when it is selected a classic style, it will try to search for pieces that are black or white and combine them to other white or black pieces. If not, it will combine pieces that have a similar background colour. In this step, we already have the pieces that fill the occasion and/or emotional status and temperature and weather. It will basically search on the pieces retrieved by those previous queries the ones that belong to the classic style database. This is not enough because having a classic style and fullfilling the previous requirements are not the desired. We could not have random combinations of classic style pieces there is why we have to apply rules to combine them.

The main difference between the casual and classic style is the type of pieces whilst in a casual look it is usual to wears a pair of jeans, we do not see that in a classic style. It also accepts to incorporate some minimalist patterns. The way to combine them is exactly the same in terms of colours, the only difference is the fact that it can have a patterned item in an outfit.

The modern style embraces colour and patterns that is why the main difference between the other styles is the fact that it accepts the diversification and combining pieces with same pattern, different textures. Here we can mix a patterned item with a plain piece with the same background colour or white or black colour or with one with the same pattern.

Even though they are different combination algorithms, the colour combination rules are the same: a white or black piece with other coloured piece, pieces with a similar background colour. The colour of a clothing piece is defined by the HSL model meaning that different colours have different HSL codes. Because we want a similar colour, we analysed and defined a threshold in the sum of the difference of each field. If it is less than that threshold it will have a similar colour and can be used in the combination.

$$(\|hue_1 - hue_2\|) + (\|saturation_1 - saturation_2\|) + (\|lightness_1 - lightness_2\|) \leq threshold$$

The main difference between the three different algorithms is the pieces in which they execute the algorithm and then each one of them have some specific characteristics which made it necessary to separate the combination algorithm to each style.

Because here our focus is not on the scalability, performance and velocity of our algorithm, it will only retrieve our combinations after analysing all the possibilities meaning that we can have, depending on the virtual closet, hundreds of possible outfits available to the user. This is good because we do not restrict the user to a limited set of outfits in case we defined a limit of maximum outfits. Thus, the user has the possibility to analyse each ouftit and decide which one he/she will wear and save it in an album. If we set a limit, it could happen that the combination that he/she would prefer were not retrieved but if that limit was not implemented it would see them.

An important piece on any recommendation system is being able to build a user profile because that is what distinguish between users and another element that provides personalisation.

To generate a user profile, we implemented a technique for the user to be able to provide implicit feedback. Whenever our system retrieves suggestions, the user has the possibility to express if his/her impression is negative. If he/she dislikes an outfit, he/she can press the dislike button meaning that that specific outfit will never be retrieved again independently of the circumstances (options selected: date, occasion, emotional status and style).

What our system does is getting the ids of those outfit pieces and store them in a database. Which means that each user has a database of outifts that she/he does not like and will never be retrieved by our system again. That database is composed by entries corresponding to the outfits – set of pieces' ids – that he/she previously disliked. Before our system suggests outfits, it will search if they match any outfit on that database and only if they do not match, they will be retrieved.

We made another study presenting some outfits that our algorithm retrieved in order for test if the users were able to distinguish between styles.

OTHER FEATURES

Social Network

We wanted to develop an application that was not just a recommendation system but that also could be used to get some inspiration from others. We believe that when we

connect to other people, we learn and the way we could obtain this from our application was if it had a social network incorporated.

A user has a username associated and can create his/her own virtual wardrobe by adding his/her items to it. The main goal of having the possibility to create albums is the fact that it helps to organise not only the items but the outfits for later inspiration and use. But the albums' purpose goes beyond that, at least in the social media context. The user's albums are the way that other users see him/her because each user has a profile composed by his/her albums.

Our social network goal is to get inspiration from other users which sometimes means that we can have some interest in a user but it does not have an interest in our profile that is why we implemented a follower dynamic. We have a feed feature in which we are able to, whenever someone we follow adds an item or outfit to his/her albums, it appears in our own feed.

Voice recommendation

We decided to incorporate a voice recognition API: Web Speech API in HTML5 and chosen to recognise two languages: portuguese and english. This engine was incorporated in recommendation task because it was the one that needed less input. Before we decided to use this one, we tested other ones but after deeply analysing and performing tests, we decided to incorporate this one.

With the help of this API, we are able to know the text from the user' speech. The key piece here was building an algorithm that would compare the text from the user speech to the available options. But the general algorithm could easily failed if we only allowed the user to say exactly the same words available in the checkboxes. Because of that, we build three different knowledge bases to each dimension: style, occasion and emotional status. In each one of them, to each option, we searched and stored synonyms in english and portuguese. Therefore, whenever a user presses the microphone button and speaks, our application will search if those words are in our knowledge bases. If there are in the knowledge bases, it will associate to each dimension the value of that word.

Interface

We developed our application iterativly and incrementally centered on our main tasks: annotation, recomendation, social network. With that in mind, before creating a functional prototype, we designed two low fidelity prototypes. After analysing those prototypes, we developed a final solution, one of the screens is in the figure 6.

Figure 6 - Application screenshot

EVALUATION

The fact that we had an application it meant that we needed to develop an easy to use interface and we had the need to evaluate if it was in fact easy to use. The first evaluation to be performed was an heuristic evaluation done by three experts which goal was to identify which violations our system had regarding usability principles.

Table 1

Severity	Frequency
0	0
1	2
2	23
3	5
4	0

Table 2

Heuristic	Frequency
Visibility of system status	5
Match between system and the real world	0
User control and freedom	6
Consistency and standards	3
Error prevention	8
Recognition rather than recall	6
Flexibility and efficiency of use	3
Aesthetic and minimalist design	0
Help users recognise, diagnose, and recover from errors	3
Help and documentation	0

In the 30 problems seen in table 1 detected by three experts, 18 regarded the same situation. Most of them had a severity of 2. 8 of them regarded error prevention, 6 were about recognition rather than recall and 6 about user control and freedom violations, as seen in table 2.

After analysing the results of that test, we needed to overcome all the problems identified.

When all the problems were solved, usability tests were needed to be performed. In this phase, we needed to evaluate our application in the user context with real users. In this evaluation, the user will perform a set of tasks that represent the main tasks possible to perform in our application.

These usability tests were performed with 17 users in order to gather quantitative and qualitative usability metrics. Overall the results made us realise that the errors made during the evaluation process were not major. The fact that the set of questions with higher mean and maximum values were the ones regarding the recommendation is due to the fact that a user can have a recommendation based on just three variables (date, location and style).

With those results in mind, we proceed to the changes needed to have a more intuitive interface. After performing the questions given on the script, it was performed a quick but accurate usability questionnaire with 10 questions using the system usability scale (SUS) invented by John Brooke [4]. The minimal rating given was 67.5 and the maximum was 97.5 on a 100 points scale. After doing the mean of all the values obtained, our application achieved a 91.61 score as we can see in figure 6, which corresponds to an A in an A-F grade scale (A is the best and F the less).

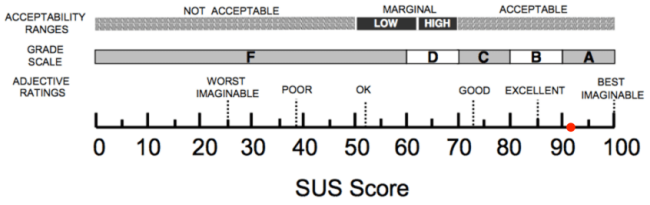

Figure 7 - SUS application score

FUTURE WORK AND CONCLUSION

Future work

One thing that we considered in our recommendation system was the user feedback. Whenever a user receives a recommendation, he/she has the possibility to store it in one of his/her albums and we store the outfits that he/she disliked not retrieving them again. But we could do that to the likes meaning that we would have to analyse the outfits that he/she liked and implement an algorithm able to retrieve suggestions similar to the ones that were liked. We implemented the means to be possible to do that: user storing the outfit itself in an album but we did not integrate the likes into our algorithm given the fact that we had already a lot of variables to consider. With that in mind, we could also take advantage of the improvement of liking other users' posts to consider in his/her past behaviour. In this case, we would be using explicit feedback because we consider his/her likes and the previous recommendations that he/she stored into his/her own albums to provide the recommendation. To go even further, we could use hybrid feedback if we integrated the implicit feedback and considered monitoring the user activity - implicit feedback - that in this case would be the time he/she passed looking to other users posts: we could track which post was shown at that moment and the time spent and analyse it, incorporating the results of that analysis and the previous explicit feedback into his/her user profile.

To improve our algorithm regarding its efficiency, whenever a recommendation is retrieved, we could store it an a database with the inputed conditions. Meaning that in a next recommendation, the system would analyse that database and see if those conditions were met in the past, it would retrieve the recommendations stored in the database. Going even further, it could be developed a similarity function that would analyse the inputed conditions and see the similarity between the ones stored in the database and if that similarity was higher than a certain value, it would adapt the case stored in the database to the actual one.

The fact that we developed a web-app with a view opens the possibility to use our application not only on smartphones but in tablets and computers transforming our application into a cross-plataform one. f our application target would also be computers we could add an infovis module with two views, figure 8 and 9, that we built in low fidelity prototypes.

Figure 8 – First view

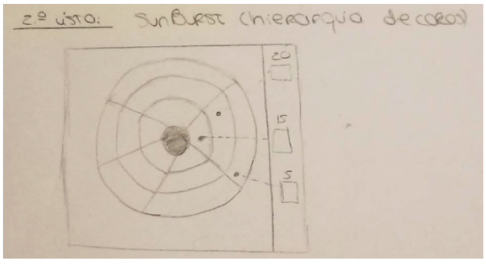

Figure 9 – Second view

Concluson

Developing an entire system is hard and time-consuming. We developed a complete solution that included not only the recommendation algorithm but the combination algorithm, the social network, the voice engine and the whole application development. We also gave attention to

the interface itself with the purpose to provide an application easy to use. Even though, it was designed and developed iteratively and incrementally and it is in its final version and if we wanted we could provide it as a final product and market it, there are obviously improvements that could be done.

Regarding the interface itself, it is in a final version being corrected after the heuristic evaluation and usability tests. We also developed a social network which main goal is to obtain inspiration thus making sense that it would be a platform with followers, not friends. The searching tool for users profiles is well defined, showing the username and his/her albums. The feed functionality shows whenever a user adds a piece or an outfit to an album or creates an album. The improvement that could be done would be giving the possibility to his/her followers to like a post. That improvement could interconnect with our recommendation and combination algorithm.

Nowadays, our recommendation and combination algorithm considers all the aspects that it should: temperature and weather of a specific location, emotional status, occasion and style. As explained before, our algorithm is divided into two parts: the first one being in the annotation phase and the other is also processed in two parts being the first one the result from performing queries in a user knowledge base and the other one the combination of pieces itself considering the chosen style.

We consider three styles but our implementation is open to receive more styles, it is only needed to analyse the ones to be added into our system and add those characteristics and rules in the annotation phase and combination.

REFERENCES

1. ADOMAVICIUS, G., AND TUZHILIN, A. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering 17, 6 (2005), 734–749.
2. ALEXANDER FELFERNIG, GERHARD FRIEDRICH, D. J., AND ZANKER, M. Recommender Systems Handbook, 1 ed. Springer US, 2011, ch. 6- Developing constraint based recommenders, pp. 187– 215.
3. BROOKE, J. Usability evaluation in industry 189 (1996), 4—-7.
4. BURKE, R. 2000, ch. Knowledge-Based Recommender Systems.
5. CHENG, C., CHUNG, M., YU, M., OUHYOUNG, M., CHU, H., AND CHUANG., Y. Chromirror: a real-time interactive mirror for chromatic and color-harmonic dressing. CHI 2008 (April 2008).
6. Fashion industries in the EU. https://brightside.me/article/ men-women-we-are-so-different-64555/.
7. CROLL, J. Fashion That Changed the World, 1 ed. Prestel, September 2014, pp. 6–8, 180–188.
8. DUMELJIC, B., LARSON, M., AND BOZZON, A. Moody Closet: Exploring Intriguing New Views on Wardrobe Recommendation.
9. Fashion and high-end industries in the EU. https://ec.europa.eu/growth/sectors/fashion/ high-end-industries/eu_en/. Accessed: 05-10-2016.
10. Fashion industries in the EU. https://fashionunited.uk/uk-fashion-industry-statistics. Accessed: 05-10-2016.
11. FUKUDA, M., AND NAKATANI, Y. Clothes Recommend Themselves: A New Approach to a Fashion Coordinate Support System. Proceedings of the World Congress of Engineering and Computer Science2011 1 (2011).
12. GABRIEL DE SOUZA PEREIRA MOREIRA, G. A. D. S., AND CUNHA, A. Comparing Offline and Online Recommender System Evaluations on Long-tail Distributions. Poster Proceedings of the 9th ACM Conference on Recommender Systems (2015).
13. GLEB BELIAKOV, T. C., AND JAMES, S. Recommender Systems Handbook, 1 ed. Springer US, 2011, ch. 22-Aggregation of Preferences in Recommender Systems, pp. 705–734.
14. GOLDBERG, D., NICHOLS, D., OKI, B., AND TERRY, D. Using Collaborative Filtering to Weave an Information Tapestry. Communications of the ACM 35, 12 (1992), 61–70.
15. ISINKAYE, F. O., Y. O. F., AND OJOKOH, B. A. Recommendation systems: Principles, methods and evaluatio. Egyptian Informatics Journal 16, 3, 261–273.
16. IWATA, T., WATANABE, S., AND SAWADA, H. Fashion coordinates recommender system using photographs from fashion magazines. Proceedings of the Twenty-Second international joint conference on Artificial Intelligence (July 2011), 2262–2267.
17. JONATHAN L. HERLOCKER, JOSEPH A. KONSTAN, L. G. T., AND RIEDL, J. T. Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems 22, 1 (2004), 5–53.

18. JURE LESKOVEC, A. R., AND ULLMAN, J. Mining of Massive Datasets, 1 ed. Cambridge University Press., 2015, ch. 9-Recommendation Systems, pp. 307–341.

19. KALANTIDIS, Y., KENNEDY, L., AND LI, L. Getting the look: clothing recognition and segmentation for automatic product suggestions in everyday photos. Proceedings of the 3rd ACM Conference on International Conference on Multimedia Retrieval (2013), 105–112.

20. LEE, H., L. S. Style Recommendation for Fashion Items using Heterogeneous Information Network.

21. LIU, S., SONG, Z., ZHANG, T., LU, H., AND XU, C., Y. S. Hi, Magic Closet, Tell Me What to Wear! Proceedings of the 20th ACM international conference on multimedia (2012), 619–628.

22. MCLAUGHLIN, M. R., AND HERLOCKER, J. L. A collaborative filtering algorithm and evaluation metric that accurately model the user experience. SIGIR '04: Proceedings of the 27th annual international conference on Research and development in information retrieval 17, 6, 329–336.

23. MCNEE, S. M. Meeting user information needs in recommender systems. PhD thesis, University of Minnesota, 6 2006.

24. Most used social network. http://www.businessinsider.com/ facebook-dominates-most-popular-social-media-apps-chart-2017-7. Accessed: 05- 08-2017.

25. NAGAO, S., TAKAHASHI, S., AND TANAKA, J. Mirror appliance: Recommendation of clothes coordination in daily life. HFT '08 (2008), 367–374.

26. NEIL RUBENS, D. K., AND SUGIYAMA, M. Recommender Systems Handbook, 1 ed. Springer US, 2011, ch. 23-Active Learning in Recommender Systems, pp. 735–768.

27. NIELSEN, J., AND MOLICH, R. ACM CHI'90 Conf. (1990), 249—-256.

28. PASQUALE LOPS, M. D. G., AND SEMERARO, G. Recommender Systems Handbook, 1 ed. Springer US, 2011, ch. 3-Content-based Recommender Systems: State of the Art and Trends, pp. 73–99.

29. Pinterest social network. http://fortune.com/2015/07/13/pinterest-ceo-ben-silbermann/. Accessed: 05-08-2017.

30. Portugal social networks. http://observador.pt/2016/06/29/ uso-das-redes-sociais-em-portugal-triplicou-em-sete-anos-mas-empresas-utilizam-nas-pouco/. Accessed: 05-08-2017.

31. RESNICK, P., IACOVOU, N., SUCHAK, M., BERGSTROM, P., AND RIEDL. GroupLens: an open architecture for collaborative filtering of netnews. ACM CSCW '94 (1994), 175–186.

32. SHANI, G., AND GUNAWARDANA, A. Recommender Systems Handbook, 1 ed. Springer US, 2011, ch. 8-Evaluating Recommendation Systems, pp. 257–298.

33. SHEN, E., LIEBERMAN, H., AND LAM, F. What am i gonna wear?: scenario-oriented recommendation. IUI (2007).

34. TINTAREV, N., AND MASTHOFF, J. Recommender Systems Handbook, 1 ed. Springer US, 2011, ch. 15-Designing and Evaluating Explanations for Recommender Systems, pp. 479–510.

35. XAVIER AMATRIAIN, ALEJANDRO JAIMES, N. O., AND PUJOL, J. M. Recommender Systems Handbook, 1 ed. Springer US, 2011, ch. 2-Data Mining Methods for RecommenderSystems, pp. 39–72.

36. YU-CHU, L., KAWAKITA, Y., SUZUKI, E., AND ICHIKAWA, H. Personalised Clothing-Recommendation System based on a Modified Bayesian Network.

37. ZHAO, Y., AND ARAKI, K. What to Wear in Different Situations? A Content-based Recommendation System for Fashion Coordination.

38.