GetDressed: A Personalized and Contextually-Aware Clothing Advisor for the Home

Zhijiao Liu zhijiaoliu@tamu.edu
Jesus Suarez jesussuarez@tamu.edu
You Wu urianawu@tamu.edu
Feiyu Yu ifeiyuyu@tamu.edu

Texas A&M University
College Station, TX 77843, USA

ABSTRACT

This paper presents our work on GetDressed, a contextually-aware clothing recommendation system. There is an unmet need for automated and personalized advice about the daily task of selecting an outfit. This task may seem small, but it has a real impact on people's lives and many people are unsatisfied with the outfits they select and the time and effort they spend deciding on them. The system we developed uses contextual information together with a personal user profile and wardrobe model to advise the user what outfits are appropriate given the day's events and weather conditions, the user's preferences and history, and a style coordination metric. We performed a two-part user study to evaluate our system and the outfit recommendations it produces. Comparing three conditions for each of the two modes in GetDressed (Virtual Wardrobe Builder and Outfit Advisor), we found that our full system is better liked than versions with key missing components, as well as versions with static and randomly generated content, though the difference is smaller than we expected. In evaluating the outfit recommendations, we found that they are considered more stylish than randomly generated outfits, but again, by a smaller margin than expected. Nonetheless, we conclude from the user feedback that GetDressed can be useful to many people.

INTRODUCTION

Every morning, people everywhere check the weather, consider their schedule for the day, and think about what to wear. What to wear is perhaps a small decision, but it is one that is important to many people and has the potential to affect anyone's day. From early data gathering, we determined that many people are in a position where they care about what they wear and how they present themselves, but feel that they lack the time and expertise to create outfits they consider fashionable. Many of the potential users we spoke with expressed that they would appreciate having easy access to personalized clothing advice as part of their daily routine, even if the advice is imperfect.

Many clothing recommendation systems exist – especially for fashion advice and virtual shopping – but none fully addresses the basic needs of the morning outfit selection routine. Most existing systems either suggest clothes to buy or allow one to share or record outfits they have worn, but do not provide recommendations about one's own clothes, much less take into account contextual information to make such a recommendation. On the other hand, contextual notification systems (e.g. for weather and calendar events) provide useful information, but do not leverage this information to produce tangible advice. Additionally, most systems don't make recommendations from the user's own wardrobe, making it difficult to make their advice actionable (especially as part of a daily routine).

We've built a new system, called GetDressed, which integrates live contextual information with a personality profile of a user and a model of their wardrobe to generate outfit recommendations to meet their needs. The system utilizes web-sourced metadata to populate the user's wardrobe model, thus alleviating them of tedious manual data entry to make the system work. GetDressed helps users express themselves through their clothing, simplifies an important part of their daily routines, and helps them find new styles tailored to their preferences. The goal is to help users feel more confident in making fashionable outfit decisions, as well as help them be more expressive and adventurous in what they wear.

GetDressed consists of two main modes: a virtual shopping mode, and an outfit advisor mode. The first is used to quickly populate a virtual wardrobe with clothing similar to what the user actually owns. This eliminates the need for tedious hand labeling, though at a cost of reduced realism. The wardrobe that the user builds is populated from online sources, and from these sources metadata attributes about the clothing are also gathered. These attributes, along with history information make up the wardrobe model of the system, which is used by the outfit advisor mode to actually produce recommendations based on what the user likes, as well as the day's conditions, as determined from two contextual sources: weather, and the user's calendar events. Rudimentary stylistic metrics also impact the recommendations produced.

We hypothesize that users will consider the complete GetDressed system to be useful to them, that the virtual shopping mode will be regarded as a easy and practical way to build a personalized virtual wardrobe, and that the outfit recommendations that the system generates are appealing to users. We evaluated the GetDressed system and the recommendations it produces in a two-part user study with

in-person and online subjects. We conclude that GetDressed can be valuable and useful to many people, especially people who lack confidence in making their own stylish outfits. However, further personalization options and access to greater clothing variety may be needed before GetDressed can make a sizeable impact on people's lives.

PRIOR WORK

Several existing systems and publications are related to GetDressed. A common these in these works is the understanding that many people have a desire for clothing recommendations, so similar needs and expectations are addressed by the various systems. In each work identified here, there are aspects that we found valuable and which influenced components of the system we developed. However, we also identify points of departure from how these systems address some of the needs of the users and use cases we envision. We've organized this related work into three categories: specialized clothing recommendation systems, context-aware recommendations, and clothes matching and similarity. Here we discuss them and their relationships to each other and to GetDressed.

Specialized Clothing Recommendation Systems

In specialized clothing recommendation systems, there exist several systems that provide recommendations for specific occasions or situations. Liu et al. [8] and Shen et al. [12] for example, developed systems that make clothing recommendations for specific occasions or scenarios, such as weddings or dating (Figures 1 and 2, respectively). In each case, user input was used to define the situation. These systems are conceptually similar to ours, especially with respect to GetDressed's context-aware recommendations, but they differ in that they only consider context in a single dimension. GetDressed will incorporate contextual information from both weather and calendar events in making its recommendations, as well as a user profile and wardrobe model.

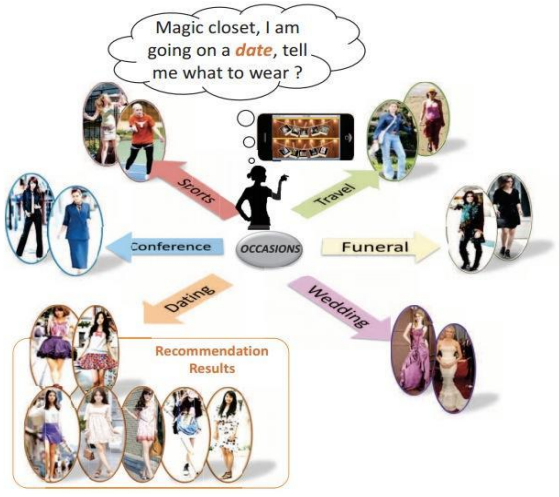


Figure 1: The motivating problem of finding an appropriate outfit for a specific occasion, from "Magic Closet" by Liu et al. [8]

Table 1: Example results of the fashion recommendation system

I am going to the beach	
I am going to have dinner	
I am going to have dinner. I want to look more casual.	
I am going to have dinner with my boss	

Figure 2. A sample recommendation set from "Scenario-Based Recommendation" by Shen et al. [12]

Non-academic examples of clothing recommendation systems include SuitUp![1], a social fashion-sharing and recommendation system, and Civia's What To Wear guide[2], a static catalog of garment recommendations for cyclists. Civia's What to Wear example (Figure 3) is interesting because, though static, it incorporates a feature that we hope to integrate into our own system: expert knowledge. In GetDressed, there is a need for a notion of fashionability and coordination quality to produce good recommendations and to evaluate outfit options. The knowledge for producing these metrics will come from evaluative qualities elucidated from fashion experts and formalized into a set of measurable properties and rules.

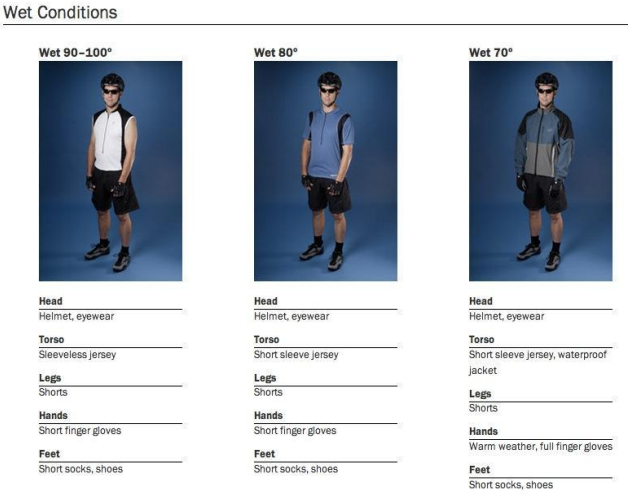


Figure 3. Static outfit advice for different weather conditions from Civia Cycle's What to Wear page[2]

Context-Aware Recommendations

Understanding contextual information and using it intelligently is a concept that extends to many areas. Campos et al. [1] for example, uses information about time and social company to make movie recommendations. Hong et al. [6] describe a context-aware personalization framework that is applicable to many activities including shopping and dining. Chen [3] meanwhile, describes a collaborative filtering system that uses contextual

[1] http://www.kerrykao.com/suitup
[2] http://www.civiacycles.com/whattowear

information to predict user preferences. And in a patent filed by Chakravarty and Jain of IBM [2], a system is described for making recommendations for wearable attire and digital program recording based on calendar events.

This last example is most similar in function to GetDressed, and it is in fact one of very few systems we found that actually makes attire recommendations based on a user's own wardrobe (rather than an arbitrary collection, as in a virtual shopping example). Though this patented system requires embedding RFID tags in clothing to work, it highlights the need for a better and more portable solution. Embedding RFID tags (or similar markers) in clothing is far too big of a burden to place on users, and is precisely what we try to avoid by using online-sourced metadata to collect the information we need about a user's wardrobe.

Clothes Matching and Similarity

A critically important consideration in ensuring that our system produces valuable outfit recommendations is the need for a computational model of garment matching and similarity. Prior research in this area includes work by Zhang et al. [14] and Kim et al. [7] on clothes matching based on color harmony and texture analysis, as well as Cheng et al.'s ChroMirror system [4], which helps users explore clothing color combinations. These systems considered real-world lighting and despite inconsistencies

Figure 4. Images tested with a Color harmony measure incorporating shape information from Zhang et al. [13]

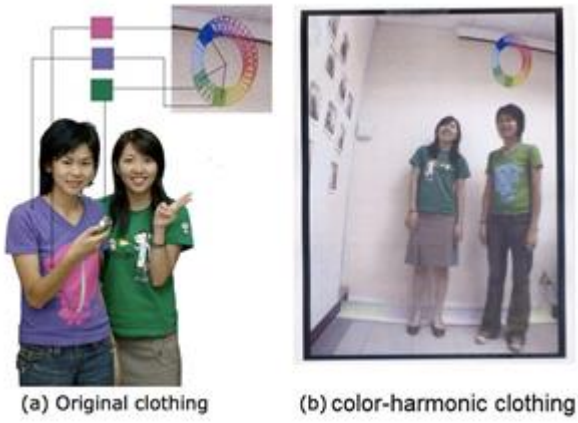


Figure 5. A comparison of clothing colors in ideal vs real conditions from ChroMirror by Cheng et al. [4]

DESIGN PROCESS

GetDressed was built using iterative design and development. We began with ethnographic data gathering to determine the needs and requirements for our system, then developed motivating scenarios, storyboards and a low-fidelity prototype to elicit user feedback about the direction we were going. This feedback was acquired in a first user study and from it we made several design changes and changed focus for the development of our final functional prototype.

Ethnographic data gathering

In early data gathering, we focused on understanding user's habits and desires. We determined several main points from data elicited from potential users: Users have an intuition for clothing and outfits that they think look good, even if they do not consider themselves fashion-centric. Users have much more trouble generating good-looking outfits, even if they do consider themselves fashion-centric. The most fundamental reasons for outfit choices that users make are functionality (e.g. rain boots), appropriateness for an occasion (e.g. formal attire), and laundered state of clothing. Users frequently observe others' clothing, and are often inspired by what they see.

Scenarios

We developed three motivating scenarios to help us understand the needs and requirement of our system, and how users might interact with it and use it in their daily lives. From these scenarios, we sketched storyboards to show to users and elicit feedback about them.

Scenario 1. A student is accustomed to wearing whatever clothing he happens to have at hand. One day, he dresses poorly for an interview, suddenly becomes self-consciousness about his attire, which affects his self-confidence, and he ultimately does poorly in his interview. He began using GetDressed, and benefited from the system's recommendation because it always reminds him of the day's calendar event and gives him the appropriate suggestions for how to dress for the day..

Scenario 2. A busy single mother raising two children worries about how to dress her children with the limited time she has every morning. She is especially concerned with picking appropriate outfits for the day's weather. Last time she didn't check the weather forecast, she didn't dress her children warmly enough for a very chilly cold front that passed through that afternoon. She began to use GetDressed, and appreciates that by simply glancing at her phone for a few seconds, she can see the day's weather and an easy list of suggestions for how to dress her children. GetDressed also makes it easy for her to make sure her children always look their best, by not repeating outfit suggestions too frequently.

Scenario 3. An aspiring fashionista, who already knows a lot about fashion rules, uses GetDressed because it makes it

easy for her to get quick feedback on some of her fashion ideas. The system's recommendations – though not perfect – often provide her inspiration. She also likes to use GetDressed to very quickly visualize many different outfit combinations.

Lo-fi prototype description

We developed a low-fidelity prototype that covered the main points of our system, but with limited functionality. The prototype include 3 modes: a Profile Input Mode, an Outfit Advisor Mode, and a Virtual Shopping Mode.

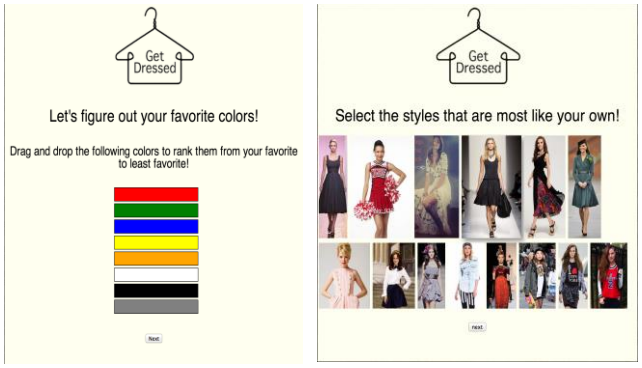


Figure 6: In the profile input mode, a users inputs some basic information about themselves and their preferences about colors and clothing styles.

Figure 7: In the outfit advisor mode, a user is presented with 3 recommended outfits and can switch between them. Dummy weather and calendar information are displayed, as well as a mood input and placeholders for like/dislike and customize buttons.

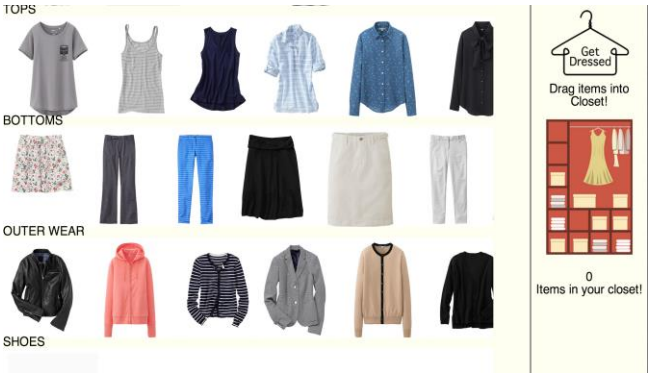


Figure 8: In the virtual shopping mode, users are presented with a catalog of clothing images in several categories. They can drag clothes they like into their virtual wardrobe, and see the items they're already added.

Feedback on scenarios and lo-fi prototype

We showed the scenarios we developed to a set of users in the form of storyboards. The feedback we received can be summarized in two main points: Most users agreed that GetDressed would be helpful to the characters in the scenario, and they associated themselves at least partly with one of these characters, showing that the system we envisioned may have broad appeal. And users agreed that the student character would benefit most from our system, providing supporting evidence for our belief that fashion-unaware users may benefit the most from our system.

Feedback we received about the low-fidelity prototype was useful to us in updating our design decisions for GetDressed. There was fixed responses concerning the color ranking input and favorite style selection components of the profile input mode, so we made the decision to refocus our efforts on the two other modes of GetDressed: the virtual shopping mode and the outfit advisor mode. Our users generally liked the advisor mode, particularly the overall layout and the outfit visualization. Some users didn't think mood would be a useful input, and we ultimately agreed to lower the priority of this contextual input in our development plan. Most users also liked the virtual shopping mode, and expressed suggestions for improvements, such as adding a search option and the ability to see the wardrobe items in categories. Both of these features were added to our design spec and made it into our final functional prototype.

User Study 1

After implementing some of GetDressed's functionality (particularly gender- and style-specifications), and making some of the changes requested in our user feedback, we ran a first user study to more fully evaluate our prototype to that point. We focused on answering several several questions: How do the users decide the clothing they buy and wear? Do users think our system will be useful? Are

there any additional requested features that make sense to add to our system?

The most commonly reported sources of inspiration for fashion were TV/movies, and what they see other people wear in public, but we saw that our subjects varied highly in how often they shop for new clothes. One cluster of subjects shop about once every month or two, while another group reported shopping about once or twice a year or "very rarely". Additionally, most of our subjects reported that they shop for clothes online, but only rarely. The most commonly reported factors influencing clothes buying were price, style, comfort, fit, and practicality.

Almost all of our subjects reported that they would find it "helpful" or "very helpful" to have easy- to -access clothing advice. They also said that they'd be equally comfortable or more comfortable receiving such advice from a program than from a person. When asked whether this advice would still be helpful if it came from a program with lower-quality suggestions than a person, most of the subjects responded that it would still be "helpful" to "very helpful", though two responded that it would not be helpful if that were the case. Most subjects responded that they would like such a program to run on a mobile device with a touch screen. We decided to ensure that the system does in fact work on mobile devices.

Design Revisions

Following feedback from our first user study, we decided to again re-evaluate the focus for our functional prototype implementation. We left the profile input mode unchanged to work on the remainder of the system. For the outfit advisor mode, we focusing on making sure we can generate likeable outfit based on both contextual information and stylistic metrics. For the virtual shopping mode, we focused on ensuring that users can use the mode to build a virtual wardrobe they are happy with.

Figure 9: A simple icon/button change that addressed a breakdown in usability

One simple problem with our design which was revealed in the user study is we needed better navigation affordances in the virtual shopping mode. In the early prototype, we use a logo image to direct the user back to the other modes. However, most users didn't realize they could click the logo image and didn't know what to do when they finished choosing clothes for their wardrobe. To address this affordance breakdown, we add a button called "done" for the user to click when they finish building their virtual wardrobe.

In addition to implementing the search feature in our virtual shopping mode, another change we made was displaying categories in the virtual wardrobe view. We also revised our clothing categories in response to feedback from our users saying that the prior categories were too coarse. In the outfit advisor mode, we focused primarily on ensuring the recommendation engine incorporates the metadata we extract.

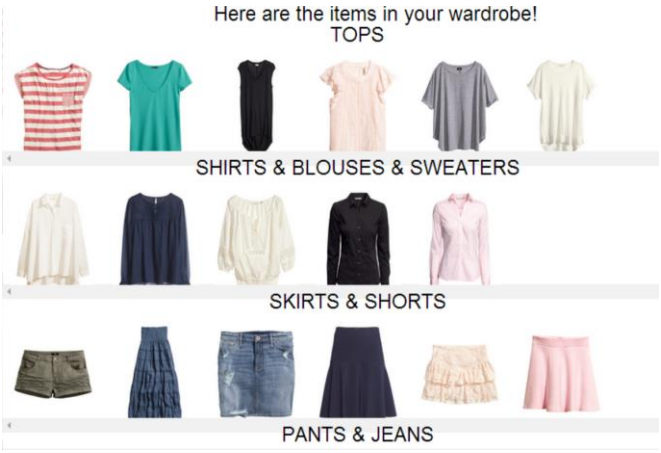


Figure 10: The category view we added to the virtual wardrobe visualization.

There are also some things we decided not to implement due to time constraints and less attention from users. We didn't implement mood contextual input, as it was not well-received in early user feedback, and implementing it would have added substantial complexity. We also didn't implement outfit customizations, due to time constraints. Outfit history scores were implemented and are evaluated by the recommendation engine, but we did not implement history score updates according to calendar events, since we knew that the participants in our user study would likely not have enough time with the system to repeat outfits.

METHODS and IMPLEMENTATION

GetDressed has two modes of functionality. First is the automatic daily outfit advisor that incorporates information about weather conditions, calendar events, user preferences and history, and a metric of color, style and texture matching to produce outfit recommendations generated from a model of the user's own wardrobe. And second is an on-demand virtual shopping guide that will over time produce shopping suggestions based on the user's dressing habits and preferences. This mode is used at this point in the prototype for building the user's virtual wardrobe, and we have not tested it for virtual shopping after the wardrobe is built.

Outfit Advisor Mode

Figure 11: Outfit Advisor mode of GetDressed

The first – and most visible – mode of GetDressed is the outfit advisor (Figure 11). GetDressed will check the weather and the user's calendar events for the day and uses these contexts together with other information to generate a set of outfit recommendations and present visually them to the user. The user can then quickly swipe through the list of recommendations and select one to wear if they like. The weather and calendar information being evaluated are prominently shown alongside the outfit suggestions.

Virtual Shopping Guide Mode

Figure 12: Virtual Shopping mode of GetDressed

The second interactive mode in GetDressed is the virtual shopping mode (Figure 12). This mode is used to select items for the user's own wardrobe from on online catalog, and build a metadata-rich wardrobe model for the recommendation system to use in the outfit advisor mode. Though it is understood that no clothing catalog will be able to contain exact matches to all the clothes in a person's wardrobe, we believe that close approximations suffice to produce useful recommendations. The rationale behind using a virtual shopping system to build the users wardrobe model is that by doing so, we can extract useful metadata about the clothing from the catalog source, and can avoid requiring the user to tediously enter information about their clothes.

System Architecture.

Contextual Information Sources

Three types of contextual information are gathered to produce useful outfit recommendations to the user: weather status, calendar events, and the user's mood. The first two of these are collected automatically from web APIs, while mood is left optional and is elicited from user input. To collect weather information for the user's location, we use a simple jQuery plugin called SimpleWeather. It provides geolocation information, temperature, precipitation, cloud coverage, and wind speed. Calendar events are gathered from Google's calendar API. Only the events for today's date are gathered. Mood is optionally selected by the user from a graphical prompt that shows emoticons for a set of moods. The information from all three context sources are shown visually to the user and stored in HTML5 local storage for access by the rest of the GetDressed system.

Wardrobe Model and Metadata

The next module of our system is the wardrobe model and its associated metadata. To populate the user's wardrobe model, we use GetDressed's virtual shopping mode. Each item in the virtual shopping mode consists of an image for the clothing item, and some metadata associated with it. This metadata currently consists of the intended gender of the item, its category (t-shirt/dress shoes, etc.), color, and material, and a history score. To collect this metadata, we use a custom set of BigSemantics wrappers. We developed several wrappers specifically designed for online clothing sites. We use these wrappers to automatically extract image source url, clothes category, color, description, and material.

Since color information has a large range of variety, and it's not easy to be categorized, we came up with a color palette generator. The generator takes an image file as input and outputs the dominant color and color palette for this image. The color values are then transformed and stored in HSV format in order to facilitate future processing. Our color palette generator code is based on Color Thief .

Outfit Evaluative Metrics

This module of the GetDressed system consists of two kinds of metrics: comparative metrics and similarity metrics. Comparative metrics for color, category, and material are used to determine evaluative scores for outfit candidates. A similarity metric is also included in order to recommend similar clothes from online shops. The similarity score considers the color of the clothing

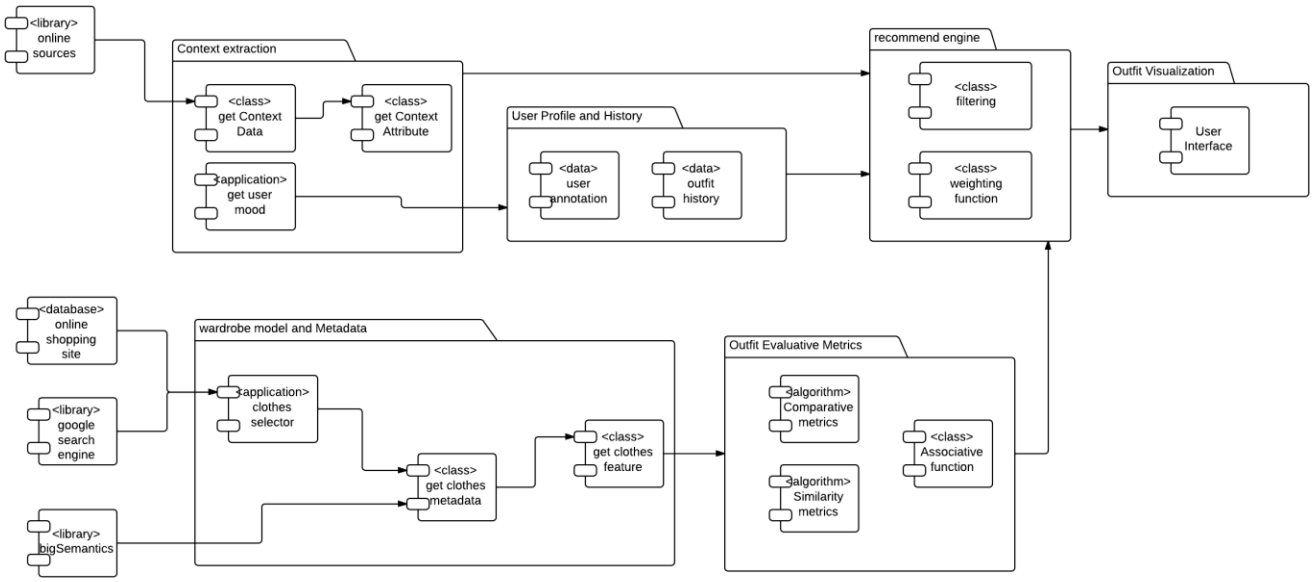


Figure 13: System Architecture diagram for GetDressed

elements, their materials, and the categories they fall into. The categories are designed such that clothing formality (such as shirt sleeve length) is incorporated into the categories.

User Profile and History

We store and use attributes about the user's profile and preference, and their outfit history as part of the recommendation system. The user's profile is first generated from user input, and their preferences are determined in two ways. First, during the profile input stage, the user is asked to select their favorite colors and favorite types of outfit styles. These direct inputs are used to initialize their preference profile. The second way the user's preferences are gathered is through their interaction with GetDressed. When they select an outfit, when they customize an outfit, and when they "like", "dislike" or make an annotation about an outfit, we update their preferences accordingly.

The user's outfit history is stored in two ways: per clothing item, and for complete outfits. A clothing item's clothing history is stored as a history score, which represents how frequency and recently it has been worn. An outfit's history is stored similarly, but it includes links to the individual items it consists of. The user's profile and history are currently stored in HTML5 local storage, but will be moved to a server-side database.

Recommendation Engine

This is the module of our system that actually produces outfit recommendations. It consists of two parts: an outfit generator, and an outfit evaluator. The generator produces outfit candidates from the items in the user's wardrobe. The items are first filtered according to the contextual information gathered from the context extraction module, and to reduce the number of outfit candidates to produce, the process is seeded with prior liked outfits and new outfits are only generated as they are needed by the evaluation step. In the outfit evaluation step, an outfit candidate is scored according to the user's preferences and history, and the evaluative metrics from the outfit evaluative metrics module. An outfit's score will be a weighted sum of scores from the clothing attributes collected from the wardrobe metadata module, and these weights will be updated overtime and associated with the user's profile.

Outfit Visualization and User Interface

Finally, the outfits that are scored highest by the recommendation engine are shown to the user as part of the web application. The visualization consists of images of the outfit's clothing elements arranged to show the entire outfit. The web application can be run on both pc and mobile devices, and the user interface is built in JavaScript with jQuery plugins.

EVALUATION

The evaluation of GetDressed consisted of two main methods: system evaluation and recommendations evaluation. The goal of the system evaluation is to evaluate users' satisfaction with the wardrobe model and the system's recommendations. For the recommendations evaluation, we try to get a broad sense of the quality of our system's outfit recommendations, especially considering whether they are regarded as fashionable.

System Evaluation

To evaluate user experience and whether users find GetDressed compatible with their own fashion sense, we observe the user interacting with our system and then

collect questionnaire results afterward. This process is divided into four steps:

1. Give user the task of building a virtual wardrobe that is meant to resemble his or her own wardrobe, while observing the user's interaction with the system.
2. Have the user fill out the first post-questionnaire (containing mostly Likert and semantic differential scales) about their experience with the wardrobe model.
3. Move on to the advisor mode, give user the task of choosing one outfit to wear today, while observing the user's interaction with the system.
4. Have the user fill out the second post-questionnaire containing the questions about their experience on advisor model and satisfaction of system's outfit recommendation.

In total, we had 46 participants for this part of the evaluation. Eight of these participants we ran in-person under direct observation, and the remainder were run remotely and independently through Amazon Mechanical Turk. We found that the participants were mainly students, professors, and recent graduates who are working.

We created six study conditions to assess key parts of our system (three for each of the two major modes of operation in the system). The study is performed between subjects, and conditions to be run were selected randomly. The six conditions and what they evaluate are described in Table 1. The dependent variable we measure in each case is how much the user likes the given system functionality. We also measure the user interaction time, and gather open-ended responses about their interaction.

Recommendations Evaluation

In the recommendations evaluations, we ask users only to rate individual outfit candidates. This evaluation attempts to balance one person's clothing tastes against those of the public. The users here only need to see the recommendation outcomes and will provide their subjective feedback to each one. By randomly showing them our recommendation outfit result and randomly generated outfit result, and letting them rate these outfits, we can get quantitative data on how our recommendation performs.

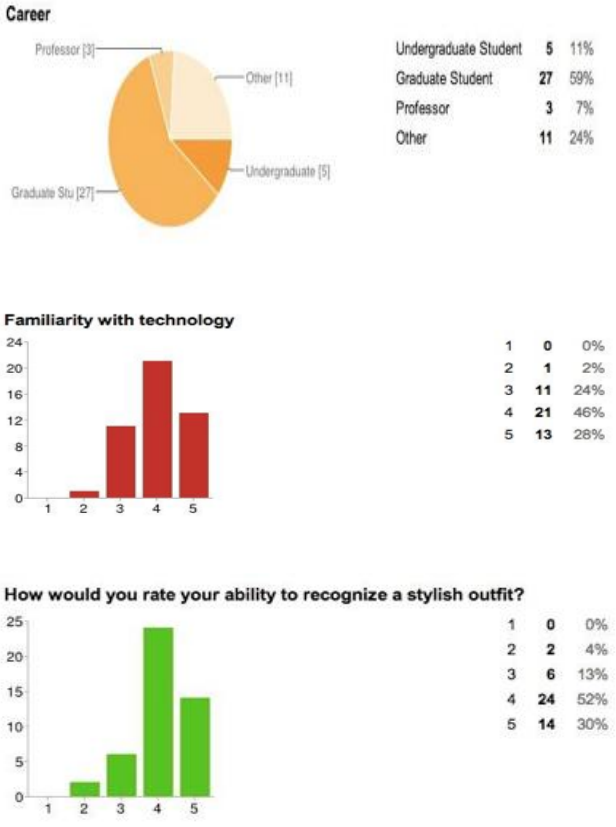
There are 74 participants in this crowd-sourced evaluation. Nine of these participants were contacted directly, and the remainder was from AMT from all around the world. Again, the participants were mostly young (18-35) students and workers. For this part of the study, there are two conditions: outfits generated by our system's recommendation engine, and randomly generated outfits. This study is within subjects, with randomly selected outfits presented to the user to evaluate. The dependent variable is the rating for outfit represented by Likert scale. By calculating and aggregating them, we can find out how our system performs. The only further information we gathered was a self-assessment asking "how would you rate your ability to recognize a stylish outfit?"

Results

For user evaluation, our data comes from survey which contains 6 parts: backgrounds, shopping and outfit habits, questionnaire on wardrobe mode and advisor mode, interaction, and general feedback. Most questions give us quantitative data while the rest are user comments that can be considered as qualitative data. For all the diagrams, the horizontal coordinates denote the scale or rating (from 0 to 5); the vertical coordinates denote counts (number of participants). Here, we examine the most significant results.

Users' Background:

Most of the participants' ages are between 18 and 35. It can also be inferred from the distribution on career and familiarity of technology. Male and female participants are evenly distributed. 59% of the participants are graduate students, and 74% claims to be very familiar with technology. So we can suggest that these are real potential users of our online clothes recommendation system. Regarding to shopping habits, more than half of the participants claim that they have above average fashion knowledge, and about 30% are very good with fashion. So this fits our system's objective, to make it useful for both fashionista and common people.



Wardrobe Model:
User's response to male wardrobe and female wardrobe don't differ much, so here we'll use male wardrobe to analyze the user response. Mean and Deviation of user ratings of system under different conditions are shown in the following table.

Condition	Mean	Deviation
Without Search	7.44	0.8464
With Search	7.61	1.2581
Static closet/ without customization	7.59	2.2219

Table 1: User responses concerning usability of different Wardrobe Model conditions

By comparing the mean value, we can conclude that users prefer wardrobe model with search function than one without search. However, static closet has higher mean value and larger deviation than dynamic closet without search function. This suggests that users have diverse opinions on static closet. 74% of users finds it more useful if they're able to search for more online clothes, even when the search result is not optimized.

Advisor Mode:
Mean and Deviation of user ratings of system under different conditions are shown in the following table.

Condition	Mean	Deviation
Full recommendation	7.39	2.7179
No context	6.95	4.6705
Random recommendation	7.21	2.3259

Table 2: User responses concerning usability of different Advisor Mode conditions

By comparing the mean value, we may find that user actually prefer random recommendations. One possible explanation may be the limitation of clothing data, since we extract our clothes from one website, therefore, even random results don't seem unreasonable. However, the following pie chart shows that 65% of users find the system with full recommendations more useful.

System Interaction:
Most users spend less than 5 minutes on interaction with advisor mode, and they spend less than 6 minutes on wardrobe mode. User's rating for all these three parts of our system are similarly distributed. Most responses lie within the range of 3 to 5. More than 50% of users vote for 4 "above average", and approximately 30% of users vote for 5 "good".

Many users commented that the interaction with the system was easy - in particular, easy to navigate and search. However, there were concerns that clothing choices is limited. Users commented that the outfit recommendations were helpful, looked good, and usually matching well. Users found our profile input easy to use, and felt that our system know them better after they finish the profile input part. Some users however were annoyed that they had to input information about themselves before using the system.

Quality of Recommendations.
For the recommendations evaluation, we received 74 valid responses. From these data, we ran the single-tail, mixed-variance T-test, trying to figure out whether our recommendation is better than random result. We have a small effect size (0.17), but strong statistical significance ($p = 0.034860787$ for mean-only comparison, and $p = 0.006507725$ for direct value comparison). Since $p <= 0.05$ is considered statistically significant, and less than $p = 0.01$ is considered very significant. It confirms that our users liked the actually recommendations more than the random recommendations, though only a very little bit more. However, we are sure that they definitely liked our system more. There is only a 0.65% chance that those preference results were due to random error.

DISCUSSION

Firstly, most people found our system is helpful. Over, 67% people think our system is useful to decide what to wear for the day's weather and events and your mood. And there are still over 50% of participants think our system can provide the help to be more stylish and expressive. Participants liked advisor mode more than wardrobe mode. The reason may be setting up wardrobe takes time. Responses for different conditions in our evaluation tended toward high scores (averaging about 7 overall), and we were surprised to see that the difference between the conditions in the virtual shopping mode and the outfit advisor mode were very small (Tables 1 and 2). We attribute these differences to bias in our study methodology and

For virtual wardrobe mode, one significant thing is searching function is really useful. This function remains me think of an instant messaging application called Tencent QQ. This instant messaging software has a great function of adding stickers from searching Internet. You can input the description of the stickers you want and the search result can be added into your own stickers gallery. I still remember when this new function of searching stickers has been put online, many of users of this software using this function. The feedback of this function is really great. That's why we need searching function for our recommendation system. User needs their ability of changing default wardrobe.

We were surprised that the perceived difference between randomly generated outfits and those produced by our system were so low. We attribute this small effect size to a low level of personalization in the prototype, and the fact that the limited clothing dataset we used includes mostly stylish clothing and it could be argued that almost any outfit generated from it may look good. To improve on the quality of the outfit recommendations, we need to have more personalized information about the user - perhaps information about fabrics, patterns and textures they like, and more parseable information about style categories they like. Additionally, metadata parsing of more information sources would be needed to ensure that the clothing that users actually want to see in their wardrobes can be evaluated correctly by the recommendation system.

CONCLUSIONS

We've present GetDressed: an outfit recommendation system that uses both contextual information and a user-generated virtual wardrobe to provide actionable outfit suggestions. We do this by filtering out the clothing items that are inappropriate for the day's weather and calendar events, then scoring a set of outfit candidates with evaluative metrics based on basic color harmony and clothes matching rules, and outfit history to produce a small set of recommended outfits. A key innovation of GetDressed is that it uses an easy-to-use virtual shopping mode to populate a wardrobe model for the user so that outfit recommendations can be similar to clothing they actually own, and therefore the recommendations are actually attainable.

We determined in our evaluation of the system that many users would find it helpful and that they find the virtual wardrobe model as a practical and acceptable way to tell GetDressed about the clothes they own without having to spend a long time manually entering metadata themselves. However, we also found that in order to make a bigger impact on people's lives, and in particular in order to to produce more likeable outfit recommendations, we'd need to improve the personalization options in GetDressed, and be able to extract relevant clothing metadata from more sources.

As a system investigating the use of contextual information from multiple sources to produce personalized recommendations, we believe GetDressed may be a valuable contribution to the expert systems research community. Additionally, we believe that the virtual shopping mode method for building a metadata-rich wardrobe model without demanding too much of the user is a promising endeavor that could use further exploration. Since most data sources do not provide relevant metadata that could be useful to a system like GetDressed (clothing color, material, patterns, texture, sleeve-length, etc.), we believe that this could be an interesting problem for computer vision research, particular from the fine-grained visual categorization sub-community.

FUTURE WORK

Ideas for future work on GetDressed fall into three categories: data extraction, personalization, and live visualization. Automatic clothing metadata extraction from different sources (in particular from search results) is needed for need for the recommendation system to evaluate the clothing that people want to see if. No one source will be broad enough to cover every user's needs, and many sources do not have associated metadata for their catalog items that could be useful to us. Additionally, using the user's mood as a contextual input of our recommendation system is something that we purposefully backpedaled on due to the added complications of understanding user intent and producing computational rules for the effect of mood on desired clothing style. This contextual source and other potential ones (e.g. social influence) would also require a deeper understanding of psychology and sociology than we have available to us at present.

The personalization and customization functions in our system have significant room for improvement. For example, users can recombine the outfits if they don't like the existing recommendations. And additional information about the user can be leveraged in the outfit advisor - such as their thoughts on different fabrics, patterns and textures, as well as more granular information style categories they like.

Finally, providing a live visualization of a recommended outfit is something that we thing would be very valuable to users and could significantly augment the power of our system to allow users to try out different outfit combinations and shop for new clothes. Such live visualization of outfits would be displayed superimposed on users' bodies using visual body tracking, such as that made possible by the Microsoft Kinect and other systems.

ACKNOWLEDGEMENTS

We would like to thank Rhema Linder, Andruid Kerne, and Nic Lupfer for their helpful comments and suggestions throughout the design and development of GetDressed.

REFERENCES

1. Campos, P. G. et al. "Context-Aware Movie Recommendations: An Empirical Comparison of Pre-Filtering, Post-Filtering and Contextual Modeling Approaches." E-Commerce and Web Technologies. Springer Berlin Heidelberg. 2013.
2. Chakravarty, P. and Jain, A. "Calendar based personalized recommendations." U.S. Patent Application 12/147,597. 2008.
3. Chen, A. "Context-aware collaborative filtering system: predicting the user's preference in the ubiquitous computing environment." International Workshop on

Location- and Context-Awareness. Springer Berlin Heidelberg. 2005.
4. Cheng, C. M. et al. "Chromirror: a real-time interactive mirror for chromatic and color-harmonic dressing." CHI'08 Extended Abstracts on Human Factors in Computing Systems. ACM, 2008.
5. Giovanni, S. et al. "Virtual Try-on using Kinect and HD camera." Motion in Games. Springer Berlin Heidelberg, 2012.
6. Hong, J. et al "Context-aware system for proactive personalized service based on context history." Expert Systems with Applications 36.4. 2009.
7. Kim, D. et al. "A scoring model for clothes matching using color harmony and texture analysis." Graphics Recognition. New Trends and Challenges. Springer Berlin Heidelberg. 2013.
8. Liu, S. et al. "Hi, magic closet, tell me what to wear!" Proceedings of the 20th ACM international conference on Multimedia. ACM. 2012.
9. Moon, P. and Spencer, D. E. "Aesthetic measure applied to color harmony." Journal of the Optical Society of America, 34(4), 234-242. 1944.
10. Ou, L. C and Luo, M. R. "A colour harmony model for two-colour combinations." Color Research & Application 31.3. 2006.
11. Pereira, F. et al. "Virtual Fitting Room Augmented Reality Techniques for e-Commerce." ENTERprise Information Systems. Springer Berlin Heidelberg, 2011.
12. Shen, E. et al. "What am I gonna wear?: scenario-oriented recommendation." Proceedings of the 12th international conference on intelligent user interfaces. ACM. 2007.
13. Zhang, Q. et al. "A color harmony measure model with shape information." International Joint Conference on Computational Sciences and Optimization. IEEE. 2009.
14. Zhang, W. et al. "An intelligent fitting room using multi-camera perception." In Proceedings of the 13th international conference on Intelligent user interfaces. ACM. 2008.
15. Zhou, Z. et al. "Image-based clothes animation for virtual fitting." SIGGRAPH Asia 2012 Technical Briefs. ACM, 2012.