Hotei: Stress Management Platform via Activity Recommendation

Tim K. Chan, Ryan Dowse, Akshay Garigiparthy, Sagar Patel, Nick Robertson

***Abstract*—Stress and general emotional well being, are important issues in the context of student life where around 63% suffer from stress, and 27% of all students suffer from a mental problem of some sort. This pilot study introduces Hotei, a wearable stress detection system that uses personalised recommendations to help reduce user's stress levels. Hotei was tested over a period of three days on a selection of Imperial College London students. Our initial findings were inconclusive given the hardware and time constraints to test the system as a whole. However, preliminary experiments on the system's subcomponents indicate that mobile devices are suitable tools for detecting and managing stress.**

I. Introduction

In a market that is increasingly populated by wearables [1] and apps targeted at improving aspects of physical well being - namely fitness, sleep quality and nutrition, conversely there are very few solutions targeted at improving consumers' emotional well being. In relation, an increasingly relevant concern which can have a strong negative effect on emotional well being is stress. These are issues commonplace for students, where 27% of all students suffer from a mental health problem, and 63% of students report suffering from stress levels that interfere with their day to day lives [2].

Stress is a term used to generalise the human body's response to a demand or a threat. Such demands can include the pressures of work, a sudden negative change in a persons life, or trauma from a major incident. In the short term, stress can be a motivator however, the survival response activated by stress can cause health problems if prolonged. Continued strain on the body can cause problems such as heart disease, high blood pressure, diabetes and mental health conditions such as anxiety and depression. The latter conditions are, in fact, the most common conditions for students with mental health problems [2].

There are steps one can take to reduce or manage their stress levels so that they can avoid long term physical & mental harm. As stated by the National Institute of Mental Health [3], such steps include: recognising the signs of stress, exercising, trying a relaxing activity or staying connected with people who can provide emotional support. However, since each individual handles stress in different ways there is currently no definitive method, for combating stress, that can be recommended to them.

Just as awareness has increased of the importance of fitness, sleep and nutrition with regards to overall physical well being, it is important to address the issues of stress management and provide greater awareness of an individuals emotional well being. This project involved the development of a smart phone app which works in conjunction with a wearable device to detect when a user is stressed based on their heart rate. It then offers tailored recommendations of activities a user can do to help manage such stress. Tailored recommendations are provided to better suit an individual's preferred activities to deal with stress. It is through such methods to help manage stress and by correlating a users emotions with activities they perform throughout the day, that an individuals overall emotional wellbeing will be positively affected.

II. Hypothesis

The following hypothesis is proposed: that through the use of a personalised activity recommendations system the user's frequency of stress will be reduced.

III. Background and Related Work

A. Heart Rate Variation and Stress Detection

The most widely used methods for measuring a subject's heart activity is through **Electrocardiography (ECG)** and **Photoplethysmography (PPG)**. ECG is the process of recording the electrical activity of the heart, for a given subject, over a period of time with electrodes placed on the skin. PPG reads the subjects heart rate using a pulse oximeter which illuminates the skin and detects the changing levels in light absorption in order to measure the rate of blood flow. In recent years, this method has become more widely featured in wearable technology, such as the Apple Watch. An overview of the history of PPG and recent developments in wearable pulse rate sensors is reviewed in [4].

It has been shown that there is a correlation between psychological stress and an increase in heart rate (HR). An example of such a paper is [5] where participants are tested for psychological stress, with the general findings that HR consistently increased among the different test groups. Similarly, in a trier social stress test, a moderate psychological stress test was conducted in a laboratory setting [6], subjects were found to significantly increase in heart rate throughout the experiment which shows the correlation between HR and stress. Hence, HR would be an ideal feature for measuring the users response to stress with Hotei.

A similar quantity is **heart rate variability (HRV)** which can be measured as the average deviation from the mean R-R, where R corresponds the the peak of an ECG or PPG wave, and is commonly done over a 5 minute period. The change in beat-to-beat variation is determined mainly by the current activity of the cardiovascular system. A link has been shown between high HRV values to overall good health and fitness, while low HRV values are associated with fatigue [7], stress [8] [9] and poor health. Links have also been shown between low HRV and severe time pressure, emotional strain [10] and an elevated state of anxiety [11]. All these are common factors that are capable of affecting a students mental state and this hence makes HRV an important feature to measure.

Machine learning algorithms have been applied in a variety of ways to measure and detect stress in individuals. An example of such a method is featured in [12], where personalised stress detection is developed using a number of different

extracted features, such as, the mean heart rate, heart rate deviation and respiration rate. This paper proposes a **support vector machine (SVM)** model to binary classify whether the user is stressed, mapping the inputs to a feature space and seeking a separating hyperplane. SVMs are a state of the art machine learning technique and have been shown to provide great performance for a variety of different applications. An issue highlighted is that their generalization can be poor if there is a large variation between different subjects results, hence this is considered within our ML algorithm design.

To summarize, the important features to measure in relation to stress are the users HR and HRV, which can be measured with ECG and PPG. For our purposes we shall focus on binary classification of stress using SVMs.

B. Recommendation System

Recommendation engines have been widely used for many applications from films to products to predict a user's preference towards previously unseen items. The recommendation process often starts with an initial data-set describing a users preference to a small number of items. Preference is indicated through explicit data such as user ratings or implicit information such as browsing time[13]. The data-set takes the form of a utility matrix (Figure 1) where r_{ij} represents the rating given to item j by user i, unrated items are often represented by a 0

$$\begin{bmatrix} r_{11} & r_{12} & r_{13} & \dots & r_{1j} \\ r_{21} & r_{22} & r_{23} & \dots & r_{2j} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ r_{i1} & r_{i2} & r_{i3} & \dots & r_{ij} \end{bmatrix}$$

Fig. 1: Utility matrix

1) Collaborative Filtering: The most common type of recommendation system is a Collaborative Filter. Collaborative filtering (CF) techniques make the assumption users who have previously agreed in the past are likely to agree in the future, recommendations are generated by taking the ratings that other similar users have given for an item into account [14].

Resnick et al [15] use the GroupLens data-set to build a user neighbourhood N for each user u with the k most similar users residing in N. The similarity between a pair of users is calculated using Pearson's correlation (equation 1). The preference $P(u,i)$ for an unrated item i by u is attained by taking a weighted average of the ratings of i by users in N (equation 2). This type of recommendation system has been widely used in industry but they tend to suffer from two significant problems, **cold-start** & **sparsity**.

$$s(u,v) = \frac{\sum_{i \varepsilon I_u \cap I_v} (r_{u,i} - \overline{r_u})(r_{v,i} - \overline{v_v})}{\sqrt{\sum_{i \varepsilon I_u \cap I_v} (r_{v,i} - \overline{r_v})^2 \sum_{i \varepsilon I_u \cap I_v} (r_{u,i} - \overline{r_u})^2}} \quad (1)$$

$$p_{u,i} = \overline{r_u} + \frac{\sum_{v \varepsilon N} s(u,v)(r_{v,i} - \overline{r_v})}{\sum_{v \varepsilon N} |s(u,v)|} \quad (2)$$

Where $r_{x,i}$ represents a user x rating for item i and $\overline{r_x}$ is the average rating for all items by x

Cold Start [16][17], occurs when a new user has no recorded activity so finding similar users is impossible, ultimately leading to poor recommendation quality. Vozalis and Margaritis [18] make use of demographic correlations between users to assist in the recommendation process by combining the similarity of a users demographic with the similarity of their rating profiles, helping to pair similar users even if the rating profile is non-existent. Their results showed significant improvement over classic CF algorithms, but noted that the quality of demographic data collected is significant in providing accurate responses. Rosli et al.[19] proposed a novel method to alleviate the cold start problem in film recommender systems. By extracting a user's Facebook Like data they were able to determine the films they enjoyed and thus were able to construct an initial profile for a new user. Ahn [20] described the limitations of existing similarity metrics and proposed the PIPS measure, a new heuristic similarity metric which combines, the arithmetic distance between two ratings (proximity), the extent to which the item is preferred or disliked (impact), and to what degree two ratings deviate from the average rating of an item (popularity). Experiments proved significant gains over existing methods of calculating similarity in cold start situations. Zhou et al. [21] proposed that a new user profile should be learnt through a preliminary interview process, they introduced a novel method using functional matrix factorization which adaptively queries the user according to previous responses whilst building a more accurate profile of the new user.

Sparsity [22] occurs when the rating profile for a user is predominantly empty, making it harder to find definite similar users. Two users who may be very similar may be overlooked if the data-set is particularly sparse leading to poor recommendation quality. As with [18], Pazzani [23] proposed combining external information describing users to increase the likelihood of finding similar users in a sparse data set. Sarwar et al. [24] proposed using SVD to reduce the dimension of the utility matrix with the intention of discovering latent factors in the observed ratings. This reduced matrix was then used to predict the rating of all the unrated items in the matrix. Although dimensionality reduction techniques have their merits their drawbacks are that by reducing the dimension some useful information may be loss, so careful consideration must be made as to what dimension to reduce to.

2) Context-Aware Systems: When recommending activities to a user, contextual information such as time is just as significant as preference to that activity, for example a user may enjoy playing football, but recommending this activity at midnight may not be a relevant in the current context. Context-aware recommendation systems take the context into account when producing recommendations.

Many methods of incorporating context into recommender systems have been suggested[25]:

Contextual Pre-Filtering, the context in this case is used to filter out ratings data, this filtered data can then be used with any recommendation technology which is a significant advantage given the amount of research in this field. The disadvantage to this method however is the limited data-set that is used to provide recommendation, this can often lead to inaccurate recommendations.

Contextual Post-Filtering, the context is used to adjust the ranked list of recommendations that are produced using common recommendation techniques, the significant advantage of this method over contextual pre-filtering is that it allows the entire data set to be leveraged, leading to more accurate

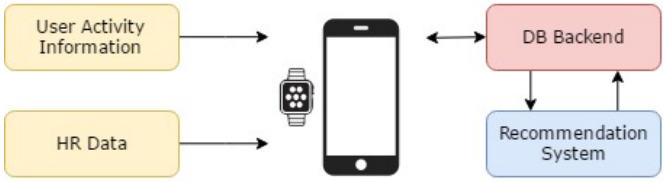


Fig. 3: Abstract Architectural Diagram of Hotei System

recommendations.

An important consideration is the level of context to filter, in some cases the context may be too narrow, imagine a scenario with the context $c = \{Saturday, 7pm\}$, using this exact filter could be problematic for two reasons, firstly this overly specific context may be insignificant, the user may find a sufficient classification being the Weekend as apposed to Saturday. Secondly we may not have sufficient data in this context for accurate prediction[26].

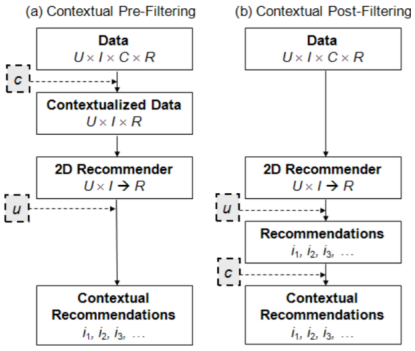


Fig. 2: Context pre-filter & post-filter data flow

IV. System Design

A. System Overview

Hotei is a mobile application that enables a user to record their stress level, and their response to activities they have participated in. Combining these two sources of information, Hotei is then able to recommend activities to assist a user in reducing their stress level. Hotei has been developed using iOS for the following reasons.

1) iOS permits a single application that works seamlessly on every device [1]. Android devices & other mobile platforms share the same OS but operate on different hardware with varying capabilities. Factors such as DPI, aspect ratio and performance have to be considered when developing an application, thus leading to a more complex specification and increased development time.
2) Apple's effort on security and Health Kit allows developers to shift focus from low-level programming such as security issues to the high-level functionality of the application.

Figure 3 depicts a high level system overview of the Hotei architecture. Inputs to the system are both explicit and implicit, users enter information about activities they have completed, heart rate data is collected at regular time intervals and information such as accelerometer data is take from a smart watch. This information is then utilised to provide bespoke activity recommendation to a user experiencing stress.

[1]Assuming updated iOS firmware

B. Hardware

There are three hardware components in the Hotei architecture (figure 4), an Apple iPhone, Apple Watch and a Polar H7 heart rate monitor, with each device communicating within the system via Bluetooth. The main application for user interaction is found on the iPhone, the Apple Watch provides user notification and collects data via the accelerometer. These devices then communicate using the iOS watch connectivity framework.

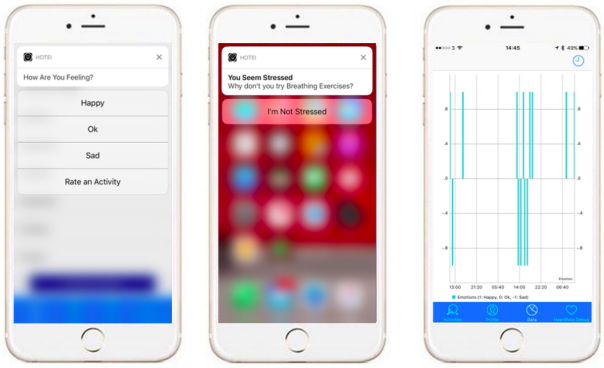
The Polar H7 chest strap heart rate monitor gathers the users heart rate and current R-R (beat to beat) intervals, which is used to calculate their HRV. This device is used due to the current state of the functionality provided by Apple for the watchOS. Currently, it is not possible to receive the raw PPG signal or the R-R intervals with the Apple Watch. The only information available from the sensors related to the heart is the heart rate, which is sampled at a low rate, hence why the Polar H7 is used instead.

Fig. 4: Hotei Hardware Components

C. Application & User Interface

From a user-facing perspective, the Hotei application is built around three core functions: emotional state recording, activity recommendation and data presentation. Emotional state recording allows users to rate how they feel throughout the day and correspond how they are feeling to an activity they have just completed. Activity recommendation displays a recommended activity to the user helping them relieve their stress, and data presentation provides the user with a graphical view of their emotional state recording throughout the day.

Whilst it is possible for a user to input their emotional state of their own accord by manually opening the application, we made use of actionable notifications that regularly prompt users for their emotion state throughout the day, as shown in figure 5a. This type of notification allows them to perform such tasks without having to open the application, aiding convenience. In the event that a user wishes to record both an emotional state and link it to an activity, an option is provided ('Rate an Activity') to open the app to perform this action.



(a) (b) (c)

Fig. 5: Screenshots of Core App Functionality

Notifications are also useful for the delivery of activity recommendations. With stress detection happening the background, an activity recommendation notification can be triggered when a stress event is detected, thereby resulting in a completely automated system without requiring the user to manually request an activity to perform when they are stressed. As with recording emotion state, it is still possible for the user to manually request an activity within the application. The actionable notification further allows users to provide feedback regarding whether the stress event notification is correct with an action 'I'm Not Stressed'. An example of this is shown in figure 5b

Data presentation is the final section in the app displaying a time series of emotion states throughout the day in the form of a bar chart, shown in figure 5c. Additionally, the user can choose to view a list of past activities they have done to better understand the charts displayed. The intention is that this data helps users to learn about their emotional state and develop habits themselves that improve their stress management. Furthermore, this data might be able to assist a CESD test and aid doctors to help a (potential or not) depression patient better.

D. Stress Detection

The stress detection functionality is performed using an SVM that predicts whether the user is currently stressed or not. This model is trained locally on the iPhone using OpenCV (Open Source Computer Vision), a library of functions mainly aimed at real-time computer vision but also provides support for general machine learning algorithms. The SVM algorithm finds the optimal separating hyperplane that maximises the margin between the training data.

The data gathered to train the algorithm contains two features: the users mean HR and HRV, which is calculated in the time domain as the standard deviation of the NN or beat to beat intervals (SDNN). The equation for this is shown as follows, where r is the beat-to-beat interval, μ_r is the mean of r and N is the number of samples.

$$SDNN = \frac{1}{N}\sum_{i=0}^{N}(r_i - \mu_r)^2 \tag{3}$$

These features were chosen from the background research acquired, which showed a strong correlation between a change in HR, HRV and stress. Since SVM is a supervised learning model, the training data requires an existing classification in order for the algorithm to train. In this case, each data point is associated with a binary label as to whether the user is, or is not, currently stressed. We consider this feature set to be a good method for learning stress since it is expected that the users stressed state will not rapidly change and can hence be captured within the given time period.

Within the app, the machine learning model is specific to each user. This is due to the fact that each user will have a different resting heart rate along with a different increase in heart rate once stressed, hence why the model is only applicable to them. With typical use of the app, the SVM model is retrained daily using the most recent weeks worth of data, so that the model is kept up to date. Once the model is trained it is used to make predictions on current data samples and send notifications to other core system components related to the current stress state of the user.

An important aspect of training the machine learning model is to ignore inaccurate data samples that could result in false positive readings. For example, if the user is exercising then their HR will appear elevated as if they were stressed. This is avoided by using the CoreMotion framework on the Apple Watch to identify the user's activity state. The range of states detectable include stationary, walking, running, cycling and driving. In our case, we choose to acknowledge stress events only when a user is in the stationary state.

E. Recommendation Engine

The recommendation system is built using C# and Microsoft Azure, the decision behind using a public cloud was that it allows for immediate access to infrastructure such as databases and servers at low cost while also providing industry level security. A global server and database is required as the recommendation engine performs collaborative filtering, which uses data from every user to provide recommendations, having an easy way to access this information allows for fast user recommendation.

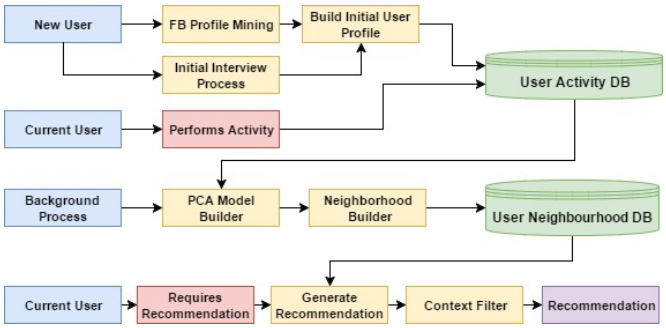


Fig. 6: Hotei recommender system component diagram

Figure 6, displays the key components of the Hotei recommendation system, the systems can be split into three key streams: *New User*, *Current User* & *Background Process* each of which will be described and justified below.

1) Current User: A current user has two interactions with the recommender system, when they perform an activity and when they require a recommendation. Once a user performs an activity the rating for that activity is updated in our activity database table. As a user can perform an activity multiple times, we represent the rating of an activity in our data-table as an aggregated score.

When a user requires a recommendation we first utilise the neighbourhood model built in the background process.This model determines the most similar users of our current user, where pair wise similarity is represented by $s(u,v)$. Using this information we are able to use a predictor function (equation 2) to predict a users rating for previously unrated items. The output of the Generate Recommendation subsystem (figure 6) is a ranked list of all the possible activities in the recommendation engine.

The context filter is responsible for adjusting the list based on the current context, the rating of an item j by user i adjusted for the current context c is given by:

$$Rating_c(i,j) = Rating(i,j) \times P_c(i,j) \quad (4)$$

Where $P_c(i,j)$ is the total number of neighbours of i who rated the item j in the same context divided by the total number of neighbours [27].

After producing a context adjusted list, the system then provides a recommendation. We have chosen to apply fitness proportionate selection [28] when choosing an item from the recommendation list, this decision diverts from traditional recommendation systems which would choose the highest ranked item as a recommendation. The decision behind this is that our recommendation system differs from tradition recommendation systems in the fact that we allow repeat recommendations, if we did not randomly select an activity from the list we run the risk of the same activity being recommended constantly.

2) Background Process: The purpose of the background process is to determine the most similar users for all users in the system, this computation is intensive particularly as the number of users grow, therefore we opt to generate this model at fixed times throughout the day. This functionality is implemented using a Microsoft Azure scheduler, which allows us to update our model at scheduled times throughout the day.

To build the model, we first perform Principle Component Analysis (PCA) on the utility matrix stored in the User Activity Database, with the intention to reduce the effect of sparsity on the data-set and thus increase the likelihood of detecting similar users. PCA determines the best low rank matrix that estimates the existing data-set, we perform this calculation using SVD [29].

This method negates the need to make assumptions about demographic data (eg. males prefer football and females prefer netball) as in other methods to reduce effect of sparseness.

After the model is built, the background process will build a neighbourhood of similar users for each user, the similarity $s(u,v)$ of two users will be determined using Pearson's correlation coefficient (equation 1)

The most similar neighbours for each user are stored in the user neighbourhood database table. Building a model in this format increases the scalability of this application as the model can be calculated offline without the need to perform similarity calculations every time a user requires a recommendation.

3) New User: This subsystem is designed to mitigate the user cold start problem, in order to maximise our ability to determine the best activities for our user. We decided to make a user perform an initial selection process, this is a simple form that the user completes that indicates their favourite activities.

V. Experimental Setup & Methodology

A. Stress Detection

1) Setup: We evaluated the performance of the stress detection system by conducting short 40 minute experiments with a set of 5 users. The time period was divided into two equal parts, a training segment and a testing segment for the SVM model. The data gathered from the users consisted of their current HR and R-R intervals which were used to calculate the mean HR and HRV features over a one minute period. In addition to this, during the experiment the user provided feedback to the app as to whether or not they currently felt stressed.

The basis of this experiment uses a reliable method to generate stress. In the past, options such as the cold pressor test or the Stroop test were popular options for inducing stress in laboratory conditions. The decision of the group was instead to use the *Trier Social Stress Test* [30]. The *Trier Social Stress Test* is broken down into two sections: a 10 minute anticipation stage and a 10 minute test period. In the 10 minute anticipation stage the test subject is given a pen and paper and told to prepare a 5 minute speech aimed at convincing a mock interview panel why they are the optimal candidate for a job. When the test period begins the test subject will be placed in a room in front of a panel of three people, a video-camera, and a visible microphone. Their notes will be taken away from them and they will be asked to present. The panel will not react to the participants speech, and if the participant finishes before the 5 minutes is over they will be asked to continue speaking til the end of the 5 minutes. Once their presentation is complete the test subject is asked to serially subtract 13 from 1022 as fast and as accurately as possible, and told that if they make a mistake they will have to start again. Every time the subject makes a mistake an interviewer will say "Stop, 1022". This method has been shown in numerous independent studies to induce stress in participants.

The test was conducted at Imperial College, South Kensington Campus, by volunteers from the student population. At the beginning of the experiment subjects wore the smart heart rate monitoring device that is connected to the application. These assumptions are made: the test subjects are not in a stressed state at the beginning of the test, the test subjects find the interview and arithmetic components of the test stressful. If the application reports that the user is stressed, the detection engine will be deemed a success.

The data gathered from the first segment of the experiment was then used to train the SVM model. After a short break, the experiment was then repeated, applying the same process for gathering data. This data was then used as the test set for the trained model, comparing the models prediction to the users actual feedback on their current stress state.

2) Results: The results for this experiment are shown in the following figures and tables. Figure 7 shows an example of the linear separator produced by the trained SVM model for a single user's experimental data. The graph show the trend that a low HR and high HRV is associated with a state of no stress and conversely a high HR and low HRV is associated with a state of stress, which is as expected from the background research conducted. Figures 8 and 9 show how for this particular user their HR and HRV varied over the course of the training period of the experiment. It can be seen that the user undergoes a period of stress from the 6th

minute to the 16th minute, where they experience a generally elevated HR and decrease in HRV.

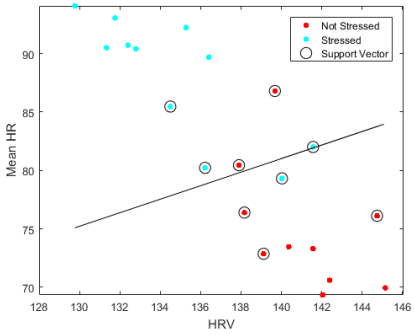


Fig. 7: SVM plot for the user.

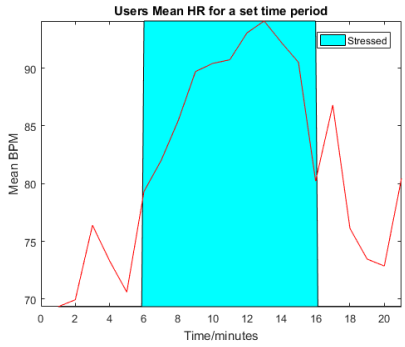


Fig. 8: Mean HR over time for the user.

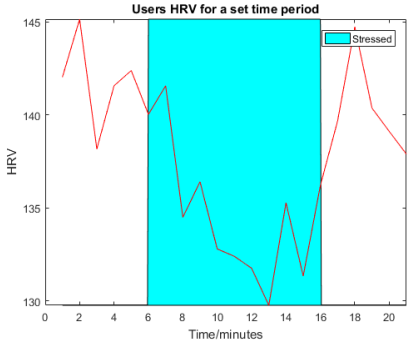


Fig. 9: HRV over time for the user.

Figure 10 shows a confusion matrix showing the accuracy of the SVM model for classifying the test data set. The label of 1 is associated with a stressed state while a label of 0 is associated with a non-stressed state. For this particular user an 80% accuracy was achieved.

Table I shows the accuracy for all users during the experiment, along with the percentage of time the user was stressed during the training phase. The results show that for all users the model achieves a better than random accuracy when predicting the users stress. Ideally, for training there would be an equal number of samples for the user in both states, but this is difficult to achieve for something as abstract as stress. For some users the trained model can be biased to

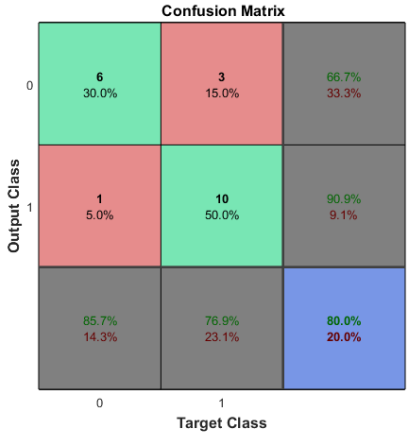


Fig. 10: Confusion matrix for the SVM and test data.

a single state, such as with user 5 where during the training phase the majority of the time they were not stressed.

Throughout the experiment, an issue that arose was that it was ambiguous for users to state whether or not they were currently stressed. Stress is relatively abstract and so it can be hard for a user to decisively categorize that they are stressed or not. This can explain the noise seen within the data sets where characteristically similar data points are labeled differently.

User Stress Detection Testing		
User ID	Time Stressed (%)	Accuracy (%)
1	50	80
2	30	65
3	30	70
4	45	75
5	5	95

TABLE I: User experiment results

3) Further Work: To improve this method for testing and evaluating the model we would expand the number of users and test over multiple weeks. This would be done to test the model during normal daily use and to gather a wider variety of data samples so that the model generalizes better.

B. Recommendation System

1) Setup: Testing a recommendation system is inherently difficult to do in a time constrained environment, it would take many weeks to collect preference data about multiple users such that the system could provide accurate recommendations. As this time was not available, the following evaluation will act as an introductory setting for future research.

To test the recommender system as an individual component of our system, a survey was conducted on 20 Imperial College Students to gather information about their favoured activities. Within the survey a user was required to indicate their preference toward a selection of 35 activities, preference was indicated through a discrete rating system between 1 & 5, with 5 indicating a very strong preference. The user was not required to give a response to an activity, indicated by a 0 rating.

We adopt a cross validation testing methodology outlined in [14], the data gathered from the user survey is split into two subsets, the training set and test set. A recommendation model is built using only the training set, each user profile in the test set is then considered in turn, the profile is split into two parts, the query set and target set. The recommender is given the query set as the current user profile, the predicted ratings for new items are compared against items in the target set. The performance of the recommender is evaluated based on the difference between the predicted & actual rating.

2) Results: Figure 11 depicts the original rating given by our test set for items in the target set and the corresponding predicted rating given by our recommendation engine. Compared to a randomised rating we observe that our recommendation engine performs relatively better.

User	Item	Rating	Predicted Rating	Random Rating
1	1	3	2	4
	2	5	5	1
	3	1	3	5
	6	2	1	5
	7	3	3	4
12	2	5	4	4
	3	1	1	4
	17	3	1	2
	22	4	1	4
	25	4	4	1
4	1	3	2	4
	3	1	1	1
	5	3	3	2
	10	1	4	1
	12	2	2	1
20	4	2	1	5
	6	4	3	4
	8	5	3	2
	10	1	1	5
	11	2	2	1
		MSE	1.8	4.25

Fig. 11: Recommendation Engine Performance.

C. Overall System

To test our overall system, extended use of the application by many users would be required. Given the time constraints and the lack of hardware components available to us, mainly the Polar H7, we were unable to conduct our experiment with several users simultaneously. This section presents a preliminary evaluation of our hypothesis.

1) Setup: To test the hypothesis, that through personalised activity recommendation the frequency of stress of a user experiences over time will reduce, we conducted a test of our application on three individual users. These users are a subset of those used in our stress detection experiment outlined earlier. The choice was made to use the linear separator determined in our stress detection test for these three users as it allowed us to train the SVM model once and focus resources on collecting data to test the hypothesis in question. Users were instructed to use the application in their daily lives over a period of 3 days, and additionally asked to confirm whether they did in fact perform the activities that were recommended.

2) Results: Figure 12 depicts the number of stress events detected for each user over the three day test period. We can see that the stress frequency over this time for each user is variable and no significant conclusion can be drawn over the performance of our application. The reason for the high variability can be attributed to several factors. For example, data on the second day for user 3 was collected on a Sunday, and further investigation indicated that this user was naturally relaxed as they spent the day away from campus doing non work related activities. Additionally, given that the nature of a students' workload differs everyday, it is natural for their stress levels to fluctuate accordingly. Future experiments would have to be conducted at a much larger scale and over an extended period of time. A thought would be to experiment over several weeks and compare the frequency of stress events for each day over each of the weeks, and calculate the average number of stress events per week.

Figure 13 shows the percentage of activities completed by each user over the three day test period. It can be seen that even though users were stressed they did not perform all activities that were recommended to them. This can be explained by the fact that the experiment was conducted during exam season at Imperial College London - a time where subjects were naturally stressed and unable to find the time to undertake activities that require a significant amount of time and effort. Additionally, this observation can be explained by the possibility that that users were not provided with accurate recommendations at the first event of stress detection, further highlighting the need for more data to be collected.

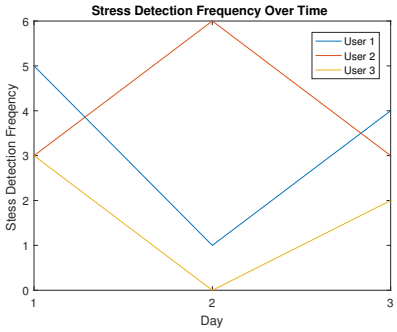


Fig. 12: Overall stress classification over 3 days.

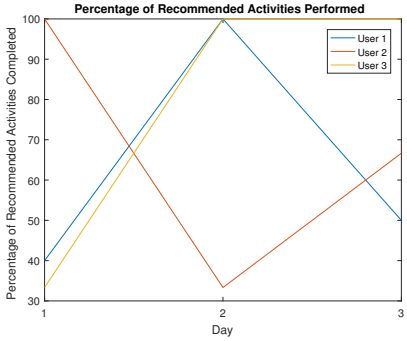


Fig. 13: Overall recommended activities completed over 3 days.

VI. Future Work

Based on our findings from testing the various sub-components of our system as well as the system as a whole, we have determined several areas of improvement for our application. Firstly we identified that, whilst the classification rate of stress was good there were several factors not originally considered that can affect HRV, such as sleep deprivation, alcohol consumption and dehydration. Just as we use the CoreMotion framework to check whether HR increase has been caused by stress or exercise, a possible solution would be to make use of iOS extensions to other health apps which monitor these factors to further enhance the accuracy of our stress detection based on HR and HRV.

In the testing of the overall system, it was observed that users often did not complete all activities recommended to them. An improvement to the recommendation system module would be to take into consideration both location and calendar data so that more suitable recommendations are provided during, for example, exam period. Additionally, we came to an assumption that users would not engage in a high intensity activity (running, swimming, etc) if they have already engaged in exercise for the day. This can be gauged by data from the CoreMotion framework and utilised to provide less intense activity recommendations in this scenario. The recommendation system module would also be further improved by gathering more data to provide more accurate recommendations.

VII. Conclusion

Hotei is a wearable stress detection system that utilises heart rate detection to detect stress and provide personalised activity recommendations through an iPhone application. In this report we tested our hypothesis: That through the use of a personalised activity recommendation system the user's frequency of stress will be reduced. Having tested our overall system with 3 Imperial College London students over a period of three days we observed that our findings were inconclusive due to the limited testing time and sample size. However, when tested in its sub-components, it was found that the stress detection system performed with greater than random accuracy and the personalised recommendation system performs better than a randomised system. Finally, we outlined several improvements that can be made in future iterations of the system. As a result of reducing the frequency of stress in students, we hope that this system can ultimately have a positive effect on an individuals emotional wellbeing.

References

[1] Laurie Beaver. The smartwatch report: Forecasts, adoption trends, and why the market isn't living up to the hype. *Business Insider*, 2016.

[2] Matthew Aronin, Scott Smith. One in four students suffer from mental health problems. *YouGov*, 2016.

[3] National Institute of Mental Health. 5 things you should know about stress. *Nimh.nih.gov*, 2017.

[4] Toshiyo Tamura, Yuka Maeda, Masaki Sekine, and Masaki Yoshida. Wearable photoplethysmographic sensorspast and present. *Electronics*, 3(2):282–302, 2014.

[5] Brigitte M Kudielka, Nicole C Schommer, Dirk H Hellhammer, and Clemens Kirschbaum. Acute hpa axis responses, heart rate, and mood changes to psychosocial stress (tsst) in humans at different times of day. *Psychoneuroendocrinology*, 29(8):983–992, 2004.

[6] Clemens Kirschbaum, K-M Pirke, and Dirk H Hellhammer. The trier social stress test–a tool for investigating psychobiological stress responses in a laboratory setting. *Neuropsychobiology*, 28(1-2):76–81, 1993.

[7] LJM Mulder. Measurement and analysis methods of heart rate and respiration for use in applied environments. *Biological psychology*, 34(2):205–236, 1992.

[8] Xavier Bornas, Jordi Llabrés, Miquel Noguera, López, Francesca Barceló, Miquel Tortella-Feliu, and Miquel Àngel Fullana. Self-implication and heart rate variability during simulated exposure to flight-related stimuli. *Anxiety, Stress & Coping*, 17(4):331–339, 2004.

[9] Nis Hjortskov, Dag Rissén, Anne Katrine Blangsted, Nils Fallentin, Ulf Lundberg, and Karen Søgaard. The effect of mental stress on heart rate variability and blood pressure during computer work. *European journal of applied physiology*, 92(1-2):84–89, 2004.

[10] Peter Nickel and Friedhelm Nachreiner. Sensitivity and diagnosticity of the 0.1-hz component of heart rate variability as an indicator of mental workload. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 45(4):575–590, 2003.

[11] Peter Jönsson. Respiratory sinus arrhythmia as a function of state anxiety in healthy individuals. *International journal of psychophysiology*, 63(1):48–54, 2007.

[12] Yuan Shi, Minh Hoai Nguyen, Patrick Blitz, Brian French, Scott Fisk, Fernando De la Torre, Asim Smailagic, Daniel P Siewiorek, Mustafa alAbsi, Emre Ertin, et al. Personalized stress detection from physiological measurements. In *International symposium on quality of life technology*, pages 28–29, 2010.

[13] Robert Kass and Tim Finin. Modeling the user in natural language systems. *Computational Linguistics*, 14(3):5–22, 1988.

[14] Michael D Ekstrand, John T Riedl, Joseph A Konstan, et al. Collaborative filtering recommender systems. *Foundations and Trends® in Human–Computer Interaction*, 4(2):81–173, 2011.

[15] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186. ACM, 1994.

[16] Le Hoang Son. Dealing with the new user cold-start problem in recommender systems: A comparative review. *Information Systems*, 58:87–104, 2016.

[17] Gediminas Adomavicius and Alexander Tuzhilin. 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):734–749, 2005.

[18] Manolis Vozalis and Konstantinos G Margaritis. Collaborative filtering enhanced by demographic correlation. In *AIAI Symposium on Professional Practice in AI, of the 18th world Computer Congress*, 2004.

[19] Ahmad Nurzid Rosli, Tithrottanak You, Inay Ha, Kyung-Yong Chung, and Geun-Sik Jo. Alleviating the cold-start problem by incorporating movies facebook pages. *Cluster Computing*, 18(1):187–197, 2015.

[20] Hyung Jun Ahn. A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem. *Information Sciences*, 178(1):37–51, 2008.

[21] Ke Zhou, Shuang-Hong Yang, and Hongyuan Zha. Functional matrix factorizations for cold-start recommendation. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 315–324. ACM, 2011.

[22] Miha Grčar, Dunja Mladenič, Blaž Fortuna, and Marko Grobelnik. Data sparsity issues in the collaborative filtering framework. In *International Workshop on Knowledge Discovery on the Web*, pages 58–76. Springer, 2005.

[23] Michael J Pazzani. A framework for collaborative, content-based and demographic filtering. *Artificial intelligence review*, 13(5-6):393–408, 1999.

[24] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Application of dimensionality reduction in recommender system-a case study. Technical report, DTIC Document, 2000.

[25] Gediminas Adomavicius and Alexander Tuzhilin. Context-aware recommender systems. In *Recommender systems handbook*, chapter 10. Springer, 2015.

[26] Gediminas Adomavicius, Ramesh Sankaranarayanan, Shahana Sen, and Alexander Tuzhilin. Incorporating contextual information in recommender systems using a multidimensional approach. *ACM Transactions on Information Systems (TOIS)*, 23(1):103–145, 2005.

[27] Umberto Panniello, Alexander Tuzhilin, Michele Gorgoglione, Cosimo Palmisano, and Anto Pedone. Experimental comparison of pre-vs. post-filtering approaches in context-aware recommender systems. In *Proceedings of the third ACM conference on Recommender systems*, pages 265–268. ACM, 2009.

[28] Thomas Back. *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms*. Oxford university press, 1996.

[29] Ian Jolliffe. Principal component analysis. 2002.

[30] C. Kirschbaum, K. M. Pirke, and D. H. Hellhammer. The trier social stress testa tool for investigating psychobiological stress responses in a laboratory setting. *Neuropsychobiology*, 28(1-2):76–81, 1993.