1K1-IOS-1a-3

A Model for Sitting Postures in Relation to Learning and Non-learning Behaviors

Anh Mai, Roberto Legaspi, Paul Inventado, Rafael Cabredo, Satoshi Kurihara, Masayuki Numao

The Institute of Scientific and Industrial Research, Osaka University

As more and more information find their way to the internet, people are able to do more at their own desk than ever before, all in the comfort of a private environment. But as more activities, especially learning, are able to be done through the personal desktop space, the question is then raised of whether or not one is really engaged and/or learning and not being distracted by other things that the internet offer. For this, we propose a model that will associate various sitting postures with a person's level of engagement and/or learning. Said model will know what kind of postures usually indicate a state of engagement to a person's work and learning, and which postures indicate a falling out from that state. We apply machine learning techniques to a database of silhouette images, captured using a Microsoft Kinect, in order to extrapolate patterns that would help link a user's postures to his learning state. Our model can be used to assist users regain learning postures and suggest for a change of activity if prolonged periods of non-learning are detected so that users will gain the most out of their time.

1. Introduction

Students exhibit different behaviors while they learn. These behaviors can be observed by looking at the actions they perform and can be identified as either learning related on non-learning related activities. We assume that students become distracted and are less likely to complete their learning objectives when they spend much time engaging in non-learning related activities. Thus, there is a need to manage these activities to improve learning.

The first step in managing these activities would be identifying them. Since asking students to identify them manually adds more cognitive load and causes them to be distracted, activities need to be identified automatically for them. Activities done on a computer while learning can easily be identified by logging all activities in a computer. However, activities outside of the computer which are equally as important, are not as easy to capture. For example, it is difficult to identify when a student reads a book, sends a text message or drinks coffee.

Postures can be used to identify students' activities since they are quite distinct for different actions. For example, a subject's posture when reading and writing is different compared to other actions such as sending a message using a cellphone or making a phone call. In this research, we collected the subject's posture while learning using data extracted from a Microsoft Kinect sensor. After the learning session, the subjects then annotated their activities as either learning related or non-learning related and served as labels for the posture data. We then developed a model that mapped the subject's posture to the type of activity they engaged in.

The model created in this work can be used by future systems to detect certain activities and provide necessary feedback to help students manage the activities they engage in while learning and help them maximize the amount of time they spend learning.

2. Related Works

Many existing learning environments keep track of student actions. Examples of such systems include the Cognitive Tutor [Aleven 10], SQL Tutor [Mitrovic 10] and AutoTutor [Graesser 99] which use student actions to maintain a student model. This student model represents the students' understanding of the concepts presented in the learning environment and is used as basis for providing feedback. These systems however keep track only of activities done within the learning environment. These are not capable of identifying actions outside the learning environment and thus are not capable of providing feedback for these cases.

Also, [Inventado 11] reported that when students study on their own, they do not only engage in learning-related activities but also non-learning related activities. The students' affective state and the type of activity they engaged in were found to affect their learning behavior. This further emphasizes the need to analyze student behavior outside the learning environment and the importance of providing feedback for these instances to support learning.

Although most systems track student activities by looking at what they do on the computer, many of their activities are also done outside of the computer [Foehr 06]. There are different ways of identifying a person's activities in a physical space but capturing an image of the person is the least obtrusive. [Jaimes 06] used a regular webcam to capture images of a person in front of a camera. From the captured images, silhouettes were extracted using background subtraction and were used to create a model for identifying the person's activity. [Wientapper 09] also used postures to identify activities but used a time-of-flight (TOF) camera which was capable of capturing a three-dimensional representation of the environment and making it easier to distinguish the person from the scene giving more

Contact: Anh Mai, The Institute of Scientific and Industrial Research, Osaka University, 8-1 Mihogaoka, Ibaraki, Osaka, 567-0047, Japan, Tel: +81-6-6879-8426, Fax: +81-6-6879-8428, abmai@berkeley.edu

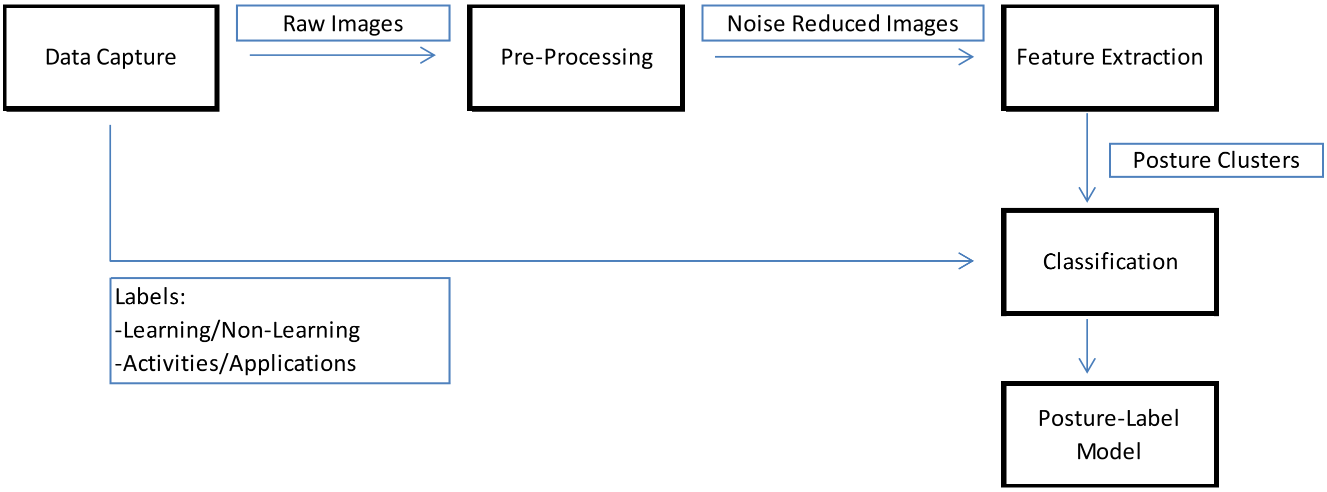


Figure 1. Framework of research. Raw Images, Noise Reduced Images, Posture Clusters, and Labels are items passed between the respective stages of the data flow.

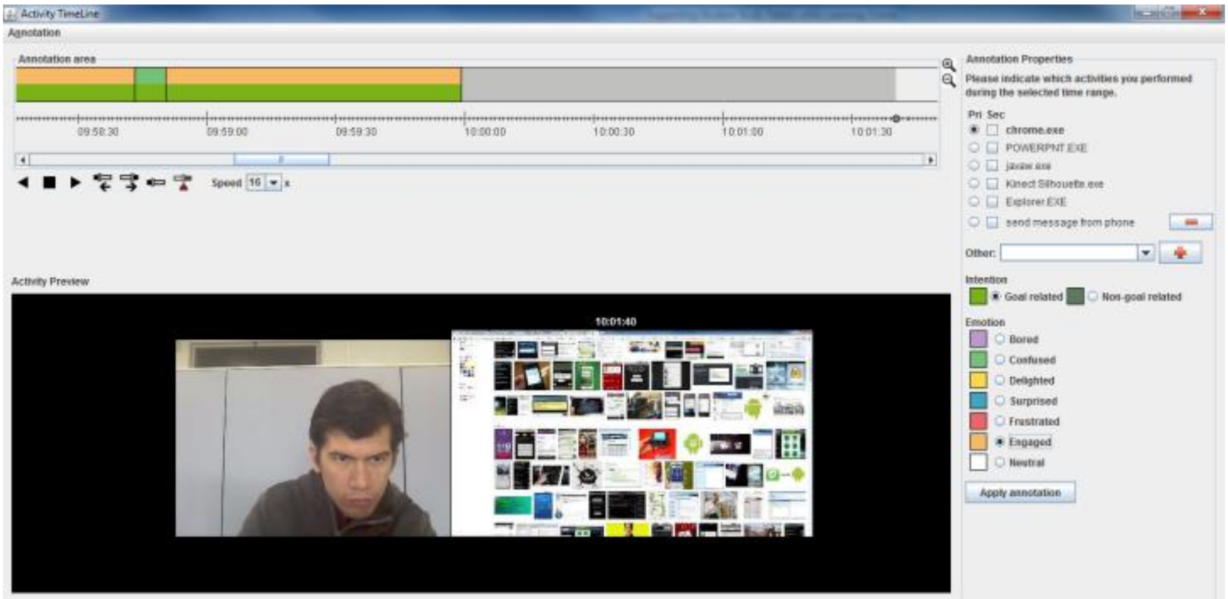
accurate results. Similarly a model was created using the time-of-flight images for building a model for identifying the person's activity in an Ambient Assisted Living environment. [Ray 12] also used posture to identify human activity but used data from the Microsoft Kinect. They also created a model which identified construction worker activities using posture data. Our approach uses the Microsoft Kinect to capture sitting postures, and our image processing involves removing extraneous noise from the captured images, which are only the silhouettes of the subject.

3. Posture Modeling

The broad perspective of our goal is to be able to create a model that would allow us to tell whether or not a user is engaging in learning or non-learning activities based on his posture. We take care in differentiating between computer and non-computer based tasks. However, we are also very interested in a user's behavior while using an application on the computer. Posture can also possibly help us determine whether a user is using a certain application for learning or non-learning activities. Figure 1 shows the framework that we used for this particular research.

3.1 Methodology

In order to automate the identification of student activities as they learned, we had to collect data from actual learning situations. Two male students were asked to engage in their usual learning habits while data regarding the activities they performed and their postures were collected. Specifically both subjects engaged in research related learning activities. During the data collection session, silhouette images taken from a Microsoft Kinect sensor, still images from a web camera focused on the subject and screenshots of the subjects' desktop were collected. After the data collection session, the subjects used an annotation software to annotate their activities as either learning or non-learning, and the emotions they felt while performing these activities. They were also instructed to select the most prominent



Annotation Properties

Please indicate which activities you performed during the selected time range.

Pri Sec
- ● □ chrome.exe
- ○ □ POWERPNT.EXE
- ○ □ javaw.exe
- ○ □ Kinect Silhouette.exe
- ○ □ Explorer.EXE
- ○ □ send message from phone

Other:

Intention
🟩 ● Goal related 🟩 ○ Non-goal related

Emotion
- ○ Bored
- ○ Confused
- ○ Delighted
- ○ Surprised
- ○ Frustrated
- ● Engaged
- ○ Neutral

Apply annotation

Figure 2. Software used for annotating and providing labels.

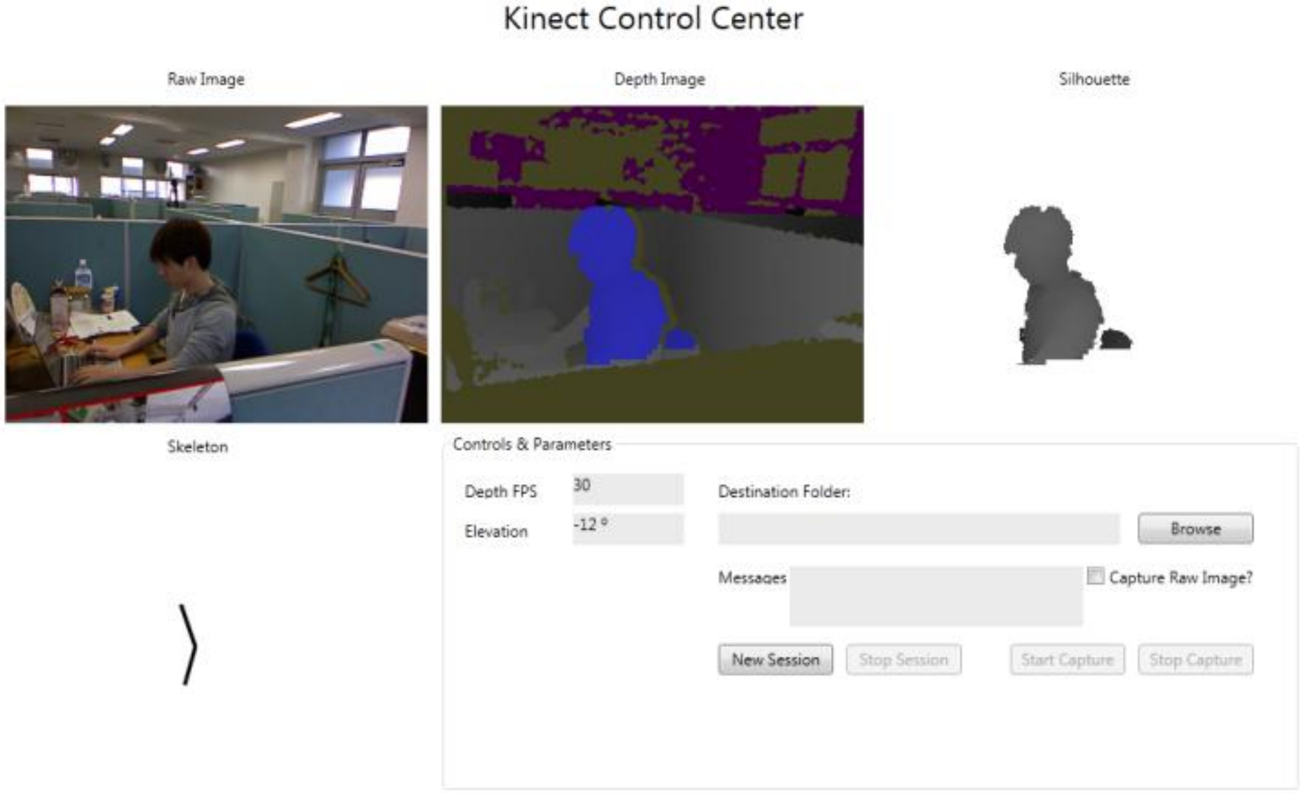
emotion they felt in cases when they felt more than one emotion while performing an activity. The annotation software created for the data collection allowed the subjects to review their activities over time using the desktop and webcam screenshots captured. Figure 2 shows a subject using the annotation software where he has selected a time range and provided an annotation for it. All annotations were mapped to the images extracted from the Microsoft Kinect sensor using its timestamps and served as labels to the posture data. Around four hours of data were collected from each subject.

3.2 Image Extraction and Pre-processing

Our approach to posture based activity recognition uses silhouette images and their pixel data. Up until recently, features extraction from an image required extensive pre-processing, such as background removal, and hence became a field of research in itself. The Microsoft Kinect, introduced in 2010, which was designed solely for the purpose of gaming, became a powerful tool to collect human silhouette images from an environment and is becoming more widely used in researches. The Kinect allows for skeletal tracking and recognition of person(s) moving in the environment - called Player Recognition. Recognized players are colored differently on the depth map that the Kinect produces. Microsoft provides an extensive Software Development Kit[1] to accompany the Kinect that makes it very easy to extract a person's depth silhouette using the information above, Figure 3 shows the software we developed for capturing data. The raw image is captured using the Kinect's regular RGB camera. The depth image is produced by using the Kinect's infrared beamer and receiver. Infrared particles are beamed out to the environment and receive back at the receiver. Depth is calculated by examining how long the particles took to travel to their destination and come back to the receiving infrared camera. And with Kinect's development kit, each pixel in the depth image can be colored differently based on its depth information and whether or not a player is present for that pixel. Using this, we simply opted to not color, or color white any pixel that is not recognized as a player, thus extracting only the person's silhouette. Using the Kinect, we captured approximately two images every second of the subject while he is work at his desk.

But the nature of infrared particles, as light particles, makes it difficult to receive clear images in real-time. Particles might bounce off one another, inducing noises within the captured depth images, and subsequently, the silhouette images. To remedy this issue, some image processing techniques are employed with the captured silhouette images to remove unwanted noise and smooth out certain areas.

For noise reduction, we use a majority-rule pixel removal method to "white out" noise pixels across the image. We setup a blank image of the same resolution as the original image, and process each pixel individually. For every pixel, we take a look at a window of surrounding pixels, and total up the number of colored and white pixels. If the surrounding pixels are mostly white, then we color the pixel white, otherwise, we simply copy over the pixel color value from the original to the new image. The window to look at can be adjusted at any time, but we have



Kinect Control Center

Depth FPS: 30

Elevation: -12 °

Figure 3. Data collection software for the Kinect. The pictures displayed from top left are the raw, depth, depth silhouette, and skeletal images. Depth FPS is the Frame Rate per Second of the depth images being displayed to the screen. And Elevation is the angle of the Kinect.

[1] http://www.kinectforwindows.org/

found that a window of 11 by 11 pixels give us the best results, that is the main pixel at the center and 120 pixels surrounding it. So if 61 or more of those pixels are white, then the pixel in question will be colored white. We do not choose to color in a pixel even if the majority of its surrounding pixels are white because there are white pixels in the image that we would like to preserve for context. Figure 4 shows an example of an image before and after it undergoes noise reduction.

```
Pseudo code for Noise Reduction algorithm:
  var whiteTotal
  var colorTotal
  for every pixel in image
    determines starting and ending point of window
      for every pixel in window
        if pixel != pixel in question
          if pixel is white
            whiteTotal += 1
          else
            colorTotal += 1
    if whiteTotal > colorTotal
      pixel Color = white
```

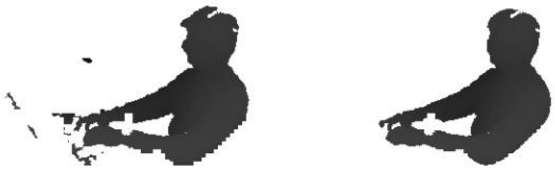


Figure 4. The left image is the raw depth silhouette captured by the Kinect, and the right image is the same image after going through our noise reduction.

3.3 Feature Extraction

We define a posture as any frequently occurring body positions. Hence, we can discard any sequence of images that do not occur enough, considering them as extraneous movement done when the user attempts to switch from one posture to another. In order to understand the entire image captured to accurately determine whether or not the user is moving, we chose to use a pixel-by-pixel approach, which is needed to be done to understand the entire picture. The depth silhouette image captured is rendered in a resolution of 320 by 240 pixels, thus giving us 76800 feature pixels.

Due to the fact that the data was captured chronologically, most of the postures are already grouped together. What we needed to do was to group together the same postures that occurred in different instances of time. For this, we employed a simple clustering algorithm that looked for the number of different pixels across images. The centroid of each cluster is the first image inside that cluster. Any subsequent image that has a pixel difference below a certain threshold in relation to the centroid image of that cluster will fall into the same cluster; otherwise, a new cluster will be created with that image as the centroid. The threshold is the percentage of pixel differences between an image and the centroid of the cluster. We have yet to run into a situation where this approach would be a cause for confusion between similar postures. Each cluster's uniqueness is then also measured by the strictness of the threshold; the smaller the threshold the closer the clusters will be in terms of similarity, but also the members inside the clusters will be more closely connected.

3.4 Posture Classification

First off, the images from the data collected are fed into our noise reduction algorithm to get rid of any extraneous noise that would otherwise hinder the clustering procedure. After clustering, and just before classification, we also pruned the dataset, removing any clusters that did not meet our criteria of a posture, frequently occurring body positions in time, here, we have pruned any clusters that did not total up to at least 20 images.

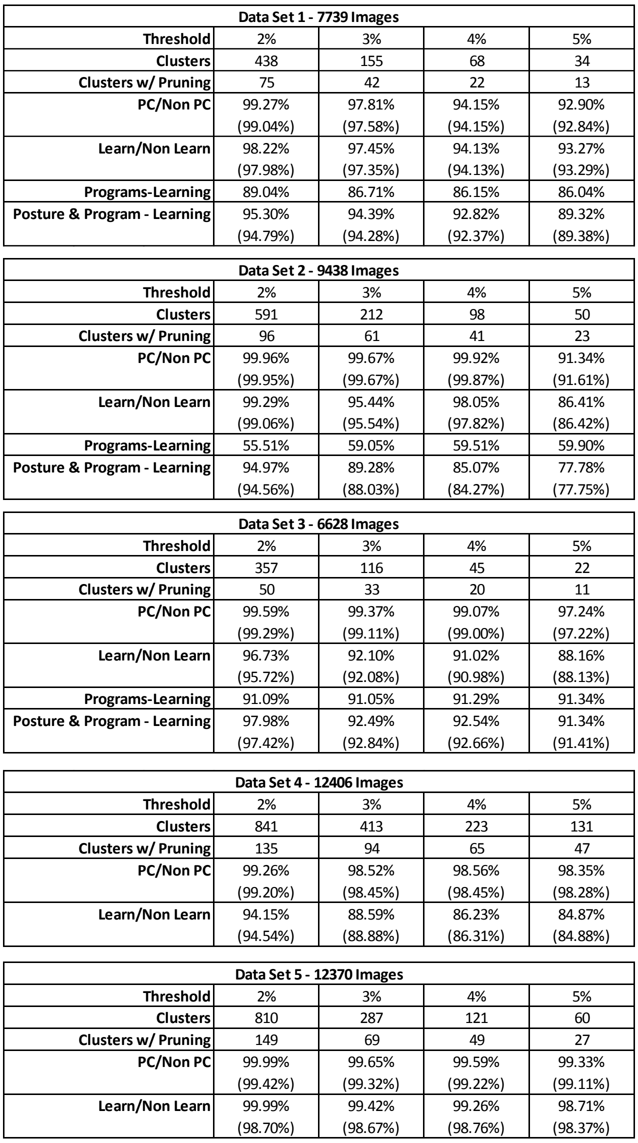
There are a few approaches we tried in classifying the postures, represented by the silhouette images. Each image received a cluster and two other labels: activity, and an indication of learning related or non-learning related. A CSV (comma-separated value file) is built out of that information, having one row per image. Firstly, we tried to classify each cluster of postures using the activity – an indication of whether the user was doing computer-related or non-computer related work. Secondly, we also classified posture clusters to the user's indication of whether they were engaging in a learning related or non-learning related activity at the time the images in each cluster were captured. Lastly, using the activity's detailed list of the main programs the user was using during his work session, we wanted to classify the user's posture, along with the program he is using, with the learning related or non-learning related label. The purpose is to use posture to give an indication on whether a user is using a program, say a web browser, for learning related purposes or not. Classification was done using the C4.5 algorithm, specifically Weka's J48 implementation [Hall 09] inside of RapidMiner [Mierswa 06], using 0.25 for the pruning confidence threshold and 2.0 for minimum number of instances per leaf. And model validation was done at the same time using 10-fold cross validation. The main idea is that every image is represented by its 76800 pixels, and those pixels are used to find the cluster that will best fit the respective image and apply that cluster's label to the image.

3.5 Results

Our current results are polled from five sessions of data, ranging from one and a half to two hours each. One experiment was run for each cluster, per label. And we were able to test clusters with different threshold values for each data set, giving us an insight into what is the best threshold to use for further testing.

- In the first test, all activities done on the computer are labeled as computer-based activities (e.g. chrome.exe, javaw.exe, etc…). As we wanted to label the features with simply computer-based or not.
- The second test was a simple classification of the features (clusters) using the learning and non-learning labels.

- And the third test, we only examine the applications that the users used. Only those applications where the user indicated that he used for both learning and non-learning purposes were considered, hence further scaling down the dataset. In Dataset four and five, the user indicated that each application was exclusively used for either learning or non-learning purposes; hence we excluded that data from classification. Specifically, the user for Dataset one through three might have indicated that he used the application "chrome.exe" for both purposes, learning (e.g. watching a lecture, researching information) and non-learning (e.g. Facebook, YouTube), whereas the user for Dataset four and five said that the only time he used "chrome.exe" was for non-learning purposes (e.g. only Facebook). For this test, we tried to classify the programs using the learning and non-learning labels attached to them.



Data Set 1 - 7739 Images				
Threshold	2%	3%	4%	5%
Clusters	438	155	68	34
Clusters w/ Pruning	75	42	22	13
PC/Non PC	99.27% (99.04%)	97.81% (97.58%)	94.15% (94.15%)	92.90% (92.84%)
Learn/Non Learn	98.22% (97.98%)	97.45% (97.35%)	94.13% (94.13%)	93.27% (93.29%)
Programs-Learning	89.04%	86.71%	86.15%	86.04%
Posture & Program - Learning	95.30% (94.79%)	94.39% (94.28%)	92.82% (92.37%)	89.32% (89.38%)

Data Set 2 - 9438 Images				
Threshold	2%	3%	4%	5%
Clusters	591	212	98	50
Clusters w/ Pruning	96	61	41	23
PC/Non PC	99.96% (99.95%)	99.67% (99.67%)	99.92% (99.87%)	91.34% (91.61%)
Learn/Non Learn	99.29% (99.06%)	95.44% (95.54%)	98.05% (97.82%)	86.41% (86.42%)
Programs-Learning	55.51%	59.05%	59.51%	59.90%
Posture & Program - Learning	94.97% (94.56%)	89.28% (88.03%)	85.07% (84.27%)	77.78% (77.75%)

Data Set 3 - 6628 Images				
Threshold	2%	3%	4%	5%
Clusters	357	116	45	22
Clusters w/ Pruning	50	33	20	11
PC/Non PC	99.59% (99.29%)	99.37% (99.11%)	99.07% (99.00%)	97.24% (97.22%)
Learn/Non Learn	96.73% (95.72%)	92.10% (92.08%)	91.02% (90.98%)	88.16% (88.13%)
Programs-Learning	91.09%	91.05%	91.29%	91.34%
Posture & Program - Learning	97.98% (97.42%)	92.49% (92.84%)	92.54% (92.66%)	91.34% (91.41%)

Data Set 4 - 12406 Images				
Threshold	2%	3%	4%	5%
Clusters	841	413	223	131
Clusters w/ Pruning	135	94	65	47
PC/Non PC	99.26% (99.20%)	98.52% (98.45%)	98.56% (98.45%)	98.35% (98.28%)
Learn/Non Learn	94.15% (94.54%)	88.59% (88.88%)	86.23% (86.31%)	84.87% (84.88%)

Data Set 5 - 12370 Images				
Threshold	2%	3%	4%	5%
Clusters	810	287	121	60
Clusters w/ Pruning	149	69	49	27
PC/Non PC	99.99% (99.42%)	99.65% (99.32%)	99.59% (99.22%)	99.33% (99.11%)
Learn/Non Learn	99.99% (98.70%)	99.42% (98.67%)	99.26% (98.76%)	98.71% (98.37%)

Figure 5. Tables of data from each data collection session. Dataset 1-3 are from subject 1 and Dataset 4-5 are from subject 2. The numbers in parenthesis are the results from using the non-pruned clusters. Each subject collected data for about four hours, giving a total of about eight hours of data.

- The fourth test was done to try and see whether using posture data can help increase the accuracy in determining whether a user was using a certain application for learning or non-learning purposes. This is similar to the third test, except this time the posture cluster data is included in the classification process as well.

4. Analysis

From glancing at the results, displayed in Figure 5, we can immediately say that the lower the cluster threshold, the higher the average accuracy becomes. This is to be expected because the data mining is done to place labels on the created clusters, hence the stricter the cluster thresholds, the more cluster we will have, and the more accurate the labels will be. We also observed that the variation in accuracy between each of the three labels start to converge as well as the amount of clusters increases. There is a great disparity between the different accuracies when the threshold is high, but become closer and closer as the threshold value decreases. We can also see that the pruning of clusters did help increase accuracy in most cases.

The high accuracies are a direct result of the dataset. As data collection was done during a period where the subjects had imminent deadlines, the data might have been more skewed than if they were collected on a regular session of work. However, the most interesting points in the results is the fact that postures do help increase the accuracy of predicting a user's learning or non-learning behavior when using the same program. We see in Dataset 2 that there were accuracies increases of twenty to forty percent after incorporating postures into the test, as opposed to guessing whether the user is doing learning or non-learning activities purely based on the program that he is using and patterns in his behavior while using that program.

5. Conclusion

Further testing and data collection still has to be done, but already there are glimpses as to what postures can do in terms of helping to predict a user's behavior while he is engaging in learning and non-learning activities at his desk. Although we did not make use of the depth data available to us in this particular experiment, mainly because we assumed that it would have contributed very little to the purpose of posture-pattern recognition. We would like to make use of such data in the future when we approach activity recognition from depth images, using the method of [Shotton 11] to segment depth images into recognized body parts. We also expect that more advancement will be made in the realm of sitting posture recognition using the Kinect to come in May as Microsoft release the next version of their Software Development Kit which they stated will offer support for sitting skeletal tracking, and skeletal tracking is far superior to processing information off of a depth image.

Acknowledgments

This work is supported in part by the Management Expenses Grants for National Universities Corporations through the Ministry of Education, Culture, Sports, Science and Technology (MEXT) of Japan, by the Global COE (Centers of Excellence) Program of MEXT and by KAKENHI 23300059.

References

[Aleven 10] Aleven, V., Rule-Based cognitive modeling for intelligent tutoring systems advances in intelligent tutoring systems, Studies in Computational Intelligence Vol. 308, Ch. 3, pp. 33-62, Springer Berlin / Heidelberg, Berlin, Heidelberg, (2010).

[Foehr 06] Foehr, U. G., Media multitasking among American youth: Prevalence, predictors and pairings, Henry J. Kaiser Family Foundation, Menlo Park, CA, (2006).

[Graesser 99] Graesser, A. C., Wiemer-Hastings, K., Wiemer-Hastings, P., and Kreuz, R., AutoTutor: A simulation of a human tutor, Cognitive Systems Research, 1(1):35-51, (1999).

[Hall 09] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I. H., The WEKA Data Mining Software: An Update, SIGKDD Explorations, vol. 11, Issue 1, (2009).

[Inventado 11] Inventado, P. S., Legaspi, R., Suarez, M., and Numao, M., Investigating transitions in affect and activities for online learning interventions, In Proceedings of the 19th Conference on Computers in Education, pp. 571-578, Chiang Mai, Thailand, (2011).

[Jaimes 06] Jaimes, A., Posture and activity silhouettes for self-reporting, interruption management, and attentive interfaces, In Proceedings of the 11th International Conference on Intelligence User Interfaces, IUI ’06, pp. 24-31, New York, NY, USA, ACM, (2006).

[Mierswa 06] Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., Euler, T., YALE: Rapid Prototyping for Complex Data Mining Tasks, in Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-06), (2006).

[Mitrovic 12] Mitrovic, A., Fifteen years of constraint-based tutors: what we have achieved and where we are going, User Modeling and User-Adapted Interaction, 22:39-72, (2012).

[Ray 12] Ray, S. J., and Teizer, J., Real-time construction worker posture analysis for ergonomics training, Advanced Engineering Informatics, 26(2):439-455, (2012).

[Shotton 11] Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., and Blake, A., Real-time human pose recognition in parts from single depth images, CVPR, (2011).

[Wientapper 09] Wientapper, F., Ahrens, K., Wuest, H., and Bockholt, U., Linear-projection-based classification of human postures in time-of-flight data. In Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics, SMC’09, pp. 559-564, Piscataway, NJ, USA, IEEE Press, (2009).