2023 IEEE International Conference on Control, Electronics and Computer Technology (ICCECT) | 979-8-3503-9902-8/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICCECT57938.2023.10140414

An Investigation into the Detection of Human Scratching Activity Based on Deep Learning Models

Kevin Wang
Shanghai High School International Division
Shanghai, China
course@mfeducation.cn

***Abstract*—Because pruritus is often overlooked and undertreated in the clinical setting, a major unmet need is objective measures of behaviors associated with scratching in order to quantify itch severity and frequency since scratch directly correlates to itch. Such methods to measure itch and how itch severity changes over time are needed to objectively study and understand pruritus, develop and assess the efficacy of new medications, quantify disease severity in patients, and monitor treatment response. Wearable sensors in the form of wrist actigraphy, which detects wrist movements over time using micro-accelerometers, are the most studied and tested method to detect scratching events. To address these issues, 7 deep learning models will be used to train and test for scratch detection, including:Convolutional Neural Network (CNN), Recurrent Neural Network (RNN) – Gated Recurrent Unit (GRU), RNN – Long Short-Term Memory (LSTM), CNN & RNN – GRU (end-to-end), CNN & RNN – LSTM (end-to-end), CNN & RNN – GRU (parallel) and CNN & RNN – LSTM (parallel). The final results show accurately detect scratching using deep learning (CNN achieved a high accuracy of 0.996) in various situations and can provide useful information (time, frequency, scratched body part, etc.) regarding the scratching behavior in day and nighttime in order to better quantify pruritus for use in the medical field.**

Keywords—scratch detection, triaxle acceleration, wearable sensor, deep learning, CNN, RNN

I. Introduction

A. Pruritus

Pruritus (itch) is a primary symptom of various medical conditions including Atopic Dermatitis [1]. The primary reaction to pruritus is to constantly scratch the affected area [2, 3], leading to constant distraction, cardinal pain, and breakdown of the skin [4], greatly troubling pruritic individuals, as shown in Fig.1.

Fig. 1. Skin Deterioration from Scratching Due to Pruritus [4]

B. Demand

Because pruritus is often overlooked and undertreated in the clinical setting [5], in AD and other conditions that cause itch, a major unmet need is objective measures of behaviors associated with scratching in order to quantify itch severity and frequency since scratch directly correlates to itch. Such methods to measure itch and how itch severity changes over time are needed to objectively study and understand pruritus, develop and assess the efficacy of new medications, quantify disease severity in patients, and monitor treatment response, possibly benefiting millions of people suffering from pruritic conditions.

Fortunately, advances in wearable sensor technology have already led to more objective measures of health, both within and outside of healthcare settings. Wearable sensors in the form of wrist actigraphy, which detects wrist movements over time using micro-accelerometers, are the most studied and tested method to detect scratching events. They are able to record longitudinally over several days, giving a more accurate representation of chronic itch. However, the output from actigraphic devices is not specific to scratching motions: they detect any movement the subject makes. As a result, several researchers have attempted to develop machine learning algorithms to distinguish scratch from other actigraphic movements recorded.

C. Related Work

For example, Feuerstein et al. utilized four signal features derived from accelerometer data obtained from wrist actigraphy with a k-means clustering technique to segment simulated scratching movements (scratching performed on command in a clinic setting) from walking and restless movements during sleep. Petersen et al. built on this approach by leveraging the same four signal features with logistic regression to also classify simulated scratching movements from walking and restless movements during sleep, as shown in Fig.2.

Fig. 2. Wrist Accelerometer Device Used by Peterson et al. (Philips Respironics PAM-RL)

Later research by Moreau et al. trained Recurrent Neural Networks (RNNs) using annotated scratch events during an overnight clinic visit to classify nighttime scratch directly from sample-level accelerometer data.

However, the mainstream approaches listed above have noticeable imperfections:

a) False Positives: Wrist actigraphy have inherent difficulties in distinguishing between non-scratching wrist movements from scratching resulting in false positives in everyday activities with similar waveforms, such as walking and waving.

b) Suboptimality: These approaches by nature offers suboptimal accuracy because wrist actigraphy struggles to identify finger-dominant scratching movements, when primarily the fingers move back and forth, instead only having the ability to detect wrist-dominant ones, when primarily the wrist moves back and forth.

c) Weak Algorithms: Deep learning is a subset of machine learning, which is essentially a neural network with three or more layers. These neural networks attempt to simulate the behavior of the human brain — albeit far from matching its ability — allowing it to “learn” from large amounts of data. Classical machine learning algorithms leverage structured, labeled data to make predictions—meaning that specific features are defined from the input data for the model and organized into tables. Deep learning eliminates some of data pre-processing that is typically involved with machine learning. These algorithms can ingest and process unstructured data, like text and images, and it automates feature extraction, removing some of the dependency on human experts. Despite the advantages of deep learning, accurate deep learning implementations of scratch detection algorithms are rare in these approaches, though deep learning have proven to be highly successful in similar applications, much better than many non-deep learning methods such as classical machine learning algorithms. For instance, a deep learning model using Convolutional Neural Network (CNN) outperformed conventional machine learning algorithms such as SVM and LDA when employed for activity recognition based on wrist worn accelerometer data. In addition, in a sleep staging study using heart rate and wrist actigraphy, a deep learning model using bidirectional Long-Short Term Memory (LSTM) based Recursive Neural Network (RNN) outperformed classic classifiers such as Support Vector Machine (SVM) and Random Forest (RF). While Moreau et al. utilized deep learning by using the RNN – LSTM architecture, they achieved a relatively low f1 score, a measure of a test’s accuracy, of 0.68. Their model also only employed one deep learning architecture, namely RNN – LSTM, which provided limited insight into the potential of other deep learning architectures for the purpose of scratch detection.

d) Usage Limitations: Current approaches are only designed to detect nighttime scratch, likely due to accuracy concerns, since these movements during sleep generally involves a noticeable amount of involuntary wrist movement and have limited non-scratching cases (most notably occasional turnings due to restless sleep).

e) Over-simplified Feedback: The feedback given by the devices and algorithms of these approaches are extremely simple (namely whether or not the current action is scratching).

Thus, this investigation aims address the limitations of previous approaches to scratch detection and find a deep learning models can be used to perform more effective scratch detection.

II. Methods and Materials

This investigation proposes a new accelerometer sensor layout design and utilization method. Instead of only placing an accelerometer on the wrist, accelerometers are also placed on each individual finger. Thus, there will be 6 sensors on each device for each hand. By capturing the movement of each finger in addition to the wrist, the data collected could be processed by the algorithms to more reliably distinguish between scratching wrist action and similar non-scratching wrist actions. The new category of finger data input also allows the algorithm to accurately recognize finger-dominant scratching activity as scratching, whereas previously since only wrist action is recorded, there is inherent difficulty in detecting these finger-dominant scratching movements. This design is low cost, lightweight, ye more comprehensive than previous devices (wrist actigraphy with only one wrist accelerometer), as shown in Fig.3.

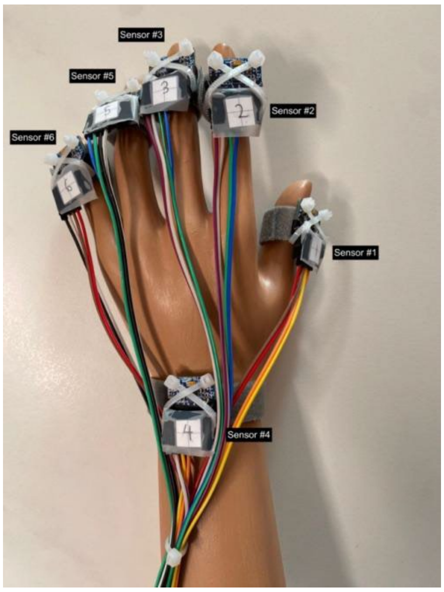

Fig. 3. Sensor Layout on Hand

A layout of the sensors on the hand is displayed in Figure 3. As seen, sensor #4 is placed on the wrist and sensors #1-3, #5-6 are placed near the tip of each finger since the it is the tip that is closest in contact with the skin during scratching.

Since both hands have the ability to scratch and the relative directions and accelerations of each hand’s scratching behavior may be slightly different, the above data needs to be collected for each hand in order to ensure accurate predictions for both hands when engaged in scratching activity.Thus, according to the specifications above, 42 total groups of data will need to be obtained:

However, the data is split into the 42 groups listed for collection purposes only; when training the models, there will only be 8 classes of data for prediction:

1) face: groups 1-3, 22-24; those that scratch the face.

2) neck: groups 4-6, 25-27; those that scratch the nape of the neck.

3) left_knee: groups 7-9, 28-30; those that scratch the

inside crease of the left knee.

4) right_knee: groups 10-12, 31-33; those that scratch the inside crease of the right knee.

5) left_elbow: groups 34-36; those that scratch the inside crease of the left elbow.

6) right_elbow: groups 13-15; those that scratch the inside crease of the right elbow.

7) others: groups 16-18, 37-39; those that scratch other parts of the body, such as the back and chest

8) none: groups 19-21, 40-42; those that perform non-scratching activities

That is, when raw acceleration data is inputted into the models, the movement will be classified into one of the 8 classes listed above. Since each of the 2 participants provided 42 groups of data, and each data contains 60 seconds of hand movements, there will be a total of 5040 seconds or 84 minutes of data overall for training and testing the models.

A. Data Preprocessing

The acquired data accelerometer raw data consists of 3-dimensional (x-y-z) acceleration data of each finger and wrist. In order to train and test machine learning models on the data, it is split into windows of a given length and stride using the sliding window technique. Then, once all of the data is processed in this manner, the data is further split 80%-20% into training and testing datasets, respectively, by randomly selecting 20% of the data from each group and inserting it into the test dataset while the remaining 80% is inserted into the training dataset.

The optimal length and stride of the sliding window for scratch detection will be determined in the following experiment by simply trying many different combinations and selecting the best one.

B. Algorithms (Models)

To address accuracy concerns c (weak algorithms), 7 deep learning models will be used to train and test for scratch detection, including: Convolutional Neural Network (CNN), Recurrent Neural Network (RNN) – Gated Recurrent Unit (GRU), RNN – Long Short-Term Memory (LSTM), CNN & RNN – GRU (end-to-end), CNN & RNN – LSTM (end-to-end), CNN & RNN – GRU (parallel) and CNN & RNN – LSTM (parallel).

1) Convolutional Neural Network (CNN)

CNN is a deep learning neural network designed for processing structured arrays of data and is particularly successful at learning patterns in the input arrays. Structurally, CNN is a feed-forward neural network, though what makes it unique is the hidden convolution layers within the network, which are stacked on top of each other and is able to progressively recognize more sophisticated patterns in the data. Convolutional layers are typically followed by a pooling layer, allowing the model to summarize the features generated by the convolutional layers and decreased the size of convolved feature map to reduce computational costs. Ultimately, these layers lead to a fully connected classification layer that utilizes the output from the convolution feature extraction process to predict the class of the original input based on the features extracted in previous layers, as shown in Fig.4.

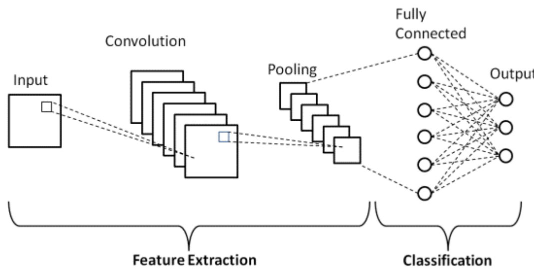

Fig. 4. Basic CNN Architecture

2) Recurrent Neural Network (RNN) – Gated Recurrent Unit (GRU)

RNN is a type of neural network that contains loops, allowing information to be stored within the network. Thus, RNNs have the ability to use reasoning from previous information along with those currently being processed to predict certain structures or upcoming events. Because of this, RNN is also most effective in processing sequential data for prediction.

Its structure is different from the traditional feed-forward neural network (which has an input layer, hidden layers, and output layer) in that an additional loop is added to the neural network to pass prior information forward. Such a looping mechanism in RNN is what allows information to flow from one step in the data to the next. This information is known as the hidden state, as shown in Fig.5.

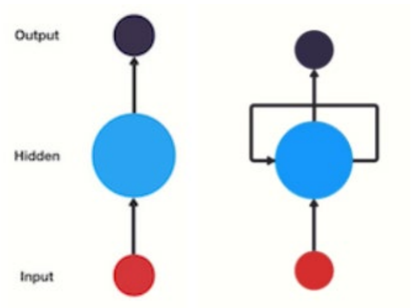

Fig. 5. Basic Feed-Forward Neural Network and RNN Structure

An RNN unit takes information from previous steps and the current input, as shown in Fig.6.

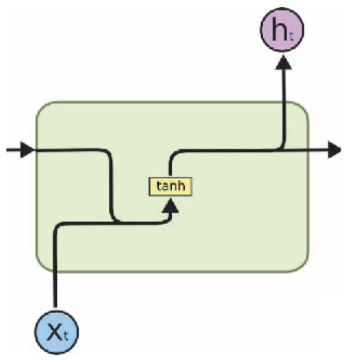

Fig. 6. Basic RNN Unit Architecture with a tanh Activation Function

To train an RNN network it is backpropagated through time, and at each step, the gradient is calculated, which updates the weights in the network. If the effect of the previous layer on the current layer is small then the gradient value will also be small. However, this makes the gradients shrink exponentially as backpropagation occurs, which might be too such that it will have essentially no effect on weight updates. Due to this, the network would have difficulty learning from the effects of earlier inputs, causing what is referred to as the vanishing gradient problem. GRU is a specialized version of RNN to overcome this problem, making use of memory cells to store the activation value of previous information in a long sequence. Gates thus are used to control the flow of information in the network by learning which inputs in the sequence are important and storing their information in the memory unit. Therefore, they can pass information in long sequences and the network can use them to make predictions.

Specifically, there are 2 gates inside a GRU unit: the reset gate (r_t) and the update gate (z_t). The reset gate is used to decide whether the previous cell state is important or not, and the update gate decides if a cell state should be updated with the candidate state (c_t). The candidate state is simply the activation value from the current input in the sequence. The final state (c^t) is thus dependent on the update gate as it may or may not be updated with the candidate state, as shown in Fig.7.

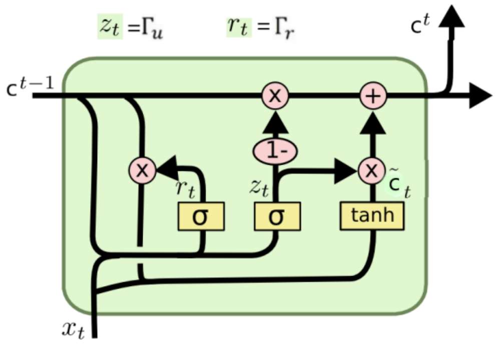

Fig. 7. Basic GRU Unit Architecture

3) RNN – Long Short-Term Memory (LSTM)

LSTM is another approach for RNN to solve the vanishing gradient problem. It has the same workflow as GRU but the difference is the operations performed in the unit. Each LSTM unit contains 2 gates: the forget gate (f_t) and output gate (o_t) instead of the reset gate. The forget gate controls what is being kept or forgotten from all previous cell states, and the output gate controls which parts of the cell are outputted to the hidden state, which determines the next hidden state, as shown in Fig.8.

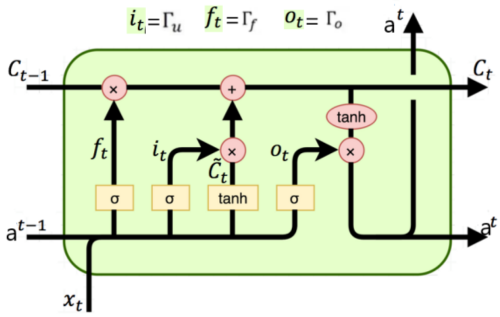

Fig. 8. Basic LSTM Unit Architecture

4) CNN & RNN – GRU (end-to-end)

An RNN – GRU is connected end-to-end with a CNN network. First, the CNN extracts patterns through the data, then the GRU network predicts the final output using the sequential patterns in the input data.

5) CNN & RNN – LSTM (end-to-end)

The same as the above, except that the GRU network is replaced by the RNN – LSTM network.

6) CNN & RNN – GRU (parallel)

The input is processed by CNN, extracting major patterns within the data. At the same time, the input is also processed by RNN – GRU, analyzing the data sequentially. These outputs of these two networks are concatenated together and pass through a fully connected layer to determine the final prediction.

7) CNN & RNN – LSTM (parallel)

The same as the above, except that the GRU network is replaced by the RNN – LSTM network.

C. Training and Testing Models

The training and evaluation of the various models is performed using the PyTorch v1.10 machine learning framework built by Meta AI, and the entirety of the code is written and tested in the Python programming language using the PyCharm IDE. In addition, all of the processing, training, and evaluation of the machine learning models are run on an Nvidia GeForce RTX 3080 GPU using cloud GPUs hosted by Featurize.

Repeat the following steps for each of the 7 models listed above and record the best results for each:

a) Input and process the raw tri-axial acceleration data into sliding windows of a specified window length and stride

b) Split the data into 80% - 20% for training and testing, respectively, as well as generate the relevant datasets

c) Initialize and train the model using the training data and test for performance after each epoch and output the accuracy to the console for a total of 50-200 epochs

d) Vary model properties including the kernel, kernel size, dropout rate, presence of batch normalization, hidden size, number of layers, whether or not bidirectional, and repeat step 3 until the a relatively good combination for each model is found

e) Record the maximum accuracy for each model

Note: Through various testing, the optimal window length is determined to be 5 seconds and the window stride is 1 second. This equates to 3864 training and 924 testing samples. The batch size is also set to be 256 samples. In addition, the best-performing optimizer and loss function is Adam and cross-entropy loss, respectively. For the optimizer, the learning rate is set to be 0.001. These properties will remain the same for all models trained and tested.

III. RESULTS

A. Scratch Detection

Through continuously varying model properties and following the steps listed above, the model results were steadily improved until a relatively stable and good one was reached. The best performance of each model is measured by the accuracy of the model on a randomized test dataset, where accuracy is the number of correct classifications made by the model over all the test samples.

Not only was each model tested on an overall randomized dataset with samples from each hand, each body part, each type of wrist-finger movement, the models were also tested on their accuracy in specifically classifying different movements in terms of wrist-dominant, finger-dominant, or wrist & finger. Such test samples were generated by selecting the relevant samples (say, wrist-dominant movements) within the overall test dataset and testing them on the model again, but separately as a smaller test subset, as shown in Table I.

TABLE I. EACH MODEL'S OVERALL ACCURACY AND ACCURACY FOR EACH SPECIFIC MOVEMENT TYPE (MODEL IS TRAINED ON DATA COLLECTED FROM THIS INVESTIGATION'S DEVICE, CONTAINING EACH FINGER AND WRIST'S ACCELERATION DATA)

Models	Overall Accuracy	Wrist-Dominant Movement Accuracy	Finger-Dominant Movement Accuracy	Wrist & Finger Movement Accuracy
CNN	0.996	1	1	0.996
RNN – GRU	0.702	0.473	0.371	0.367
RNN – LSTM	0.694	0.303	0.25	0.292
CNN & RNN – GRU (end-to-end)	0.54	0.25	0.208	0.212
CNN & RNN – LSTM (end-to-end)	0.404	0.174	0.186	0.193
CNN & RNN – GRU (parallel)	0.868	0.652	0.591	0.735
CNN & RNN – LSTM (parallel)	0.839	0.792	0.777	0.864

To further demonstrate the advantage of the method this investigation proposed over previous research utilizing wrist actigraphy, wrist acceleration was specifically isolated from the data collected previously and all the models were trained and tested on solely wrist acceleration data to perform scratch detection, simulating previous mainstream wrist actigraphy approaches, as shown in Table II.

TABLE II. EACH MODEL'S OVERALL ACCURACY AND ACCURACY FOR EACH SPECIFIC MOVEMENT TYPE (MODEL IS TRAINED ONLY ON WRIST DATA COLLECTED)

Models	Overall Accuracy	Wrist-Dominant Movement Accuracy	Finger-Dominant Movement Accuracy	Wrist & Finger Movement Accuracy
CNN	0.767	0.882	0.614	0.712
RNN – GRU	0.484	0.235	0.186	0.242
RNN – LSTM	0.413	0.299	0.189	0.227
CNN & RNN – GRU (end-to-end)	0.26	0.186	0.152	0.152
CNN & RNN – LSTM (end-to-end)	0.332	0.261	0.152	0.152
CNN & RNN – GRU (parallel)	0.621	0.409	0.333	0.39
CNN & RNN – LSTM (parallel)	0.66	0.617	0.326	0.557

This investigation utilized various deep learning architectures to perform scratch detection. When the models are trained on finger and wrist acceleration data that is collected by this investigation's device, overall accuracies were immediately improved for all models and accuracies for specific movements mostly improved. CNN, for example, saw its overall accuracy increase from 0.767 previously to 0.996 after training on finger and wrist acceleration (table I, II), becoming the most accurate model in this investigation and this investigation's final result. Its accuracies for detecting all movements (wrist-dominant, finger-dominant, wrist & finger) all increased significantly as well.

IV. CONCLUSION

This investigation utilized various deep learning architectures to perform scratch detection. Overall, deep learning models proved to be significantly better than non-deep learning models, such as SVM and RF, similar to those employed by most mainstream approaches. When the models are trained using this investigation's device, the most accurate deep learning model, CNN, reached an overall accuracy of 0.996 surpassing others, by a wide margin (Table I). CNN's high accuracy might have resulted from the advantage of constantly learning patterns in the data and obtaining higher level information related to scratching movements. This observation is also true when the models are only trained on wrist acceleration data (simulating previous approaches), as CNN obtained an overall accuracy of 0.767, noticeably better than others (Table II), proving the potential of deep learning in scratch detection applications, especially CNN, which was not even explored in previous research. Hopefully, this project is able to provide more insights into the task of scratch detection, demonstrating the potential of deep learning models (including CNN, which had not been used before), usage of finger movement data alongside wrist movement acceleration, and new objective metrics to quantify scratch and itch severity. By identifying imperfections of previous research, this investigation is able to address them and propose a better method that allows medical professionals to accurately detect and monitor patient scratching activity; in doing they can not only appropriately assess patient condition and provide more effective treatments, but also to develop and test new therapies for pruritus-related diseases that will benefit millions of young children and adults.

REFERENCES

[1] Kapur, S., Watson, W. & Carr, S. Atopic dermatitis. Allergy Asthma Clin. Immunol. 14, 52 (2018)

[2] Ebata, T., Aizawa, H. & Kamide, R. An infrared video camera system to observe nocturnal scratching in atopic dermatitis patients. J. Dermatol. 23, 153–155 (1996)

[3] Endo, K., Sano, H., Fukuzumi, T., Adachi, J. & Aoki, T. Objective scratch monitor evaluation of the effect of an antihistamine on nocturnal scratching in atopic dermatitis. J. Dermatol. Sci. 22, 54–61 (1999)

[4] Camfferman, D., Kennedy, J. D., Gold, M., Martin, A. J. & Lushington, K. Eczema and sleep and its relationship to daytime functioning in children. Sleep Med. Rev. 14, 359–369 (2010)

[5] Staff, S. X. (2020, July 10). Nemolizumab beats placebo for reduction of pruritus in eczema. Medical Xpress. Retrieved November 12, 2022, from https://medicalxpress.com/news/2020-07-nemolizumab-placebo-reduction-pruritus-eczema.html.