Rock beats Scissor: SVM based gesture recognition with data gloves

Philipp Achenbach
Multimedia Communications Lab
Technical University of Darmstadt
64283 Darmstadt, Germany
0000-0003-4948-4440

Philipp Niklas Müller
Multimedia Communications Lab
Technical University of Darmstadt
64283 Darmstadt, Germany
0000-0001-9660-691X

Tobias Alexander Wach
Multimedia Communications Lab
Technical University of Darmstadt
64283 Darmstadt, Germany
0000-0002-1521-9801

Thomas Tregel
Multimedia Communications Lab
Technical University of Darmstadt
64283 Darmstadt, Germany
0000-0003-0715-3889

Stefan Göbel
Multimedia Communications Lab
Technical University of Darmstadt
64283 Darmstadt, Germany
0000-0003-3657-8744

***Abstract*—Hand gestures play an important role in human communication, particularly when auditory communication is limited. Akin to speech recognition, hand gesture recognition can therefore be a useful tool to facilitate communication and for more immersive computer interaction. In this paper, we examine the mobile recognition of hand gestures using data recorded with sensor gloves. We design a system based on Support Vector Machines (SVM), capable of recognizing 5 different hand gestures. In an experiment with 11 participants, we determine applicable hyperparameters based on performance on the training set which translates into 100% classification accuracy on the test set. In an additional practical experiment with 9 participants, our system achieves up to 98% in a personalized and up to 87.5% in a generalized model setting.**

***Index Terms*—gesture recognition, wearable, machine learning, data glove, support vector machine, rock-paper-scissor**

I. Introduction

Gestures - especially hand and finger gestures - are an essential part of our human communication. They help us to express thoughts and feelings and also visually support our spoken language. With the help of sign language, it is even possible to express whole sentences with gestures only.

As a form of communication, gestures also play an increasing role in Human-Computer-Interaction (HCI). Wipe and touch gestures are already well established in the field of smartphones and tablets.

With the spread of Virtual Reality (VR) and Augmented Reality (AR), the desire to dive deeper into the virtual world is also growing. An essential part of this is a more intuitive way of interaction. Modern controllers and other input devices are mostly limited to motion sensors, joysticks and buttons. This restricts the user's input possibilities since intuitive operating options are lacking. However, the hand and finger gestures often used in everyday life have not yet been able to establish control of a technical device. The reason for this is the difficulty of reliably recognizing and distinguishing the different gestures and hand movements with conventional approaches.

The focus of this paper is on gesture recognition using a data glove. As an example, we have decided to implement the well-known game of rock paper scissor. Fig. 1 shows the gestures used by us including two additional gestures *well* and *match*.

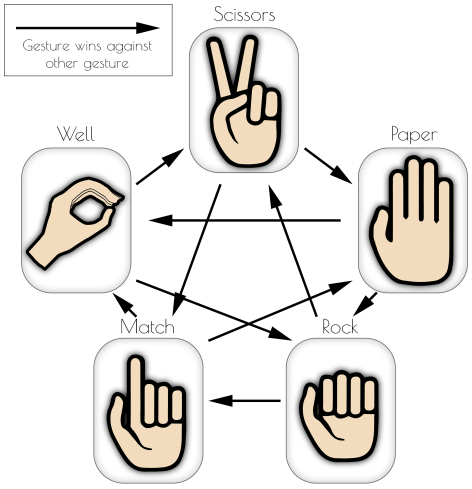

Fig. 1. Rock paper scissor game with additional gestures for *well* and *match*

II. Background & Related Work

To recognize hand gestures, we first investigate which hardware is suitable to capture gestures for our purpose. Then, related works with the same hardware are examined to choose the appropriate machine learning approach to classify the gestures.

A. Gesture capturing

Approaches to capture gestures can be divided into two types: Vision-based and sensor-based approaches [1]. Since we want to achieve a mobile approach with a high degree of

978-0-7381-4348-4/21/$31.00 ©2021 IEEE

suitability for everyday use and for reasons of data privacy, we decided against an external observer like a camera. Therefore we will focus on a sensor-based approach in form of a data glove, due to their good availability.

Individually developed devices with sensors are a less frequently used method to obtain data on the current hand and finger positions. These devices are developed specifically for each project and are therefore costly and time-consuming in both development and production. An alternative is offered by various data gloves, which are commercially available, especially in the field of VR.

The data gloves used for this work are the *Senso Glove: DK2* from *Senso Device Inc*[1]. By using 8 Inertial Measurement Units (IMUs) and two magnetometers per glove, it offers the possibility to detect hand and finger movements accurately and cameraless. Fig. 2 shows the *Senso Glove* with 5 IMUs mounted on the middle phalanx of each finger. An additional second one on the thumb is located a bit lower because the human thumb has more freedom in its movement than the other fingers such that there are two sensors needed to provide accurate information. As for the back of the hand and the wrist, there is one IMU sensor located at each of those as well.

All of the 8 sensors measure the current orientation of their respective part of the hand precisely through a combination of accelerometers and gyroscopes. The sensors on the wrist and the palm also contain a magnetometer, whose values can be accessed as well. All sensors provide data at 10 Hz. The exact type of data can be found in the later part (see Section III-A) of this paper.

Fig. 2. Data glove used in this work: *Senso Glove DK2*

B. Gesture recognition using data gloves

Ma et al. have investigated different approaches to classify finger gestures with the help of a custom-designed data glove [2]. The glove uses flex sensors to measure the bending of each finger and one IMU to measure the rotation and acceleration of the wrist.

Five different methods were compared for the classification: Feed-forward Neural Network (FNN), SVM, k-Nearest Neighbor (KNN), Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU). The last two methods used the data without feature extraction.

For the evaluation, data were collected from 10 people who bent each finger one after the other, resulting in 5 very simple finger gestures. The obtained sensor data was first segmented to determine the beginning and end of a gesture and then features were extracted.

The comparison showed that the 4-layer FNN had the highest accuracy of 94.3%, followed by the SVM with an accuracy of 89.3% and KNN with an accuracy of 85.4%. The two methods without Feature Extraction performed by far the worst with accuracies of only 37.1% and 38.4%.

Plawiak et al. investigated hand gesture recognition using the commercial data glove *DG5 VHand* from *DGTech Engineering Solutions*[2] with different classifiers [3]. This glove uses a bending sensor for each finger and measures the accelerations of the hand for each axis as well as the angular accelerations for *roll* and *pitch*.

Data were collected from 10 people performing 22 hand body language gestures, each of them 10 times. Examples of gestures were the common “okay” gesture and the gesture of cutting a sheet of paper with two fingers.

The classification was performed using 3 different methods: Probabilistic Neural Network (PNN), SVM and KNN. Parameters for SVM were optimized by using a genetic algorithm, parameters for the other two methods were optimized manually.

The result of the evaluation shows that the use of a SVM with hyperparameter optimization gave the best results, with an accuracy of 98.32%, followed by KNN with 97.36%.

A paper that used a self-developed data glove for the recognition of rock paper scissor gestures was presented in 2009 by Kim et al. [4]. The data glove contained 3 3D-accelerometers which were attached to the thumb, middle finger and back of the hand.

The recognition of the gestures was purely rule-based. Beginning with a starting position, rules were developed as to how the signals of the 3 acceleration sensors must behave to recognize the gestures.

Using this recognition method, the authors achieved an accuracy of 100% for each gesture. Each gesture was executed 50 times.

In another paper, two different static gestures were recognized in real-time with a data glove [5]. They used the same data glove as in our approach and also SVM for classification. After training their model with a total of 100 samples for each gesture, an accuracy of, on average, at least 89% could be achieved.

[1]https://senso.me (Last visited on 30. November 2020)

[2]http://www.dg-tech.it/vhand3/products.html (Last visited on 30. November 2020)

C. Gesture classification

The classification algorithm provides the foundation for the planned hand and finger gesture recognition. The final goal is to create a model that is trained by a given amount of training data. Algorithms from supervised learning create a mathematical model from given training data [6]. During the learning process, an attempt is made to learn the relationships and rules of the training data and to represent them in the model to be able to distinguish new data according to these if possible. Therefore, algorithms of supervised learning represent a suitable selection for our specific case.

Due to the low complexity of the problem of hand gesture recognition as well as the comparatively small amount of available training data, a method has to be chosen which provides satisfactory accuracy despite these two points. The methods KNN and SVM have already been used in several papers with good results as shown in II-B. Some papers also implemented more than one classification algorithm and compared their achieved accuracies. When using a data glove, the trend can be seen that a SVM achieves the highest accuracy among the two classification algorithms mentioned above.

Support Vector Machines (SVM) were introduced in 1964 by Vapnik and Chervonenkis [7] and can separate a set of data into two different classes. Therefore the data are mapped in vector space. The SVM then generates a hyperplane that separates the data of both classes. The hyperplane is chosen to maximize the margin to the next element of each class. Unknown data can now be represented in vector space and assigned to one of those two classes. Fig. 3 shows this approach.

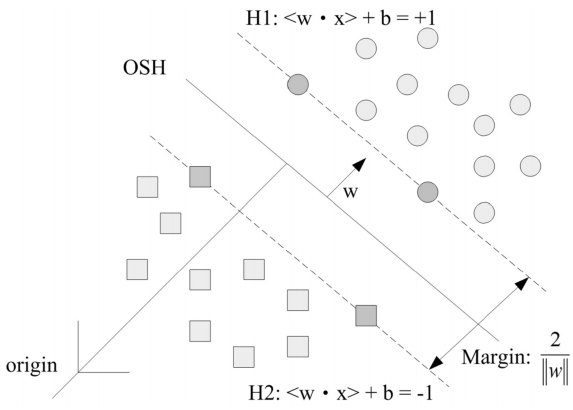

Fig. 3. OSH: Optimal separating hyperplane. By maximizing the margin between data of both classes, the error rate of classifying unseen data can be minimized [8].

Since data are not always linearly separable, Cortes et al. have introduced soft margins in 1995 [9]. This can be seen as an aberration around the hyperplane, where no clear assignment is possible. Also by using soft margins, a generally high separability of the data must be available.

The "kernel trick" is typically used to improve classification of data that cannot be separated linearly without errors. With this method, the vector space is transformed into a higher-dimensional space to perform a linear separation by a hyperplane [10].

In order to separate more than two classes, a combination of multiple SVMs can be used. Among the various ways to combine SVMs, the most commonly used methods are *One-against-All* [11] and *One-against-One* [12].

With *One-against-All* one classifier is trained for each class, which separates whether the data belong to the considered class or not, thus k classifiers. With *One-against-One* each SVM should be able to separate a pair of classes and is trained only with the data which belongs to either of it's both classes. For each combination of classes there is one SVM, thus $\frac{k(k-1)}{2}$ classifiers. To classify new data, each SVM is applied to the data and gives a "point" to the class that it predicts such that the class with the highest score will be the final classification.

To choose between these methods, we refer to two different papers. The first paper is from Hsu and Lin [13] which compares different methods for multi-class SVMs, including both methods in question. They showed that the method *One-against-One* had both the highest accuracy and the lowest training duration. All other methods that were compared in addition to the two methods mentioned above performed much worse. The second comparison is from Milgram et al. [14], who compared the two methods in question in a handwriting recognition scenario. As for their results, they state that both methods performed about the same, but the training process using the *One-against-All* method took much longer than with the *One-against-One* method.

III. Towards data acquisition and processing

A. Data acquisition and preprocessing

The *Senso Glove: DK2* provides its data via one of the two plugins for *Unity* and *Unreal Engine 4* or directly through a custom program that makes use of the network protocol documented on their website. In our case, we use the provided plugin for the *Unity* engine.

For both, the training data and the test data, the users were instructed to perform the gestures as shown in Fig. 1. The rotation and position of the hand, as well as the body, were not specified. Also, which hand (left or right) should be used was not specified. These decisions were based on the fact that gesture recognition should be as natural and universal as possible, i.e., that the players should be given as much freedom as possible in executing the gesture.

This also includes potential arm movements during execution a gesture. To minimize their influence, the data were not recorded over a period of time but at a fixed point in time. This point in time was announced to the test person via a countdown and conveyed via visual and haptic feedback.

As mentioned before, the glove itself has 8 IMU sensors, located at the fingers, the back of the hand and the wrist. Table I provides an overview of all values that are accessible through the *Unity* plugin that is being used in our application. Additionally, the values that will be used for the actual gesture recognition are marked. As shown there, we only use the

values that come from the sensors on the fingers. The main reason for this is that the finger values provided by the glove already represent the local orientation and position of the fingers, meaning that the processor on the glove already converts the global values that an IMU sensor usually provides into local values using the information from the palm and wrist. As a result, the values are independent of the rotation and position of the hand itself and thus the sensor values for the palm and the wrist are not needed to classify the gestures.

TABLE I
OVERVIEW OF AVAILABLE AND USED SENSOR VALUES

Value	Datatype	Used	$\sum$ Floats
Thumb Angles	Vector2	✓	2
Thumb Quaternion	Quaternion	✓	4
Thumb Bend	Float	✓	1
Index Finger Angles	Vector2	✓	2
Middle Finger Angles	Vector2	✓	2
Ring Finger Angles	Vector2	✓	2
Little Finger Angles	Vector2	✓	2
Palm Position	Vector3	✗	3
Palm Rotation	Quaternion	✗	4
Palm Magnetometer	Vector3	✗	3
Wrist Rotation	Quaternion	✗	4
Wrist Magnetometer	Vector3	✗	3

Since the glove provides the sensor data as already preprocessed digital data (refer to Table I), there is very little data preprocessing required from our side. Every value can be split up into individual floating-point numbers, resulting in 15 of those every 100 ms as input data for the classification algorithm. The only preprocessing we add is a linear scaling of the data, as this is highly recommended when using a support vector machine (see Section III-B) for classification [15]. We use a min-max-scaling to scale the range of our data to $[-1, 1]$.

Let D be a data set containing n points $(\vec{x}_1, y_1), \ldots, (\vec{x}_n, y_n)$, each point for one static gesture y_i. For every feature $x_i \in \vec{x}$, which represent the used sensor values from Table I, there is a maximum and a minimum value x_{max} and x_{min} in this data set, using which we scale the data as follows:

$$f_{scaling}(x) = 2\frac{x - x_{min}}{x_{max} - x_{min}} - 1 \tag{1}$$

The values x_{min} and x_{max} from the training data are saved in a file and later used to scale every new data point that will be classified because we need the scaling of the training data and new data to be the same.

B. Data processing through SVMs

To classify the different gestures, a robust and reliable classification method is needed. In this paper, we decided to use a c-classification SVM, which has proven to be a good algorithm for gesture recognition in various other research papers. As our data will not be linearly separable, the so-called *kernel trick* is needed to use a support vector machine for our data. In our case, we chose the radial basis function kernel defined as:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\gamma \left\|\mathbf{x}_i - \mathbf{x}_j\right\|^2\right), \gamma > 0 \tag{2}$$

This decision is based upon a comparison of four different kernel functions (Radial Basis Function, Linear, Polynomial, Sigmoid) in a paper from Hsu et al. [15], which showed that the radial basis function kernel shall be used in almost any given scenario except for when the amount of features is much less than the amount of instances or if both amounts are very high. Seeing that neither is true in our case, we stick to the recommendation of the radial basis function kernel.

Additionally, a method is needed to make the SVM algorithm usable for more than two classes (gestures), since standard SVMs only allow the classification of two different classes. Based on the aforementioned two comparisons (II-C), we use the *One-against-One* method as follows:

Assuming we have $k = 5$ different gestures, we train $\frac{k(k-1)}{2} = 10$ different SVMs, where each one shall differentiate between a pair of gestures. During the training process, each SVM is only trained using the data from either of its classes.

As recommended by Hsu et al [15] the SVM hyperparameters C and γ will be finally optimized with a hyperparameter optimization using the grid search algorithm with the discrete values:

$$\begin{aligned} C &= 2^{-5}, 2^{-3}, \ldots, 2^{15} \\ \gamma &= 2^{-15}, 2^{-13}, \ldots, 2^{3} \end{aligned} \tag{3}$$

For the implementation of the SVM we used the open source machine learning library *LIBSVM* [16].

IV. EXPERIMENTS

To reliably assess our system's performance across multiple users, we recorded gesture data for 11 people between 20 and 54 years old, under the conditions already mentioned before. Each person was told to conduct each of the 5 gestures 10 times, resulting in a total of 550 data samples, 50 for each person and 110 for each gesture class.

Before conducting the hyperparameter optimization and training our model, we removed one randomly chosen participant's data from the data set to form a test set. The remaining data in the data set then form the training set of our model. Therefore, the performance on the test set indicates how the system performs for a new user, i.e., a user whose data hasn't been seen by the system during training.

Fig. 4 shows the classification accuracy of the model for different values of C and γ and during hyperparameter optimization. Each value represents the mean accuracy for one 10-fold cross-validation. Other measures such as the F-score are omitted here as we have an even class distribution and prioritize each class equally.

It can be seen that a large number of different hyperparameter combinations allow for classification accuracy of 100%, usually suggesting potential overfitting to the training set. If this were to be the case, we would expect that models

trained with these hyperparameters on the training set perform significantly worse on the test set. We could not, however, observe this behavior on our test set. Whereas models trained with hyperparameters netting below 100% accuracy during cross-validation would perform very erratically on the test set, the 42 models with 100% accuracy during cross-validation also netted 100% accuracy on the test set.

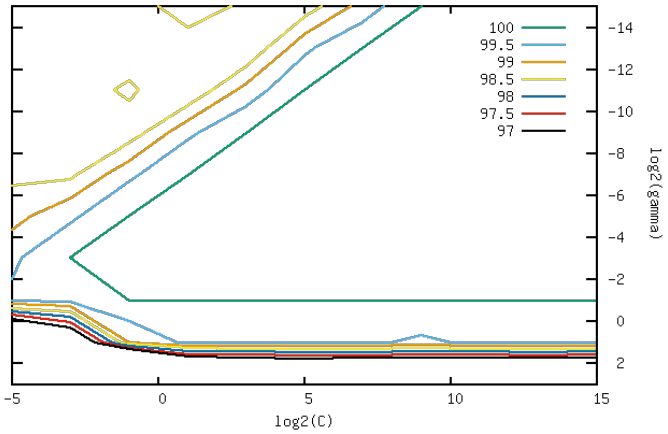

Fig. 4. Cross validation accuracy during hyperparameter optimization for different values of C and γ.

To examine whether these results would hold up in practice, we conducted another experiment during which 9 people, 4 of which had not previously participated when collecting training data, were told to freely play 20 games of rock paper scissor each while wearing the data glove. After each round, the participants wrote down which of the 5 gestures they had intended to do. We then compared their intended gestures to those predicted by our model which had 100% accuracy during cross-validation and on the test set.

For the 5 people who had already participated during the collection of training data, 98 out of 100 gestures (98%) were correctly classified, with *well* being wrongly classified as *stone* twice. For the 4 people who had not previously participated, 70 out of 80 gestures (87.5%) were correctly classified, with the *match* gesture being classified as another gesture 6 out of 20 times.

V. Discussion

Whereas our initial experiment suggested that a 100% classification accuracy would be feasible even for people with no previously by the model seen data, the follow-up experiment confirmed our suspicions that this would not hold true in practice. We suspect that this is due to at least two different factors.

First, we expect the recorded data to differ on a per-user basis, depending on both the user's physique (height, muscularity, flexibility) and how they perform each individual gesture (speed, orientation, extent). The model will only generalize well to new people, i.e., people for which no training data exists, if the training data includes data from people with similar characteristics.

Second, we expect the recorded data to differ on a case-by-case basis since users are likely to perform the same gesture slightly differently based on their current situation. For example, a gesture in rock paper scissor may be performed differently if the user decides to change the gesture at the last moment. This may also play a large role in other use cases such as fitness where exhaustion can vastly affect how an exercise is performed.

Overall, our results match related work in that SVMs appear to be well suited to classifying hand gestures recorded with data gloves. Directly comparing our model's performance to models in related work, however, seems disingenuous to us since we could not confirm our initial experiment's 100% accuracy in a practical setting. In the future, we would like to further evaluate the applicability of SVMs in hand gesture recognition for larger and more diverse sets of users and for more complex problems such as sign language detection.

VI. Conclusion

In this paper, we have shown the applicability of SVMs to data glove based hand gesture detection with 5 different gestures. In a practical scenario, we were able to achieve 98% accuracy for whom there was data in our training set and 87.5% accuracy otherwise.

We expect future work in this area to look into how such systems can be utilized for more complex detection tasks such as detecting sign language. In such a scenario, it would also be interesting to incorporate additional sensors or more complex models to also detect the hand position as well as full movement patterns.

References

[1] M. J. Cheok, Z. Omar, and M. H. Jaward, "A review of hand gesture and sign language recognition techniques," *International Journal of Machine Learning and Cybernetics*, vol. 10, no. 1, pp. 131–153, 2019.

[2] W. Ma, J. Hu, J. Liao, Z. Fan, J. Wu, and L. Liu, "Finger gesture recognition based on 3d-accelerometer and 3d-gyroscope," in *International Conference on Knowledge Science, Engineering and Management*, pp. 406–413, Springer, 2019.

[3] P. Pławiak, T. Sośnicki, M. Niedźwiecki, Z. Tabor, and K. Rzecki, "Hand body language gesture recognition based on signals from specialized glove and machine learning algorithms," *IEEE Transactions on Industrial Informatics*, vol. 12, no. 3, pp. 1104–1113, 2016.

[4] J.-H. Kim, N. D. Thang, and T.-S. Kim, "3-d hand motion tracking and gesture recognition using a data glove," in *2009 IEEE International Symposium on Industrial Electronics*, pp. 1013–1018, IEEE, 2009.

[5] P. M. P. de Melo, "Gesture recognition for human-robot collaborative assembly," 2018.

[6] M. Maynard, *Machine Learning: Introduction to Supervised and Unsupervised Learning Algorithms with Real-World Applications*. Advanced Data Analytics, Independently Published, 2020.

[7] V. Vapnik and A. Chervonenkis, "On a class of algorithms of learning pattern recognition," in *Automation and Remote Control*, vol. 25, pp. 937–945, 1964.

[8] Y.-T. Chen and K.-T. Tseng, "Multiple-angle hand gesture recognition by fusing svm classifiers," in *2007 IEEE International Conference on Automation Science and Engineering*, pp. 527–530, IEEE, 2007.

[9] C. Cortes and V. Vapnik, "Support-vector networks," in *Machine Learning*, vol. 20, pp. 273–297, 1995.

[10] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *COLT '92: Proceedings of the fifth annual workshop on Computational learning theory*, pp. 144–152, 1992.

[11] V. Vapnik, "Statistical learning theory."

[12] S. Knerr, L. Personnaz, and G. Dreyfus, "Single-layer learning revisited: a stepwise procedure for building and training a neural network," in *Neurocomputing*, pp. 41–50, Springer, 1990.
[13] C.-W. Hsu and C.-J. Lin, "A comparison of methods for multi-class support vector machines," in *IEEE Transactions on Neural Networks*, vol. 13, pp. 415–425, 2002.
[14] J. Milgram, M. Cheriet, and R. Sabourin, 2006.
[15] C.-W. Hsu, C.-C. Chang, and C.-J. Lin, "A practical guide to support vector classification," 2003.
[16] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011.