Predicting competition performance of age group swimmers

Miriam Anschütz
TUM Department of Informatics, Chair for Applied Software Engineering
Technical University of Munich
Munich, Germany
m.anschuetz@tum.de

***Abstract*—Age group swimmers' performance is hard to predict as their performance is highly dependent on their current training status and body development. In this paper, we propose a machine learning model used in an Android application that helps coaches to estimate the performance of their athletes. Therefore, the coaches can measure the 50m freestyle times in training and use our app to predict the corresponding 100m and 200m freestyle time. We use a linear regression network that predicts these times based on the 50m time as well as the swimmer's age and training age, i.e. how long the swimmer has been training on a competitional level. With this approach, we can predict the 100m time with an error of 1.83s and the 200m time with an error of 7.97 on average. We encounter problems due to non-professional swimmers being unpredictable with regards to their competition performance. Moreover, our dataset provides insufficient information about the swimmer and his training status. Nevertheless, we consider our approach as a valuable contribution to help coaches of non-professional junior swimmers to estimate their swimmers' performances.**

I. Introduction

In swimming competitions, every swimmer has his lane; thus, the competition is organised in heats. These heats are ordered by speed so the first heats are the slowest and the last are the fastest. To organise these heats, the coaches have to submit estimated times when registering their swimmer for the respective competition. A swimmer has a psychological benefit if he swims in a heat together with swimmers of the same speed; thus, it is essential that the estimated time is correct. For junior swimmers, however, it is hard to estimate the correct time as junior swimmers react strongly to stress and environmental influences[1] whats makes them more unreliable in competitions. Moreover, a new training input, e.g. an intense training camp, or growing up has a stronger impact for junior swimmers than for professionals thus, their performance changes more often. A possible solution for this could be to measure the times in training. For short distances this works very well, for longer distances, however, the training and competition times diverge. In this paper, we propose a mobile application that helps swim trainers to find the best registration times. The coaches can measure the 50m freestyle time in training and use the app to get the corresponding 100 and 200m freestyle times. To do so, we use a machine learning model that predicts the 100 and 200m time based on the 50m time as well as the age, gender and training age of the athlete. Besides, we investigate the relationship of different swimming distances and their times with a focus on junior swimmers. This paper is structured as follows: In section II we will explain the sports theory background and compare our work to current research. The results of this paper are explained in three separate parts. Each of the three parts explains its underlying approach as well as reports and discusses the respective results: In section III, we focus on the new dataset we generate for this paper. Section IV then explains and evaluates our machine learning approach. In section V we showcase the mobile application that uses the resulting machine learning model.

II. Related Work and background

In the following, we will explain relevant sport theory concepts that our work is based upon and compare our approach to related research.

A. Sports theory background

There are three different ways of energy provision in the muscles. The first and most efficient energy production comes from ATP that is stored in the cells. However, this only works for about 2 seconds. When this storage is depleted, the body generates ATP from glycogen that is also stored in the muscle cells. This way of energy provision is also very efficient but does not last for long, either. The long-time energy is produced by consuming oxygen (aerobic energy production) [2]. Figure 1 shows how the energy in the muscle cells is provided over time. In the first seconds, the energy comes from the ATP stored in the muscle cells, and the anaerobic energy production starts. The longer the activity lasts, the more energy comes from aerobic production. 50m sprints last for about 25s to 50s thus more than 60% of the energy comes from anaerobic energy production [3] . As this energy production is independent of oxygen and its transport, the energy provision is independent of fatigue; thus, the 50m times in training are close to those in competition. This is an important fact on which we base our assumption that we can simulate the 50m times in training. In general, a single training session can be split into multiple phases. During and shortly after the training session, the athlete is fatigued. After the training session has ended, the recovery starts. Recovery includes restoring energy stores in the muscles and reduce the lactic acid in the blood circulation. After many training

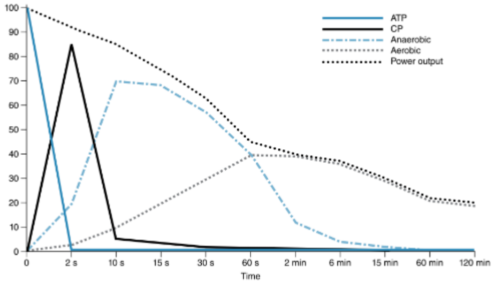


Fig. 1: Energy provision by time [2]

sessions, the athlete's body starts to adapt to the constant exertion, i.e. the energy stores in the muscles increase and the heart muscle becomes stronger. This adaption process leads to improvement of the athlete's performance [4]. Figure 2 visualises this adaption effect. The blue curve shows the performance of the athlete. With every stimulus, i.e. training session, the performance reduces to a minimum due to fatigue. After recovery, the maximum performance increases due to adaption. In this paper, we assume that athletes that have been training for a longer time perform better in competition. This assumption is based in the adaption process in the athlete's body.

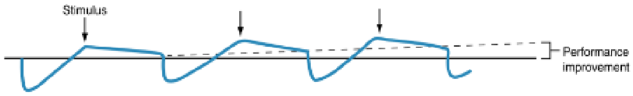


Fig. 2: Performance improvement by continuous training [5]

B. *Critical power concept*

In sports theory, critical power is the maximum rate that can be sustained for a very long time without fatigue [6, 7]. Monod and Scherrer [6] introduced this concept and showed in their study that muscle power output and time to exhaustion follows a hyperbolic relationship while the total work performed is linearly connected to the time to exhaustion. Therefore, the critical power can be used as an indicator of the athlete's endurance and anaerobic capacity. With this indicator, coaches can optimise their training by setting loads and intensity in the aerobic area and evaluating the athlete's anaerobic performance without employing expensive material [8]. As Wakayoshi et al. [9] have shown that these concepts are applicable for swimming in general. Moreover, Hill et al. [10] proved the applicability even for junior swimmers . The critical power concept is of interest for our work as it gives us an idea of how the distances and the corresponding times are related.

C. *Machine learning for swim result prediction*

To the best of our knowledge, there is no related work that uses the same approach as we propose. However, some papers use similar data representation or try to achieve the same result but based on a different setting.

Silva et al. [11] use neural nets to predict the swimmers performance in 200m individual medley and 400m freestyle based on how the swimmer performs in multiple fitness, strength and flexibility tests. The authors submitted 138 junior swimmers of national level to four test categories: kinanthropometric, general functional and specific functional tests as well as an evaluation of their swimming technique. With the results of these tests, they predict the swimmers competition performance using Artificial neural nets. Moreover, they determine the physical factors that affect this performance the most. Although the ground truth, i.e. the data the model uses for training, is different from our, the machine learning approach itself is similar to our approach: They train a multilayer perceptron with one hidden layer and mean-squared error as loss function. Edelmann-Nusser et al. [12] use this machine learning approach as well. However, they try to predict the Olympic 200m backstroke performance of only a single swimmer. Therefore, their training data contains specific knowledge about the amount and intensity of training in the weeks before the respective competition. With this approach, they were able to predict the performance with a mean-absolute error of $\pm 0.62s$.

In order to predict how junior swimmers perform when they are older,, Xie et al. [13] divide 12- and 13-year-old top swimmers from the US into four performance level: the top 25%, 25%-50%, 50%-75% and the last 25%. A swimmer is represented by a single record of `(stroke, course, age, time, powerpoint)` where the powerpoint is a score that relates the swimmer's performance to other performances in this age group. Using SVM and neural networks, they determine the level change rate that is the probability that the swimmer will (still) be in the top-level when he is 18 years old.

In contrast to the aforementioned papers, we focus on junior swimmers of any level, not only of national level. In our approach, we try to predict the 100m and 200m freestyle times. As freestyle is the most common strokes in competition, we focus on this stroke for a first evaluation.

III. Data generation

As we could not find a dataset that matches our requirements, we generate a new dataset using competition result published by the DSV (Deutscher Schwimm-Verband e.V.). All swimmers that are eight years or older and take part in competitions in Germany must register at the DSV. The DSV publishes all official competition results as well as information about registered swimmers online [1]; thus, we can access a considerable amount of data for our dataset.

Similar to the records in *Prediction on Performance of Age Group Swimming Using Machine Learning* by Xie et al. [13] our dataset contains records that look like this: `(gender, age, training age, 50m time, 100m time, 200m time)`. To get a continuous representation of our data, we express all times in seconds with two decimal places.

[1] http://www.dsv.de/schwimmen/wettkampf-national/schwimmerabfrage/, accessed 02.01.2020

We include the gender and the age of the swimmer as boys and girls develop differently at different ages. Golle et al. [14] showed in their study that the fitness performance improves when children get older. While boys outperformed girls of the same age-group in upper-extremity muscular power and endurance, girls performed better in flexibility test. All of the three aforementioned physical aspects are important for swimmers; thus, we expect different performances by swimmers of different age groups and gender. We use the swimmer's age group to determine his age instead of his actual age at the respective competition. Our dataset covers ages from nine to 14 years

The training age describes how long the swimmer has been training competitive swimming. The longer the swimmer has been training, the better is the swimmer's endurance and his technique as explained in section II-A. On top of that, he gained experience in competitions; thus, he may be less nervous during competition. In our dataset, we consider training ages ranging from one to seven years.

Wakayoshi et al. [9] have shown that for college swimmers, the distance and the corresponding time have a linear connection thus we can calculate the longer distance time based on any shorter distance. However, Golle et al. [14] observed a curvilinear improvement in upper-body strength and endurance for older junior swimmers. Putting this together, we expect the correlation between 50m, 100m, and 200m time to be non-linear, dependent on the age, training age and gender of the swimmer.

Using the aforementioned considerations, we generate two different datasets, a training and a test dataset. The training dataset consists of 116 samples with 49 female and 67 male swimmers. The test dataset contains 19 samples, ten of which are female. The swimming times in the dataset were achieved at the same competition on the same weekend in a 25m pool. Figures 3 and 4 show the correlation of 50m and the longer distance times. In order to plot the samples in 2D, we omit the information about the swimmer. We can see that the times have a strong linear correlation with a Pearson correlation coefficient of 0.983 for the 100m training data and 0.972 for the 200m training data. Therefore, our initial assumption that age is essential for the prediction was wrong. Even for junior swimmers, the times follow a linear correlation as introduced by the critical power concept in section II-B, independent of the age, the training age or the gender.

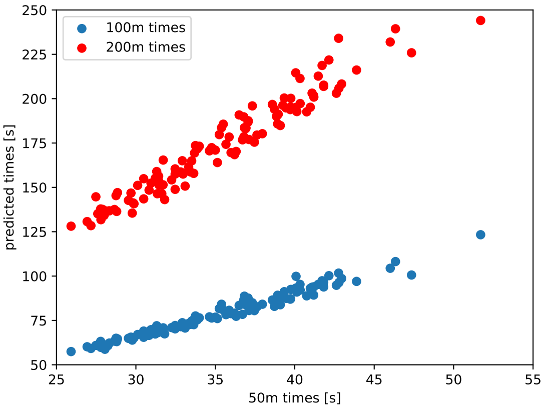


Fig. 3: Training data: correlation of 50m and longer distance times

Also, we observe a higher variance in the long-distance times for higher 50m times. As we can see in Figure 5, these higher 50m times are mostly from swimmers that are in their first or second training year. These swimmers have less competition experience or a rather bad technique. Moreover, young swimmers are often very nervous at competitions; thus, they make mistakes more likely. This makes it harder to predict their performance. The better a swimmer is, i.e. the longer he trains and the better his technique is, the more reliable his performance will be.

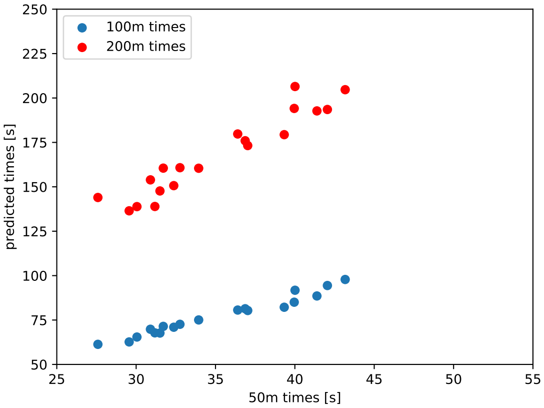


Fig. 4: Test data: correlation of 50m and longer distance times

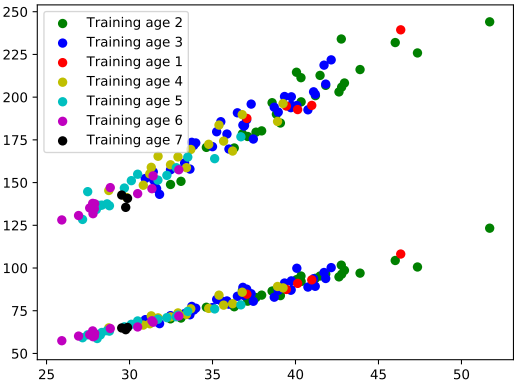


Fig. 5: Training data: influence of training age

Model	Training data	Test data	Error
■ original	swimmer info, 50m	swimmer info, 50m	7.97
•	swimmer info, 50m, true 100m	swimmer info, 50m, true 100m	5.77
●	swimmer info, 50m, true 100m	swimmer info, 50m, predicted 100m	8.30
▲	swimmer info, 50m, predicted 100m	swimmer info, 50m, predicted 100m	7.67

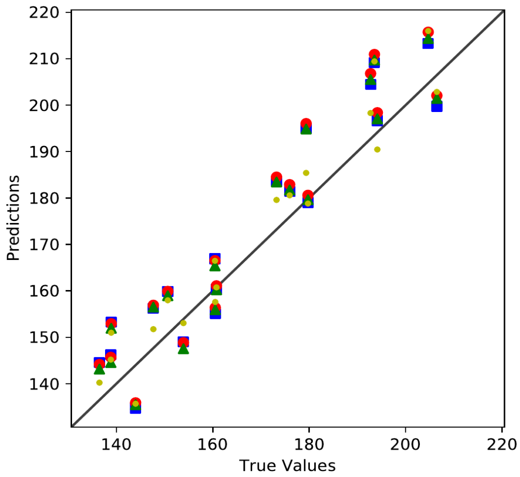


Fig. 6: Comparison of different 200m approaches

IV. Machine learning approach

Using these datasets, we train machine learning models to predict the longer distance times. We use Tensorflow[2] for training our machine learning models. Tensorflow offers a well-documented and flexible ecosystem as well a the tensorflow-lite package that makes it possible to use trained models on mobile devices.

A. Model selection

We will train two separate models, one for the 100m and one for the 200m prediction. Our models must return continuous output values; thus, we need to solve a regression instead of a classification task. Therefore, we use the regression model explained in the Tensorflow tutorials [3]. As we have seen in the previous section, our data is linearly separable thus our model does not need a hidden layer to fit the data. For the training itself, we randomly split our training data into a training and validation subset with a fraction of $\frac{1}{3}$ and use the validation set to evaluate the current model. As our output must be continuous, we can only use activation functions that do not saturate, e.g. linear function or ReLU. The linear function can return negative values, while ReLU can only return values greater or equal to zero. As the predicted times are greater than zero, it makes no difference which of these activation functions we use. Besides, as per common knowledge [15], we use mean-squared error as loss function.

We evaluate the final models with a separate test dataset and mean absolute error as metric.

In order to find the best model, we vary two parameters: The number of neurons in the Dense input layer and the optimiser used to fit our data. Therefore, we train models using $\{16, 32, 64\}$ neurons and try different optimisers offered by the keras library[4], i.e. Adam or RMSprop. For both optimisers, we use their default learning rates of 0.001 The models are trained using the training data and tested against the validation data. The results are presented in Table I.

		Units		
		16	32	64
rmsprop	100m	1.94	1.90	1.97
	200m	5.35	5.49	5.92
adam	100m	4.88	5.28	2.48
	200m	10.40	8.55	8.47

Table I: Mean absolute error for different units in input layer

We can see that RMSprop is the best optimiser; thus, we use RMSprop with 32 units for the 100m prediction and RMSprop with 16 units for the 200m prediction. We train the model in 500 epochs and splting the data into a batch size of 32.

B. Model evaluation

We evaluate the resulting models using the test dataset. Figure 7 shows the scatter of the predicted and the correct 100m times as well as the predicted and correct 200m times. The black line in each plot represents the optimum, where predicted and correct times are identical thus the closer the dots are to this line, the better is the model. The models achieve a mean-absolute error of 1.83 for the 100m model and

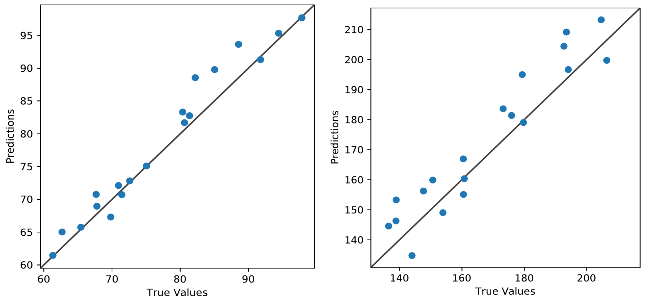


Fig. 7: Evaluation of 100m (left) and 200m (right) model

[2] https://www.tensorflow.org/ accessed 03.01.2020

[3] https://www.tensorflow.org/tutorials/keras/regression, accessed 02.01.2020

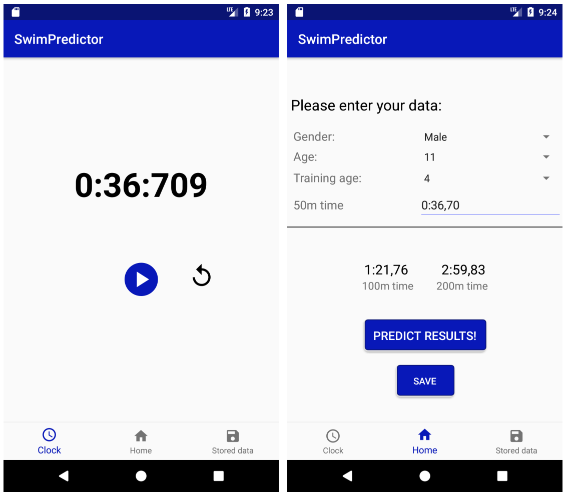
[4] https://keras.io/optimizers/, accessed 03.01.2020

7.97 for the 200m model. We consider the 100m prediction as successful thus we use this model in our application.
The 200m prediction, however, has deficits. A potential improvement could be to include the 100m times in the 200m prediction. In order to evaluate this idea, we train two additional models. For the first model, we add the correct 100m times to the training data and for the second model, we use the 100m model to predict the 100m times and add these times to the training data. Figure 6 evaluates the four resulting models. The small yellow circles use the correct 100m times for training and prediction. It is important to mention that this model is only for theoretical analysis and does not apply to our real-world problem as we do not know the correct 100m times when we predict new samples. However, this model shows that using the 100m times for the 200m prediction can indeed improve the results. The models using the predicted 100m times for the 200m prediction do not perform better than the original model. Moreover, we can see in the plot that these models almost ignore the 100m time as their predicted times are nearly identical. A reason for this is that the 100m prediction model is not good enough to be helpful for the 200m prediction. Finally, using the 100m times for the 200m model is a good idea but does not work in practice; thus, we use the original 200m model in our application.

V. Android Application

We implement the Android application ”SwimPredictor” using Java, a minimum SDK version of 22 and a target SDK version of 28. The app implements three functionalities: The usage of the trained models, a stopwatch to measure the times and a database to store the predicted results and swimmer information to access it later when he makes the competition registration. All three functionalities are implemented as fragments and can be accessed from a bottom tab navigation whats makes changing from one functionality to another very easy and intuitive. Figure 8 shows screenshots of the final Android application.

In the core or home fragment the coach can enter the information of the swimmer and the 50m time he measured in training and let the app predict the longer distance times based on this information. It loads the pre-trained tensorflow-lite model it needs for the predictions from the app's assets folder. To use these models, we need to include the tensorflow-lite library in our app's `build.gradle` file. The coach can enter the information about the swimmer by choosing values from a dropdown list. This prevents the coach from entering unknown values or values in the wrong format. Moreover, the core fragment has a predict button, with which the user can trigger the prediction, two text views to display the predicted times and a store button to store the current record.



0:36:709

Please enter your data:

Gender: Male
Age: 11
Training age: 4
50m time 0:36,70

1:21,76 100m time
2:59,83 200m time

Fig. 8: Screenshots of the app's stopwatch and home view

The stopwatch function implemented in the `org.swimpredictor.ui.clock` package has two buttons, a play/pause button and a reset button, as well as a text to display the measured time. We explain the stopwatch logic in Figure 9. At first the stopwatch is set to zero. When the play button is clicked, the time starts to run and the play icon on the play/pause button changes to a pause icon. When clicking the pause button, the stopwatch stops running, displays the current time and the button shows the play icon again. The time is saved into a shared preference object and loaded into the 50m time field of the prediction view thus the user does not need to transfer the time himself. The stopwatch can be reset at any time. If the reset button is clicked, the displayed time and the time stored in the shared preferences are set to zero again. We simulate the running stopwatch using a java *Runnable* object that displays the difference between the start time and the current system time [5].

We implement the store functionality in the `org.swimprodictor.database` package using Android's Room [6] and a SQLite database behind it. Although we only have one table, we consider a database the best way to store our data as it is rather easy to implement with Room and it assures consistency by built-in constraints. A `DataSample` object represents an entity in this table and contains all the information from the data records, as explained in section III, together with a unique description that the coach enters when saving a record and human-readable String representation of the times. These Strings have the format `m:ss,dscs`, e.g. `1:23,45`. The database itself offers the functionality to insert and delete a sample. The database tab displays all samples stored in the database.

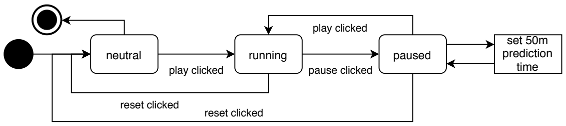
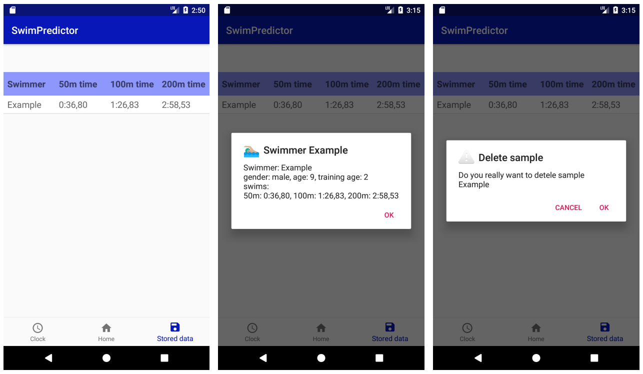


Fig. 9: Activity diagram for stopwatch function

[5] https://www.android-examples.com/android-create-stopwatch-example-\ tutorial-in-android-studio/, accessed 31.12.2019

[6] https://developer.android.com/training/data-storage/room, accessed 12.01.2020

We had problems to display the entries in the database as the number of entries in the database changes thus a static TableView is not applicable. Therefore, we use the Open-Source version of the SortableTableView[7]. Due to the small display on a mobile device, we were not able to show all information of all samples at the same time. Therefore, the table only shows the description and the three times. The remaining information about the swimmer can be shown in a pop- up message when clicking at the respective sample. To delete a sample, the user needs to long click on the sample he wants to delete. The app then requires a confirmation of the user. If he confirms the deletion, the sample is removed from the database, and the table is reloaded. Figure 10 shows the TableView of the database entries as well as the pop-up messages with detailed information and deletion confirmation.



Swimmer	50m time	100m time	200m time
Example	0:36,80	1:26,83	2:58,53

Swimmer Example

Swimmer: Example
gender: male, age: 9, training age: 2
swims:
50m: 0:36,80, 100m: 1:26,83, 200m: 2:58,53

OK

Delete sample

Do you really want to detele sample
Example

CANCEL OK

Fig. 10: Screenshots of the app's database functionality

VI. Future Work & Conclusion

In this paper, we introduced an appealing application that helps coaches to estimate their swimmers' performances. However, the underlying machine learning model does not achieve the results we hoped. One reason for this is that we focus on non-professional swimmers; thus, their performance is less constant and therefore, less predictable. Nevertheless, the coaches of these swimmers need the most assistance; thus, a reliable prediction model is of high interest. A possible improvement could be to include more data, especially for the slower 50m times to find a correlation in our data, or use data augmentation to generate more samples from the existing data.

Another problem is that the swimmer's performance changes within a season. In our approach, we use an age representation based on the age group of the swimmer instead of his actual age. Therefore, it would be better to include the swimmer's age in month and, more importantly, the information at which point in the season the competition took place.

As the related work suggests, the swimmer's anthropometry is of high importance for his performance. Therefore, it would be good to include the height or flexibility in our records. We are currently working on a different approach that includes information about the training status of the swimmer. Therefore, we collect data about the four weeks before the respective competition. This data includes the total amount of training in kilometres as well as the intensity of the training sessions, i.e. the amount of training in the aerobic area or the number of technique sessions. The first results are already promising. However, for this approach, it is indispensable to work together with a group of swimmers as the needed data is not publicly available.

To get a first evaluation of our approach, we had to make some constraints. If we find a good data representation to predict results, we can enlarge our predictions to different strokes and distances, as well as include times swam in a 50m pool.

[7] https://github.com/ISchwarz23/SortableTableView, accessed 12.01.2020

References

[1] P. J. McCarthy, M. S. Allen, and M. V. Jones, "Emotions, cognitive interference, and concentration disruption in youth sport," *Journal of Sports Sciences*, vol. 31, no. 5, pp. 505–515, 2013.

[2] T. O. Bompa and C. Buzzichelli, *Periodization: theory and methodology of training*, 6th ed. Human kinetics, 2018, p. 20ff.

[3] Dr. Patrizia Mayer, Haral Ochwat, weitere BSV Dozenten, *Trainerhandbuch: Breiten-, Freizeit- und Gesundheitssport, Leistungssport Schwimmen*, 2nd ed. Munich: Bayerischer Schwimmverband e.V. pages = 224f, 2007.

[4] T. O. Bompa and C. Buzzichelli, *Periodization: theory and methodology of training*, 6th ed. Human kinetics, 2018, p. 8ff.

[5] ——, *Periodization: theory and methodology of training*, 6th ed. Human kinetics, 2018, p. 18.

[6] H. Monod and J. Scherrer, "The work capacity of a synergic muscular group," *Ergonomics*, vol. 8, no. 3, pp. 329–338, 1965.

[7] D. W. Hill, "The critical power concept," *Sports medicine*, vol. 16, no. 4, pp. 237–254, 1993.

[8] J. Dekerle, M. Sidney, J. M. Hespel, and P. Pelayo, "Validity and reliability of critical speed, critical stroke rate, and anaerobic capacity in relation to front crawl swimming performances," *International journal of sports medicine*, vol. 23, no. 02, pp. 93–98, 2002.

[9] K. Wakayoshi, K. Ikuta, T. Yoshida, M. Udo, T. Moritani, Y. Mutoh, and M. Miyashita, "Determination and validity of critical velocity as an index of swimming performance in the competitive swimmer," *European journal of applied physiology and occupational physiology*, vol. 64, no. 2, pp. 153–157, 1992.

[10] D. W. Hill, R. P. Steward, and C. J. Lane, "Application of the critical power concept to young swimmers," *Pediatric Exercise Science*, vol. 7, no. 3, pp. 281–293, 1995.

[11] A. J. Silva, A. M. Costa, P. M. Oliveira, V. M. Reis, J. Saavedra, J. Perl, A. Rouboa, and D. A. Marinho, "The use of neural network technology to model swimming performance," *Journal of sports science & medicine*, vol. 6, no. 1, p. 117, 2007.

[12] J. Edelmann-Nusser, A. Hohmann, and B. Henneberg, "Modeling and prediction of competitive performance in swimming upon neural networks," *European Journal of Sport Science*, vol. 2, no. 2, pp. 1–10, 2002.

[13] J. Xie, J. Xu, C. Nie, and Q. Nie, "Prediction on performance of age group swimming using machine learning," in *International Conference on High Performance Computing and Applications*, 2015, pp. 178–184.

[14] K. Golle, T. Muehlbauer, D. Wick, and U. Granacher, "Physical fitness percentiles of german children aged 9–12 years: findings from a longitudinal study," *PLoS One*, vol. 10, no. 11, p. e0142393, 2015.

[15] R. Reed and R. J. MarksII, *Neural smithing: supervised learning in feedforward artificial neural networks*. Mit Press, 1999, p. 155.