**Monitoring of body posture during computer use**

Jana Kovaličová

Supervisor: RNDr. Juraj Šebej, PhD.

Consultant: RNDr. Martin Lukáň, PhD.

# Introduction

We live in a time where more and more jobs require either full or partial computer work. An intensive computer work often means that we stop focusing on proper posture and forget about taking regular breaks. Many people are starting to experience various medical conditions, which are directly related to the sedentary type of work. The most frequent problems caused by the slouching and leaning for long hours using computers are back problems, headaches, visual impairment, or carpal tunnel syndrome.

In recent years, quite a few sitting posture monitoring systems have been developed to prevent back pain related to the bad sitting posture. In this work, we will focus on proper sitting and posture when working with a computer and will review some available systems for posture monitoring. We will have a look at different technologies and methods used, to help us design our own posture monitoring system. We will then describe a process of designing a network of IoT sensors that will be attached to the back of the chair and will be used to monitor the posture of the computer user. The data we collect will then be processed and annotated so that it can be used for the artificial intelligence model that we will use in this work.

# Existing posture monitoring systems

In this section we describe the different existing systems for posture monitoring. We will have a closer look at the technologies used for each system because initially we must decide what sensors would be suitable for our system. Our other focus will be on the machine learning models used in a different works.

## Vision cameras

The basic and easiest way to monitor people is to use vision cameras. Authors in [1] used specially installed webcam, which was positioned at a 90 degrees on the side, where the worker held the mouse while using the computer. The picture of a correct posture was taken. The camera was used to photograph workers every 20–25 minutes. The system randomly selected a time from this interval. The workers were then showed a pop-up window on their computer showing the photo of the ideal position and the current position so they can adjust their posture accordingly. No machine learning model was used in this paper, as workers received the information about their posture only from the picture. Authors used Rapid Upper Limb Assessment (RULA) measuring tool, which is used to quantify the grade of musculoskeletal risk of the sitting posture. The RULA grade is calculated based on the degree of angles between various body parts and recommended postures. The semi-automated software was built to access a database and calculate the RULA score from the photos. This model using a vision webcam improved the sitting posture of the workers, however, the vision-based systems can face privacy problems.

Another approach using only simple depth camera placed on a desk was proposed by Kulikajevas et al [2]. Authors did not use a single photo but a time sequence with four frames and decided on deep convolutional recurrent neural network based on *MobileNetV2* as their machine learning model (shown on Figure 1). There were 11 subjects (seven men and four women) to perform the posture emulation tasks. Because the camera was used in this study, the informed consent was obtained, while they followed strictly the requirements. To collect the data, they filmed each subject for thirty seconds in each position deliberately to label the data, and then they let subject get into their usual bad position. The data was then labelled manually by an expert; however, the labeling of such data is a challenge due to its subjectiveness as bad data labels may poison the network and cause it to overfit instead of generalizing. They divided different postures into eight positions. Sitting straight was a correct position, there were three forward postures – lightly hunched, hunched over and extremely hunched, and two backward postures – partially lying and lying down. The model firstly classified sitting postured into eight categories with accuracy of 68.33 %. All largest misclassification values occur between neighboring classes (extremely hunched vs hunched over—49.5%), (hunched over vs extremely hunched—40.66%), (partially lying vs lying down—28.15%), (lying down vs partially lying—19.47%), suggesting that perhaps the need for some fuzzification of class definitions and interpretation of results, or that these posture classes should be combined. After additional analysis they changed the classification and created three categories with the accuracy of 91.47 %. Their network is accurate enough that it can suggest the labels in further labeling processing. This would change their solution from being supervised machine learning into semi-supervised or even completely unsupervised machine learning approach.

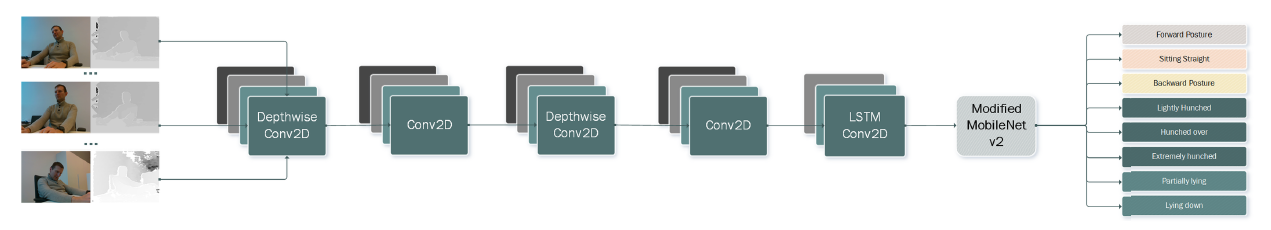


Figure 1 The recurrent hierarchical ANN architecture using MobileNetV2 as the main backbone

## Pressure sensors

There are different approaches of how to use pressure sensors when building a posture monitoring system. Jongryun et al [3] in their work used four pressure sensors, that were placed on the seat plate of the chair. They replaced the seat frame of an existing office chair with a new frame with the load cell inserted. The location of each load cell was marked as the left (S1) and right (S2) sides of the thigh position, as well as the left (S3) and right (S4) sides of the buttock position. All the load cells were placed at 70 mm from each corner of the seat plate. The real-time data on the load (kg) measured on the four load cells were transferred to a personal computer via the Arduino board. Two webcams were used to confirm that the sitting posture was correct. This study involved 24 healthy adult males (age: 27.6 ± 5.6 years, height: 174.5 ± 6.2 cm, and body weight: 71.9 ± 8.7 kg). The selected subjects regularly worked with a video display terminal, sitting on office chairs for eight or more hours a day, and had no apparent severe musculoskeletal deformity or nervous system abnormality.

Authors tried to classify six seating positions shown in Figure 2. The experiment was divided into a preliminary test (15 subjects) and the main test (nine subjects). Subjects in preliminary test were used to train the model and help define the body weight ratio range for sitting posture. Each subject held every position for 30 seconds, and this was repeated five times. Another nine subjects participated in the main test for classification using machine learning. In this test, each of the six positions were changed randomly every ten seconds and the subject was shown on the screen the picture of the actual position. There were 30 changes of positions, and it was repeated three times so total of 90 positions performed by each subject.

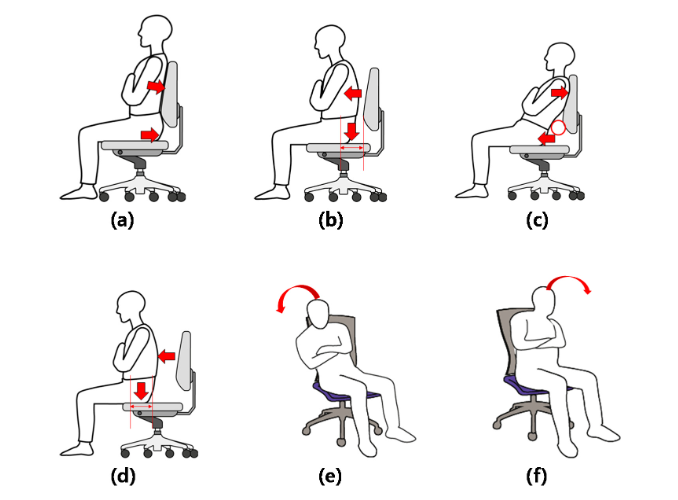


Figure 2 Types of sitting postures : (a) upright sitting with backrest; (b) upright sitting without backrest; (c) front sitting with backrest ; (d) front sitting without backrest ; (e) left sitting ; and (f) right sitting

After data was collected, the authors made some calculations and used seven different machine learning methods for the posture classification - support vector machine using the radial basis function kernel; support vector machine using the linear kernel; linear discriminant analysis; quadratic discriminant analysis; Naïve Bayes classifier; random forest classifier; and decision Tree. The model with the best results was SVM with RBF kernel with an accuracy of 97.20 %, which is a very good result considering they used only four pressure sensors.

Authors in [4] created a system of six force sensitive resistors placed on the chair seat, in a way that they covered most of the low part of the body in contact with the seat for all participants. A soft seat pillow was placed over the sensors for a homogeneous distribution of weight and as a protector of the sensors and wiring. As the microcontroller unit they used NUCLEO-F411RE STM32 Nucleo-64 development board and a proper firmware was developed for gathering, handling and then sending this data via Bluetooth to a PC host. They designed a user-friendly application for displaying measured data using JavaScript and node.js and vue.js frameworks, as well as a heat map graphic representation library to visualize the changes in the sensors in real time.

In this study, one correct and six incorrect positions were defined according to similar articles and the most common sitting habits shown in Figure 3. The participants were not provided illustrations of the postures to be carried out, but rather the postures were described with the phrases indicated in the table.

Table, timeline

Description automatically generated

Figure 3 Seven sitting positions [4]

The postures were performed by 12 volunteers. The participants were 8 males and 4 females aged between 19 and 50, with heights between 1.58 and 1.92 m and weights between 45 and 117 kg. They were informed about the postures to be carried out, and records were acquired without prior training. For the registration of all the positions considered, a chair with a back and armrests was used. They asked each participant to perform two repetitions for each pose for approximately 15 seconds, so that when a repetition of each position considered had been recorded, a second iteration was carried out to collect the second repetitions. During the recording of each posture, the system collected the values sensed by the device. Each sample obtained corresponds to the values of the sensors. Approximately 120 samples were obtained for each repetition, which means a sampling frequency of about 8 Hz. Then used neural network architecture for classifying seven different seating position with an accuracy of 81 %. The effectiveness of the classification system was measured using different and well-known metrics: accuracy, sensitivity (also known as recall), specificity, precision, and F1-score. The problem was that their system could not distinguish between the correct upright position and the one with the back not resting on the backrest.

A similar system was designed by Matuska et al [5] who used 6 force-resisting sensors, four of them were place on the seat and two of them at the backrest of the chair. Their primary goal was to design a system, which could be easy to implement in any office space where the person does not have to use the same chair every day. The overall system consists of a variable number of chairs, the cloud server, and client stations, smartphones. Each chair has an electronic device based on the Arduino microcontroller, external battery power source, and six flexible force sensors. In this experiment, the person chooses a free chair in the office and sits down. The Arduino hardware will wake up from sleep at this point and connect to the cloud. The person turns on the mobile application and logs in to the chair. Each smart chair has an identification number to login. The information about the sitting posture with additional data is displayed in the smartphone application. After finishing the work, the person logs out from the chair. Finally, you can view the daily report.

Authors defined nine sitting postures, first one was a correct sitting posture, next four represented bad sitting posture, and last four postures were heavy load for the backbone (shown on Figure 4). Twelve test subjects participated in this experiment. For each subject, they recorded 10 measurements for each posture. Then they calculated the average standard deviation for each pose. The next step was the average calculation for each posture. Afterward, they determined the 3 threshold values based on the collected data. The user then receives the notifications about the sitting posture correctness on the mobile application on Android. No machine learning model was used in this work, authors just tried to create a simple rule based on calculating the standard deviation, to distinguish between different posture categories.

A collage of a person sitting in a chair

Description automatically generated with low confidence

Figure 4 Nine sitting positions [5]

Another approach of using pressure sensors were proposed in [6] where authors placed pressure sensors under each leg of a normal office chair shown in Figure 3. The experiment had two phases. Firstly, they tried to demonstrate the ability of their system to recognize a set of predefined postures and simple actions in controlled lab experiments. The subject is seated on the chair in front of a desk with a computer and asked to perform the activity displayed on the screen, in total 12 postures and actions × 20 times each. The postures are sitting straight, leaning forward / backward / left / right, sitting with one leg cross the other knee, and sitting with one hand raised in the air. The actions are, nodding, clapping hands, typing on the keyboard, moving, and clicking the mouse. Each activity lasts for 15 seconds, followed by a small pause, where the subject returns to the default posture (sitting straight). The activities’ order is randomized in each round. Overall, 5 healthy subjects (1 female, 4 males, aged 23-34 years) participated in data recording. Classification was performed using 10-fold cross-validation method with an LDA classifier. The average accuracy with subject dependent training is 0.826.

Then they investigated the usefulness of their concept for the discrimination of seven more complex, high-level activities in real life data streams: working on PC, eating, playing video games, watching movie, talking with others, browsing the Internet and vacant seat. They recorded data from 4 healthy subjects (1 female, 3 males, aged 24–34 years, the same people as in the former experiment except subject 5) during at least 8 hours of normal day in the lab. Thus, the subjects have been asked to sit on the equipped chair and perform their normal work routine for at least 8 hours × 3 days. They were also asked to play games and watch movies. For privacy and practicability issues the experiment was not video recorded and not precisely labeled. Instead, the subjects kept a log of their major activities throughout the day in an experience sampling like approach. The same classifier was used and The accuracy is 0.783 for the user dependent case.

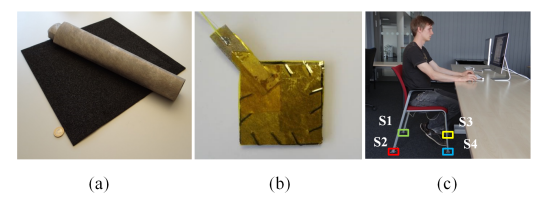


Figure 5 System Design; a) raw materials for the pressure sensor; b) a sensor prototype; c) a chair equipped with sensors

## Inertial sensors

The inertial sensors provide information about the tilting angle relative to the gravity and linear acceleration. They can be easily integrated into people clothes due to their small size and high portability. However, the system measurements accuracy is sensitive to the sensors position and some people might find those systems invasive as the sensing devices must be stuck to the human body.

Authors in [7] developed a portable trunk posture analysis system to monitor a daily posture habits of a user. They used three inertial triaxial accelerometer sensors attached onto the back of the subjects at the upper trunk(T1/T2), middle trunk (T12) and pelvic (S1) levels. These devices monitor the spinal curvature during the trunk movement. The parameters of the trunk posture measurement were calculated based on several axis systems.

A Smart Rehabilitation Garment (SRG) for posture monitoring was introduced in [8] consisting of two inertial measurement sensors embedded in the person clothes. Sensors were placed at C7-T1 spinal segment and at T4 and T5 vertebrae. Evaluation of the garment was dome by measuring the thoracic angle. The purpose of this garment was to prevent the spinal pain and avoid compensatory movements during arm-hand training. The posture feedback was directly provided on the jacket using vibration motors and graphically on a connected Android smart phone app.

Azin Fathi and Kevin Curran [9] designed an intelligent wearable system for the displacement of the spine detection. The system continuously monitors a user’s posture using three sensors. Sensor 1 was placed on Cervical spine; second sensor was placed in between all Thoracic spine and sensor 3 was placed on lower Lumbar spine. Authors analyzed collected data using the SAX (Symbolic Aggregate approXimation) method for posture classification to identify the hunched and slouching back posture. A user interface was created to send feedback signals when an incorrect posture is detected.

# Machine learning models

The goal of this work is to create some machine learning model to classify correct and incorrect sitting posture, alternatively make a classification into more sitting positions. Machine learning [10] is a subset of artificial intelligence which allows a machine to automatically learn from past data without explicitly programming an algorithm. The goal of artificial intelligence is to create computers which would be able to simulate human cognitive functions and behavior and solve complex problems. Machine learning has dramatically progressed over last few years, and it has been used in different fields like science, technology, even in healthcare, education, manufacturing, or financial modeling. It helped develop different practical software for computer vision, speech recognition, natural language processing, robot control, and other applications. We can see machine learning software all around us without even realizing it.

Machine learning can also be described as the scientific study of algorithms and statistical models that computer systems use to perform a specific task without being explicitly programmed [11]. That is why decided to take advantage of this discipline. Figure 10 illustrates different algorithms possibility that machine learning can provide. We would like to point out that there is not one universal algorithm that is suitable for all the problems. It always depends on data and a problem we want to solve.

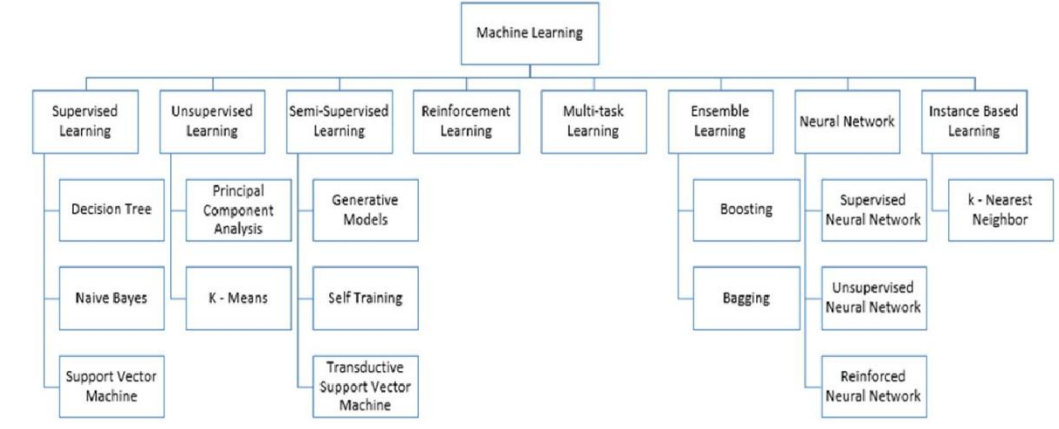


Figure 6 Machine learning algorithms

## Supervised learning

Supervised learning algorithms need an external assistance, and it is based on predicting labels of new data learnt on a set of training data. Input dataset is divided into training and testing samples. Firstly, the algorithm is trained on training dataset, where it learns some pattern, which is then applied to testing samples to predict correct labels. The workflow of supervised learning is shown on Figure 11.

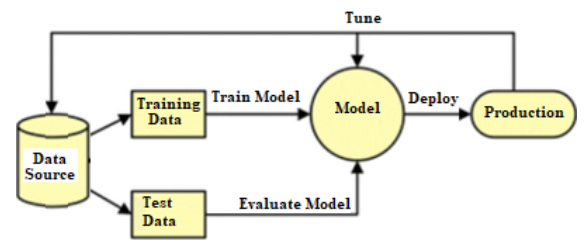


Figure 7 Supervised learning workflow

The most known supervised learning algorithms are Decision tree, Naïve Bayes, and Support Vector Machine.

### Decision tree

Decision tree is classification algorithm which is represented in form of a tree graph. The nodes in the graph represent an event or choice and the edges of the graph represent the decision rules or conditions. Each node represents attributes in a group that is to be classified and each branch represents a value that the node can take.

### Naïve Bayes

It is a classification technique based on Bayes Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. Naïve Bayes mainly targets the text classification industry. It is mainly used for clustering and classification purpose depends on the conditional probability of happening.

### Support vector machine

In machine learning, support-vector machines are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis.

It basically, draw margins between the classes. The margins are drawn in such a way that the distance between the margin and the classes is maximum and hence, minimizing the classification error, shown on Figure 8.

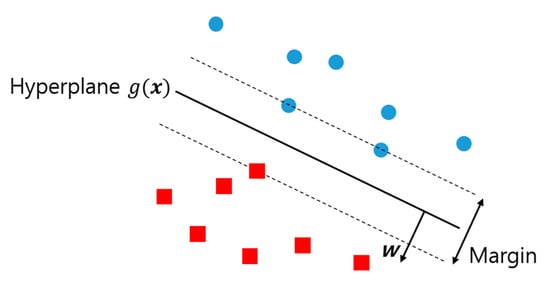


Figure 8 Support vector machine algorithm

## Unsupervised learning

The unsupervised learning means that there are no correct answers and there is no teacher. Algorithms are left on their own to find a data classification or discover some interesting structure in the data. The unsupervised learning algorithms learn few features from the data. When new data is introduced, it uses the previously learned features to recognize the class of the data. It is mainly used for clustering and feature reduction.

### Principal component analysis

PCA is a statistical procedure that allows you to summarize the information content in large data tables by means of a smaller set of “summary indices” that can be more easily visualized and analyzed. The most important use of PCA is to represent a multivariate data table as smaller set of variables (summary indices) to observe trends, jumps, clusters and outliers. This overview may uncover the relationships between observations and variables, and among the variables.

### K-Means clustering

K-means is one of the simplest unsupervised learning algorithms used for solving clustering problems. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters. The main idea is to define k centers, one for each cluster. A cluster refers to a collection of data points aggregated together because of certain similarities. The algorithm then allocates every data point to its nearest cluster.

## Semi-Supervised learning

Semi-supervised machine learning is a combination of supervised and unsupervised machine learning methods. It combines a small amount of labeled data with a large amount of unlabeled data during training. With more common supervised machine learning methods, you train a machine learning algorithm on a “labeled” dataset in which each record includes the outcome information.

### Generative models

A Generative model is the one that can generate data. It models both the features and the class (i.e., the complete data). If we model P (x, y): I can use this probability distribution to generate data points - and hence all algorithms modeling P (x, y) are generative. One labeled example per component is enough to confirm the mixture distribution.

### Self-training

In self-training, a classifier is trained with a portion of labeled data. The classifier is then fed with unlabeled data. The unlabeled points and the predicted labels are added together in the training set. This procedure is then repeated further. Since the classifier is learning itself, hence the name self-training.

### Transductive support vector machines

Transductive support vector machines algorithm (TSVM) has been widely used as a means of treating partially labeled data in semi-supervised learning. It is used to label the unlabeled data in such a way that the margin is maximum between the labeled and unlabeled data.

## Reinforcement learning

Reinforcement learning is an area of machine learning concerned with how software agents ought to take actions in an environment to maximize some notion of cumulative reward. It is a technique that enables an agent to learn in an interactive environment by trial and error using feedback from its own actions and experiences.

## Multi-task learning

Multi-Task learning is a sub-field of machine learning that aims to solve multiple different tasks at the same time, by taking advantage of the similarities between different tasks. Rather than training independent models for each task, we allow a single model to learn to complete all the tasks at once. Formally, if there are *n* tasks, where these *n* tasks or a subset of them are related to each other but not identical, Multi-Task Learning (MTL) will help in improving the learning of a particular model by using the knowledge contained in all the *n* tasks.

## Ensemble learning

Ensemble learning is the process by which multiple models, such as classifiers or experts, are strategically generated and combined to solve a particular computational intelligence problem. Ensemble learning is primarily used to improve the performance of a model or reduce the likelihood of an unfortunate selection of a poor one. Other applications of ensemble learning include assigning a confidence to the decision made by the model, selecting optimal features, data fusion, incremental learning, nonstationary learning, and error-correcting.

### Boosting

The term „Boosting‟ refers to a family of algorithms which converts weak learner to strong learners. Boosting is a technique in ensemble learning which is used to decrease bias, not variance. In boosting, a random sample of data is selected, fitted with a model, and then trained sequentially—that is, each model tries to compensate for the weaknesses of its predecessor. With each iteration, the weak rules from each individual classifier are combined to form one, strong prediction rule.

### Bagging

Bagging is used when the goal is to reduce the variance of a decision tree classifier. Here the objective is to create several subsets of data from training sample chosen randomly with replacement. Each collection of subset data is used to train their decision trees. As a result, we get an ensemble of different models. Average of all the predictions from different trees are used which is more robust than a single decision tree classifier.

## Neural network

A neural network is a series of algorithms that is used to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. In this sense, neural networks refer to systems of neurons, either organic or artificial in nature. Neural networks can adapt to changing input; so, the network generates the best possible result without needing to redesign the output criteria. The concept of neural networks, which has its roots in artificial intelligence, is swiftly gaining popularity in the development of trading systems. An artificial neural network behaves the same way. It works on three layers. The input layer takes input. The hidden layer processes the input. Finally, the output layer sends the calculated output.

### Supervised Neural Network

In the supervised neural network, the output of the input is already known. The predicted output of the neural network is compared with the actual output. Based on the error, the parameters are changed, and then fed into the neural network again. Supervised neural network is used in feed forward neural network.

### Unsupervised Neural Network

The neural network has no prior clue about the output the input. The main job of the network is to categorize the data according to some similarities. The neural network checks the correlation between various inputs and groups them.

### Reinforced Neural Network

Reinforcement learning refers to goal-oriented algorithms, which learn how to attain a complex objective (goal) or maximize along a particular dimension over many steps; for example, maximize the points won in a game over many moves. They can start from a blank slate, and under the right conditions they achieve superhuman performance.

## Instance Based learning

Instance-based learning refers to a family of techniques for classification and regression, which produce a class label based on the similarity of the query to its nearest neighbors in the training set. Instance-based learning algorithms do not create an abstraction from specific instances, but rather, they simply store all the data, and at query time derive an answer from an examination of the queries nearest neighbors.

### K-Nearest Neighbor

The k-nearest neighbors (KNN) algorithm is a simple, supervised machine learning algorithm that can be used to solve both classification and regression problems. It is a non-parametric, supervised learning classifier, which uses proximity to make classifications or predictions about the grouping of an individual data point. It is typically used as a classification algorithm, working off the assumption that similar points can be found near one another.

# Sensor network design

First, we needed to define what we considered to be the correct posture and what is the wrong one. To be able to do this, we needed to obtain appropriate information from a reliable source – domain expert. Figure 9 is showing how spine is divided into groups of vertebrae. Cervical spine has 7 vertebrae, thoracic spine has 12 vertebrae, lumbar spine has 5 vertebrae, sacrum has 5 vertebrae and coccyx has 4 vertebrae.

Obrázok, na ktorom je mapa

Automaticky generovaný popis

Figure 9 The vertebral column [12]

## Correct posture

We made an appointment with the domain expert, an orthopedist, and he helped us to have a better understanding of the posture problem. He explained and showed us a correct sitting posture, told us all about the mistakes, people usually make when sitting behind the computer for a long period of time. Figure 10 shows what is the correct sitting posture (on the right), and what is a bad posture when using computer for a long time (left). When people come to work and sit behind the computer, firstly they tend to be sitting correctly, as it is something, they are focusing on. However, after spending some time sitting, their attention and focus is drawn towards work tasks, and they stop keeping the correct upright posture. Another problem is that because of busy work, people tend to forget taking regular breaks, which help stretching tighten muscles and regain their focus on correct sitting when coming back to the table. That is why, when sitting correctly, one should have all the surface of one’s back pressed against the back of the chair, because it does not require as much focus to keep the back straight. One’s sacral spine should be as far back as possible; this will make sure that the lumbar and the rest of the spine will be upright. Arms should be bent in elbows in 90 degrees, supported by a table. Legs are supposed to be relaxed, also bent in about 90 degrees angle, feet situated on a small pad. We can call this posture the correct sitting posture no. 1.

These are the rules for an ideal posture, that most people do not follow. The correct sitting posture is also considered an upright posture, displayed in Figure 10 on the right, not necessarily with the back pressed against the back of the chair. One can sit correctly without leaning on the chair back, even though it is harder without a back support, as it requires much more focus to stay in a correct position. This position would be called the correct sitting posture no. 2.

Two positions described above will be considered a correct sitting posture in our work. All the other positions will not be correct. The most common wrong sitting posture are sitting with a hunched or slouched back with the head tilted forward, another position would be sitting at the front part of the chair, with the top of the back resting on the backrest.



Figure 10 Wrong vs. correct posture

## IoT sensors network

After meeting the domain expert, we took into consideration all his recommendations, and based on those, started working on our model of IoT sensors. For our experiment we will be using ergonomic chair.

### Types of sensors

For our model we would be using two types of sensors – sensors for distance measurement and pressure sensors. As we planned to mount sensors on the back of the chair, we had to consider the comfort of a person sitting. Hence, we had to choose the types of sensors, that would not be too thick and would almost merge with the chair. Regarding sensors for distance measurement, after doing some research we decided that Time-of-Flight type of sensors would be the most suitable for out model. There might be some more precise sensors on the market, but ToF sensors were chosen mainly for their thin shape. Time-of-Flight sensors [13] contain very small invisible laser source and a matching sensor. Those sensors use the time that it takes for photons to travel between two points to calculate the distance between them, that is what “time of flight” means. We decided to use VL6180 sensor (shown on Figure 11), which can measure 50 - 1200 mm and VL53L0X with 5mm to 200mm of range distance. As pressure sensors we are using Force Sensitive Resistors SEN-09375 (shown on Figure 12) [14] with combination with 3.3 ohm resistors to create a voltage that can be read by a microcontroller's analog-to-digital converter input.

Obrázok, na ktorom je elektronika

Automaticky generovaný popis

Figure 11 ToF sensor VL6180

Figure 12 Force sensitive resistor SEN-09375

### Microcontroller

To collect data from our sensor network we initially used Arduino Uno R3 microcontroller (Figure 13) with 32 kB of memory. However, after connecting all different types of sensors we are using, we found out, that memory was not sufficient, therefore we had to find an alternative. We researched some available options and decided to use Arduino Mega 2560 + Wi-Fi R3 microcontroller (Figure 14) which has sufficient memory of 256 kB. Since this microcontroller also has a Wi-Fi, it would make it easier to transfer our data into the database, as we will not have to use any additional cables.

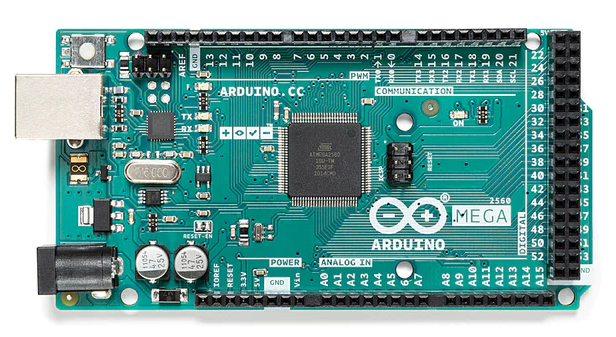
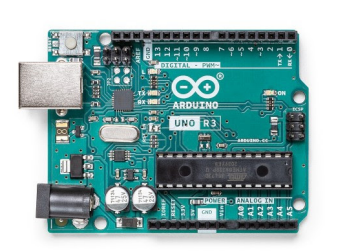


Figure 13 Arduino Uno R3

Figure 14 Arduino Mega 2560 + WiFi R3

### Model of IoT sensors

After the sensors were chosen, the next step was to design IoT sensors network we would use. From the information gained form the domain expert, we had a concept in mind. Sensors would be placed at the tops of the curves of the ergonomic chair, approximately on the levels of the C3 vertebrae of the cervical spine, the T7 vertebrae of the thoracic spine and the L4 of the lumbar spine. This layout should be suitable for an average person’s height.

Originally, we proposed a model of IoT sensors shown on Figure 15. The green dots represent ToF sensors and yellow dots represent pressure sensors. This network would help us identify incorrect postures, like tilting right or the left shoulder, and tilted lower back. After connecting all the sensors to the microcontroller and testing their functionality, we found out, that our force sensitive resistors do not respond well to a light pressure, which can cause a problem while measuring pressure on the bottom of the chair backrest. This realization led us to reconsider our initially proposed model of sensors.

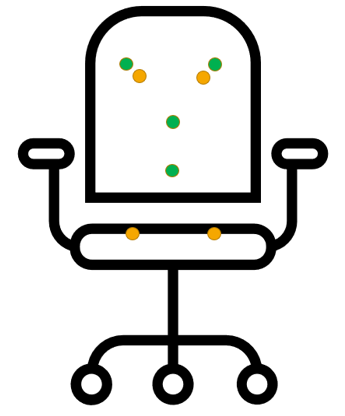
Obrázok, na ktorom je sedadlo, nábytok, stolička

Automaticky generovaný popis

Figure 15 Initial proposed model of IoT sensors

We agreed that the better option would be to place those two pressure sensors at the back of the chair seat, one on the right side, and one on the left, to obtain better results. Eventually we decided, that tilting positions of a computer user are not as common and being able to monitor them would not give us much advantage as we had thought. We directed our attention to simplification of our model and tried to focus on more common incorrect sitting postures. Newly proposed model is shown on Figure 16. However, we will try to use the model without two pressure sensors on the chair seat, because we would like to minimize number of sensors used in our model.

On the Figure 17 we can see our model mounted on the chair. We used rubber-like cover and stuck the sensors on it with a double-sided tape. Sticking sensors on the chair would be a little difficult and we would have to use good enough adhesive tape to make sure that sensors stay in place. Otherwise, it could cause an inaccurate measurement. We positioned four of them (two ToF and two pressure sensors) approximately at the level of shoulder blades, one at the middle curve of the ergonomic chair and last one at the level of the lumbar spine. We tried to approximate the layout of the sensors using three people with similar height (172 cm – 182 cm), one female and two males.

 Obrázok, na ktorom je vnútri, kancelária

Automaticky generovaný popis

Figure 16 Updated model of IoT sensors Figure 17 Model

On the back of the chair, the Arduino board is assembled. We welded cables with the same output together and at the end of each output was welded to the header pin which connects straight to the board. We used header pins for simpler manipulation with cables. They also stayed in place better than cables themselves. Our first idea was to make the chair cover portable, thus it could have been use on different chairs in different places. However, the cable management caused us some complications, as the cables had to be quite tightly wrapped around the chair in order not to be too long and in a way.

## Correct posture measurement

The idea of the proposed system of sensors is, that pressure sensors would provide us information about the correct sitting posture no. 1. It means that two pressure sensors on the chair seat would be activated, as well as two pressure sensors on the backrest of the chair. Another correct sitting position is the correct sitting posture no. 2, which is upright sitting without the backrest. To determine this position Time-of-Flight sensors come to use as indicate Figure 18.



Figure 18 Distance measurement

## Data processing

The next step in our work is to collect and process data. As we assembled all the sensors together and fastened them on the chair cover, we could start with the measurements.

### The database

The data would be stored in the InfluxDB database. Influx DB is an open-source time series database written in Go language which is developed by InfluxData. It is optimized for high-availability retrieval of data, faster and storage of time series data in fields such as operations monitoring, application metrics, IoT sensor data, and real-time analytics. InfluxDB is a high-performance Time Series Database which can store data ranging from hundreds of thousands of points per second. The InfluxDB is a SQL-kind of query language which was built specifically for time series data [15]. InfluxDB is a great option, as we have six IoT sensors and each of them is sending data multiple times per second.

### Data collection

We are using Arduino board, so firstly we made a script that collect the data from all the sensors. It is written in C++ as that is the standard programming language for Arduino. Subsequently we upload this script to the Arduino board. Then we wrote a Python script (Figure 19) which receives the data from Arduino and sending them on our university server, saving them in the database.

Figure 19 Script for data collection

def get\_write\_api(client):  
 try:  
 return client.write\_api(write\_options=SYNCHRONOUS)  
 except Exception as e:  
 return e  
  
def connect(port):  
 try:  
 arduino = serial.Serial(port,timeout=.1,baudrate=115200) *#create Serial object \*REMEMBER to check the number of COM* return arduino  
 except:  
 print(**"Check the port"**)  
  
def prep\_record(record):  
 try:  
 reading = float(record[1].strip().replace(**'**\r\n**'**,**""**))  
 return Point(**"record"**).tag(**"sensor"**,record[0]).field(**"reading"**, reading)  
 except:  
 return None  
  
if \_\_name\_\_ == **"\_\_main\_\_"**:  
 if len(sys.argv) > 1:  
 port = sys.argv[1]  
 else:  
 port = **"/dev/cu.usbserial-1130"** write\_api = get\_write\_api(client)  
 records = []  
 arduino = connect(port)  
 if not arduino:  
 logging.error(**f"Could not connect to arduino"**)  
 else:  
 while True:  
 raw = arduino.readline().decode(**'utf-8'**).split(**':'**)  
 if len(raw)==2:  
 records.append(prep\_record(raw))  
 if len(records) == 100:  
 try:  
 write\_api.write(bucket=bucket, record=records)  
 logging.info(**f"Wrote** {records}**"**)  
 records = []  
 except Exception as e:  
 logging.error(**f"Could not write record to influx,**{e}**"**)

Our data records are simple. They only consist of the name of the sensor, measured value and the timestamp. We were discussing two options we could use while working with data. The first would be using supervised machine learning. It means that we would have to label our data. We would perform an experiment similar than in [3]. We could either distinguish only between correct and incorrect position, so we would only have two labels, or decide on multiple incorrect sitting posture we would like to use. The subject would be sitting in the different positions for 30 seconds, and those would be labelled according to the position. We would not be using camera, but a person who would be supervising each subject and subsequently label the data. Then we would be able to use some supervised machine learning algorithms.

Another approach we would like to try is to let a person sit on the chair all day at work, without being watched. Then, without any data labeling, we would use different unsupervised machine learning algorithms to help us classify the positions into groups.

# Conclusion

In this work we addressed the problem of keeping the correct sitting posture while working with a computer which has become a worldwide problem. We researched this topic and tried to gain as much information as possible. We made an appointment with the domain expert, the orthopedist, who introduced us to this topic, which gave us a good base to be able to design our own model. We also had a look on different machine learning techniques to help us work with the data we collect and make it easier for us to decide, what model would be suitable for us.

In the first chapter we described different posture monitoring systems, we compared different sensors used and reviewed various machine learning models that authors used in their work. We focused more detailed on the pressure sensors monitoring systems described in 1.2. This was because those were the most like our proposed system as the sensors were placed on the chair. Our focus was on the types of sensors, placement on the chair, the overall course of experiment (e.g., amount of subject, different sitting positions, length of measurement for each position, and algorithms used).

The second chapter focused on introduction to machine learning, its meaning and importance in today’s world. Different types of existing machine learning were introduced along with the most known algorithm for each type of learning. This gave us a good overview of what algorithms we could focus on.

The last chapter was devoted to the posture monitoring system we designed. We started with describing the anatomy of the spine, we provided the information we obtained from an orthopedist, the domain expert, and based on that information we defined two correct sitting posture. We proposed a design of the initial system with an appropriate layout of IoT sensors, then the second, improved model which we will be using. Technical details of the sensor used and the type of microcontroller which our model uses were described in this chapter, along with the problems we encountered while designing and creating the model. We also mentioned the database we chose to use for data collection. The process of measurement and data collection with two different possible approaches we are about to take to analyze data.

The next step would be data collection and annotation. We have everything ready to start collecting data, so the forthcoming task would be to find volunteers to collect the data and then process those data. The last important thing will be trying different types of machine learning models and finding the most suitable for our data for classification of correct an incorrect sitting posture of our subjects.

# References

[1] M. Taieb-Maimon, J. Cwikel, B. Shapira, and I. Orenstein, “The effectiveness of a training method using self-modeling webcam photos for reducing musculoskeletal risk among office workers using computers,” *Appl. Ergon.*, vol. 43, no. 2, pp. 376–385, Mar. 2012, doi: 10.1016/J.APERGO.2011.05.015.

[2] A. Kulikajevas, R. Maskeliunas, and R. Damaševičius, “Detection of sitting posture using hierarchical image composition and deep learning,” *PeerJ Comput. Sci.*, vol. 7, pp. 1–20, Mar. 2021, doi: 10.7717/PEERJ-CS.442/TABLE-4.

[3] J. Roh, H. J. Park, K. J. Lee, J. Hyeong, S. Kim, and B. Lee, “Sitting Posture Monitoring System Based on a Low-Cost Load Cell Using Machine Learning,” *Sensors 2018, Vol. 18, Page 208*, vol. 18, no. 1, p. 208, Jan. 2018, doi: 10.3390/S18010208.

[4] F. Luna-Perejón, J. M. Montes-Sánchez, L. Durán-López, A. Vazquez-Baeza, I. Beasley-Bohórquez, and J. L. Sevillano-Ramos, “Iot device for sitting posture classification using artificial neural networks,” *Electron.*, vol. 10, no. 15, pp. 1–15, 2021, doi: 10.3390/electronics10151825.

[5] S. Matuska, M. Paralic, and R. Hudec, “A Smart System for Sitting Posture Detection Based on Force Sensors and Mobile Application,” *Mob. Inf. Syst.*, vol. 2020, 2020, doi: 10.1155/2020/6625797.

[6] J. Cheng, B. Zhou, P. Lukowicz, and M. Sundholm, “MONARCA View project SocialWear View project Smart Chair: What Can Simple Pressure Sensors under the Chairs’ Legs Tell Us about User Activity?,” Accessed: Jun. 19, 2022. [Online]. Available: https://www.researchgate.net/publication/290552685.

[7] W. Y. Wong and M. S. Wong, “Trunk posture monitoring with inertial sensors,” *Eur. Spine J.*, vol. 17, no. 5, pp. 743–753, May 2008, doi: 10.1007/S00586-008-0586-0/FIGURES/12.

[8] Q. Wang, W. Chen, A. A. A. Timmermans, C. Karachristos, J. B. Martens, and P. Markopoulos, “Smart Rehabilitation Garment for posture monitoring,” *Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBS*, vol. 2015-November, pp. 5736–5739, Nov. 2015, doi: 10.1109/EMBC.2015.7319695.

[9] A. Fathi and K. Curran, “Detection of spine curvature using wireless sensors,” *J. King Saud Univ. - Sci.*, vol. 29, no. 4, pp. 553–560, Oct. 2017, doi: 10.1016/J.JKSUS.2017.09.014.

[10] M. I. Jordan and T. M. Mitchell, “Machine learning: Trends, perspectives, and prospects,” *Science (80-. ).*, vol. 349, no. 6245, pp. 255–260, Jul. 2015, doi: 10.1126/SCIENCE.AAA8415.

[11] B. Mahesh, “Machine Learning Algorithms-A Review Machine Learning Algorithms-A Review View project Self Flowing Generator View project Batta Mahesh Independent Researcher Machine Learning Algorithms-A Review,” *Int. J. Sci. Res.*, 2018, doi: 10.21275/ART20203995.

[12] “The Vertebral Column.” https://antranik.org/the-vertebral-column/ (accessed Jun. 19, 2022).

[13] “Arduino Code | Adafruit VL6180X Time of Flight Micro-LIDAR Distance Sensor Breakout | Adafruit Learning System.” https://learn.adafruit.com/adafruit-vl6180x-time-of-flight-micro-lidar-distance-sensor-breakout/wiring-and-test?view=all (accessed Jun. 19, 2022).

[14] “Force Sensitive Resistor Hookup Guide - learn.sparkfun.com.” https://learn.sparkfun.com/tutorials/force-sensitive-resistor-hookup-guide (accessed Jun. 19, 2022).

[15] “Introduction to InfluxDB: A time-series database.” https://wearecommunity.io/communities/india-java-user-group/articles/891 (accessed Jan. 11, 2023).