Diagnosing Knee Osteoarthritis Using Artificial Neural Networks and Deep Learning

Jean de Dieu Uwisengeyimana¹,², *, Turgay Ibrikci¹

¹Department of Electrical and Electronics Engineering, Cukurova University, Adana, Turkey
²Department of Electrical and Electronics Engineering, University of Rwanda, Kigali, Rwanda

Email address:
uwisengeyimana@gmail.com (J. de D. Uwisengeyimana)
*Corresponding author

To cite this article:

Received: January 24, 2017; Accepted: February 18, 2017; Published: March 29, 2017

Abstract: Among various medical diagnostic tests performed to identify osteoarthritis in the knee, most of them are invasive and expensive. Therefore, in this study, another methodology for diagnosing osteoarthritis in the knee in a more quick, non-invasive and cheap manner was proposed. For that purpose, surface electromyography signals recorded from the four muscles surrounding the knee, the recording of the flexion degree in the knee and pattern recognition algorithms were used. The datasets of this experiment comprised 22 subjects among whom 11 subjects had normal knee and other 11 Subjects had an osteoarthritis-affected knee. The total sample size was 1,048,576 samples and were processed using segments of overlapping-windows of 5000 samples. Time-series features were then extracted from each segment and were used to train, test and validate 7 different learning classifiers and 7 variants of deep learning networks. In this study, the best performance measure of 99.5% was achieved by multilayer perceptron. Quadratic support vector machine and complex tree performed as well with accuracy of 99.4% and 98.3% respectively. In contrast, the use of deep learning networks which were investigated over a wide range of hidden size of the sparse autoencoders, showed accuracy of 86.6% with final softmax layer and accuracy of 91.3% by replacing the final softmax layer with k-nearest neighbour. By comparison, artificial neural networks outperformed deep learning networks and it is therefore concluded that the knee pathology can be diagnosed more efficiently and automatically using surface electromyography signals and artificial neural network algorithms.

Keywords: Diagnosis, Electromyography, Deep Learning, Feature Extraction, Artificial Neural Network

1. Introduction

Osteoarthritis is one of joint pathologies that mostly affects cartilage and the majority of cases of cartilage damage involve the knee joint [1, 2]. Critical knee injuries such as those that damage the pairs of cruciate ligaments in the knee have been of the most interest in researches [3]. These knee damages, when not diagnosed and treated the soonest, may result in damage and deterioration resulting in disappearing of meniscus and weak alignment [4]. It is therefore very important to early diagnose pathology of the knee so as to treat it the soonest.

Several studies have studied the efficient way for diagnosing osteoarthritis. From [5], one method was by analysing the knee images recorded by X-Ray and to automate the detection of osteoarthritis using image classification techniques. Actually X-Ray technique is an efficient way to diagnose any default in bones and joints, but according to [6], this way is expensive and is not always preferred the earliest possible. Another technique is by using Magnetic Resonance Imaging (MRI), which is used to take many images of the knee cartilage and its ligaments. Studies [7] and [8], discussed the recent start of using MRI and quantitative image analysis technology to give information on the state of the cartilage, bone and degenerative changes in osteoarthritis. However, MRI tests are also expensive and most of the hospitals in developing countries do not have these equipments.

Hence, in this study we propose an automated process to diagnose any abnormal knee by using muscle signals recorded by the Surface Electromyography (SEMG) along with signal processing techniques and classifying algorithms.
In the literatures, the main use cases of Electromyography (EMG) signals are met in neurology, rehabilitation [9], ergonomics [10], and sports [11]. These signals are also used as diagnostic tool for neuro-muscular and motor control problems [12, 13]. According to [14], the state of quadriceps muscles can suggest whether the ligaments in the knee are unstable, or that there are problems with the knee.

To realize the purposes of this study, which is about building a prototyping system to automatically diagnose knee pathology to support related medical diagnosis, the combination of SEMG signals recorded from the four muscles of interest (rectus femoris, vastus medialis, biceps femoris and semitendinosus), goniometry signals and machine learning techniques were used. Also a comparison between Artificial Neural Network (ANN) and deep learning has been done.

2. Materials and Methods

2.1. Lower Limb EMG Data Sets

In this study, we have used the newly uploaded datasets which are publicly available at UCI [15]. In these datasets, twenty two people were hired to record the data. From those people, eleven subjects had knee pathologies confirmed by the physiotherapist. Three positions which are marching, sitting, and standing were undergone by the subjects while analyzing the behavior associated with the knee muscles. The muscles of concern were the vastus medialis, semitendinosus, biceps femoris and rectus femoris. The data sets were acquired from a subject to a computer’s storage through bluetooth. Also, 14-bit resolution and fs=1000 Hz were used. Finally, the total sample size acquired was 1,048,576 samples.

2.2. EMG Feature Extraction

Time series features can be extracted based on either time domain, frequency domain or a combination of the two signal domains. Each of these different types of features are used in a specific application. From many previous studies especially those concerned with EMG signals, the frequently used features were based on time-domain statistical features. [16-19]. The time-domain features were widely used in many literatures due to their relatively easy construction and high efficiency.

In this study, before feature extraction, EMG data acquired from all the four electrodes were segmented resulting in a chain of analysis windows. As shown in figure 1, time overlapping windows method was used. The window overlapping was 90% and the window length was 5000 samples.

After signal windowing, extraction of time domain features was made from each of the analysis windows. It should be noted that the processes of signal windowing and feature extraction acted as dimensionality reduction which in the end resulted in 2642 samples and 30 features.

Below are formulas for the representative techniques in time-domain used for extracting signal features from each signal window:

- Mean power of the raw data:
  \[ P = \frac{1}{N} \sum_{n=1}^{N} X^2(n) \]  
  (1)

- Peak Value (PV) of the raw signal:
  \[ PV = \text{MAX} \left( \sqrt{X^2(n)} + \sqrt{X^2(n)} \right) \]  
  (2)

Where, \( \hat{X}^2(n) \) is Hilbert transform of \( X(n) \), \( X_s(n) \) is the pre-envelope of \( X(n) \) and \( \sqrt{X^2(n)} + \sqrt{X^2(n)} \) is the envelope of \( X(n) \)

- Mean of the raw data \( (\mu_X) \):
  \[ \mu_X = \frac{1}{N} \sum_{n=1}^{N} X(n) \]  
  (3)

- Standard deviation of raw data \( (\sigma_X) \):
  \[ \sigma_X = \sqrt{\left( \frac{1}{N} \sum_{n=1}^{N} (X(n) - \mu_X)^2 \right)} \]  
  (4)

- Variance of the raw data \( (\sigma^2_X) \):
  \[ \sigma^2_X = \frac{1}{N} \sum_{n=1}^{N} (X(n) - \mu_X)^2 \]  
  (5)

Where, \( X(n) \) is the vector of the data points and \( \mu_X \) is the mean of the data points.

2.3. Classification Algorithms

After processing the SEMG signals, several classifiers were applied using Matlab toolbox [21].

2.3.1. Multilayer Perceptron (MLP)

Multilayer Perceptron was implemented to tackle the limitation of single layer Perceptron. To train the multilayer perceptron network, the error back-propagation learning algorithm is adopted [22]. The basis of this learning algorithm is on the error-correction learning rule, described by the below equation.

\[ w(n+1) = w(n) + \eta [d(n) - y(n)] x(n) \]  
(6)

Where in the equation above, \( x(n) \) is the examples’ input vector, \( w(n) \) is the weight vector, \( b(n) \) is the bias, \( y(n) \) is the system response, \( d(n) \) is the target response, \( \eta \) is the learning rate constant, which is a constant less than a unit and...
$n$ is the time step, $n = 0, 1, 2, ...$

2.3.2. Support Vector Machine

i. Linear Support Vector Machines (SVMs)

SVMs were previously employed in classification of two categories by setting a maximum margin hyperplane between the two classes. Assuming a linear classifier with a function

$$f(w, b) = signum(w^T x + b)$$

Then, the distance from the point $x$ to the hyperplane separating the classes is given by

$$w^T x + b$$

Thus, SVMs consist of the following constrained optimization:

$$\min_{w, \xi} \frac{1}{2} w^T w + C \sum_{i=1}^{N} \xi_i$$

Where, in the equations (7)- (9), $x_i$ are support vectors, $w$ is the normal vector to the plane, $y_i$ is the desired class response, $b$ indicates the bias, $C$ is the normalization constant and, $\xi_i$ are slack variables that are inserted into the equation to give the classifier the ability to handle some data that could not be well separated, like those data containing noise. [23]. The of equation (9) is referred to as the primal form of norm L1-SVM

ii. Quadratic Support Vector Machine

The fact that the equation (9) is not differentiable leads to another most used variation known as the dual formor as the norm L2-SVM which minimizes the squared hinge loss:

$$\min_{w, \xi} \frac{1}{2} w^T w + C \sum_{i=1}^{N} \xi_i^2$$

Hence, since norm L2-SVM is differentiable, this is what shows a difference between quadratic and linear SVM. For Kernalized SVMs, like quadratic SVM, optimization must be done in dual rather than primal form [24, 25].

2.3.3. Logistic Regression

Assuming that we have a function $g: X \rightarrow C$, where $C$ is the class label, and $X = (X_1, X_2, ..., X_N)$ is examples’ input vector; this method considers a distribution in the form of $P(C|X)$ and immediately approximates its parameters from the examples [26, 27]. This shows that logistic regression is a parametric learning model and the parameters that it surmises separating the classes is given by

$$\frac{w^T x + b}{\|w\|}$$

The fact that the equation (9) is not differentiable leads to another most used variation known as the dual formor as the norm L2-SVM which minimizes the squared hinge loss:

$$\min_{w, \xi} \frac{1}{2} w^T w + C \sum_{i=1}^{N} \xi_i$$

With logistic regression, the distribution $P(C|X)$ is thought of as to keep track of the shape of logistic function. So considering the case of linear classification, the model sets category $C= 0$ if $X$ satisfies the equation:

$$0 < w_o + \sum_{i=1}^{n} w_i X_i$$

and sets $C= 1$ otherwise.

2.3.4. Linear Discriminant Analysis

The concept of Linear Discriminant Analysis (LDA) is to separate the samples from two different classes by making the inter-class distance large while setting small intra-class variances. For long time, LDA has been an important way for either classification or dimension reduction tasks. For classification tasks, LDA is based on mahalanobis distance ($D_M$) [28]:

$$D_M = \sqrt{(x - \mu)^T C^{-1} (x - \mu)}$$

Where, $x = (x_1, x_2, ..., x_n)^T$ is the predictors’ vector, $\mu = (\mu_1, \mu_2, ..., \mu_n)$ is the class centroids’ vector and $C$ is the pooled within-class covariance of the predictors.

2.3.5. Complex and Medium Decision Trees

Decision Trees (DTs) are tree-like decision techniques employed to construct classification or regression systems [29]. In each iteration of decision tree learning algorithms, a dataset is fed and a variable is sorted out and is used to split up the dataset into subsets; where every subset is considered as the provided data set for the next iteration. Now the concept of decision tree algorithm is based on using information gain to decide which best variable to select to test each node. To introduce information gain, we first introduce the entropy, which measures the amount of information and noise present in a signal [30].

Given a binary classification problem with a set $D$ of positive ($p_+$) and negative ($p_-$) examples, the entropy of a set $D$ relative to this binary categorization is given by:

$$Entropy(D) = -p_+ \log_2 (p_+) - p_- \log_2 (p_-)$$

Then, the information gain which measures the expected reduction in entropy, or uncertainty caused by partitioning the examples according to the selected attributes, is given by

$$Gain(D, A) = Entropy(D) - \sum_{v \in Values(A)} \frac{|D_v|}{|D|} Entropy(D_v)$$

Where, $Values(A)$ is a set of all possible values for attribute $A$ and $D_v$ is the subset of $D$ for which attribute $A$ has a value $v$.

The main difference between complex tree and medium tree is determined by the number of leaves; where for medium tree, this number does not go beyond 20 splits while for complex tree it can come up to 100 splits.

2.3.6. k-Nearest Neighbours

k-Nearest Neighbours (k-NN) is a lazy learning classifier, which means that it hardly learns anything from the training examples [31]. To classify any new data sample, the k-NN algorithm will have to first estimate the k closest neighbors from the training examples to the new sample. Then, the system class label for the new sample, will be the same as the class label of the k closest neighboring points. If $k = 1$, the
new sample will be automatically set to the class of its nearest neighbor.

To find k-neighbors closest to the new data sample k-NN, uses a metric for measuring the distance between the new point and cases from the examples. The most popular distance functions to measure this distance are defined below:

\[ \text{Euclidean} = \sqrt{\sum_{i=1}^{N} (x_i - y_i)^2} \]  
(17)

\[ \text{Manhattan} = \sum_{i=1}^{N} |x_i - y_i| \]  
(18)

\[ \text{Minkowski} = (\sum_{i=1}^{N} |x_i - y_i|^q)^{1/q} \]  
(19)

Where, \( x \) and \( y \) are the new point and a case from the examples, respectively.

### 2.3.7. Deep Learning Networks and Algorithm

A deep learning model is basically an artificial neural network that has more than one hidden layers [32-34]. The first breakthrough results in deep learning appeared since early 2000 and used Deep Belief Networks (DBNs) to pre-train deep networks [35-37]. In many past studies, autoencoders have regained the prominence in the deep learning approach designed for the pre-training task [38, 39].

Algorithm for autoencoders is influenced by the concept of a good depiction of the data. Suppose the input to an autoencoder is a vector \( x \in \mathbb{R}^k \), then the encoder maps the vector \( x \) back and forth to another vector \( z \in \mathbb{R}^k \) as follows:

\[ z^{(i)} = W^{(i)}_1 x + b^{(i)}_1 \]  
(20)

\[ \hat{z}^{(i)} = W^{(i)}_2 x + b^{(i)}_2 \]  
(21)

Where, the superscript \( (i) \) indicates the layer, \( W^{(i)}_1 \in \mathbb{R}^{k \times d} \) and \( W^{(i)}_2 \in \mathbb{R}^{d \times k} \) are weight matrices for encoder and decoder respectively and \( b^{(i)}_1 \in \mathbb{R}^k \) and \( b^{(i)}_2 \) are bias vectors for encoder and decoder respectively.

We can set up the following objective function, which is the sum of squared differences between \( \hat{z}^{(i)} \) and \( x^{(i)} \):

\[ J(W_1, b_1, W_2, b_2) = \sum_{i=1}^{m} (\hat{z}^{(i)} - z^{(i)})^2 \]  
(22)

\[ = \sum_{i=1}^{m} (W_2 z^{(i)} + b_2 - x^{(i)})^2 \]  
(23)

\[ = \sum_{i=1}^{m} (W_2(W_1 x^{(i)} + b_1) + b_2 - x^{(i)})^2 \]  
(24)

Where, in the formulas (22)-(24) again, \( x^{(i)} \) are the input vectors to the autoencoder, \( z^{(i)} \) are the coded input vectors, \( \hat{z}^{(i)} \) are the decoded input vector, \( W_1 \) and \( b_1 \) are respectively weight and bias parameters for the encoder and \( W_2 \) and \( b_2 \) are also respective weight and bias parameters for the decoder.

### 3. Results and Discussions

In this study, the main goal was to build a diagnosing system which would be able to automatically distinguish between the patients having abnormal knee and those having normal knee with least error rate. To do this, EMG signals acquired from four muscles surrounding the knee were first filtered using a second order Chebyshev filter which removed signals that were outside the bandwidth of the electromyography signals which roughly ranges from 20Hz to 500Hz [40]. For this filter design, the attenuation ripple was 3 dB and the attenuation of unwanted signals was 60 dB.

From the filtered signals, thirty time-domain features were extracted and were used to train, test and validate 7 classifiers and 7 deep learning networks. The results from the first seven classifiers presented in table 1, where different classifier evaluation metrics such as classification accuracy, precision (specificity), recall (sensitivity), F-measure and the error rate were calculated from the confusion matrix. For this purpose, the formulas below were used.

**Classification accuracy:** The number of correct predictions from all the predictions made.

\[ \text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \]  
(25)

**Recall:** The proportion of actual positive which are predicted positive.

\[ \text{Recall (Sensitivity)} = \frac{TP}{TP+FN} \]  
(26)

**Precision:** The proportion of predicted positive which are actual positive.

\[ \text{Precision} = \frac{TP}{TP+FP} \]  
(27)

**F-measure:** It is harmonic mean between Precision and Recall and also known as F-score.

\[ F = 2 \times \frac{\text{Precision} \times \text{Recall}} {\text{Precision} + \text{Recall}} \]  
(28)

**Error Rate:** The number of mistakes made by the classifier.

\[ \text{Error Rate} = \frac{FP+FN}{TP+TN+FP+FN} \]  
(29)

Where in the formulas above; \( TP \) is True positive, \( TN \) is True Negative, and \( FP \) is False Positive, and \( FN \) is False Negative.

#### Table 1. The statistical performance averages for the 7 different classifiers.

<table>
<thead>
<tr>
<th>Classifier Group</th>
<th>Classifier</th>
<th>Accuracy%</th>
<th>AUC</th>
<th>Recall</th>
<th>Precision</th>
<th>F-measure</th>
<th>Error Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decision Trees</td>
<td>Complex Tree (CT)</td>
<td>98.3</td>
<td>0.99</td>
<td>0.98</td>
<td>0.99</td>
<td>0.98</td>
<td>0.017</td>
</tr>
<tr>
<td></td>
<td>Medium Tree (MT)</td>
<td>97</td>
<td>0.98</td>
<td>0.93</td>
<td>0.99</td>
<td>0.96</td>
<td>0.03</td>
</tr>
<tr>
<td>Discriminant Analysis</td>
<td>Linear Discriminant Analysis (LDA)</td>
<td>65.9</td>
<td>0.72</td>
<td>0.73</td>
<td>0.59</td>
<td>0.65</td>
<td>0.341</td>
</tr>
</tbody>
</table>
In addition to the performance metrics described above, a Receiver Operating Characteristic (ROC) curve was another important measure used to select the best classifier [41-43]. To compare classifiers ROC performance was expressed as a single scalar value standing in place of expected performance by calculating the area under the ROC curve (AUC). Noting that the value of AUC is always between 0 and 1, then the best classifier is the one with a relatively high value of AUC. Combining the results of all these performance metrics, we can see that the most three competitive algorithms for this application with their respective accuracies were shown up to be multilayer perceptron (99.5%), Quadratic SVM (99.4%) and Complex Tree (98.3%).

From the column bar chart of figure 2, which compares the performance metrics among the seven classification learners, it is seen that MLP is outstanding due to its highest accuracy, recall, precision, F-measure and its lowest error rate. Furthermore, MLP systems actually have one more other advantage over kernelized SVM that might make it more applicable: Regarding the training time, MLP is quicker than SVM and the reason for the relative slowness of SVM is that its training requires calculating the solutions of the associated Lagrangian dual problem rather than primal problem [44].

Apart from shallow classification learners, 7 deep learning models were also investigated. To make the first basis of deep learning system, we made an artificial neural network that had two hidden layers and train the hidden layers separately in unsupervised manner using two sparse autoencoders. Then, by use of the output of the last autoencoder, we trained a final softmax layer, and matched the layers together [45]. Then, different performances of this deep learning model were achieved by varying the size of the hidden layer for the two autoencoders. Where, the highest accuracy of 86.6% was achieved with 900 nodes in sparse autoencoder 1. and 200 nodes in sparse autoencoder 2, as shown in figure 3.
architecture were made by replacing the final softmax layer with other standard learning algorithms. Hence, instead of feeding the output of the second autoencoder to the final softmax layer, it was rather fed to complex tree, medium tree, logistic regression, k-NN, quadratic SVM and linear discriminant. By this way, we achieved an accuracy of 91.3% with a deep learning network made by combination of sparse autoencoder’s output with k-NN. Detailed results are presented in table 2. and the performance comparison has been shown in figure 5.

<table>
<thead>
<tr>
<th>Base algorithm</th>
<th>Final layer algorithm</th>
<th>Accuracy%</th>
<th>AUC</th>
<th>Precision</th>
<th>F-Measure</th>
<th>Error Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Softmax</td>
<td></td>
<td>86.6</td>
<td>0.91</td>
<td>0.90</td>
<td>0.89</td>
<td>0.134</td>
</tr>
<tr>
<td>Complex Tree (CT)</td>
<td></td>
<td>86.5</td>
<td>0.89</td>
<td>0.86</td>
<td>0.9</td>
<td>0.135</td>
</tr>
<tr>
<td>Medium Tree (MT)</td>
<td></td>
<td>81.9</td>
<td>0.85</td>
<td>0.82</td>
<td>0.88</td>
<td>0.111</td>
</tr>
<tr>
<td>Logistic regression (LR)</td>
<td></td>
<td>88.9</td>
<td>0.88</td>
<td>0.86</td>
<td>0.88</td>
<td>0.107</td>
</tr>
<tr>
<td>k-NN</td>
<td></td>
<td>91.3</td>
<td>0.91</td>
<td>0.88</td>
<td>0.93</td>
<td>0.087</td>
</tr>
<tr>
<td>Quadratic SVM (QSVM)</td>
<td></td>
<td>89.8</td>
<td>0.94</td>
<td>0.84</td>
<td>0.93</td>
<td>0.088</td>
</tr>
<tr>
<td>Linear Discriminant Analysis (LDA)</td>
<td></td>
<td>64.6</td>
<td>0.54</td>
<td>0.55</td>
<td>0.81</td>
<td>0.65</td>
</tr>
</tbody>
</table>

Table 2. Results of new variants of deep learning by different final layer classification algorithms.

![Figure 5. Comparing results obtained by combining sparse autoencoders with different learning algorithms.](image)

4. Conclusion

Automatic diagnosis is very important in medical field. In this study, the main purpose was to distinguish healthy people from people with knee abnormality. For this purpose, signals from four lower limb muscles recorded using a four-channel Surface Electromyography (SEMG) was used along with goniometer signals which measures the flexion at the knee. From EMG and goniometer data obtained, statistical features were extracted and applied to train, test and validate 7. learning classifiers and 7 variants of deep learning systems. In this study, the best performance measure of 99.5% was achieved by multilayer perceptron. Quadratic support vector machine and complex tree also achieved a good accuracy of 99.4% and 98.3% respectively. Whereas, the combination of deep learning network with k-NN showed a relatively lower accuracy (91.3%) in comparison to artificial neural networks. Therefore, it is concluded that the knee pathology can be diagnosed using surface Electromyography signals and Artificial Neural Network (ANN) algorithms.

Acknowledgments

The authors would firstly like to appreciate the research funds provided by Çukurova University with the Project Number: FYL-2017-7864. We also thank the Scientific and Technological Research Council of Turkey (TÜBİTAK)-BIDEB2235 and the University of Rwanda (UR) for their help, which made this work possible.

References


