Classification of Corneal Nerve Images using Machine Learning Techniques

Tooba Salahuddin[1], Uvais Qidwai[1*]

[1]Department of Computer Science and Engineering,
 College of Engineering, Qatar University, Doha, QATAR

*Corresponding Author

Abstract: Recent research shows that small nerve fiber damage is an early detector of neuropathy. These small nerve fibers are present in the human cornea and can be visualized through the use of a corneal confocal microscope. A series of images can be acquired from the subbasal nerve plexus of the cornea. Before the images can be quantified for nerve loss, a human expert manually traces the nerves in the image and then classifies the image as having neuropathy or not. Some nerve tracing algorithms are available in the literature, but none of them are reported as being used in clinical practice. An alternate practice is to visually classify the image for neuropathy without quantification. In this paper, we evaluate the potential of various machine learning techniques for automating corneal nerve image classification. First, the images are down-sampled using discrete wavelet transform, filtering and a number of morphological operations. The resulting binary image is used for extracting characteristic features of the image. This is followed by training the classifier on the extracted features. The trained classifier is then used for predicting the state of the nerves in the images. Our experiments yield a classification accuracy of 0.91 reflecting the effectiveness of the proposed method.

Keywords: Machine learning, image classification, corneal confocal microscopy, medical image analysis, neuropathy

1. Introduction

A debilitating complication of many chronic diseases is neuropathy. It is a disorder of the central nervous system which degrades the quality of life of the patient. Early and timely diagnosis of neuropathy is beneficial in many ways. It can help determine the severity level of nerve damage and allow the monitoring of disease growth. Peripheral neuropathy is distinguished by numbness in the limbs and is the most prevalent complication of diabetes. Other visible effects of neuropathy include foot ulceration [1]. One of the earliest hidden symptoms of neuropathy is small fiber nerve damage and is apparent in a very early stage prior to the occurrence of visible symptoms [2]. On the other hand, visible symptoms of neuropathy occur only when the damage has reached the long nerve fibers. Therefore, precise and prompt diagnosis of neuropathy is necessary for prognosis, early recognition of subclinical neuropathy, monitoring disease growth, classifying disease severity and suggesting relevant therapy plans [3].

State-of-the-art techniques for detecting nerve damage include electrophysiology, quantitative sensory testing, skin biopsy and nerve conduction studies. Most of these techniques are unable to detect small nerve fiber loss and provide subjective and inaccurate results [4]. Although skin biopsy has been successful in detecting small nerve fiber loss, the technique itself is invasive and therefore cannot be conducted frequently. Moreover, it is time consuming and requires expert skill [5].

Recently, in vivo corneal confocal microscopy (CCM) has emerged as a non-invasive, objective surrogate and imaging biomarker for detecting nerve fiber deficits. Due to the scientific fact that small nerve fibers are present in the human cornea, an insight into the subbasal nerve plexus of the cornea can detect very early neuropathy. The

transparency of the epithelium enables the laser to penetrate into the different layers of the cornea and give a clear visualization. Thus, CCM images reveal a detailed and magnified structure of the densely innervated cornea of the human eye.

Studies have demonstrated the effectiveness and reproducibility of CCM in detecting neuropathy in diabetic patients [3] and subjects with Parkinson's disease [6], Multiple Sclerosis [7], chronic migraine [8], chemotherapy induced neuropathy [9], human immunodeficiency virus [10] and acute ischemic stroke [11].

CCM provides a detailed and magnified visual representation of the corneal nerve structure. Current limitations of such a promising tool include the tedious process of manual nerve tracing by clinicians for nerve parameter quantification and classification of images to define the degree of nerve damage. Rapid, accurate and automated quantification of CCM images by exploiting image processing techniques tends to be a challenging task. Nevertheless, significant research has been conducted in this domain [12]–[17] attempting to address the challenge by employing different techniques.

To the best of our knowledge, currently there is no fully automatic system for classification of CCM images captured from the sub-basal nerve plexus of the cornea. One research group [18] has proposed the idea of neuropathy classification through convolutional neural networks, but it is a pilot study and lacks an in-depth analysis of the results.

Therefore, the primary contribution of this research is to evaluate machine learning techniques for classifying corneal nerve images, using adaptive neuro fuzzy inference system (ANFIS), support vector machines (SVM), naïve Bayes (NB), linear discriminant analysis (LDA), classification trees and k-nearest neighbours (KNN). The classifier can distinguish between the state of nerves in the corneal images as normal or abnormal. The proposed system significantly speeds up the classification process which allows for early diagnosis of neuropathy.

This paper is organized as follows. The subsequent section presents related work on CCM image segmentation and classification. This is followed by a detailed description of the proposed method including nerve segmentation along with a description of machine learning algorithms in Section III. The evaluation of the classification technique is reported in Section IV which explains the experimental setting and achieved results. Finally, Section V concludes the paper with possible future research directions.

2. Related Work

A factor hindering the advancement of CCM for neuropathy detection is the absence of precise and automated systems for image analysis and disease prediction. Precise nerve segmentation and quantification techniques are required for the establishment of reliable and consistent standards for nerve measurements. Researchers have approached nerve segmentation of CCM images using various methods. Ruggeri et al. [17] proposed a nerve recognition and tracing method based on vessel segmentation in retinal images [15]. This method starts with fixed locations for seed point extraction and tracks nerve pixels by expanding the region of interest. Conflicts at nerve intersections are rectified by using a technique called bubble analysis which identifies nerve pixels by going through concentric circles from the center point. Then, fuzzy k-means clustering is applied to classify pixels as nerve or non-nerve. The proposed algorithm was evaluated on 12 CCM images captured from a sit-lamp CCM. The algorithm showed a tendency for increased false positives, possibly due to the existence of other structures in the background. The segmentation time per image was $4-5$ minutes. The same algorithm was modified by Scarpa et al. [16] to include the use of Gabor filters before nerve tracking. A further enhancement of the algorithm was performed by Poletti and Ruggeri [14], by allowing multiple orientations of the lines for seed points extraction. The algorithm was tested on 30 corneal nerve images and the segmentation time was reduced to 25 seconds per image.

Dabbah et al. [19] developed a dual model algorithm for nerve segmentation by applying Gabor and Gaussian filters. A comparative analysis of the proposed algorithm with another previously reported method for detecting asbestos fibers [20] showed improved performance for the dual model approach. Later, the method was modified with multiscale enhancement in [21], and pixel classification was approached through neural networks and random forest classifiers. In another study [22], classification was performed using SVM. Al-Fahdawi et al. [12] approached nerve segmentation through morphological operations. They applied coherence and Gaussian filters for contrast enhancement followed by dilation and erosion operations for noise reduction. Canny edge detection is employed for detecting nerve edges. The algorithm required about 7 seconds per image and was tested on approximately 1500 images.

Colonna et al. [18] introduced the idea of using U-net for nerve segmentation. U-net is a convolutional neural network originally designed for segmentation of biomedical images [23]. During the preprocessing stage, 10 pixels from each side of the image were removed and then the image was resized to the input size required by U-Net using a bicubic transformation. The training was performed on 8909 images obtained from diabetic and healthy subjects. 30% of this dataset was used for validation only. The maximum training epochs of the network was set to 6 epochs. Due to the lack of manual tracings for the training images, they created segmented images by applying a previously proposed algorithm for segmentation [24] and used it as ground truth for comparison. The performance of the trained model was tested on 30 test images and compared against manual tracings by an observer. In order to allow for slight shifts in the nerve position, a tolerance of 3 pixels was allowed. Results revealed a sensitivity of 97% and a false detection rate of 18%. However, since the number of test images is so small, the ability of the model to generalize cannot be determined.

The authors also used U-Net for neuropathy classification of images. They tested the trained model on 100 images and obtained an accuracy of 83% for binary classification (healthy/pathological). No further details on the experimental results were provided.

In summary, the related research focusses on the segmentation techniques for CCM images and the domain of neuropathy classification of CCM images is yet to be explored. We show the potential of machine learning techniques for automatic classification of CCM images. In the following sections, we discuss our methodology.

3. Materials and Methods

3.1 Dataset

The performance of the classifier was evaluated on 297 CCM images taken from the dataset in [25]. 93 images belong to the normal class, while the rest belong to the abnormal class. The images were captured using a laser scanning corneal confocal microscope, Heidelberg Retinal Tomograph, equipped with Rostock Corneal Module (HRT-RCM: Heidelberg Engineering, Heidelberg, Germany). The images are of size 384x384 pixels and saved in JPG format.

3.2 Nerve Segmentation

During nerve segmentation, the images first undergo discrete wavelet transform to reduce the size of the image to one-fourth of the original. For the elimination of background noise and enhancement of linear structures Gaussian and coherence filters are applied. The diffusion scheme for coherence filter was chosen to be optimized derivative kernels, because it gave the best results. The resultant image is passed through a Gaussian filter with a variance of 0.5. this is followed by binarization with a threshold of 0.35. A number of morphological operations are applied to the binarized image to remove further noise and link broken segments. The final step is skeletonization which reduces the detected nerves to one-pixel wide segments. Further details on the image segmentation process are described in [26].
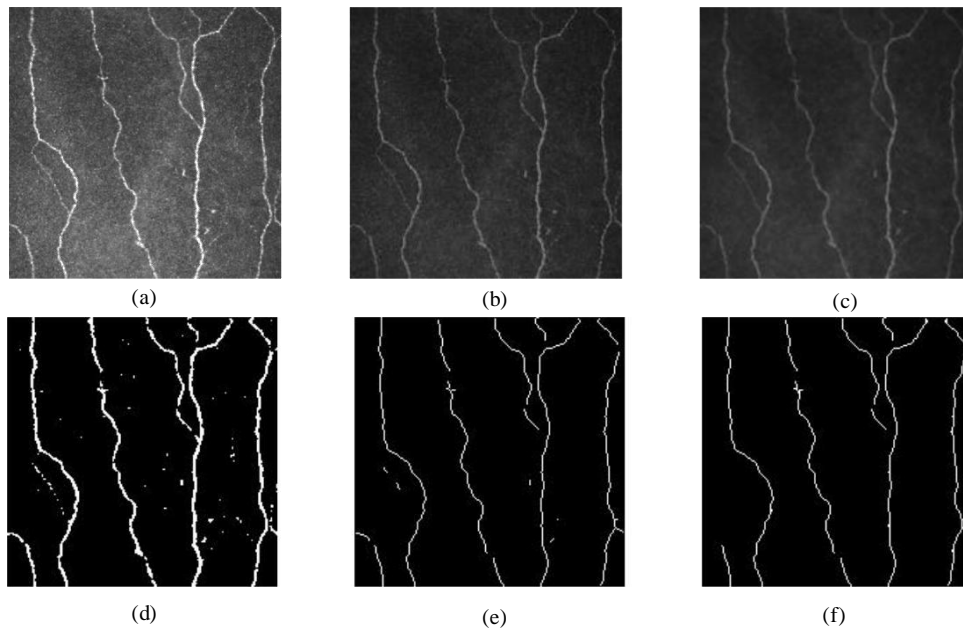

(a) (b) (c)

(d) (e) (f)

Fig. 1. Nerve segmentation outputs (a) original CCM image, (b) discrete wavelet transform, (c) coherence and gaussian filter output, (d) binary image, (e) skeleton image, (f) final segmented image

3.3 Feature Extraction

In this step, features representing the image are extracted from the binary segmented nerve image. We extracted three features for each image:
1) Total nerve fiber length (NFL) calculated as summation of all nerve pixels,
2) Entropy of the image, and
3) Area occupied by the nerves.

3.4 Machine Learning

This section briefly explains the machine learning algorithms used in this study.

3.4.1 Adaptive Neuro Fuzzy Inference System

ANFIS is a machine learning algorithm which combines the learning power of fuzzy inference systems and artificial neural networks into a robust framework [27]. It consists of five layers. In the first layer, the membership functions specify the membership degree of each input variable. These membership functions are formulated during the training phase. Using these membership functions, ANFIS creates a fuzzy inference system (FIS) which map the inputs to their corresponding outputs. The inferences from the rule base are used in the second and fourth layer to adjust the firing strength of each rule. The fourth layer generates the outputs using a linear polynomial equation. The last layer concatenates all outputs into a single output.

A two-pass learning algorithm is implemented during the learning stage [28]. The forward pass consists of updating the parameters using least squares estimation to produce the output. During the backward pass, error is computed across all layers and parameter values are updated accordingly using gradient descent algorithm.

The ANFIS network builds a FIS from the three input features, mapping them to the output using the membership functions. Hence, the FIS is trained on the randomly selected training data. The architecture of the ANFIS network is displayed in Fig. 2. The figure shows only four layers because the second and third layers are displayed as one, namely the rule layer.
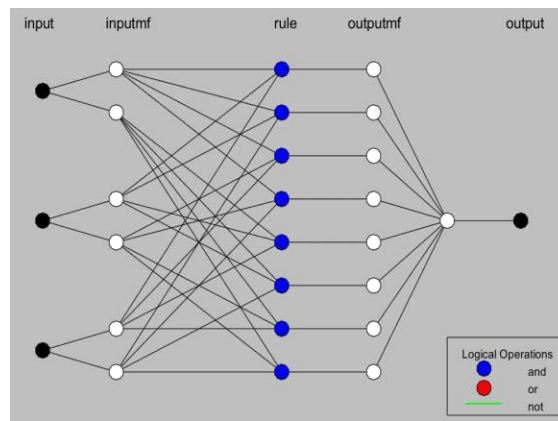

Fig. 2. Architecture of ANFIS

3.4.2 Support Vector Machine

SVMs are known as universal learners because they usually perform well in most classification problems. SVM aims to create an optimal hyperplane with maximum margin, that separates the two classes of data. The points closest to the hyperplane are called support vectors, and they determine the position of the hyperplane. Consider a set of n training samples, $x_1, \ldots, x_n$, each having a label from a set or r labels, $Y = y_1, \ldots, y_r$. The SVM classifier creates a classifier of the form [29]:

$$y(x) = sign\left[\sum_{i=1}^{n} \alpha_i y_i \psi(x, x_i) + b\right],$$ (1)

Where α_i belongs to a set of real constants, b is the bias and ψ is a kernel function. Commonly used kernel functions are: linear $x_i^T x$, polynomial with degree d $\left(x_i^T x + 1\right)^d$, and radial basis function $exp\left(-\left|x - x_i\right|_2^2 \Big/ \sigma^2\right)$.

The lines that separate the data are defined by:

$$\omega^T \varphi(x_i) + b \geq 1, \text{ for } y_i = 1$$ (2)

$$\omega^T \varphi(x_i) + b \leq -1, \text{ for } y_i = -1$$ (3)

This is equivalent to the non-linear function:

$$y_i\left[\omega^T \varphi(x_i) + b\right] \geq 1, \text{ for } i = 1, \ldots, n$$ (4)

This maps the input data to a high dimensional space and finds the hyperplane that perfectly separates the classes.

3.4.3 Naïve Bayes

Naïve Bayes is a simple probabilistic classification algorithm that classifies based on the likelihood of occurrence [30]. It assumes that features are independent given the class. During training, probabilities are calculated for each feature value given a class label. These probabilities are used to predict the label of a test sample.

Consider a feature vector, $X = \left(X_1, \ldots, X_n \right)$, where each feature value is taken from a distribution D_i. The set omega contains all feature vectors: $\Omega = D_1 \times \ldots \times D_n$. Let C be the class label of an example.

The class posterior probabilities given a feature vector can be defined as a discriminant function: $f_i^*(x) = P(C = i | X = x)$. This can be rewritten after applying Bayes rule:

$$P(C = i | X = x) = \frac{P(X = x | C = i)\, P(C = i)}{P(X = x)} \qquad (5)$$

Here, $P(X = x)$ is the same for all classes and can be eliminated. Thus, Bayes discriminant functions can be written as the following: $f_i^*(x) = P(X = x | C = i)\, P(C = i)$, where $P(X = x | C = i)$ is termed as the class-conditional probability distribution.

Finally, the Bayes classifier can be defined as:

$$h^*(x) = argmax\, P(X = x | C = i)\, P(C = i) \qquad (6)$$

$h^*(x)$ finds the maximum a posteriori probability for any example x. Extending this to simplified naïve Bayes assumption that features are independent given class, we get the following form:

$$f_i^{NB}(x) = \prod_{j=1}^{n} P\left(X_j = x_j | C = i \right) P(C = i) \qquad (7)$$

3.4.4 K-Nearest Neighbors

One of the classical and simplest nonparametric classification algorithms is the k-nearest neighbor (KNN) classifier, which classifies new examples based on nearest sample observation. It is based on the assumption that when feature vectors for training data points are projected into a subspace, any new data point can be classified based on its proximity to its k nearest neighbors [31].

Consider a set of n training samples, $x_1, \ldots, x_n$, each having a label from a set or r labels, $Y = y_1, \ldots, y_r$, and m features. The feature vector for x_i is represented as $x_{im}, \ldots, x_{nm}$. A new sample s is assigned label y_i if a majority of k nearest neighbors of s possess the label y_i.

Nearness can be measured using any of the several distance measures. The most common ones are Euclidean distance (L2 norm), Manhattan distance (L1 norm) or Max norm.

The Euclidean distance between two samples x_a and x_b is defined as:

$$D\left(x_a, x_b \right) = \sqrt{\left(x_{a1} - x_{b1} \right)^2 + \ldots + \left(x_{am} - x_{bm} \right)^2} \qquad (8)$$

The number of nearest neighbors in the neighborhood, k, is usually tuned as a hyperparameter. Empirically, as k increases, the accuracy of the prediction decreases.

Several variations of KNN exist in the literature. Weighted KNN adds weight to the vote of each label in the neighborhood based on its distance from the test sample [32]. Epsilon-ball KNN is a method that selects neighbors within a distance from the test sample.

3.4.5 Classification Trees

Classification trees split the training data into partitions, based on mapping of inputs to the outputs. Thus, by creating partitions it learns the different patterns occurring in the data. Commonly used split criteria include gini index, information gain and entropy. Partitioning the data results in the creation of a tree, where the root of the tree is one feature value, and subsequent nodes are other feature values. Each level contains feature values corresponding to one feature. The leaf nodes predict the class of a given sample. As the tree goes deeper, the learning represents overfitting. Consequently, pruning the tree to a certain depth is a tunable hyperparameter.

3.4.6 Linear Discriminant Analysis

A concept similar to Linear Discriminant Analysis is Principal Component Analysis, which is mainly used for dimensionality reduction by creating new dimensions representing the original ones. LDA works in a similar way, by creating new dimensions from pairs of original dimensions. It aims to maximize the distance between the means of the

two categories and minimize the variation within each category. The class of a test sample is predicted using Bayes' Theorem as explained in Section 3.4.3.

4. Experiments and Results

4.1 Experimental Setting

All implementation was done using MATLAB. In the preprocessing stage, data are normalized by dividing each value by the maximum of its column. Data is randomly selected for training and testing using a ratio of 3:2 for train and test sets. For generating an initial FIS, the grid partitioning method is used. ANFIS is trained on the initial FIS for 30 training epochs. The trained FIS is used to calculate the training error and trained repeatedly to generate the best FIS. The FIS with the highest accuracy is selected as the best one.

Besides ANFIS, classifiers are also trained using Support Vector Machines (SVM), Naïve Baiyes (NB), linear discriminant analysis (LDA), classification trees (Tree) and k-nearest neighbors (KNN). For SVM, a linear kernel with a scale of 1 was used. Sequential momentum optimizer (SMO) was used as the optimization method which uses second order polynomial information to speedup convergence. The model identified 42 support vectors from the training data. The kernel smoothing type for NB was tuned to be Gaussian which is defined by the following formula:

$$f(x) = \frac{1}{\sqrt{2\pi}} exp(-0.5x^2) \tag{9}$$

The number of nearest neighbors in KNN was set to 5 and Euclidean distance was used to determine nearness between the samples. The same train and test subsets are used for all. For all models, hyperparameters were optimized to give the best results.

4.2 Performance Measures

The following performance measures are used in this study to evaluate the results of the ANFIS classifier:

In this study, the performance measures of accuracy, precision, recall and macro F1 were used to evaluate the performance. Accuracy is defined as TP+TN/TP+TN+FP+FN, precision as TP/TP+FP and recall as TP/TP+FN, where TP stands for true positives, FP for false positives, TN for true negatives and FN for false negatives. Macro F1 is the harmonic mean of precision and recall.

4.3 Results and Discussion

The predicted values by ANFIS are scaled to confine between the range [0,1] using the following equation.

$$ys_i = \left| \frac{y_i}{max(|y_i|)} \right| \tag{10}$$

where y_i is the predicted ANFIS output for the ith data point. Fig. 3 compares the actual output and the predicted output by ANFIS on the testing data. Tuples (data points) from the testing data are represented by the x-axis (84 tuples). The y-axis shows the actual and scaled predicted output for each tuple. For example, for the second data point, the actual output, y_3, is 1, and the scaled predicted output, ys_3, is 0.98 (almost 1).

After calculating the scaled predicted output, the most satisfying range for each class is selected. The highest accuracy for this experiment is acquired given by the rule:

IF $ys_i > 0.8$ THEN $ys_i = 1$ ELSE $ys_i = 0$,

where class 'normal'=0, 'abnormal'=1, and the cutoff point is 0.5. Hence, the ranges for class 1 and class 0 are (0.5,1] and [0,0.5] respectively. The cutoff line is shown in Fig. 4 as a dotted line where Y=0.5. For example, for the first data point, the actual output, y_2=1, and it coincides with the predicted output, ys_2. Since the predicted output is greater than 0.5, thus, according to the rule, ys_2=1, which makes it a true prediction.

Table 1 presents the results from all 6 algorithms. For all six classifiers, the performance is more or less the same. It can be observed that NB, KNN, LDA and SVM achieved a better accuracy as compared to ANFIS, whereas classification trees are worse. However, we hypothesize for future experiments that as we move beyond binary classification, the performance of ANFIS will dominate others because data will get complicated. Moreover, KNN was able to detect the highest of number of neuropathy images. Since the train subset only contains 213 samples, the prediction time of KNN is tolerable. However, as the training data increases, which is expected with further detailed studies, the prediction time of KNN will eventually increase. This is due to the known fact that during the testing phase, KNN compares the distance of the test sample with all training samples to get the minimum distance label. Fig. 4 shows

the decision surfaces formed by all classifiers except ANFIS. Since it is a 2d plot, two features were considered for plotting the surface. The red region shows neuropathy while the green shows normal.
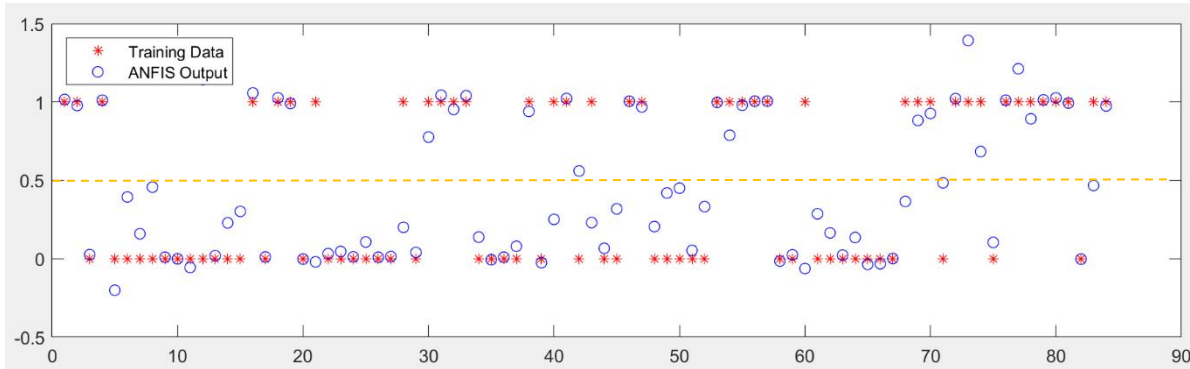

Fig. 3. ANFIS Predicted and Actual Outputs Plot. Cut-off point = 0.5

Table 1 - Results

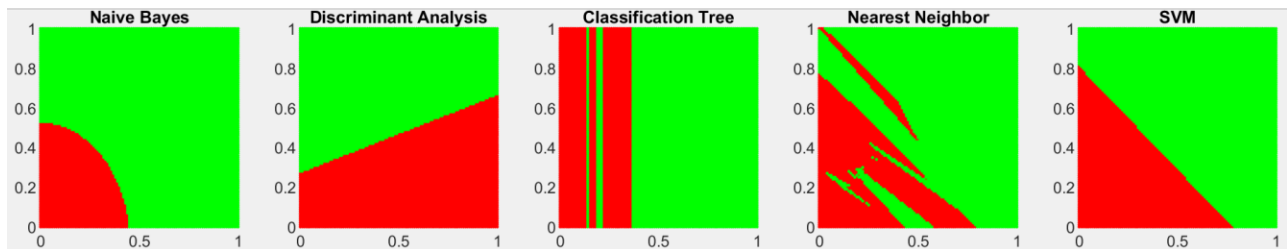| Machine learning algorithm	TP	TN	FP	FN	Accuracy	Precision	Recall	MacroF1
SVM	44	32	6	2	0.90	0.88	0.96	0.92
NB	44	32	6	2	0.90	0.88	0.96	0.92
KNN	44	33	5	2	0.92	0.90	0.96	0.93
LDA	43	33	5	3	0.90	0.90	0.93	0.91
Tree	39	32	6	7	0.85	0.87	0.85	0.86
ANFIS	44	29	2	9	0.87	0.96	0.83	0.89

Fig. 4 Decision surfaces formed by the classifiers

Since the classification relies heavily on the extraction of features from the images, which is preceded by the nerve segmentation procedure, classification accuracy can be increased by improving the segmentation output and extracting the right features. Performance can also be improved by extracting different kinds of features, and by combining clinical features with the hand-crafted image features.

5. Conclusion

We have presented an evaluation of six machine learning algorithms for the problem of neuropathy classification of corneal nerve images. In our experiments, we achieved the highest accuracy of 0.92 using the k-nearest neighbors algorithm. The automated process of classification solves the limitations posed by the current manual process. Further evaluations on other datasets are required to achieve better results. In future, we plan to continue our experiments and widen our research in this domain by extracting different kinds of features, incorporating deep learning, and observing the results on different kinds of datasets.

References

[1]   C. Quattrini *et al.*, "Surrogate markers of small fiber damage in human diabetic neuropathy," *Diabetes*, 2007.
[2]   M. A. Dabbah, J. Graham, R. Malik, and N. Efron, "Detecting and Analyzing Linear Structures in Biomedical Images: A Case Study Using Corneal Nerve Fibers," in *Medical Image Processing*, 2011.
[3]   N. Efron, "Corneal Confocal Microscopy Is Emerging as a Powerful Diagnostic Tool for Assessing Systemic

Neurologic Disease," *Invest. Ophthalmol. Vis. Sci.*, vol. 58, no. 9, pp. 3682–3682, 2017.

[4]   P. Hossain, A. Sachdev, and R. A. Malik, "Early detection of diabetic peripheral neuropathy with corneal confocal microscopy," *Lancet*. 2005.

[5]   M. Brines *et al.*, "Corneal nerve fiber size adds utility to the diagnosis and assessment of therapeutic response in patients with small fiber neuropathy," *Sci. Rep.*, 2018.

[6]   P. J. Podgorny, O. Suchowersky, K. G. Romanchuk, and T. E. Feasby, "Evidence for small fiber neuropathy in early Parkinson's disease," *Park. Relat. Disord.*, 2016.

[7]   J. Mikolajczak *et al.*, "Patients with multiple sclerosis demonstrate reduced subbasal corneal nerve fibre density," *Mult. Scler.*, 2017.

[8]   R. Shetty, R. Deshmukh, R. Shroff, C. Dedhiya, and C. Jayadev, "Subbasal nerve plexus changes in chronic migraine," *Cornea*, 2018.

[9]   M. Ferdousi *et al.*, "Corneal confocal microscopy detects small fibre neuropathy in patients with upper gastrointestinal cancer and nerve regeneration in chemotherapy induced peripheral neuropathy," *PLoS One*, 2015.

[10]   H. I. Kemp *et al.*, "Use of corneal confocal microscopy to evaluate small nerve fibers in patients with human immunodeficiency virus," *JAMA Ophthalmol.*, 2017.

[11]   A. Khan *et al.*, "Corneal confocal microscopy detects corneal nerve damage in patients admitted with acute ischemic stroke," *Stroke*, 2017.

[12]   S. Al-Fahdawi *et al.*, "A fully automatic nerve segmentation and morphometric parameter quantification system for early diagnosis of diabetic neuropathy in corneal images," *Comput. Methods Programs Biomed.*, 2016.

[13]   R. A. Dabbah, M. A., Graham, J., Tavakoli, M., Petropoulos, Y., & Malik, "Nerve fibre extraction in confocal corneal microscopy images for human diabetic neuropathy detection using gabor filters," *Med. Image Underst. Anal.*, vol. 254–258, 2009.

[14]   E. Poletti and A. Ruggeri, "Automatic nerve tracking in confocal images of corneal subbasal epithelium," in *Computer-Based Medical Systems (CBMS), 2013 IEEE 26th International Symposium*, 2013, pp. 119–124.

[15]   A. Grisan, E., Pesce, A., Giani, A., Foracchia, M., & Ruggeri, "A new tracking system for the robust extraction of retinal vessel structure," in *Engineering in Medicine and Biology Society, 2004. IEMBS'04. 26th Annual International Conference of the IEEE*, p. Vol. 1, pp. 1620–1623.

[16]   F. Scarpa, E. Grisan, and A. Ruggeri, "Automatic recognition of corneal nerve structures in images from confocal microscopy," *Investig. Ophthalmol. Vis. Sci.*, 2008.

[17]   A. Ruggeri, F. Scarpa, and E. Grisan, "Analysis of corneal images for the recognition of nerve structures," in *Engineering in Medicine and Biology Society, 2006. EMBS'06. 28th Annual International Conference of the IEEE*, 2006, vol. 4739–4742.

[18]   A. Colonna, F. Scarpa, and A. Ruggeri, "Segmentation of Corneal Nerves Using a U-Net-Based Convolutional Neural Network," 2018, pp. 185–192.

[19]   M. A. Dabbah, J. Graham, I. Petropoulos, M. Tavakoli, and R. A. Malik, "Dual-model automatic detection of nerve-fibres in corneal confocal microscopy images," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2010.

[20]   R. N. Dixon and C. J. Taylor, "Automated Asbestos Fiber Counting," in *1979 Inst. Physics Conference*, 1979.

[21]   M. A. Dabbah, J. Graham, I. N. Petropoulos, M. Tavakoli, and R. A. Malik, "Automatic analysis of diabetic peripheral neuropathy using multi-scale quantitative morphology of nerve fibres in corneal confocal microscopy imaging," *Med. Image Anal.*, 2011.

[22]   P. Guimaraes, J. Wigdahl, E. Poletti, and A. Ruggeri, "A fully-automatic fast segmentation of the sub-basal layer nerves in corneal images," in *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBC 2014*, 2014.

[23]   O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2015.

[24]   P. Guimarães, J. Wigdahl, and A. Ruggeri, "A Fast and Efficient Technique for the Automatic Tracing of Corneal Nerves in Confocal Microscopy," *Transl. Vis. Sci. Technol.*, 2016.

[25]   I. Otel *et al.*, "Diabetic peripheral neuropathy assessment through corneal nerve morphometry," in *Bioengineering (ENBENG), 2013 IEEE 3rd Portuguese Meeting*, 2013.

[26]   T. Salahuddin and U. Qidwai, "Neuro-Fuzzy Classifier for Corneal Nerve Images," in *IEEE-EMBS Conference on Biomedical Engineering and Sciences (IECBES)*, 2018, pp. 131–136.

[27]   J. S. R. Jang, "ANFIS: Adaptive-Network-Based Fuzzy Inference System," *IEEE Trans. Syst. Man Cybern.*, 1993.

[28]   N. Talpur, M. N. M. Salleh, and K. Hussain, "An investigation of membership functions on performance of ANFIS for solving classification problems," in *IOP Conference Series: Materials Science and Engineering*, 2017.

[29]   J. A. K. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Process. Lett.*,

1999.

[30] I. Rish, "An empirical study of the naive Bayes classifier," in *International Joint Conferences on Artificial Intelligence 2001 Workshop on Empirical Methods in Artificial Intelligence*, 2001.

[31] T. M. Cover and P. E. Hart, "Nearest Neighbor Pattern Classification," *IEEE Trans. Inf. Theory*, 1967.

[32] S. A. Dudani, "The Distance-Weighted k-Nearest-Neighbor Rule," *IEEE Trans. Syst. Man Cybern.*, 1976.