Irish Sign Language Recognition Using Principal Component Analysis and Convolutional Neural Networks

Marlon Oliveira
School of Computing
Dublin City University, Ireland
Email: marlon.oliveira2@mail.dcu.ie

Houssem Chatbri
Insight Centre for Data Analytics
Dublin City University, Ireland
Email: houssem.chatbri@dcu.ie

Suzanne Little
Insight Centre for Data Analytics
Dublin City University, Ireland
Email: suzanne.little@dcu.ie

Ylva Ferstl
ADAPT Centre
Trinity College Dublin, Ireland
Email: yferstl@tcd.ie

Noel E. O'Connor
Insight Centre for Data Analytics
Dublin City University, Ireland
Email: noel.oconnor@dcu.ie

Alistair Sutherland
School of Computing
Dublin City University, Ireland
Email: alistair.sutherland@dcu.ie

***Abstract*—Hand-shape recognition is an important problem in computer vision with significant societal impact. In this work, we introduce a new image dataset for Irish Sign Language (ISL) recognition and we compare between two recognition approaches. The dataset was collected by filming human subjects performing ISL hand-shapes and movements. Then, we extracted frames from the videos. This produced a total of 52,688 images for the 23 common hand-shapes from ISL. Afterwards, we filter the redundant images with an iterative image selection process that selects the images which keep the dataset diverse. For classification, we use Principal Component Analysis (PCA) with with K-Nearest Neighbours (k-NN) and Convolutional Neural Networks (CNN). We obtain a recognition accuracy of 0.95 for our PCA model and 0.99 for our CNN model. We show that image blurring improves PCA results to 0.98. In addition, we compare times for classification.**

I. Introduction

Hand-shape recognition is a very important area in Computer Vision (CV). It has been studied for years and we still do not have a good enough implementation using only a regular camera as input, without sensors or multiple cameras. Human Computer Interaction depends strongly on the new developments in CV and hand-shape recognition is a very active research area in this field. Irish Sign Language (ISL) is known to be used by around 6,500 deaf people in the island of Ireland. ISL is not based on English or Irish; it is a different language. In addition, it is estimated that ISL is known by some another 50,000 non-deaf people [8].

Earlier works in this area have used rather smaller datasets. For instance, Farouk *et al.* proposed two ISL datasets of relatively limited size [1]. The first dataset is composed of 920 computer generated images that are produced by the Poser software by SmithMicro. The second dataset is composed of 1620 real hand images. Both datasets represent 20 ISL hand-shapes. We increase the number of real hand images from 1620 to 52,628 and we use 6 different human subjects. We add an additional 3 shapes to the 20 used by Farouk.

The images show the hand and arm of a signer against a uniform black background. As for recognition, it was achieved using handcrafted features as well as data-driven models (Sec. II).

In this paper, we are reporting a comparison of shallow features using PCA with k-NN and deep features using Convolutional Neural Networks (CNN) with Softmax on a dataset of Irish hand-shape images that we collected. PCA is considered an efficient method for dimensionality reduction and feature extraction [3]. It uses the covariance matrix of the data to create a space known as an eigenspace. Each dimension in the space is known as an eigenvector. The number of eigenvectors required to represent the full data is considerably lower than the dimensionality of the original data. CNNs are multi-layered neural networks (NNs) specialised on recognising patterns directly from images. They are widely known for robustness to distortion and having minimal or no preprocessing. They have been used for detection and recognition of different objects, including hands [10], [7].

Our main contributions in this work are twofold:

- We propose a public image dataset for ISL containing more than 50K images. This is done by recording subjects performing ISL hand-shapes and designing an iterative process to select the images that keep the dataset diverse by removing redundant frames (Sec. III).
- We show that our dataset can be used to successfully train two different classifiers. We report experiments with Principal Component Analysis (PCA) and Convolutional Neural Network (CNN) (Sec. IV). We also show that the performance of PCA can be improved by using image blurring as preprocessing (Sec. VI).

II. Related work

Hand-shape recognition has been achieved using hand-crafted features and techniques including Hidden Markov Models, Fourier Analysis, Points of Interest, Principal Component Analysis (PCA), Orientation Histograms and Kalman Filters [14], [9]. In [12], a PCA-based method has been presented. It was shown that it is possible to interpolate data created by the projection of the images into a PCA space. This interpolation was made in order to improve the accuracy of recognising a hand-shape at an unknown rotation angle. In addition, it is possible to interpolate eigenspaces, creating artificial ones [11].

A hierarchical multistage approach was presented in [2], [13]. Here, the authors described an algorithm using a multi-stage hierarchy to build a pyramid which consists of different eigenspaces at the different levels, to analyse a new incoming pattern and classify it to the nearest neighbour pattern from a set of example images. Image blurring was used to reduce the small changes between objects and to linearise the manifolds. However, blurring was only applied to computer-generated images.

[17] proposed a system for recognising American Sign Language finger-spelling using Local Binary Patterns and Geometric Features. The recognition part was done by a Support Vector Machine with a Kinect depth sensor. It was tested on two datasets and obtained an accuracy rate over 90%.

In [10], Nagi *et al.* reported a method for hand gesture classification using a CNN with three convolutional layers, two Max Pooling layers, and one fully connected layer. They collected hand gesture images of six classes corresponding to finger counts from zero to five, and and a colour-divided glove was used for segmentation and hand area extraction before classification. They compared their approach with engineered features (e.g. PHOG, FFT) classified with SVMs, and obtained improved results.

[4] proposed a vision-based hand gesture recognition system for intelligent vehicles. The long-term recurrent convolution network was used to classify the video sequences of hand gestures. John *et al.* ran around 7500 iteration in the deep learning classifier to classify 9 classes of gestures with a gesture classification around 91% accuracy.

III. Image dataset

In this work we use the Irish Sign Language alphabet which is composed of 23 static gestures. Figure 1 shows cropped images of the dataset.

A. The Irish Sign Language hand-shape dataset (ISL-HS)

The Irish Sign Language hand-shape dataset (ISL-HS) contains only real hand images. The ISL alphabet is composed of 23 static gestures (corresponding to characters from the English alphabet apart from J, X and Z) and 3 dynamic gestures (J, X and Z) [6]. However, in this work we used only the static gestures. The typology is one-handed finger spelling.

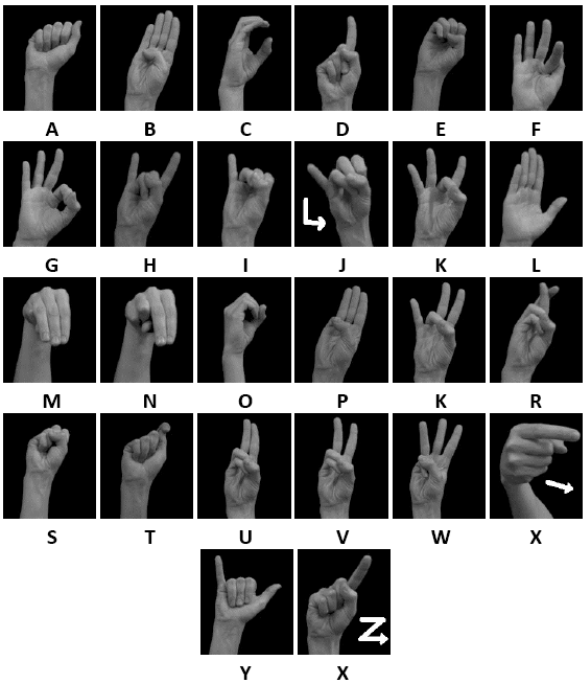

Fig. 1: Irish Sign Language alphabet.

In order to collect images for the dataset we recorded short videos. We asked 6 people (3 males and 3 females) to perform the finger spelling ISL alphabet. Each shape was recorded 3 times. Each of the 23 static gestures was performed by moving the arm from the vertical to the horizontal position. This is to include rotation as a variation in order to train classifiers to be robust to the sign rotation angle. However, for the dynamic gestures there was no arm rotation, only the motion of the gesture itself (the movement of the shape, without forcing an arm rotation).

The videos were converted into frames. Frames were converted to grayscale and the background was removed from the frames, using a pixel value threshold. Finally the frames contained only the arm and the hand.

The number of frames for each video depends on the length of the video. Videos were recorded at 30 frames per second (fps) and a resolution of 640x480 pixels. In total 468 videos were recorded. From these videos we obtained a total of 52,688 frames for the static shapes and 5,426 frames for dynamic gestures. In total 58,114 frames were obtained. However, for this work we used only the static-shape images. Note that some of these frames are naturally blurred because of the arm movement.

B. Redundant frame filtering

Since we extract all frames from the videos we filmed, numerous frames are similar due to the speed variation of the subjects. For this reason, we design a method to filter redundant frames (i.e. frames with insignificant difference). Our method works as follows: Each image I_u of the original dataset is represented with a compact feature vector $\overrightarrow{V_u}$ (Sec.

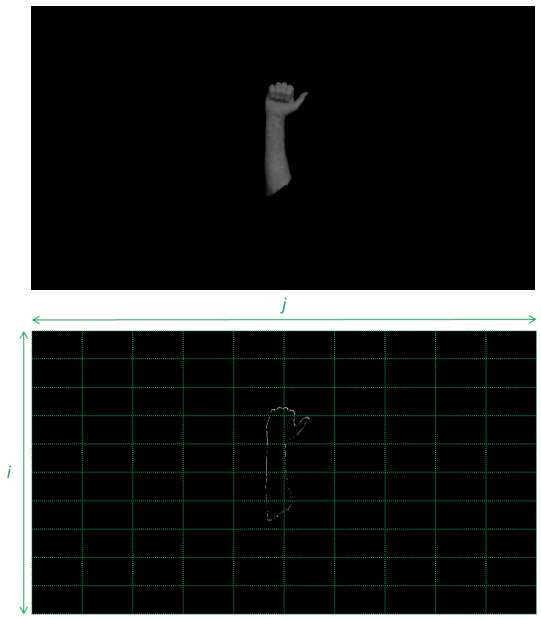

Fig. 2: A sample image from the ISL-HS dataset (a) and the feature extraction grid (b).

III-B1). Then, a *diversity score* is introduced to express image heterogeneity, and images are iteratively selected so to optimise the *diversity score* (Sec. III-B2).

1) Feature extraction: From each image I_u of the original dataset, a feature vector $\overrightarrow{V_u}$ is extracted by splitting the image into a $K \times K$ grid and calculating the number of foreground pixels in each cell of the grid after edge detection with the following 3×3 convolutional kernel (Figure 2):

$$\phi = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \quad (1)$$

Therefore, the feature vector $\overrightarrow{V_u}$ for an image I_u is a 2D histogram and it is expressed as follows:

$$\overrightarrow{V_u} = (V_{i,j}), 0 \leq i, j \leq K \quad (2)$$

where $V_{i,j}$ denotes the number of foreground pixels in bin (i, j) divided by the total number of foreground points.

Then, computing the dissimilarity between two images I_u and I_v is done by accumulating the distances between the histogram bins of their feature vectors $\overrightarrow{V_u}$ and $\overrightarrow{V_v}$ as follows:

$$d = \frac{1}{K^2} \sum_{i=0}^{K-1} \sum_{j=0}^{K-1} (V_{i,j}^u - V_{i,j}^v)^2 \quad (3)$$

2) Iterative image selection using the diversity score: After extracting a feature vector $\overrightarrow{V_u}$ from each image I_u in the original dataset, an iterative selection process is performed to select images based on a *diversity score* that is defined as follows:

$$\gamma(I, A) = \frac{1}{|A|} \sum_{i=0}^{|A|-1} |\overrightarrow{V} - \overrightarrow{V_i^A}| \quad (4)$$

Algorithm 1 Diversity-based image selection

Precondition: Original dataset D_{orig} : containing all N images

function DIVERSITYBASEDSELECTION(D_{All})
  A : final dataset, initially empty
  I : one frame selected randomly from D_{All}
  $A \leftarrow \{I\}$
  for $i \leftarrow 1$ to N **do**
    $i_{max} \leftarrow \underset{0 \leq i \leq |D_{orig}|-1}{\operatorname{argmax}} \gamma(I, A)$
    $I \leftarrow I_{i_{max}}$
  end for
end function

where A is an image (initially empty), I is an image from the original dataset and does not belong to A, $\overrightarrow{V}$ and $\overrightarrow{V_i^A}$ are the feature vectors of I and I_i^A respectively.

The iterative process is illustrated in Algorithm 1. First, an image is selected randomly from the original image dataset D_{orig} and put into the final dataset A. Then, an iterative process selects the image from D_{orig} that has the largest *diversity score* (i.e. the image that is the most different to images of A), where *diversity score* is calculated with Eq. 4. When this process finishes, we plot the curve of *image diversity score* at each selected image (Figure 3).

In order to perform this process in a reasonable time, we tweak it by applying the iterative image selection from a subset of D_{orig} with a limited size equal to 100 images selected randomly, instead of the whole D_{orig}. During the process, we observed that this does not lead to a selection bias, but makes the process achievable in a reasonable time.

After plotting the curve of Figure 3 which shows the plot of the curve of *image diversity score* at each selected image, the first 50,000 images are selected (i.e. roughly before the sharp decrease in *diversity score*). We take this subset as the final dataset for subsequent experiments. Figure 4 shows the class distribution in this dataset. The dataset is divided into a training set and a testing set, by iterating through the images and assigning every image to either the training or the testing set in an alternating manner. Thus, both our training and testing dataset contains 25,000 images. The dataset is available online[1].

IV. HAND-SHAPE CLASSIFICATION

We compare two approaches. The first approach is based on Principal Component Analysis (PCA), and the second approach uses a Convolutional Neural Network (CNN).

A. PCA based approach

PCA is an efficient method for dimensionality reduction [3]. It uses the covariance matrix of the data to create a space known as an eigenspace. Each dimension in the space is represented by an eigenvector. The number of eigenvectors

[1]https://github.com/marlondcu/ISL_50k

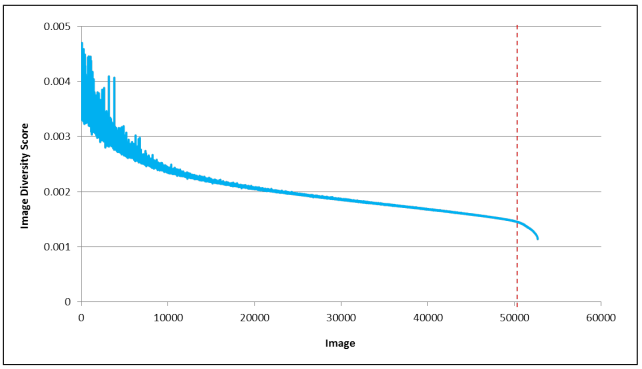

Fig. 3: The curve of image diversity score.

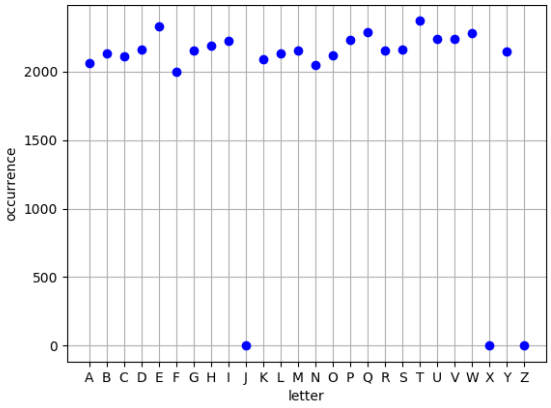

Fig. 4: Rate of occurrence by shape.

required to represent the full data is considerably lower than the dimensionality of the original data.

In order to apply PCA over our training dataset we combine all the images into the same array and then compute PCA. Each image in the dataset has 640×480 pixels, we resized them to 160×120 pixels. After vectorization every image is represented by a row array with 19,200 (pixels) entries. As a result, we have an eigenspace with 19,200 dimensions.

By projecting the images from the training set onto the most significant D_i eigenvectors, we obtain a D_i-dimensional space containing (N_{im}) points for each pose angle. Each point represents an image. In this work we tested different number of eigenvectors and how it affects the accuracy.

Images from the training dataset were blurred using a two-dimensional Gaussian kernel. In this stage, a kernel of size (36,36) was used and variance equal to 60. The use of blurring was motivated by earlier results by Farouk [1], which showed that such image filtering is beneficial for PCA accuracy.

Figure 5 shows the 2 dimensions (axes) D_1 and D_2, where each point represents one image in our training dataset.

In the same way, we project images from the testing dataset onto the eigenspace, in order to have both in the same space.

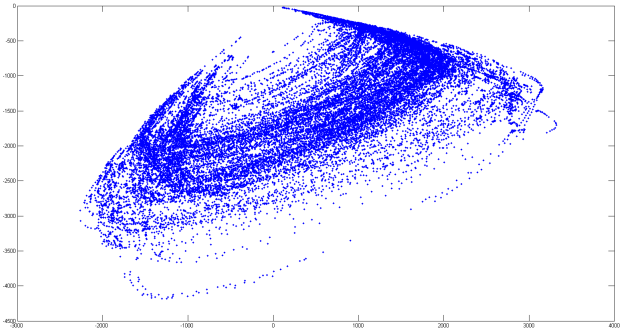

Fig. 5: Projection of training dataset images onto the PCA space in 2D.

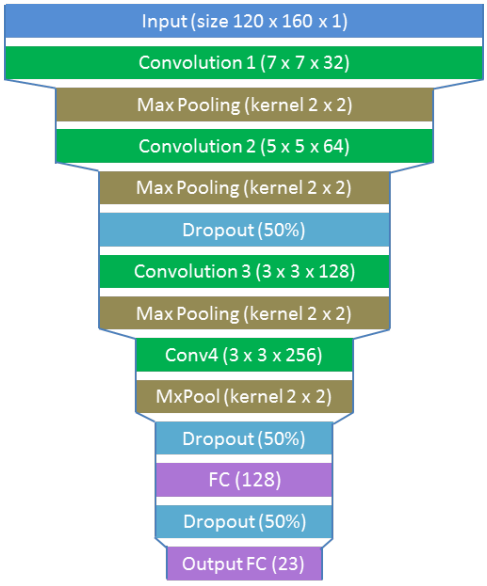

Fig. 6: Architecture of the convolutional neural network.

B. *CNN-based approach*

Convolutional Neural Networks (CNNs) are artificial neural networks that are able to automatically learn features through convolutional layers where the level of abstraction increases with each convolutional layer [7]. They are widely known for robustness to distortion and they have been applied with minimal or no preprocessing.

The architecture of our CNN model is as follows (Figure 6): It has 4 convolutional layers with ReLU non-linearity, and ends with 2 fully connected layer with 128 and 23 neurons in each layer, and Relu and Softmax non-linearity respectively. The 23 output neurons are activated correspondingly to the image class. Dropout layers are used to prevent overfitting [15], [16]. An Adadelta optimizer [18] is used with a learning rate of 1.0, and a categorical cross entropy is used as a loss function. The filter size of the convolutional layers decreases throughout the model since that has been proven to be beneficial [5].

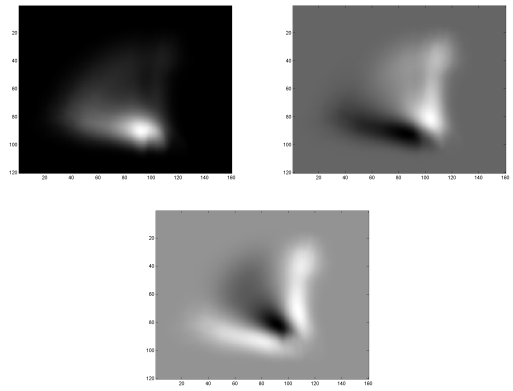

Fig. 7: Visualization of the first 3 eigenvectors for the full training dataset.

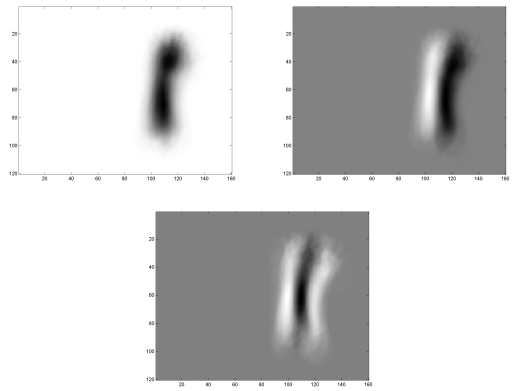

Fig. 8: Visualization of the first 3 eigenvectors for the non rotated sub-dataset.

V. Outputs of the PCA approach

PCA allows to display each eigenvector independently as an image. Each eigenvector represents some features of the set of images. Figure 7 represents the 3 first eigenvectors (in order: first, second and third) of the 25,000 training images. As we can see visually all of them represent variation in rotation. This variation makes it more difficult to identify what else the eigenvector represents.

In order to make it easier to identify what each eigenvector represents we have considered only the 9 first images of each person and each shape, in total 3,304 images were considered. This should avoid a significant variation in rotation. Thus, we plot and present images in Figure 8.

From these images, we plot a projection of all the images onto the first 3 eigenvectors (same as the Figure 8). This is illustrated in Figure 9.

Thus, we select images from each extreme, two extremes for each dimensions (3 dimensions), 6 extremes in total. We have coloured these image extremes. Note in Figure 9, yellow and green are the extremes of the first dimension; cyan and magenta for the second and black and yellow for the third. Finally we plot these images in order to analyse what the extremes mean.

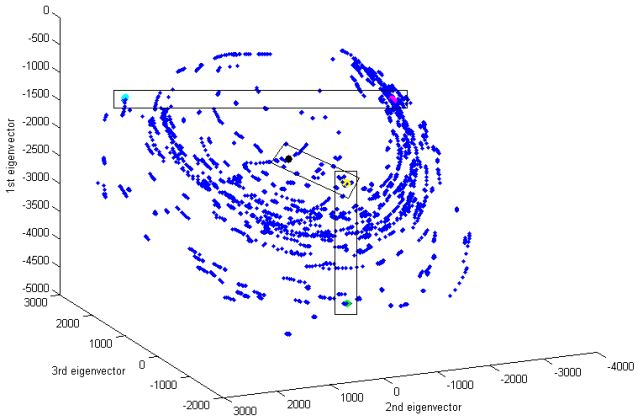

Fig. 9: Plot of the images projected in 3D (3 first eigenvectors).

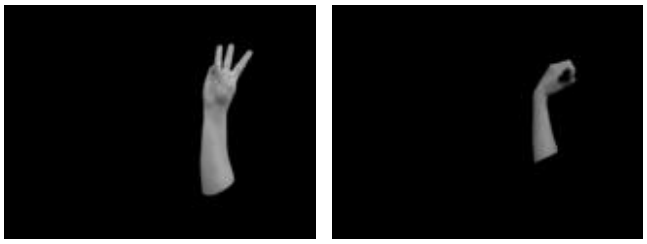

Fig. 10: Visualization of the images corresponding to the variation along the first eigenvector.

Fig. 11: Visualization of the images corresponding to the variation along the second eigenvector.

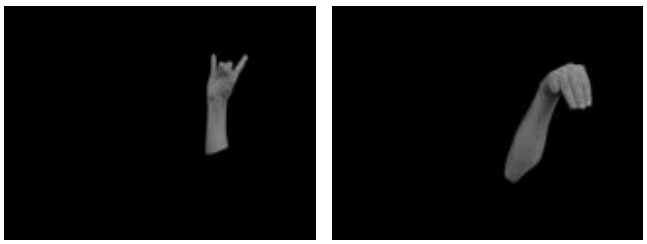

Fig. 12: Visualization of the images corresponding to the variation along the third eigenvector.

Figure 10 shows two different images at the extremes of the

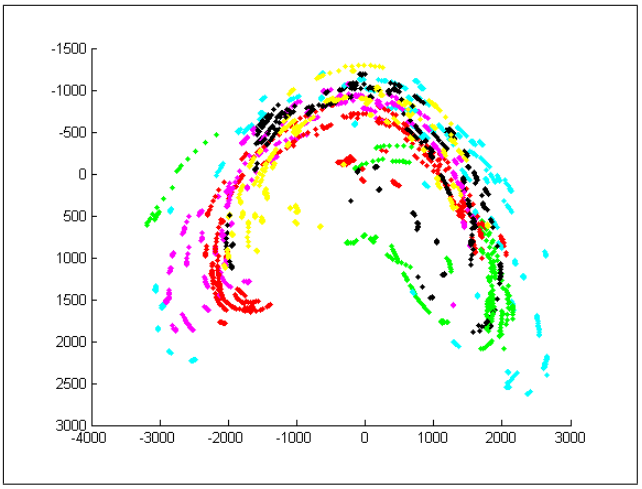

Fig. 13: Manifolds with different colours for different persons.

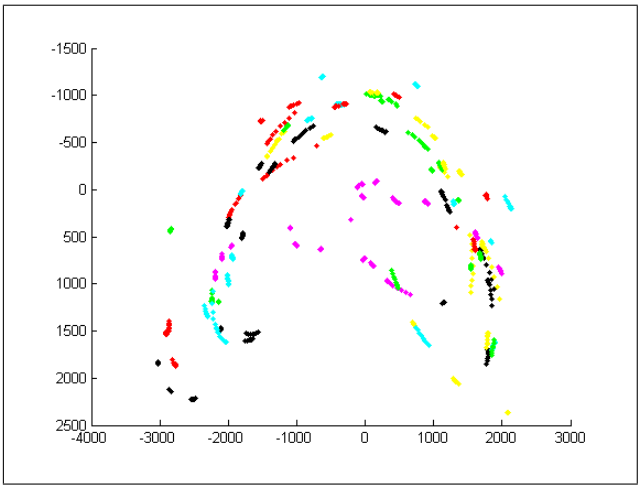

Fig. 14: Manifolds with different colours for different shapes.

first eigenvector. The image in the left is one of the yellow dots of the plot in Figure 9 and the left image, the green dots. In a similar way, images in Figure 11 refer to cyan (left image) and magenta (right image). Finally, Figure 12 show the left image corresponding to yellow dots and the right image corresponds to black dots of the plot in Figure 9.

We can infer from Figures 10, 11 and 12 that the first eigenvector is concerned with size of the hand/arm and illumination, the second mostly about translation and the third about shape and size.

In order to understand how the manifolds behave according to different shapes and different persons we have changed the colour of some points in the plot. Figure 13 shows the manifolds, with different colours for different persons.

Figure 14 shows the manifolds, with different colour for different hand-shapes, being A = yellow, G = red, M = magenta, R = cyan, W = green and Y = black.

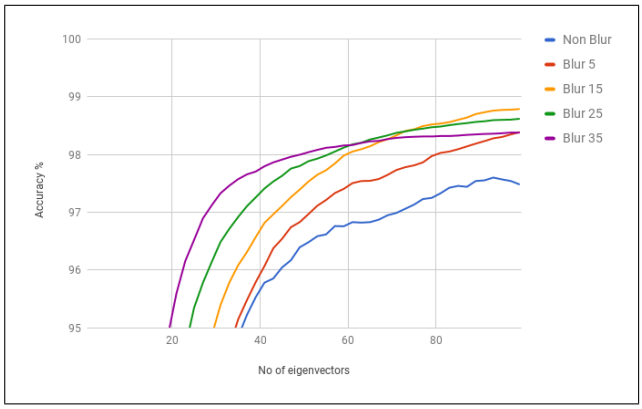

Fig. 15: Accuracy according to the blurring level.

VI. Experimental results

In this section, we report results of comparing our PCA and CNN models on the Irish Sign Language dataset. Evaluation is reported in terms of *Recognition Accuracy*, which is defined as follows:

$$Recognition\ Accuracy = \frac{1}{N_h}\sum_{N_h=1}^{N_h}(1-\frac{|\overrightarrow{y_{true}}-\overrightarrow{y_{predicted}}|}{N_h}) \quad (5)$$

where $\overrightarrow{y_{true}}$ and $\overrightarrow{y_{predicted}}$ refer to the ground truth and predicted outputs respectively.

Experiments were made in Python 3.5, and Keras 2.0.6, running on Windows 7, on a CPU with 16GB RAM.

In addition, we carried out tests with blurred images. Images were blurred using a two-dimensional Gaussian kernel. Blur was applied with different kernel sizes.

A. Classification with PCA and k-NN

In order to classify the correct hand-shape we used the k-NN algorithm, with $k = 1$ and Euclidean distance. We projected each testing image into the training dataset eigenspace and classified according to the nearest point (shortest Euclidean distance).

The accuracy in recognising the correct sign strongly depends on the number of the eigenvectors (dimensions) considered. For example, for $D_i = 15$, accuracy is 0.817. When using more eigenvectors the accuracy increases as well. e.g. for $D_i = 60$ we obtained 0.914.

Figure 18 shows the accuracy according to the number of eigenvectors. It is possible to identify that a plateau starts around the 40th eigenvector and above. This means that there is no need to consider a large (i.e. > 40) number of eigenvectors in the recognition process.

Figure 15 shows the difference in accuracy for blurred and non blurred images according to the blurring level. Note that blurring increased the accuracy and decreased the number of eigenvectors. In addition, we can note an optimal result in the blurring with kernel size $(15, 15)$.

Fig. 16: Training and testing curves of the CNN model.

B. Classification using CNN

For the CNN model, the images were used without any pre-processing or resizing, apart from dividing the pixel intensities on 255 for normalization.

Figure 16 shows the performance progress of the model during the training. The model starts to improve in the first 65 iterations, after which the testing accuracy exceeds 0.9. After 104 iterations, the testing accuracy exceeds 0.99. The best testing performance is at iteration 130 where the accuracy is 0.99875.

Figure 17 illustrates the responses of the model's intermediate convolutional layers. It can be seen that the model learns filters that respond to the salient part of the hand, that is the fingers, which results in brighter pixel values in the heat map.

We can infer from Fig. 17 that the first layer is sensitive to the holistic shape boundary and shades, while the second layer is more sensitive to the hand and fingers. As the image size gets smaller at each stage it is difficult for visualise the information in those layers.

Finally, we tested CNN with blurred images. The kernel size chosen was $(15, 15)$ because for PCA it has shown the optimum accuracy curve. It's possible that CNNs carry out their own blurring on the images. Blurring is a convolution operation and it showed an insignificant change in testing accuracy for CNN, proving that pre-processing is not important for this technique.

C. Analogy between PCA and CNN

Even understanding that PCA and CNN are different approaches, we can infer both have some similarities. PCA uses eigenvectors and CNN uses layers and iterations. More eigenvectors for PCA results in preserving more information from the original data meanwhile more iterations for CNN results in presenting more batches of data for training and adjusting the weights.

Figure 18 shows a comparison of accuracy of PCA and CNN. The y-axis show the accuracy in % and the x-axis represents the number of eigenvectors for PCA and iteration number for CNN. Note that CNN has a faster and improved accuracy.

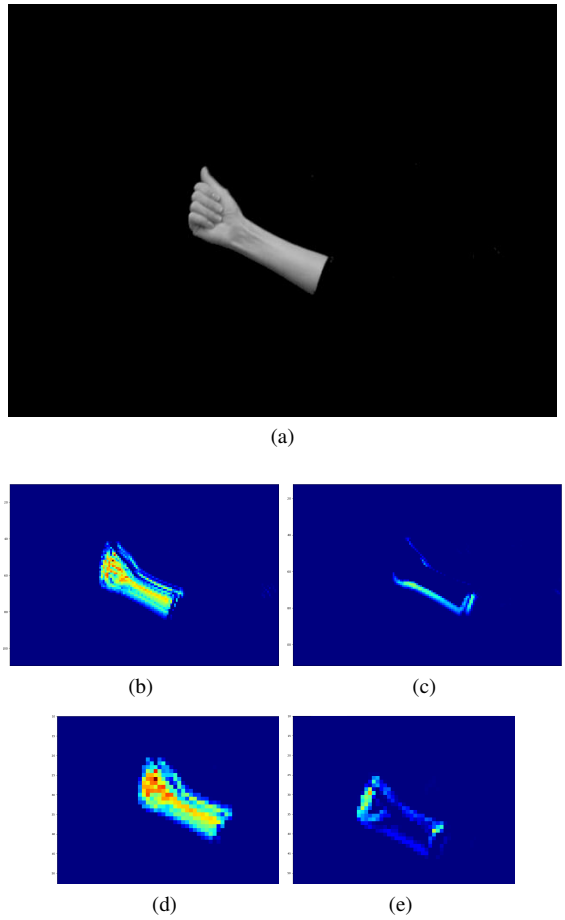

Fig. 17: Results of convolving a sample image (a) with two filters of the CNN's first convolutional layer (b,c) and second convolutional layer (d,e).

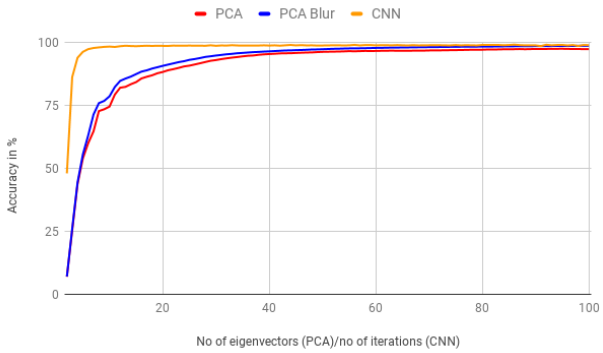

Fig. 18: Comparison of accuracy, PCA with number of eigenvectors (blurred and non blurred images) and CNN iteration number.

Figure 19 shows the time to recognize one image in seconds. We used 100 images to measure the time. For PCA we considered 100 eigenvectors and k-NN for classification, for CNN we considered 100 iterations because this number provided a good accuracy as an empirical result. Note that the

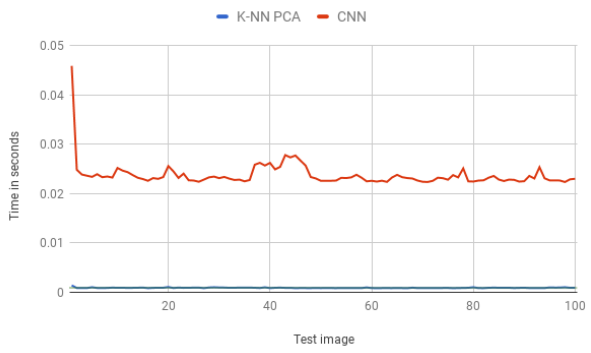

Fig. 19: Comparison of time to recognise one image using PCA with k-NN with CNN.

time is shorter for the PCA approach.

VII. Conclusions and Future Work

In this paper, we reported results of using Principal Components Analysis (PCA) with k-NN and Convolutional Neural Networks (CNNs) for hand-shape recognition. Using images that we collected depicting the hand-shape of Irish Sign Language, we obtained more accurate performances with CNNs than by PCA. Although the improvement is slight, it is worth noting that CNN did not need any preprocessing, while for PCA we need blurring to improve the accuracy.

This paper reports a stage of our current work on Irish hand-shape recognition and showed the advantages of deep features extracted with CNNs over shallow features extracted with PCA. The recognition accuracy was 0.95 for our PCA model without blurring, 0.98 for our PCA model with blurring and 0.99 for our CNN model. In addition, we have shown some outputs of the eigenvectors of the PCA approach and trace an initial analogy between PCA and CNN. On the other hand, time for classification is considerably shorter for PCA with k-NN than for CNN.

As a future application, we are working towards enabling hand-shape recognition from videos and performing comparative evaluation with shallow and deep models and a deeper analysis between the outputs of the PCA and CNN approaches.

Acknowledgements

This research was funded by CAPES/Brazilian Science without Borders, process no.: 9064-13-3. The ADAPT Centre for Digital Content Technology is funded under the SFI Research Centres Programme (Grant 13/RC/2106) and is co-funded under the European Regional Development Fund. This research also emanated from a grant in part from the IRC under Grant no. GOIPD/2016/61, in part from the EU H2020 Programme under grant agreement no. 688099 (Cloud-LSVA), and in part from SFI under Grant no. SFI/12/RC/2289 (Insight).

References

[1] M. Farouk. *Principal Component Pyramids using Image Blurring for Nonlinearity Reduction in Hand Shape Recognition*. PhD thesis, Dublin City University, Ireland, 2015.

[2] M. Farouk, A. Sutherland, and A. Shokry. Nonlinearity Reduction of Manifolds using Gaussian Blur for Handshape Recognition based on Multi-Dimensional Grids. *ICPRAM*, 2013.

[3] F. Han and H. Liu. Scale-invariant sparse PCA on high-dimensional meta-elliptical data. *Journal of the American Statistical Association*, 109(505):275–287, 2014.

[4] V. John, A. Boyali, S. Mita, M. Imanishi, and N. Sanma. Deep learning-based fast hand gesture recognition using representative frames. In *Digital Image Computing: Techniques and Applications (DICTA), 2016 International Conference on*, pages 1–8. IEEE, 2016.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[6] L. I. S. Language. Learn irish sign language, 2012.

[7] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[8] L. Leeson and J. I. Saeed. *Irish Sign Language : A Cognitive Linguistic Account*. Edinburgh University Press, 2012.

[9] H. Nagendraswamy, B. C. Kumara, and R. L. Chinmayi. Indian sign language recognition: An approach based on fuzzy-symbolic data. In *International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 1006–1013. IEEE, 2016.

[10] J. Nagi, F. Ducatelle, G. A. Di Caro, D. Cireşan, U. Meier, A. Giusti, F. Nagi, J. Schmidhuber, and L. M. Gambardella. Max-pooling convolutional neural networks for vision-based hand gesture recognition. In *International Conference on Signal and Image Processing Applications (ICSIPA)*, pages 342–347. IEEE, 2011.

[11] M. Oliveira and A. Sutherland. Interpolating eigenvectors from second-stage pca to find the pose angle in handshape recognition. In *Irish Machine Vision & Image Processing Conference*, 2015.

[12] M. Oliveira, A. Sutherland, and M. Farouk. *Manifold Interpolation for an Efficient Hand Shape Recognition in the Irish Sign Language*, pages 320–329. Springer International Publishing, Cham, 2016.

[13] M. Oliveira, A. Sutherland, and M. Farouk. Manifold interpolation for an efficient hand shape recognition in the irish sign language. In *International Symposium on Visual Computing*, pages 320–329. Springer, 2016.

[14] A. K. Sahoo, G. S. Mishra, and K. K. Ravulakollu. Sign Language Recognition : State of the Art. *Asian Res. Publ. Netw.*, 9(2):116–134, 2014.

[15] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[16] L. Wan, M. Zeiler, S. Zhang, Y. L. Cun, and R. Fergus. Regularization of neural networks using dropconnect. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 1058–1066, 2013.

[17] C. Weerasekera, M. H. Jaward, and N. Kamrani. Robust asl fingerspelling recognition using local binary patterns and geometric features. In *Digital Image Computing: Techniques and Applications (DICTA), 2013 International Conference on*, pages 1–8. IEEE, 2013.

[18] M. D. Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.