Italian Sign Language Fingerspelling Recognition

Final project for the Machine Learning course
University of Bologna

Margherita Donnici mat. 774416
Gianluca Monica mat. 786161

January 2018

Abstract

In this work, we studied the problem of recognizing images of fingerspelled letters in the Italian Sign Language (LIS). We believe that deep learning can have a major role in sign language interpretation and thus, in improving social inclusion for the deaf or hearing-impaired. In our project, we propose a recognition approach based on deep convolutional networks. Seeing there are practically no available image datasets of LIS fingerspelling, our project also included the creation of such dataset from scratch in collaboration with an italian foundation for the deaf (Fondazione Gualandi [1]). Our CNN was then implemented using Keras (a high-level API for neural networks) with Theano as back-end.

1 Introduction

.

Sign languages are languages which involve the use of hand movements, gestures, body language and facial expressions to communicate. They are predominantly used by people who are either deaf or have a hearing impairment in order to communicate comfortably. A common misconception is that all sign languages are the same worldwide or that there exists one international sign language. In fact, sign languages (just like oral languages) are specific to the geographical area: each country generally has its own, native sign language, and some even have more than one (although there are also substantial similarities among all sign languages). Over 5% of the world's population – or 466 million people – has disabling hearing loss (432 million adults and 34 million children)[4].

In Italy specifically, there is no official recognition of Italian Sign Language (Lingua dei Segni Italiana, LIS) yet, even though it is estimated about 8-10% or about 3.5 million Italians have some form of hearing loss and use LIS. Given that sign languages are independent from oral ones, this constitutes a real barrier for

[1] Fondazione Gualandi (Bologna) `http://www.fondazionegualandi.it/`

communication and social inclusions for signers. The scientific community has been studying ways to leverage technology to support this, however most existing works target American Sign Language or other non-italian sign languages.

For this reason, in our research, we concentrate on LIS. Our objective is to design a system capable of recognizing (and therefore translating) signs of the fingerspelling alphabet. A major necessity in the development of such a system is the existence of a suitable database for our objective. Seeing as there is a very limited choice, if any, of such databases for the Italian case study, at first we realized our own database with the precious help of the Fondazione Gualandi. Subsequently, we implemented a deep convolutional neural network and used images from our database to train and evaluate it.

Figure 1: Italian sign language alphabet

2 Related Work

In recent years, studies have increasingly used convolutional neural networks for human gesture recognition, due to their success in image recognition and classification. In particular, different studies have been done regarding ASL fingerspelling [5], some of them exploiting cameras that sense depth and contour, such as Microsoft Kinect [6] [7], and custom-designed color gloves [9] or sensor gloves [8]. There have also been successful attempts at using a pre-trained GoogLeNet architecture for recognizing static images of ASL letters [10]. However, all of these studies concern American Sign Language. Regarding Italian Sign Language, studies are much more rare [11] [12].

In particular, our work was strongly inspired by [1], which featured a CNN built from scratch and trained on an ASL fingerspelling static image dataset.

3 Dataset

In order to create an automatic fingerspelling recognition system using deep convolutional networks, a suitable corpus was needed to train our CNN and evaluate its performance. The lack of such a corpus for the Italian Sign Language led us to define a preliminary phase for our project whose objective was the creation of a reasonably large LIS fingerspelling image dataset from scratch to use as data for our recognition system and, eventually, to make available to the computer vision research community.

3.1 Creation of the LIS fingerspelling dataset

3.1.1 Other datasets

A google search on "italian sign language dataset" (and its italian equivalent) yields, at the time of writing, only one approximately relevant result: the A3LIS-147 database [3]. However, it is actually a video database and contains the word signs and thus is not applicable for fingerspelling recognition.

3.1.2 Characteristics

Our dataset contains 11008 images taken from 11 candidates signing 22 letters. Four letters (G, S, J and Z) were excluded from the dataset, as the corresponding sign required movement. The pictures were taken on a black background in order to avoid shadows as much as possible. For each sign, five different angles were used: front, top, bottom, left and right.

The only alteration made to the original images before adding them to the dataset was resizing them from 5184x3456 pixels to 64x64 pixels to reduce overall file size. Seeing as the pictures were taken on a uniform monochromatic background, we determined any further image preprocessing (such as background removal or cropping) was not particularly necessary. Hence, any future researchers using our dataset are also free to subject the images to the image processing they deem more appropriate for their task, and test their own combination of feature extraction methods (such as edge detection or binary images).

3.1.3 Methodology

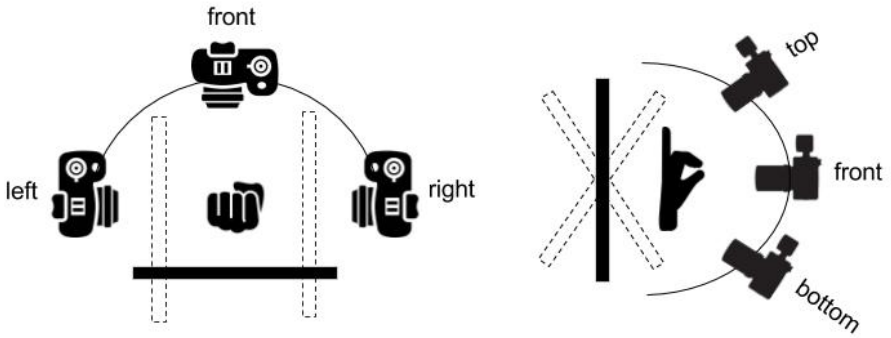
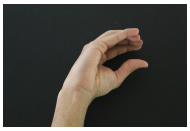


Figure 2: Set up for each angle

To acquire the images, the volunteer was asked to gesticulate each letter in front of a black panel. The different angles were obtained by rotating the camera around the volunteer's hand, which was kept still. The distance between the camera and the hand was kept more or less equal, to avoid too much variation in scale. The black panel was tilted accordingly so as to keep it parallel to the camera and maintain the black background for the pictures. Illumination conditions vary slightly for each volunteer, depending on time of day and room lighting.

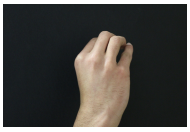
(a) Front

(b) Top



(c) Bottom

(d) Right



(e) Left

Figure 3: Sample images for the letter C

3.1.4 Comparison with ASL dataset

Since we built our CNN following the work of [1], it is important to keep in mind the characteristics of their ASL dataset [2] in order to have a measure of comparison for our own dataset's performance.

The preexisting dataset contains 2452 images taken from 5 individuals signing the 26 letters of the alphabet and 10 digits (0-9). All the pictures were taken at the same angle of rotation, perpendicular to the subject, and different lighting conditions were used for each of the five candidates (bottom, top, left, right and diffused). The images are segmented and cropped, but not altered from the original captured images.

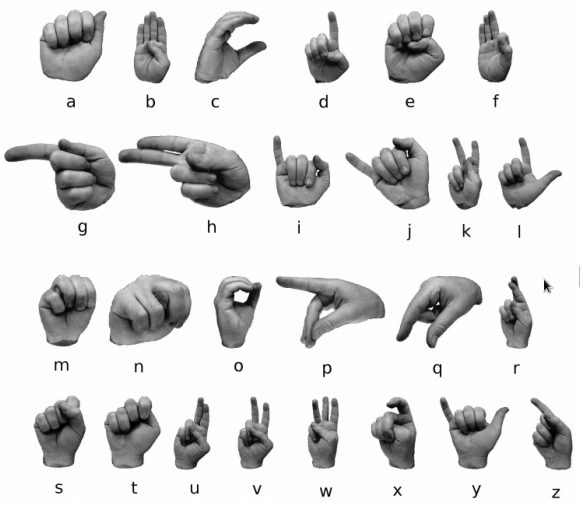


Figure 4: Sample images from the ASL dataset

	ASL pre-existing dataset	LIS constructed dataset
Number of candidates	5	11
Total number of images	2425	11008
Different lighting conditions	Yes	Yes (slight variations)
Angles of rotation	1	3
Cropped	Yes	No

Table 1: Differences between the two datasets

4 Method

Our main task was the classification of the letters of the LIS alphabet using basic supervised learning using mini-batch stochastic gradient descent. Even though most existing literature concerning similar tasks attempted an approach via transfer learning, we decided to train our own deep convolutional neural networks from scratch, inspired by [1].

4.1 Architecture

The inputs are fixed-size (64x64 pixel) images belonging to our self-constructed dataset.

The general architecture consists in multiple convolutional and dense layers, and in particular follows that of [1]:

- 3 groups consisting of 2 convolutional layers, a max-pool layer and a dropout layer
- 2 groups consisting of a fully connected layer followed by a dropout layer
- 1 final output layer

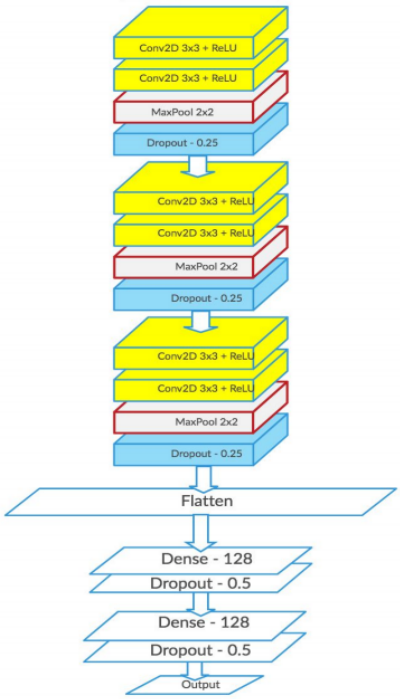


Figure 5: CNN architecture

4.2 Data

In subsequent considerations, we realized the variation in angle for images taken from the side (left and right) was a bit excessive. Hence, we generated three different datasets with our images to train and evaluate our CNN on, in order to compare performance and results based on which angles were included. Each dataset was divided into training (60%), validation (30%) and testing (10%) data.

	Angles	**Total images**
Dataset 1	Top, Front, Bottom, Left, Right	11008
Dataset 2	Top, Front, Bottom	5954
Dataset 3	Front	1869

Table 2: Different datasets characteristics

In order to make the most of our training examples, we augmented them via a number of random transformations. This was done using the Keras `ImageDataGenerator` class, which allowed us to configure random transformations and normalization operations to be done on our image data during training, instantiating generators of augmented image batches (and their labels). Transformations such as rotation, shearing and zoom were applied randomly to the images.

5 Results

Our model was trained on the three datasets mentioned previously. In this section we will discuss the different results obtained and how we tried to improve them (when needed).

We trained with a categorical cross-entropy loss function for all our datasets. For our gradient descent optimization algorithm, we chose Adam (Adaptive Moment Estimation).

In addition to training accuracy and validation accuracy, we also used a confusion matrix to describe the performance of our model on test data. To compute the confusion matrix, we used the `confusion_matrix` function of the scikit-learn library. This function also allowed us to compute the following rates (where TP are true positives, FP false positives and FN false negatives):

- **Precision**: When it predicts yes, how often is it correct?

$$recall = \frac{TP}{TP + FP} \tag{1}$$

- **Recall**: true positive rate - When it's actually yes, how often does it predict yes?

$$precision = \frac{TP}{TP + FN} \tag{2}$$

- **F1-score**: weighted average of precision and recall

$$F1 = \frac{2TP}{2TP + FN + FP} \tag{3}$$

Moreover, we also calculated the overall percentage of successful predictions by looping on our test data using Keras' `predict` function.

Both tests were carried out on two separate cases:

- **Signer Dependent**: test data consisted in pictures of a signer which was present in the training set and the validation set.
- **Signer Independent**: test data consisted in pictures of a signer which was **not** present in the training set nor in the validation set.

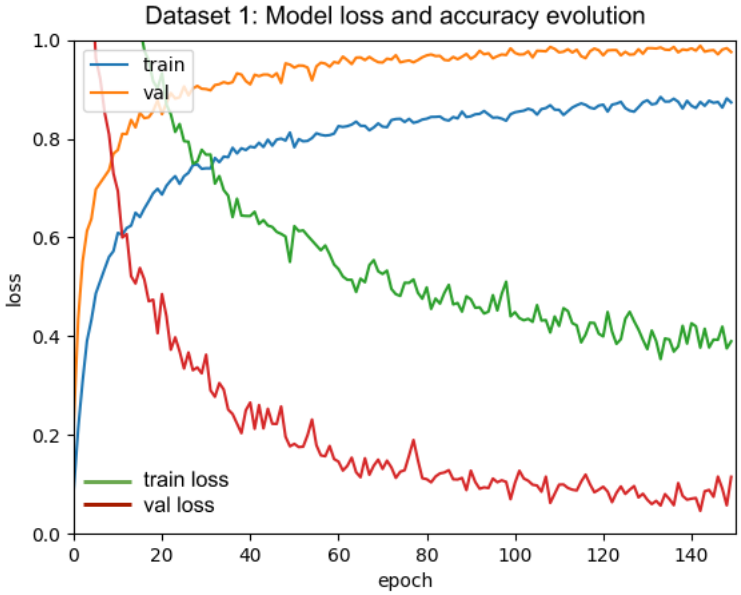
For readability reasons, the complete classification report results for each dataset have been included in Appendix A at the end of this paper.

5.1 Dataset 1 (all angles)

5.1.1 Training and validation accuracy and loss

We observed a final training accuracy of 87% and validation accuracy of 98%.

The fact that the validation accuracy is higher than the training accuracy, and thus the training loss much higher than the validation loss, can be explained by the fact that regularization mechanisms, such as Dropout and L1/L2 weight regularization, are turned off at testing time. Besides, the training loss is the average of the losses over each batch of training data. Because the model is changing over time, the loss over the first batches of an epoch is generally higher than over the last batches. On the other hand, the testing loss for an epoch is computed using the model as it is at the end of the epoch, resulting in a lower loss [13].



5.1.2 Confusion Matrix

In the signer dependent test case, the confusion matrix yielded an average F1-score of 97%, while in the signer independent test case, as expected, the average F1-score was lower (68 %).

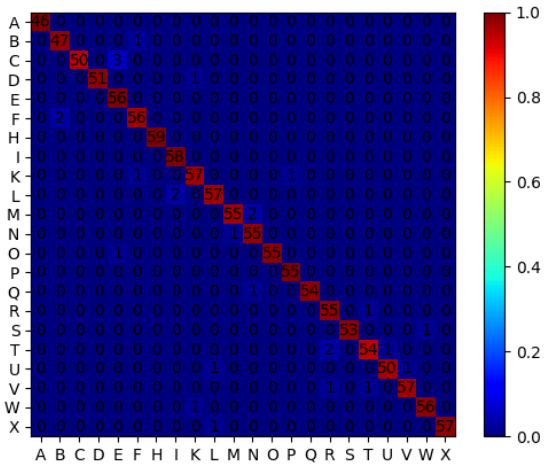


Figure 6: Confusion matrix results for dataset 1, Signer Dependent

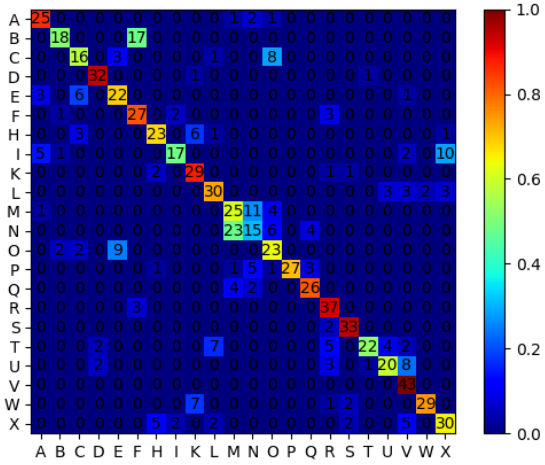


Figure 7: Confusion matrix results for dataset 1, Signer Independent

5.1.3 Predictions

Table 3: Signer Dependent prediction results for dataset 1

	Correct predictions
Overall	98% out of 1220 total images
Front	99% out of 238 total images
Top	99% out of 260 total images
Bottom	99% out of 234 total images
Left	96% out of 244 total images
Right	96% out of 244 total images

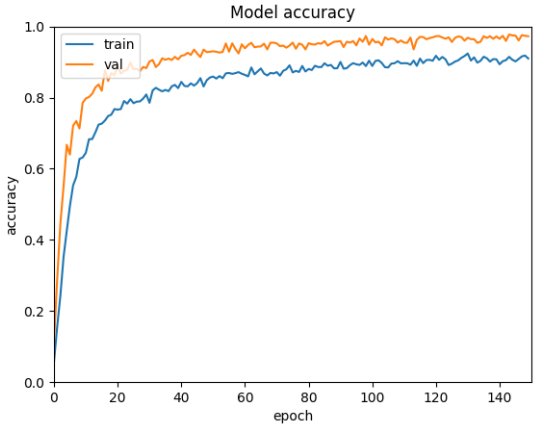
Table 4: Signer Independent prediction results for dataset 1

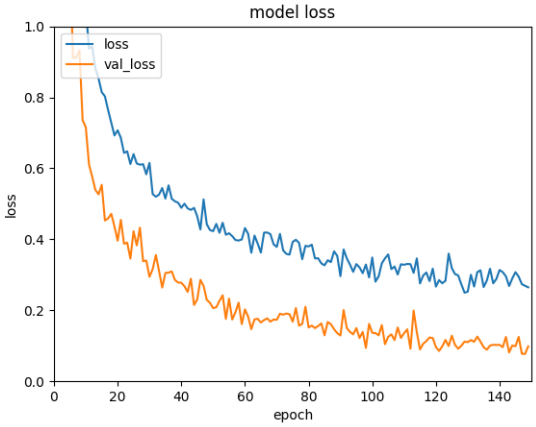
	Correct predictions
Overall	72% out of 808 total images
Front	85% out of 155 total images
Top	77% out of 160 total images
Bottom	66% out of 159 total images
Left	68% out of 161 total images
Right	64% out of 173 total images

5.2 Dataset 2 (front, top, bottom)

5.2.1 Training and validation accuracy and loss

In this case, training accuracy improved to 91% while validation accuracy remained more or less the same as the complete dataset (97%).





5.2.2 Confusion Matrix

In the signer dependent test case, the confusion matrix yielded an average F1-score of 98%, while in the signer independent test case the average F1-score was lower (77 %).

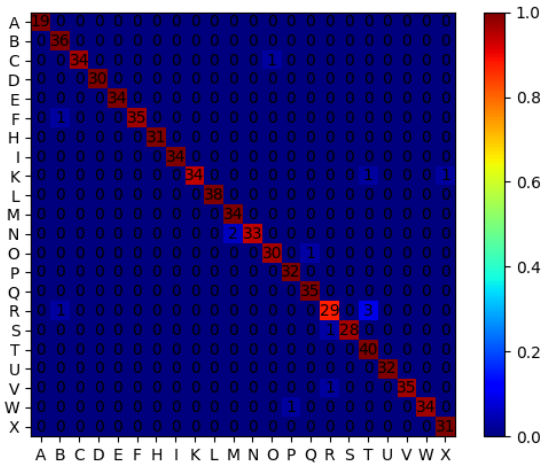


Figure 8: Confusion matrix results for dataset 2, Signer Dependent

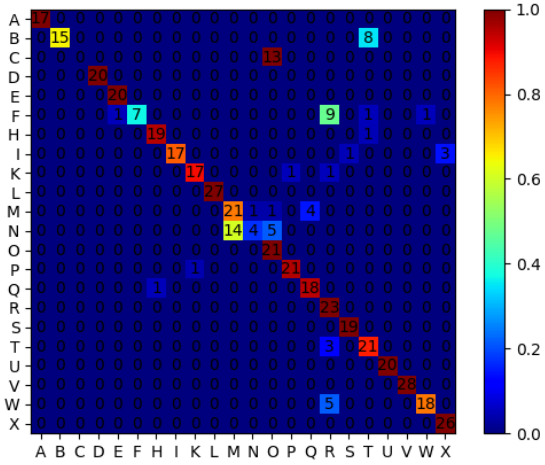


Figure 9: Confusion matrix results for dataset 2, Signer Independent

5.2.3 Predictions

Table 5: Signer Dependent prediction results for dataset 2

	Correct predictions
Overall	99% out of 732 total images
Front	99% out of 238 total images
Top	99% out of 260 total images
Bottom	99% out of 234 total images

Table 6: Signer Independent prediction results for dataset 2

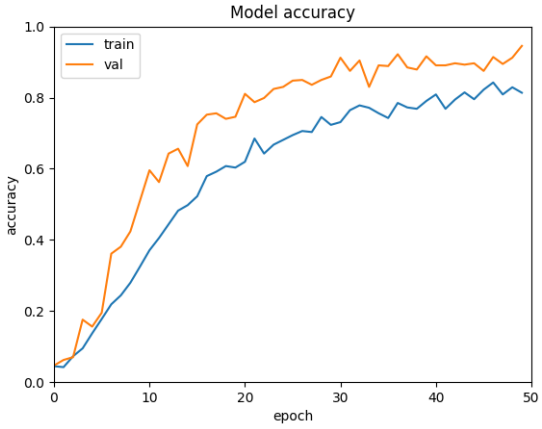
	Correct predictions
Overall	87% out of 474 total images
Front	86% out of 155 total images
Top	84% out of 160 total images
Bottom	91% out of 159 total images

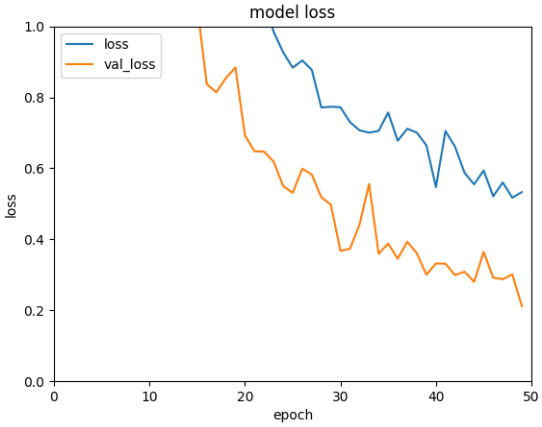
5.3 Dataset 3 (only front)

In this case as well, the validation accuracy was much higher than the training accuracy. In this case, we believe it is due to the fact that it is a much smaller dataset: smaller datasets have smaller intrinsic variance so this means that the model properly captures patterns inside of the data and train error is greater simply because the inner variance of training set is greater than validation set.

5.3.1 Training and validation accuracy and loss

We observed a final training accuracy of 84% and validation accuracy of 94%.





5.3.2 Confusion Matrix

In the signer dependent test case, the confusion matrix yielded an average F1-score of 97%, while in the signer independent test case, once again as expected, the average F1-score was lower (81 %).

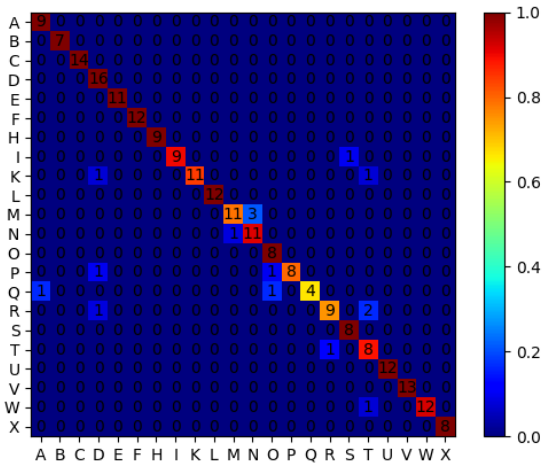


Figure 10: Confusion matrix results for dataset 3, Signer Dependent

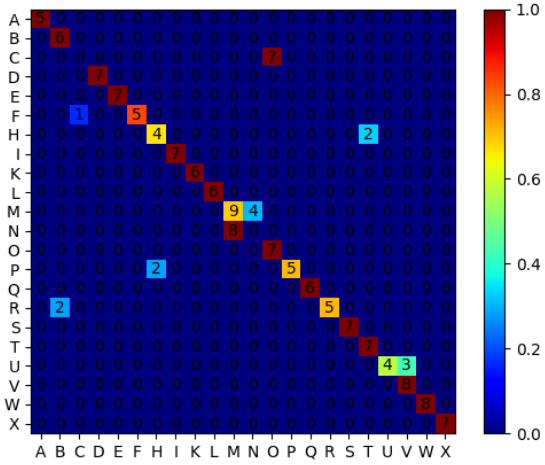


Figure 11: Confusion matrix results for dataset 3, Signer Independent

5.3.3 Predictions

Table 7: Signer Dependent prediction results for dataset 3

	Correct predictions
Overall	92% out of 238 total images

Table 8: Signer Independent prediction results for dataset 3

	Correct predictions
Overall	82% out of 155 total images

5.4 Additional tests

A few additional tests were done to compare accuracy results. Among other things, we also trained our CNN on different-sized images. Namely, the results on dataset 1 with images resized to 200x133 pixels were the same as those obtained with the 64x64 pixels (97% F1-score).

In order to prevent overfitting, we modified our original architecture adding batch normalization, that is adding a normalization layer after each convolutional layer. However, on this case, accuracy remained more or less the same (variations of $\approx 0,01$).

5.5 Summary of results

Dataset	Epochs	Batch Normalization	Accuracy	Loss	Validation Accuracy	Validation Loss
1	250	No	0.87	0,38	0,98	0,07
2	150	No	0,91	0,26	0,97	0,09
3	50	No	0,84	0,51	0,94	0,21

Table 9: Summary of training results

Dataset	Epochs	Batch Normalization	Precision	Recall	F1-score	Correct Predictions
1	250	No	0,97	0,97	0,97	98%
2	150	No	0,99	0,98	0,98	99%
3	50	No	0,92	0,91	0,91	92%

Table 10: Summary of testing results - Signer Dependent

Dataset	Epochs	Batch Normalization	Precision	Recall	F1-score	Correct Predictions
1	250	No	0,70	0,69	0,68	72 %
2	150	No	0,81	0,83	0,81	87%
3	50	No	0,77	0,79	0,77	82%

Table 11: Summary of testing results - Signer Independent

6 Conclusions

In this paper, we described the creation of a statics image dataset of fingerspelling signs of the Italian Sign Language. This dataset will be made available for other computer vision researchers. We also described a deep learning approach to recognize and classify fingerspelling gestures from the Italian Sign Language. We observed in average a higher accuracy than the results obtained on the dataset in [1] (67%). Our CNN yielded acceptable results for the signer independent test as well. In particular, the best results were observed with dataset 2: indeed, dataset 1 also contained the left-angled and right-angled pictures (which were later deemed too diverted) who could have had an impact on the accuracy. On the other hand, dataset 3 contained very few pictures.

Even though better results can be reached with more costly technologies, such as cameras with depth sensors, our work showed a potential to solve this task with a simple camera.

A Appendix

A.1 Dataset 1 results

A.1.1 Classification Report

Classification Report

	precision	recall	f1-score	support
a	1.00	0.98	0.99	46
b	0.94	0.98	0.96	48
c	0.98	0.94	0.96	53
d	1.00	0.96	0.98	52
e	0.92	1.00	0.96	56
f	0.97	0.97	0.97	58
h	1.00	0.98	0.99	59
i	0.98	1.00	0.99	58
k	0.95	1.00	0.98	59
l	0.95	0.98	0.97	59
m	1.00	0.93	0.96	57
n	0.93	0.98	0.96	56
o	1.00	0.95	0.97	56
p	1.00	0.98	0.99	55
q	0.98	1.00	0.99	55
r	0.91	0.95	0.93	56
t	0.98	0.98	0.98	54
u	0.95	0.95	0.95	57
v	1.00	0.92	0.96	52
w	0.93	0.95	0.94	59
x	0.96	0.96	0.96	57
y	0.98	0.95	0.96	58
avg / total	0.97	0.97	0.97	1220

Figure 12: Classification report results for dataset 1, Signer Dependent

Classification Report

	precision	recall	f1-score	support
a	0.76	0.86	0.81	29
b	0.68	0.54	0.60	35
c	0.63	0.43	0.51	28
d	0.94	0.85	0.89	34
e	0.76	0.81	0.79	32
f	0.53	0.70	0.61	33
h	0.74	0.74	0.74	34
i	0.71	0.43	0.54	35
k	0.60	0.91	0.72	33
l	0.74	0.76	0.75	41
m	0.56	0.68	0.62	41
n	0.44	0.33	0.38	48
o	0.57	0.72	0.63	36
p	1.00	0.61	0.75	38
q	0.59	0.81	0.68	32
r	0.67	0.82	0.74	40
t	0.89	0.94	0.92	35
u	0.73	0.45	0.56	42
v	0.60	0.53	0.56	34
w	0.73	0.95	0.83	43
x	0.94	0.74	0.83	39
y	0.62	0.61	0.62	46
avg / total	0.70	0.69	0.68	808

Figure 13: Classification report results for dataset 1, Signer Independent

A.2 Dataset 2 results

A.2.1 Classification Report

	precision	recall	f1-score	support
a	1.00	1.00	1.00	19
b	0.94	0.92	0.93	36
c	1.00	1.00	1.00	35
d	1.00	1.00	1.00	30
e	1.00	1.00	1.00	34
f	0.95	0.97	0.96	36
h	0.97	1.00	0.98	31
i	1.00	1.00	1.00	34
k	1.00	0.97	0.99	36
l	1.00	1.00	1.00	38
m	0.92	1.00	0.96	34
n	1.00	0.91	0.96	35
o	1.00	0.97	0.98	31
p	1.00	1.00	1.00	32
q	0.95	1.00	0.97	35
r	0.91	0.94	0.93	33
t	1.00	0.97	0.98	29
u	0.93	0.95	0.94	40
v	1.00	1.00	1.00	32
w	1.00	0.97	0.99	36
x	1.00	0.97	0.99	35
y	1.00	1.00	1.00	31
avg / total	0.98	0.98	0.98	732

Figure 14: Classification report results for dataset 2, Signer Dependent

	precision	recall	f1-score	support
a	0.77	1.00	0.87	17
b	1.00	0.74	0.85	23
c	0.00	0.00	0.00	13
d	0.95	0.95	0.95	20
e	0.95	0.95	0.95	20
f	0.88	0.37	0.52	19
h	1.00	1.00	1.00	20
i	1.00	0.86	0.92	21
k	0.94	0.89	0.92	19
l	1.00	0.96	0.98	27
m	0.55	0.78	0.65	27
n	0.00	0.00	0.00	23
o	0.53	1.00	0.69	21
p	0.95	0.95	0.95	22
q	0.95	1.00	0.97	19
r	0.53	1.00	0.70	23
t	0.95	1.00	0.97	19
u	0.70	0.79	0.75	24
v	0.95	1.00	0.98	20
w	1.00	1.00	1.00	28
x	1.00	0.74	0.85	23
y	0.96	1.00	0.98	26
avg / total	0.81	0.83	0.81	474

Figure 15: Classification report results for dataset 2, Signer Independent

A.3 Dataset 3 results

A.3.1 Classification Report

Classification Report

	precision	recall	f1-score	support
a	1.00	1.00	1.00	9
b	0.75	0.86	0.80	7
c	1.00	1.00	1.00	14
d	0.89	1.00	0.94	16
e	1.00	1.00	1.00	11
f	0.91	0.83	0.87	12
h	0.90	1.00	0.95	9
i	1.00	0.90	0.95	10
k	1.00	0.92	0.96	13
l	0.92	1.00	0.96	12
m	0.91	0.71	0.80	14
n	0.73	0.92	0.81	12
o	0.80	1.00	0.89	8
p	1.00	0.70	0.82	10
q	1.00	0.83	0.91	6
r	0.89	0.67	0.76	12
t	0.89	1.00	0.94	8
u	0.82	1.00	0.90	9
v	0.92	0.92	0.92	12
w	1.00	0.92	0.96	13
x	0.92	0.92	0.92	13
y	0.89	1.00	0.94	8
avg / total	0.92	0.91	0.91	238

Figure 16: Classification report results for dataset 3, Signer Dependent

Classification Report

	precision	recall	f1-score	support
a	1.00	1.00	1.00	5
b	0.67	1.00	0.80	6
c	0.00	0.00	0.00	7
d	1.00	1.00	1.00	7
e	0.88	1.00	0.93	7
f	1.00	0.83	0.91	6
h	0.50	0.33	0.40	6
i	1.00	0.86	0.92	7
k	1.00	1.00	1.00	6
l	1.00	1.00	1.00	6
m	0.53	0.62	0.57	13
n	0.17	0.12	0.14	8
o	0.50	1.00	0.67	7
p	1.00	0.71	0.83	7
q	1.00	1.00	1.00	6
r	0.80	0.57	0.67	7
t	1.00	1.00	1.00	7
u	0.67	0.86	0.75	7
v	1.00	0.71	0.83	7
w	0.80	1.00	0.89	8
x	1.00	1.00	1.00	8
y	0.88	1.00	0.93	7
avg / total	0.77	0.79	0.77	155

Figure 17: Classification report results for dataset 3, Signer Independent

References

[1] Vivek Bheda and N. Dianna Radpour *Using Deep Convolutional Networks for Gesture Recognition in American Sign Language*. 23 Oct 2017

[2] Barczak, A.L.C., Reyes, N.H., Abastillas, M., Piccio, A., Susnjak, T. (2011), *A new 2D static hand gesture colour image dataset for ASL gestures*, Research Letters in the Information and Mathematical Sciences, 15, 12-20

[3] Fagiani M, Principi E, Squartini S, Piazza F, *A new Italian sign Language database*.
In: Zhang H, Hussain A, Liu D, Wang Z (eds) *Advances in brain inspired cognitive systems, Lecture Notes in Computer Science, vol 7366*. Springer, pp 164-173, 2012

[4] World Health Organization *Deafness and Hearing loss*. Updated March 2018. `http://www.who.int/mediacentre/factsheets/fs300/en/`

[5] Taehwan Kim, Karen Livescu, Gregory Shakhnarovich, *American Sign Language Fingerspelling Recognition with phonological feature-based tandem models*.
In IEEE Spoken Language Technology Workshop (SLT), 119-124,2012

[6] Nicolas Pugeault, Richard Bowden, *Spelling It Out: Real–Time ASL Fingerspelling Recognition*

[7] Agarwal, Anant and Thakur, Manish, *Sign Language Recognition using Microsoft Kinect*.
In IEEE International Conference on Contemporary Computing, 2013.

[8] S. A. Mehdi and Y. N. Khan, *Sign language recognition using sensor gloves*, Neural Information Processing, 2002. ICONIP '02. Proceedings of the 9th International Conference on, 2002, pp. 2204-2206 vol.5.

[9] Cao Dong, Ming C. Leu and Zhaozheng Yin. American, *Sign Language Alphabet Recognition Using Microsoft Kinect*.
In: IEEE International Conference on Computer Vision and Pattern Recognition Workshops, 2015.

[10] Garcia, Brandon and Viesca, Sigberto. *Real-time American Sign Language Recognition with Convolutional Neural Networks*.
In: Convolutional Neural Networks for Visual Recognition at Stanford University, 2016.

[11] Lionel Pigou, Sander Dieleman, Pieter-Jan Kindermans, Benjamin Schrauwen, *Sign Language Recognition using Convolutional Neural Networks*

[12] N. Neverova, C. Wolf, G. W. Taylor, and F. Nebout, *Multiscale deep learning for gesture detection and localization.*
In ECCVW, 2014.

[13] Keras FAQ
https://keras.io/getting-started/faq/#why-is-the-training-loss-much-higher-than-the-testing-loss