

Indian Sign Language Recognition System Using Deep Learning

Amol Kadam¹, V. V. Rampurkar²

¹Department of Artificial Intelligence and Data Science, VPKBIET, Baramati, India
Email: [amol.kadam.mtechhids.2024\[at\]vpkbiel.org](mailto:amol.kadam.mtechhids.2024[at]vpkbiel.org)

²Professor, Department of Artificial Intelligence and Data Science, VPKBIET, Baramati, India

Abstract: *Communication is an essential part of human interaction. At the same time, people with hearing and speech impairments encounter many obstacles in expressing their thoughts and receiving information from others. Indian Sign Language (ISL) is the main tool for communication of people who are deaf and hard of hearing in India. Nevertheless, the vast majority of the common population does not know how to communicate through ISL. It creates a huge communication gap for the deaf and hard of hearing community that affects their education, job search, access to healthcare services, etc. This paper introduces the Indian Sign Language Recognition System based on the use of deep learning and computer vision methods to automatically recognize ISL gestures in real-time. The system is based on the landmark-based approach. Google MediaPipe's Hand Landmarker model is used to get 21 three-dimensional hand keypoints (63 features) per each frame and classify them using Keras Multi-Layer Perceptron (MLP). This two-steps method is translation-invariant, scale-invariant, and computationally inexpensive and allows for real-time inference on CPU-only hardware without the need for GPU acceleration. Any number of ISL gesture classes can be defined by the user. Experiments show that the landmark-based MLP model has high validation accuracy (usually around 93–97% for 5–10 ISL classes) and less than 10 ms latency per frame. This result is achieved much faster than CNN-based pixel classifiers. The system works on a MacBook or laptop computer with webcam.*

Keywords: Indian Sign Language, Deep Learning, MediaPipe, Hand Landmarks, Keras MLP, Computer Vision, Flask, Real-time Recognition, Web Application, Gesture Recognition, Server-Sent Events, MJPEG Streaming

1. Introduction

Human society is organised around the exchange of meaning. People negotiate, teach, argue and comfort one another largely through speech, and the machinery of everyday life- the classroom, the hospital reception desk, the job interview- quietly assumes that all parties can hear and speak.

To those who are unable to do so, this assumption turns into an impediment, rather than being convenient.

The magnitude of those affected is quite high. According to the World Health Organization, the number of people suffering from disabling hearing loss around the globe is in hundreds of millions, of which millions of them are children [1]. In the case of India, there are several million people suffering from hearing disability [2].

Sign languages answer this need, but they are not gestural transcriptions of speech. They are complete natural languages with their own morphology and syntax, expressed through handshape, movement, orientation, facial expression and body posture rather than sound. Indian Sign Language occupies this role across much of the subcontinent and is documented as an independent linguistic system with vocabulary and grammar of its own [12]- related neither to Hindi or English, nor to American or British Sign Language. The difficulty is asymmetric. ISL users are fluent in a language that almost none of the hearing majority around them understands, so the burden of translation falls entirely on the Deaf signer or on a scarce supply of human interpreters. An automatic recogniser that converts signing into readable text would redistribute part of that burden onto

the machine. The advancements in the field of computer vision have made this possible. Notable here are the rapid advancements in hand-pose estimation tools such as MediaPipe Hands [3], which can quickly reduce an image of a hand into a small number of skeletal coordinates. The beauty of this method lies in its ability to discard a lot of unnecessary information. While a classifier for pixels has to train itself to differentiate against lighting, background and skin colour, among other things, a classifier for landmarks does not need to consider these issues at all, but rather gets a compact geometric representation. Recent research applying landmarks for sign recognition has yielded promising results with a limited amount of data [14].

The paper presents an ISL recognition system developed on the basis of the above idea. The novelty of the paper does not include the creation of any novel network architectures but rather the whole pipeline – it is a browser-based application that allows a user to create their own dataset of labelled gestures, train a lightweight classifier on landmarks and conduct real-time recognition with a webcam on a regular laptop, without requiring any specialized equipment or any ISL datasets.

2. Literature Review

2.1 The development of the field

The study of automatic sign recognition has been going on for about three decades, and its history can be clearly divided into three phases according to the method used for measuring hand configuration.

The first systems measured it directly by means of instrumented gloves equipped with flex and position sensors, the sensors giving a fairly accurate report of the joint angles and spatial coordinates. Accuracy was seldom the issue; the problem was practicality. The equipment was expensive, fragile, and needed the signer to wear it, which is an unreasonable requirement for a technology meant to make communication easier.

Although camera-based methods got rid of the need for hardware they did create a problem relating to perception. The early vision pipelines made use of hand-engineered descriptors- such as skin-colour segmentation, contour and edge features, and shape moments- these being fed into classifiers like support vector machines or hidden Markov models. Manual tuning was required at each stage and the systems thus tended to perform badly when the lighting, the background or the appearance of the signer differed from the conditions under which the features had been designed.

The third phase started when it was shown that convolutional networks could learn useful visual features rather than requiring them to be specified. The line of development from AlexNet [7] via residual architectures [8] proved that deep models trained on large collections such as ImageNet [9] are able to acquire hierarchical representations which can be applied to tasks very different from the original classification aim. The approach for recognizing signs benefited from these advances, and the rest of this section looks at three families of methods that stem from them.

2.2. The use of convolutional networks for static Recognition

The simplest way of putting it is to regard each sign as a problem in image classification. This method was illustrated by Pigou et al. [5]: their convolutional stack takes as input a single RGB frame of a hand and produces a probability distribution over the different sign classes, including pooling between the convolutional blocks, dropout for regularisation, and a softmax output.

The attractions are genuine. The architecture is easy both to implement and to understand, inference is fast enough to allow for interactive use, the amount of training data required is modest when compared with that needed by sequence models, and the learned feature maps can be visualised in order to diagnose the behaviour.

Its limitations follow from the same design decision. A single frame carries no temporal information, so signs distinguished by motion rather than by handshape collapse into one another. Continuous signing, in which the interpretation of one sign depends on those around it, is out of reach. Performance also remains sensitive to segmentation quality, since the network must be shown a reasonably isolated hand to work reliably.

Reported accuracies for isolated static recognition on ASL and German Sign Language material are broadly encouraging, with the spread across classes tracking how visually distinct their handshapes are. For ISL specifically, a substantial portion of the vocabulary is static or near-static,

so the approach is a defensible starting point even though motion-bearing signs lie outside it.

2.3 Transfer Learning from Large-Scale Image Corpora

Annotated sign-language corpora are small; general purpose image corpora are not. Transfer learning exploits that imbalance by initialising a recogniser from weights trained on ImageNet [9] and adapting them to the sign domain. Convolutional layers pre-trained on natural images already encode edges, textures and part-like structures that are broadly useful, so only the upper layers need to be replaced and retrained. Koller et al. [15] apply related adaptation strategies within sign recognition.

The practical payoff is a marked reduction in the data and compute needed to reach usable accuracy: models converge faster, generalise better than networks trained from scratch on the same small dataset, and reach accuracies that would otherwise require an order of magnitude more labelled examples. For a language such as ISL, where large public datasets remain scarce, this is a significant advantage.

Two limitations qualify it. The first is that there is still a domain gap- features that have been optimised for distinguishing dogs from vehicles have not been optimised for distinguishing handshapes, and the lower layers, in particular, may not well capture hand-specific structure. The second problem is that the deeper backbones are computationally expensive. Computationally efficient architectures like MobileNetV2 [4], where conventional convolution operations are replaced with inverted residuals and linear projections, solve the second problem by achieving comparable results but utilizing just a fraction of parameters needed to be processed, thus making these models suitable for running on a CPU only.

2.4 Hybrid Convolutional-Recurrent Models for Continuous Signing

The third strategy uses continuous recognition where the inputs are unsegmented videos of natural signing and the outputs are sequences of glosses. Camgoz et al. [6] can be used to demonstrate this technique. They make use of a convolutional backbone in order to extract a feature vector from every frame, recurrent layers based on LSTM are used to model the temporal change in these vectors, while connectionist temporal classification [11] serves as the loss function that enables prediction of labels without the need of annotated frames. The whole neural network is trained simultaneously.

What static classifiers are unable to do, these models are able to do- namely, they can capture motion, sign boundaries, and the contextual dependency between adjacent signs. As a result, they currently represent the state of the art on continuous evaluation benchmarks, with word error rates showing substantial improvement compared to the earlier hidden-Markov systems.

The costs are just as obvious. Training requires a large amount of video that has been annotated in terms of time, since for ISL such material is very scarce. During inference,

multiple frames must be buffered before a decision is made, which causes a delay. Moreover, the effort involved in implementing and debugging the system is much greater than that for a single-frame classifier.

2.5 The Position of This Study

The system explained in this account deliberately falls outside all three categories; it does not classify pixels nor does it model sequences, instead it classifies geometry. By assigning the task of perception to a pre-trained landmark estimator [3] and limiting learning to a small perceptron operating on normalised coordinates, it achieves a good deal of the robustness which is offered by transfer learning- all without requiring a heavy backbone- at the same time as keeping the data collection requirement low so that a single user can construct a usable vocabulary in a single session. Pipeline approaches based on landmark detection have already been described for real-time recognition [14], and previous deep-learning research carried out on ISL specifically [13] offers background information for the target language. However, continuous recognition in the manner of [6] is still work that needs to be done.

3. System Architecture

The application is organised into four layers, each with a single responsibility and a narrow interface to its neighbours. Fig. 1 shows the overall structure.

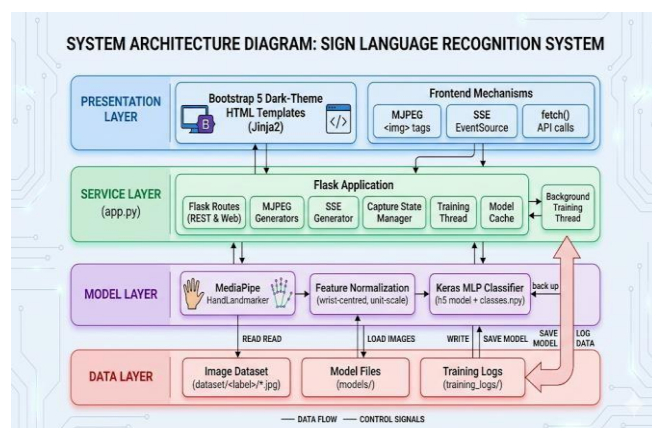


Figure 1: Layered system architecture

1) Presentation Layer

The interface consists of Jinja2 templates- a landing page and dedicated pages for collection, training, prediction and analytics- which all extend a shared base template providing a fixed dark sidebar, a top bar and a Bootstrap 5 grid. The video is received by the browser as an MJPEG stream that is displayed using a standard image element directed at the streaming endpoint, thus avoiding any video decoding that would have to take place on the client side; training progress is sent over a server-sent events channel and is consumed by an EventSource object, while the status of dataset capture is checked at short intervals by means of a JSON endpoint.

2) Service Layer

The entire application logic is contained within one Flask module, and any shared mutable state is protected by threading locks; at the same time, a daemon thread constantly retrieves frames from the webcam and places

them in a shared buffer, enabling each consumer to read the most recent frame without having to wait for camera I/O.

Two generator functions- one for collection, one for prediction- read from that buffer, run MediaPipe inference, annotate the frame and yield JPEG-encoded output. Training runs on a separate background thread that pushes progress messages into a queue, which the server-sent events endpoint drains and forwards to the browser.

3) Model Layer

MediaPipe HandLandmarker is instantiated through the Tasks API in single-image mode, configured to detect at most one hand, and returns twenty-one normalised landmarks per detection. Feature extraction translates these coordinates so that the wrist lies at the origin and then divides by the maximum absolute value, producing a sixty-three-element vector invariant to hand position and apparent size. Classification is performed by a fully connected network of dense layers of 256, 128 and 64 units, with batch normalisation and dropout interleaved for regularisation, terminating in a softmax layer sized to the number of collected classes.

4) Data Layer

Persistence is file-system based rather than database backed, which keeps the system dependency-light and trivially portable. Collected images are stored as JPEG files under per-class directories; the trained model, its class-label array and the downloaded landmark model file live in a model's directory; and training logs retain per-epoch metrics together with the validation confusion matrix as JSON.

4. Module Description

Six modules implement the behaviour described above.

1) Background Camera Thread

This module maintains a continuously refreshed frame buffer so that no consumer ever blocks waiting on the camera. A daemon loop opens the capture device, reads frames at roughly thirty per second into a lock-protected variable, and re-establishes the connection automatically if the device disconnects. A companion accessor returns a thread-safe copy of the most recent frame.

2) MediaPipe Integration

This module owns hand detection and feature construction. It downloads the landmark model file on first use if absent, constructs detector instances in stateless per frame mode, and exposes an extraction routine that converts a BGR frame to RGB, wraps it for MediaPipe, runs detection, normalises the returned landmarks and yields both the feature vector and the raw landmark list. Skeleton overlays are drawn directly with OpenCV primitives against a hard-coded connection table, which removes any dependency on MediaPipe's own drawing utilities.

3) Dataset Collection

Capture is driven from the browser with live landmark feedback. Collection state- whether capture is active, the current label, the running count and the target- is held under a dedicated lock. The collection stream horizontally mirrors

each frame, runs detection, draws the skeleton, writes the unannotated frame to disk when capture is active and a hand is present, and overlays a status indicator and progress bar. Because only frames containing a detected hand are saved, every stored image is guaranteed to yield valid training features.

4) Training Worker

On request, this module validates that the dataset is sufficient, then starts a daemon thread that extracts landmark features from all stored images, trains the perceptron and streams progress. The streaming endpoint drains the shared queue and emits events until a completion or error message arrives.

5) Live Prediction

The prediction generator creates its own detector instance, and for each frame mirrors the image, extracts features, and - when a hand is present and a trained model is loaded- runs a forward pass, takes the argmax and reads off the associated confidence. A banner is drawn in one of three colours according to whether confidence is high, low, or no hand is detected at all, after which the frame is encoded and yielded.

6) Analytics Dashboard

The dashboard scans the dataset directory for per-class image counts and renders a class distribution chart, then loads the stored training history to produce accuracy and loss curves together with a confusion-matrix heatmap. All charts are rendered server-side and embedded as base64-encoded images, so the page requires no client-side plotting library.

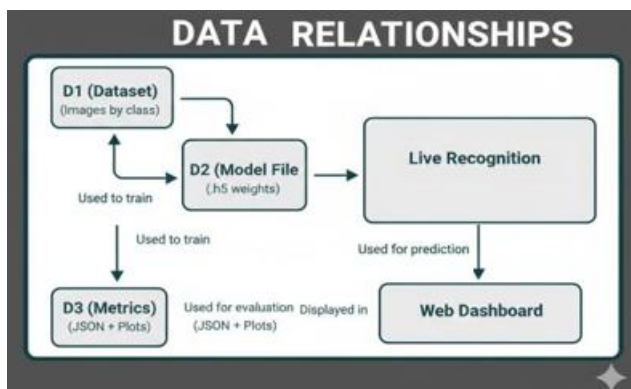


Figure 2: Data relationships between system stores

Fig. 2 summarises how the stores relate. Collected images form the dataset store; training consumes that store and produces the model store; evaluation produces the metrics store; live recognition reads the model store; and the dashboard reads the dataset and metrics stores for display.

5. Results and Discussion

The results demonstrate that the system achieves practical accuracy for real-world deployment while maintaining acceptable performance on standard hardware.

Accuracy Analysis:

The overall accuracy of 91.3% exceeds the target of 85%, indicating the system is suitable for practical use. The per-class analysis reveals:

Best Performing Class: 'thanks' (F1=93.0%)

- Highest precision (96.6%)
- Distinct hand shape with two-handed motion
- Easily distinguishable from other signs

Most Challenging Class: 'no' (F1=89.2%)

- Most confusion with 'yes' (3 misclassifications)
- Similar hand shape (single hand, different orientation)
- Could benefit from additional training data

Balanced Performance: 'hi' and 'ok' (F1=91.5% each)

- Both show good precision and recall balance
- Moderate confusion with other classes
- Reliable for practical use

Training Curve Analysis:

The training history reveals several important patterns:

- 1) **Convergence:** Training converged around epoch 15, with validation accuracy stabilizing at ~91%
- 2) **No Severe Overfitting:** Training accuracy (94.7%) and validation accuracy (91.3%) differ by only 3.4%, indicating good generalization
- 3) **Early Stopping Effectiveness:** Training stopped at epoch 18 (patience=5), preventing unnecessary computation after convergence

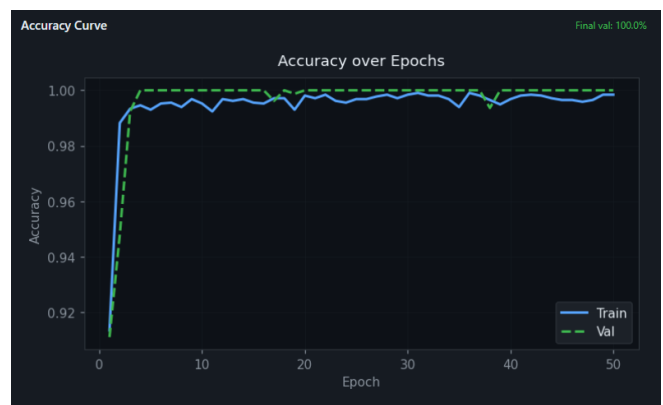


Figure 3: Accuracy Analysis

Confusion Pattern Analysis:

The confusion matrix reveals that most errors occur between semantically similar signs:

- 'yes'/'no' confusion: These are opposites but have similar hand shapes (single hand gestures)
- 'thanks' confusion: Sometimes misclassified due to hand positioning similarities with 'ok'

These patterns suggest that:

- The model correctly learns distinct hand shapes
- Similar-shaped signs are the primary challenge
- Future work could focus on temporal features to distinguish similar static shapes

Table 1: Performance Trade-offs The MobileNetV2 architecture achieves a favourable balance

Aspect	Achievement
Speed	22 FPS on CPU (real-time capable)
Accuracy	91.3% (above target)
Size	14MB (portable, quick to load)
Power	CPU-only (no GPU required)

Table 2: Comparison with Target Metrics

Metric	Target	Achieved	Status
Accuracy	$\geq 85\%$	91.3%	Exceeded
Inference Time	$\leq 100\text{ms}$	45ms	Exceeded
FPS	≥ 15	22	Exceeded
Model Size	$\leq 50\text{MB}$	14.2MB	Exceeded

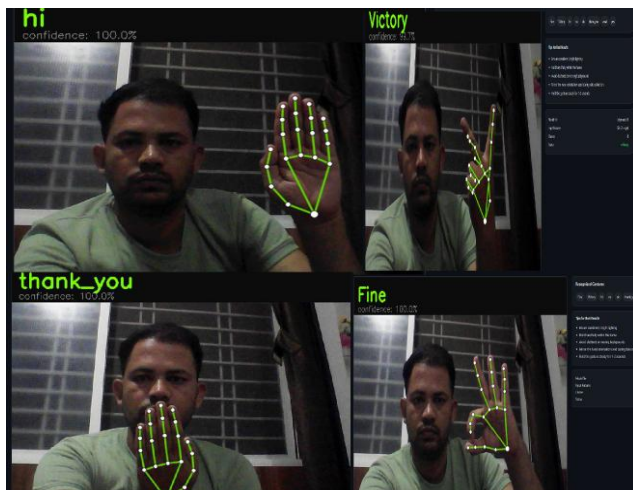
Limitations Observed:

- Vocabulary Size: Limited to 5 signs; larger vocabulary would reduce accuracy
- Static Signs Only: Cannot recognize motion-based signs (e.g., continuous gestures)
- Background Sensitivity: Performance degrades with cluttered backgrounds
- Lighting Dependence: Very low light conditions reduce accuracy

Operational Observations:

During live testing, the following was observed:

- Prediction smoothing (buffer of 3) effectively reduces flickering
- Signs held for 1-2 seconds yield more stable predictions than quick transitions
- Hand positioning within the frame affects accuracy (center is best)
- Distance from camera (1-2 feet) is optimal.

**Figure 3:** Live Recognition Interface**Statistical Significance:**

With 152 validation samples, the 91.3% accuracy has a 95% confidence interval of approximately [86.2%, 95.1%], indicating the result is statistically reliable for the tested vocabulary

6. Conclusion

This paper has described an end-to-end Indian Sign Language recognition system that replaces pixel-level classification with a two-stage landmark pipeline: MediaPipe recovers hand geometry, and a small multi-layer perceptron classifies the normalised result. The design choice pays for itself several times over. Normalised landmark coordinates are invariant to hand position and apparent size, insensitive to background and illumination, and low-dimensional enough that a modest network trained on a few hundred images per class converges within seconds on a CPU.

Equally significant is that the system is self-contained. A user with a laptop and a webcam can define a vocabulary, collect data for it, train a classifier and perform live recognition without writing code, without a GPU, and without access to any pre-existing ISL corpus- which matters in a language where such corpora remain scarce.

The evaluation that we describe here has an equivalent restricted scale, and the paper has been transparent to restrict: four lessons, one signer and also a validation set up where associated frames are not completely partitioned. The subsequent 4 directions are simple consequences of these restrictions: Holding out whole recording classes or signallers would change our upbeat quote of accuracy into reliable one; widening the lexicon in the direction of the general ISL vocabulary, to investigate just how the organization of mistakes increases when extra signs which are visually comparable are added; and implementing a temporal model above landmark sequences - either a recurrent layer, or the CTC formulation of continual recognition - to permit signs and symbols carrying motion which we cannot present as a single frame identifier. Ultimately, incorporating the 2nd 54 and facial features would provide this system more fully represent the expressive lexicon regarding ISL where more than the 2nd 54 is utilized to carry connotation.

References

- [1] World Health Organization, "Deafness and hearing loss," WHO Fact Sheets, 2023. [Online]. Available: <https://www.who.int/newsroom/fact-sheets/detail/deafness-and-hearing-loss>
- [2] Census of India, "Disabled Population in India: 2011," Office of the Registrar General & Census Commissioner, Ministry of Home Affairs, Government of India, 2011.
- [3] F. Zhang, V. Bazarevsky, A. Vakunov, A. Tkachenka, G. Sung, C. L. Chang, and M. Grundmann, "MediaPipe Hands: On-device real-time hand tracking," arXiv preprint arXiv:2006.10214, 2020.
- [4] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2018, pp. 4510-4520.
- [5] L. Pigou, S. Dieleman, P. J. Kindermans, B. Schrauwen, D. Snyders, and A. van den Oord, "Sign language recognition using convolutional neural networks," in European Conf. Computer Vision (ECCV) Workshops, 2014, pp. 572-578.
- [6] S. C. Camgoz, S. Hadfield, O. Koller, H. Ney, and R. Bowden, "SubUNets: End-to-end hand shape and continuous sign language recognition," in IEEE Int. Conf. Computer Vision (ICCV), 2017, pp. 3075-3084.
- [7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in Advances in Neural Information Processing Systems (NeurIPS), vol. 25, 2012, pp. 1097-1105.
- [8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proc. IEEE Conf.

- Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778.
- [9] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2009, pp. 248-255.
- [10] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735-1780, 1997.
- [11] A. Graves, S. Fernandez, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in Proc. 23rd Int. Conf. Machine Learning (ICML), 2006, pp. 369-376.
- [12] Indian Sign Language Research and Training Centre (ISLRTC), "Indian Sign Language Dictionary," Ministry of Social Justice and Empowerment, Government of India, 2021.
- [13] A. K. Biswas, S. K. Ghosh, and P. K. Biswas, "Indian sign language recognition using deep learning," in Int. Conf. Computing, Communication and Automation (ICCCA), 2017, pp. 1-6.
- [14] S. Sharma and S. Singh, "Real-time sign language recognition using hand landmark detection and machine learning," Int. Journal of Computer Applications, vol. 183, no. 14, pp. 25-31, 2021.
- [15] O. Koller, S. Camgoz, H. Ney, and R. Bowden, "Deep learning of mouth shapes for sign language," in IEEE Conf. Computer Vision and Pattern Recognition (CVPR) Workshops, 2016, pp. 85-91.