Weakly Supervised Training of a Sign Language Recognition System Using Multiple Instance Learning Density Matrices

Daniel Kelly, John Mc Donald, *Member, IEEE*, and Charles Markham

***Abstract*—A system for automatically training and spotting signs from continuous sign language sentences is presented. We propose a novel multiple instance learning density matrix algorithm which automatically extracts isolated signs from full sentences using the weak and noisy supervision of text translations. The automatically extracted isolated samples are then utilized to train our spatiotemporal gesture and hand posture classifiers. The experiments were carried out to evaluate the performance of the automatic sign extraction, hand posture classification, and spatiotemporal gesture spotting systems. We then carry out a full evaluation of our overall sign spotting system which was automatically trained on 30 different signs.**

***Index Terms*—HMM, multiple instance learning (MIL), sign language recognition, size function, support vector machine (SVM), weakly supervised learning.**

I. Introduction

SIGN language spotting is the task of detecting and classifying signs in a signed sentence in a set vocabulary.

In this paper, we propose a system which attempts to solve the three major difficulties in automatic sign language spotting.

The first difficulty is that, when recognizing temporal gestures, the hand(s) must move from the end point of the previous gesture to the start point of the next gesture. These intergesture transition periods are called the movement epenthesis [1] and are not part of either of the signs. Thus, an accurate recognition system must be able to distinguish between valid sign segments and the movement epenthesis. We solve this by developing a temporal gesture model which addresses the problem of movement epenthesis detection without the need for explicit epenthesis training.

The second difficulty with developing automated techniques for sign language spotting is that an ideal sign language spotting system should give good recognition accuracy for the signers not represented in the training data set. User-independent hand posture recognition is particularly challenging as a user-independent system must cope with the geometric distortions due to different hand anatomies or different performances of the gestures by different persons. We address this problem by proposing a real-time user-independent hand posture classifier which we integrate with our temporal gesture model to classify signs.

The third difficulty in developing automatic sign language recognition systems is developing the algorithms which scale to large vocabularies. A difficulty with this is that previous research has typically required manual training data to be generated for each sign. This involved a signer or interpreter hand labeling each sign or sign phoneme such that a recognition system could be trained on the isolated samples of each sign. This can be a very time-consuming and expensive procedure and makes it difficult to expand sign vocabularies. To solve this problem, we propose a weakly supervised system, using our novel multiple instance learning (MIL) density matrix algorithm, which automatically extracts isolated samples of signs which can then be used to train our hand gesture models. The main contribution of this paper is that we propose techniques to address all three of these difficulties in a single recognition framework which can automatically learn and recognize signs. Currently, there exists no work which deals with all of these problems in a sign language recognition system.

A. Related Work

Gesture recognition systems which deal with temporal gestures, hand postures, and automatic training are briefly reviewed in this section. For a comprehensive review of the automatic sign language recognition, refer to the survey paper by Ong and Ranganath [2].

1) Temporal Gestures: While many works have proposed promising isolated gesture recognition techniques, natural gestures which occur in sign language are continuous; therefore, sign language recognition requires the spotting of the gesture from continuous videos (i.e., determining the start and end points of a meaningful gesture pattern). An approach to dealing with continuous recognition is to use hidden Markov models (HMMs) for implicit sentence segmentation. Starner *et al.* [3] and Bauer and Kraiss [4] model each word or subunit with an HMM and then train the HMMs with data collected from full sentences. A downside to this is that training on full sentence data may result in a loss in valid sign recognition accuracy due to the large variations in the appearance of all the possible movement epentheses that could occur between two signs.

Wang *et al.* [5] also use HMMs to recognize the continuous sign sequences with a 92.8% accuracy, although signs were assumed to end when no hand motion occurred. Assan and

Manuscript received January 4, 2010; revised April 22, 2010; accepted July 15, 2010. Date of publication September 23, 2010; date of current version March 16, 2011. This paper was recommended by Associate Editor S. Sarkar.

The authors are with the Computer Science Department, National University of Ireland Maynooth, Maynooth, Ireland (e-mail: dankelly@cs.nuim.ie).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCB.2010.2065802

Grobel [6] model the HMMs such that all transitions go through a single state while Gao *et al.* [7] create separate HMMs that model the transitions between each unique pair of signs that occur in sequence. Vogler and Metaxas [8] also use an explicit epenthesis modeling system where one HMM is trained for every two valid combinations of signs. While these works have had promising results in gesture recognition and movement epenthesis detection, the training of such systems involves a large amount of extra data collection, manual data labeling, model training, and recognition computation due to the extra number of HMMs required to detect the movement epenthesis. Few researchers have addressed the problem of the movement epenthesis without explicitly modeling these movements. Yang *et al.* [9] proposed an American Sign Language recognition method based on an enhanced level building algorithm and a trigram grammar model. Their method was based on a dynamic programming approach to spot signs without explicit movement epenthesis models. The recognition rate was 83% with 39 signs, articulated in 25 different sentences, but it is unclear how the system would perform if the grammar model was trained on a real-world corpus where there existed more variability than there existed in the 150 sentences used to train their grammar model.

Yang *et al.* [10] develop threshold models in a conditional random field model which performs an adaptive threshold for distinguishing between signs in a vocabulary and nonsign patterns. The experiments show that their system can spot signs from continuous data with an 87.0% detection rate from a vocabulary of 48 signs wherein the system was trained on ten isolated samples of each of the 48 signs. The system was then tested on continuous sentences which contained 480 samples of the signs in the sign vocabulary. In this paper, we propose an HMM-based gesture recognition framework which accurately spots and classifies motion-based gestures and detects the movement epenthesis.

2) Hand Postures: Although there are many methods in the literature which have described promising hand posture recognition systems, most of these methods have been evaluated on signer-dependent data sets [2]. Analogous to speaker independence in speech recognition, an ideal sign recognition system should give good recognition accuracy for the signers not represented in the training data set. The user-independent hand posture recognition is particularly challenging as a user-independent system must cope with the geometric distortions due to different hand anatomies or different performances of the gestures by different persons. Triesch and von der Malsburg [11] proposed a user-independent hand posture recognition system using elastic graph matching which reported a recognition rate of 92.9% when classifying ten hand postures. The elastic graph matching method showed very promising results but was reported to have a high computational complexity with the method requiring several seconds to analyze a single image. One of the goals of this paper was to incorporate an accurate user-independent and real-time hand posture classifier into a sign recognition system.

3) System Training: The previous works for the recognition of temporal gestures and hand postures, described in Section I-A1 and A2, carry out training using manually labeled

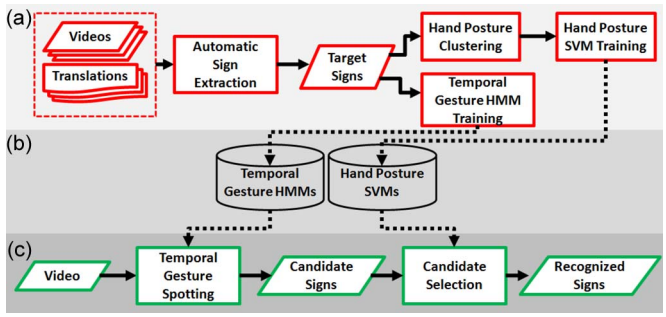


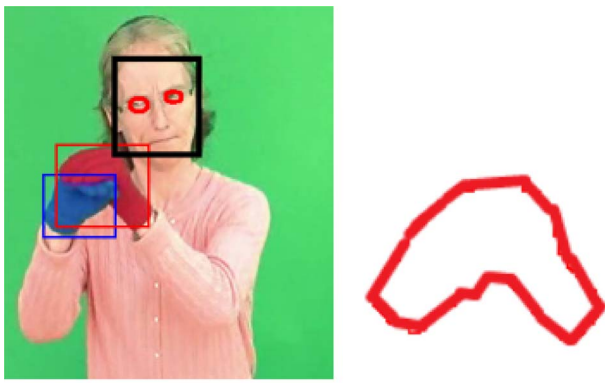
Fig. 1. Flowchart of our proposed automatic sign training and sign spotting framework. (a) System Training. (b) Sign Classifiers. (c) Sign Recognition.

training data. The automatic labeling of signs is an extremely challenging task and is demonstrated by the limited works dealing with this problem.

Buehler *et al.* [12] develop a MIL technique to extract given sign language words from videos annotated with weakly aligned subtitles. This technique allows the automatic extraction of isolated signs without manual labeling. The results show that their technique was able to find 65% of the words from a vocabulary of 210 words. Cooper and Bowden [13] also implement an automated method to extract signs from subtitles. A temporally constrained adaptation of apriori mining is used to extract similar regions of a video, with the aid of a proposed contextual negative selection method. The system was tested on 23 signs which occurred in a 30-min video. Their proposed method was able to correctly isolate, on average, 53.7% of the signs. Nayak *et al.* [14] proposed an unsupervised approach to extract the instances of continuous basic units of the signs, which are called signemes, from continuous sentences. They automatically extracted signeme models, using iterative conditional models, given a set of sentences with one common sign. The experiments showed that their method was able to correctly extract ten key signs from 136 sentences with an error rate of 13%. While these works have proposed promising techniques for the automatic extraction of target sign segments, no work has further developed these techniques to automatically train a full sign language spotting system on temporal and hand posture information.

B. System Overview

The goal of this paper is to develop a weakly supervised system which automatically extracts isolated samples of signs which can then be used to train temporal gesture and hand posture recognition frameworks. Section III presents the gesture similarity functions which will be utilized by our proposed automatic training framework. Section IV presents our automatic sign extraction system which utilizes the text translations of the videos to automatically extract isolated samples of target words. Section V presents our framework for training classifiers which model the automatically extracted signs. Section VI presents our continuous sign spotting system which utilizes the automatically trained classifiers. Section VII presents the experimental results. Fig. 1 shows a visualization of the flow of our automatic training and sign spotting systems.



(a) (b)

Fig. 2. Extracted features from image.

II. Feature Extraction

The focus of this paper is to develop a pattern recognition framework for the automatic spotting of sign language gestures. For completeness and to introduce feature notation, we briefly describe the feature tracking methods, although we do not consider it to be the innovative part of our work. The tracking of the hands is performed by tracking colored gloves [see Fig. 2(a)] using the mean shift algorithm [15].

In this paper, we build a multichannel sign language spotting system which utilizes hand movements, positions, orientations, and postures to classify sign language sentences. In the remainder of this paper, we treat these four components as two distinct information channels described in the following paragraphs.

Hand Posture Channel: The key feature used for the analysis of the hand posture channel is the external contour made by the hand [see Fig. 2(b)]. To extract this feature from the image, we segment the glove region in the image using a back projection image computed during the mean shift algorithm. We then extract the external contour of the hand blob, which we denote as C. The dominant hand is used to convey most hand shape information, and thus, we consider only the shape of the dominant hand when analyzing the hand shape. When the dominant and nondominant hands overlap, we extract a single contour made around the dominant and nondominant hands as illustrated in Fig. 2(a) and (b).

Spatiotemporal Gesture Channel: Hand positions are used to describe the spatiotemporal gestures, and the face and eye positions are also used as spatiotemporal gesture cues. The face and eye detection is carried out using a cascade of boosted classifiers working with the haar-like features proposed by Viola and Jones [16]. A set of public domain classifiers [17], for the face, left eye, and right eye, are used in conjunction with the OpenCV implementation of the haar cascade object detection algorithm. We define a raw spatiotemporal gesture feature vector F extracted from each image. Each feature vector F contains a description of the following: the right hand position (RH_x, RH_y), left hand position (LH_x, LH_y), face position (FC_x, FC_y), face width (FW), left eye position (LE_x, LE_y), and right eye position (RE_x, RE_y). During sign language communication, the parts of the face may become occluded by the hands. If a face part cannot be detected by the computer vision algorithm, then we use the previous points for the occluded parts of the face.

III. Gesture Similarity Functions

Before describing our MIL density matrix algorithm which we utilize to automatically extract isolated samples of sign language target words, we first describe the gesture similarity functions used as part of our MIL algorithm. In order to compare the spatiotemporal gestures and hand postures, we require a measure of the similarity between the gesture frames. We implement two similarity functions, one for spatiotemporal gesture similarity and one for hand posture similarity.

A. Spatiotemporal Gesture Similarity Function

As described in Section II, the raw feature vector describing the hand, head, and eye positions is defined as F. From the raw feature vector, we calculate a feature vector $\widehat{f} = \{RP_x, RP_y, V_x, V_y, D_H\}$, where RP_x and RP_y describe the position of the hands relative to the eyes, V_x and V_y describe the velocity of the movement of the hand, and D_H describes the distance between the two hands. We choose this feature vector as it holds information about the movement of the hand through the velocity vector while also holding information about the location of the articulation through the relative position vector. The distance between each hand is also useful as it stores information on how each hand is moving relative to the other hand. This feature vector is calculated for both the right and left hands, which we denote as $\widehat{f}_R$ and $\widehat{f}_L$, respectively.

Given two feature vectors $\widehat{F}$ and $\widehat{F}'$, where $\widehat{F} = \{\widehat{f}_R, \widehat{f}_L\}$, we define the similarity function $D^G()$ as the weighted sum of the distance for the right and left hands where ω_R and ω_L define the right and left weights, respectively

$$D^G(\widehat{F}, \widehat{F}') = \omega_R \left(1 - d^G\left(\widehat{f}_R, \widehat{f}'_R\right)\right) + \omega_L \left(1 - d^G\left(\widehat{f}_L, \widehat{f}'_L\right)\right). \tag{1}$$

The distance between the individual hands $d^G()$ is then calculated by first calculating three individual distances between each of the three spatiotemporal gesture features (movement, position, and hand distance) as follows:

$$d^G_{\text{pos}}(\widehat{f}, \widehat{f}') = \sqrt{(RP_x - RP'_x)^2 + \left(RP_y - RP'_y\right)^2} \tag{2}$$

$$d^G_{\text{mov}}(\widehat{f}, \widehat{f}') = \sqrt{(V_x - V'_x)^2 + \left(V_y - V'_y\right)^2} \tag{3}$$

$$d^G_{\text{dis}}(\widehat{f}, \widehat{f}') = \sqrt{(D_h - D'_h)^2}. \tag{4}$$

Using each of the three distances, the overall distance can then be calculated using a normalized sum of the three distances

$$d^G(\widehat{f}, \widehat{f}') = \omega_{\text{pos}} d^G_{\text{pos}}(\widehat{f}, \widehat{f}') + \omega_{\text{mov}} d^G_{\text{mov}}(\widehat{f}, \widehat{f}') + \omega_{dis} d^G_{dis}(\widehat{f}, \widehat{f}') \tag{5}$$

where $\omega_x = (1/\max^G_x * 3)(x \in \{\text{pos}, \text{mov}, \text{dis}\})$ and $\max^G_x$ is the maximum distance $d^G_x()$ calculated between all pairs of frames in a data set. The scale factors ω_x normalize each

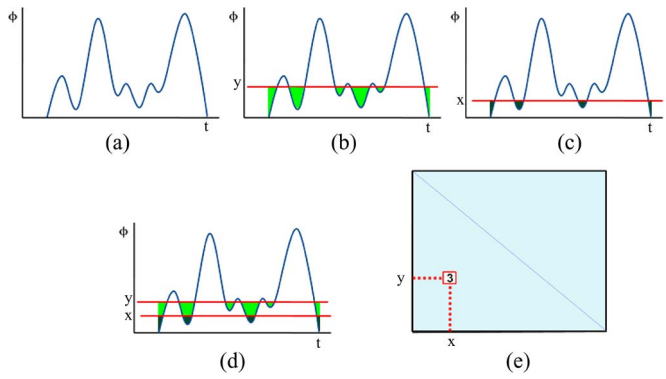


Fig. 3. (a) Graph of some measuring function φ. (b) Shaded region $\equiv \varphi \leq y$. (c) Shaded region $\equiv \varphi \leq x$. (d) Graph depicting $\varphi \leq y$ and $\varphi \leq x$. (e) Graph of size function ℓ_φ with current $\ell_\varphi(x, y) = 3$.

distance measure such that $0 \leq d^G(\widehat{f}, \widehat{f}') \leq 1$ for all pairs of features $\widehat{f}$ and $\widehat{f}'$.

B. Hand Posture Similarity Function

This paper presents a method of calculating a hand posture similarity function from a Hu moment feature set and our novel eigenspace size function shape representation calculated from the external hand contour C.

1) Hu Moments: The Hu moments [18], which are a reformulation of the nonorthogonal centralized moments, are a set of translation, scale, and rotation invariant moments. The set of Hu moments $I = \{I_1, I_2, I_3, I_4, I_5, I_6, I_7\}$ is calculated from the hand contour described in Section II. The Hu moment distance between two hand contours C_k and C_l is calculated using the Hu moment comparison metric implemented in the OpenCV library [19] and is described in

$$d^{\text{hu}}(C_k, C_l) = \sum_{I=1}^{7} \left| \Lambda\left(I(C_k)[i]\right) - \Lambda\left(I(C_l)[i]\right) \right| \quad (6)$$

$$\Lambda(\text{hu}) = \frac{1}{\text{sign}(\text{hu}) \times \log(\text{hu})}. \quad (7)$$

2) Size Functions: Size functions are integer-valued functions which represent both the qualitative and quantitative properties of a visual shape [20]. They have recently been shown to perform well at carrying out the user-independent recognition of the hand postures used in sign language [21]. For a given hand contour C, let G be a graph whose vertices are the points of the contour. Let φ, the measuring function, be a real-valued function defined on the vertices of G [see Fig. 3(a)]. The size function ℓ_φ, induced by the measuring function φ, is an integer-valued function defined on a real pair (x, y) according to the following algorithm.

1) Find the subgraph $G_{\varphi \leq y}$ of G determined by the points p with $\varphi(p) \leq y$ [see Fig. 3(b)].
2) Identify the connected components of $G_{\varphi \leq y}$ [see Fig. 3(b)].
3) The size function ℓ_φ at point (x, y) equals the number of connected components of $G_{\varphi \leq y}$ which contain at least a vertex with $G_{\varphi \leq x}$ [see Fig. 3(c)–(e)].

When identifying the number of connected components of the graphs $G_{\varphi \leq y}$ and $G_{\varphi \leq y}$, it should be noted that the graphs are circular. Therefore, in Fig. 3(d), there exist three connected components of $G_{\varphi \leq y}$, which contain at least a vertex with $G_{\varphi \leq x}$, and not four which would be the case if the graphs were not circular. This ensures that the number of connected components will remain the same independent of the start and end points for which the measuring function was computed.

The theory of the size functions does not identify a formal tool to resolve a suitable measuring function. Therefore, a suitable measuring function must be found heuristically. As defined by Stokoe's model [22], a hand posture is made up of the shape and orientation of the hand. Thus, for the application of classifying the hand postures performed in sign language, the measuring function chosen must be sensitive to the orientation changes of the hand (although a suitable classifier should not be sensitive to the minor changes in the hand orientation). With Stokoe's model in mind, the measuring function model proposed in this paper utilizes a family of measuring functions indexed by the angle $\theta \epsilon \{0, 1(2\pi/N_\Theta), 2(\pi/2N_\Theta), \ldots, (N_\Theta - 1)(2\pi/N_\Theta)\}$, where N_Θ is the total number of rotation angles used. Each measuring function $\varphi_\theta = D_\theta(p)$ is a function which rotates p about the center of gravity of G and measures the distance between the horizontal axis and a point p on the graph G_θ. The horizontal axis is a line which passes through the minimum point of G_θ. For every θ, a size function $\ell_{\varphi\theta}$ is generated, resulting in a set of size functions $\Gamma_\varphi = \{\ell_{\varphi 1}, \ell_{\varphi 2}, \ldots, \ell_{\varphi N_\Theta}\}$. The sensitivity of the system to orientation can then be controlled by means of adjusting N_Θ.

To illustrate the concept of the size functions and their application in analyzing the hand postures used in sign language, a specific example will be used. The hand contour is extracted using the method described in Section II. For this example, let $N_\Theta = 4$. The hand contour is rotated to each of the four θ values [see Fig. 4(a)], the measuring function is applied to each of the four rotated contours [see Fig. 4(b)], and the size functions are then generated from each of the measuring functions [see Fig. 4(c)].

3) PCA and Size Functions: In order to quantify the shape information held in a size function, we utilize a more robust method of shape representation, as compared to the unmodified normalized size function representation used in [20] and [23]. We apply an important improvement to the size function technique by implementing a size function feature which is more robust to noise and small changes in shape which occur from different people [21]. The technique is a method of incorporating eigenspace information into the hand posture feature using a principal component analysis (PCA). The PCA computes a linear projection from a high-dimensional input space to a low-dimensional feature space. It is a statistical technique used for finding the patterns in data of high dimensions. Since we are looking for the similarities and differences between two size functions, we can utilize the PCA in order to reduce the influence of noise and small variations in shape by different persons and to highlight the portions of the size function useful for user-independent hand posture recognition. To calculate the principal components of a size function, the size function is described as an $N \times N$ matrix $X_\theta = \ell_{\varphi\theta}$. From the size function

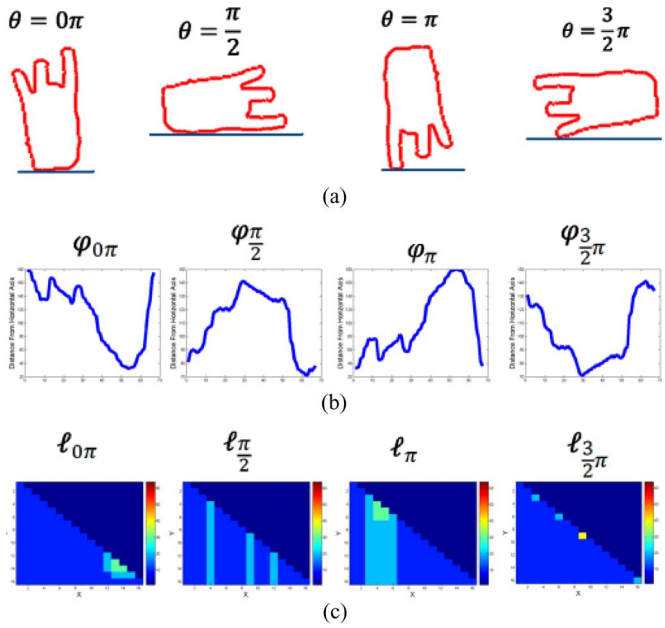


Fig. 4. (a) θ rotation applied to hand contour. (b)Measuring function φ_θ applied to hand contour. (c) Size function $\ell_{\varphi\theta}$ generated.

matrix, we calculate a covariance matrix Σ. The eigenvector matrix V_θ and the eigenvalue matrix W_θ are then computed from the covariance matrix and then sorted in order of decreasing eigenvalue. This records the components in order of significance, the eigenvector with the highest eigenvalue being the *principal component*. Therefore, the first column of V_θ, a $1 \times N$ vector, corresponds to the principal component vector. Kelly *et al.* [21] have shown that this method of combining the size functions and PCA improves upon the original size function and performs well at discriminating between the hand postures independent of the person performing them.

4) Size Function Distance: We define $d^{\mathrm{SF}}(C_k, C_l)$ as the distance measure computed between the size functions of contours C_k and C_l: For each contour C, we compute the set of size functions $\Gamma_\varphi(C)$ and then calculate the eigenvectors and eigenvalues for that set of size functions. For each θ, we choose only the first P eigenvectors, resulting in a set of matrices $M_\theta(C)$ with dimensions $N \times P$ and a set of eigenvalues $W_\theta(C)$ of dimension P. We define our weighted eigenspace size function $\zeta_\theta(C)$ in

$$\zeta_\theta(C)[p,i] = M_\theta(C)[p,i] \times \varpi_\theta(C,p) \times \varrho_\theta(C) \tag{8}$$

where $\varpi_\theta(C,x)$ is a weighting factor for each eigenvector x associated with the size function indexed by θ according to

$$\varpi_\theta(C,x) = \frac{W_\theta(C)[x]}{\sum_{k=0}^{K} W_\theta(C)[k]}. \tag{9}$$

The second weighting factor $\varrho_\theta(C)$ is calculated for each set of eigenvectors associated with the size function indexed by θ such that the size function with the greatest total variance gets a greater weighting according to

$$\varrho_\theta(C) = \frac{\sum_{p=0}^{P} W_\theta(C)[p]}{\sum_{\theta=0}^{N_\Theta} \sum_{p=0}^{P} W_\theta(C)[p]}. \tag{10}$$

The distance $d^{\mathrm{SF}}(C_k, C_l)$ between the hand contours C_k and C_l is then defined as the Euclidian distance between the weighted eigenspace size functions

$$d^{\mathrm{SF}}(C_k, C_l) = \sqrt{\sum_{\theta=0}^{2\pi} \sum_{p=0}^{P} \sum_{I=0}^{N} \left(\zeta_\theta(C_k)[p,i] - \zeta_\theta(C_l)[p,i]\right)^2}. \tag{11}$$

5) Hand Shape Similarity Function: The overall similarity between the two hand contours C and C' is then calculated using a normalized sum of the size function and Hu moment distances as follows:

$$D^H(C,C') = 1 - \left(\omega_{\mathrm{Sf}} d^{\mathrm{SF}}(C,C') + \omega_{\mathrm{Hu}} d^{\mathrm{hu}}(C,C')\right) \tag{12}$$

where $\omega_x = (1/\max_x^H * 2)(x \in \{\mathrm{Sf}, \mathrm{Hu}\})$ and $\max_x^H$ is the maximum distance $d_x^H()$ calculated between all pairs of frames in the data set. The scale factors ω_x normalize the similarity measure such that $0 \leq D^H(C,C') \leq 1$ for all pairs of hand contours C and C'.

It should be noted that in the experiments, the parameter combination, which produced the best performance, was $N = 16$, $N_\Theta = 6$, and $P = 1$, where N, N_Θ, and P correspond to the size of the size function, the number of the graph rotations, and the number of the principal components, respectively.

IV. Automatic Sign Extraction

Given a target word, a set of video sequences, and the corresponding weakly aligned text translations, the goal is to automatically identify which frames in each video, if any, contain the target word. Furthermore, we must also identify which frames contain the key hand postures related to the target word. The only information available about the video sequences is from the text translations; thus, labels are at a sequence level and not at the required frame level. The difficulty in this problem is that no one-to-one mapping exists between the English translations and the signs since the ordering of sign language is different in the English translations. Another difficulty with the learning task is that there exist ambiguities in the translation task where the same sign may have different translations or a word may appear in the text translation but the corresponding sign may not occur in the video. This introduces a significant correspondence problem between the translation and overlapping video sequence, so the supervision is weak.

In order to find the frame level labels, we can formulate the task as a MIL problem. Using a MIL notation, we define a video sequence as a bag, where the set of positive bags $B^+ = \{B_0, \ldots, B_{N_B}\}$ is the set of videos in which the target word occurs and N_B is the number of bags in the set of bags B^+. The videos contained within the set of bags should be chosen such that the target word appears with the most frequency compared to the other words in the text translations. This can be done automatically by a process of word counting. Each bag B_i represents a sequence of features for each frame $B_i = \{B_i[0], \ldots, B_i[N_{Bi}]\}$, where N_{Bi} is the number of frames in a bag B_i. To reduce the impact of the translation ambiguities, we develop a MIL solution that requires only positive bags. In

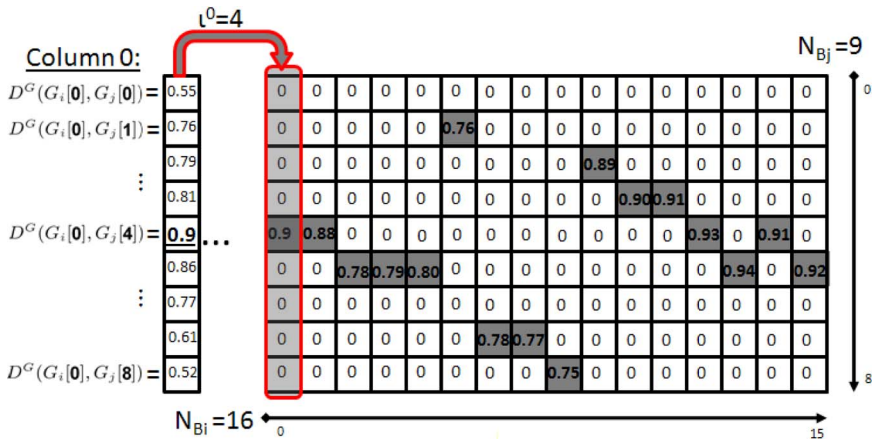


Fig. 5. Visual representation of the calculation of the spatiotemporal gesture comparison matrix $\overline{G_{ij}}$.

a traditional MIL framework, a learner uses both the positive and negative bags to learn the intended concept, where a negative bag represents a sequence of features which do not lie within the concept. In the case of the sign language videos, a negative bag would represent a set of videos which did not contain the text translation for a target word. Since there are ambiguities in the translations, there is no guarantee that the corresponding sign does not occur in the videos. The work of Buehler *et al.* [12] acknowledges the difficulty in estimating the errors in negative bags without manual labeling, and their work uses a heuristic to determine the errors in the negative bags. We address the problem of the errors in the negative bags by developing a MIL framework which requires the positive bags only. This eliminates the need to automatically identify the errors in the negative bags and thus limits the type of translation ambiguities, which our system must deal with, to the errors in the positive bags. The translation ambiguities which our system must identify are thus limited to the situations where the text translation contains a target word but the corresponding sign does not occur in the video.

A. MIL Density Matrix

We develop a novel MIL density matrix algorithm, inspired by diverse density MIL [24], to label videos at the frame level. Since hand postures and spatiotemporal gestures are independent channels, we propose a multichannel MIL density matrix algorithm. We define seperate bags for each channel, denote the set of spatiotemporal gesture bags as G^+, and denote the set of hand shape bags as H^+. The first step in our MIL density matrix algorithm is to compute the comparison matrices $\overline{G_{ij}}$ and $\overline{H_{ij}}$ in order to compare each pair of bags (G_i, G_j) and (H_i, H_j), respectively. Each of the comparison matrix $\overline{G_{ij}}$ and $\overline{H_{ij}}$ corresponds to an $N_{Bj} \times N_{Bi}$ matrix. For each column t $(0 \leq t < N_{Bi})$ in the spatiotemporal gesture comparison matrix $\overline{G_{ij}}$, we store only the best similarity measure between $G_i[t]$ and all the features in G_j. All the other entries in that column are set to zero. Each element of the comparison matrix, $\overline{G_{ij}}[t, \iota^t]$, is calculated as defined in (13) and (14), where ι^t is the frame index of G_j which is most similar to $G_i[t]$. Fig. 5 shows a visual example of the calculation of the spatiotemporal comparison matrix where we show the calculation of the similarity measures for column 0 by calculating all the similarities $[D^G(G_i[0], G_j[0]), \ldots, D^G(G_i[0], G_j[8])]^T$. In column 0, it can be seen that the index ι^0 representing the most similar gesture frame is the fourth frame of G_j. The fourth element of the first column in the comparison matrix $\overline{G_{ij}}$ then stores the similarity measure between $G_i[0]$ and $G_j[4]$ while all the other entries in that column are set to zero.

The hand posture comparison matrix $\overline{H_{ij}}$ is calculated in a similar fashion. For each column t $(0 \leq t < N_{Bi})$ in the hand posture comparison matrix $\overline{H_{ij}}$, we store the similarity between $H_i[t]$ and $H_j[\iota^t]$, where ι^t is the frame index of G_j which best matches $G_i[t]$ [as defined in (15)]. For example, in Fig. 5, it was identified that the most similar gesture frame for column 0 was the fourth frame of G_j. This means that, in the corresponding hand posture comparison matrix $\overline{H_{ij}}$, the fourth element of the first column stores the similarity measure between $H_i[0]$ and $H_j[4]$ while all the other entries in that column are set to zero. Using the most similar spatiotemporal comparison index allows us to prioritize the spatiotemporal gestures. This allows us to match the hand poses which help us to discriminate between the spatiotemporal gestures that have similar spatiotemporal patterns but differ only in the hand pose. Each element of the comparison matrices is calculated using the similarity functions D^G and D^H which we define in Section III. It should be noted that the similarity functions D^G and D^H are implemented such that similar frames return values close to one while the frames which are not similar return values close to zero. The threshold values τ^G and τ^G are implemented such that the similarity comparisons $\overline{G_{ij}}[t, \iota^t]$ and $\overline{H_{ij}}[t, \iota^t]$ only include the video frames which are within a set similarity. This reduces the effect that nonsimilar frames have on the overall detection of target words, and through experiments, we have found that a threshold value of 0.75, for both τ^G and τ^G, has performed well

$$\iota^t = \underset{0 \leq \tilde{\iota} < N_{Bj}}{\arg\max} \quad D^G\left(G_i[t], G_j[\tilde{\iota}]\right) \tag{13}$$

$$\overline{G_{ij}}[t, \iota^t] = \begin{cases} D^G\left(G_i[t], G_j[\iota^t]\right) & \text{if } D^G\left(G_i[t], G_j[\iota^t]\right) > \tau^G \\ 0 & \text{else.} \end{cases} \tag{14}$$

$$\overline{H_{ij}}[t, \iota^t] = \begin{cases} D^H\left(H_i[t], H_j[\iota^t]\right) & \text{if } D^H\left(H_i[t], H_j[\iota^t]\right) > \tau^H \\ 0 & \text{else.} \end{cases} \tag{15}$$

For convenience, we denote a bag as B hereinafter, but the following computations are calculated for both the spatiotemporal gesture bags G and the hand pose bags H by replacing the notation B and $\mathbf{b}$ with H and $\mathbf{h}$ or G and $\mathbf{g}$ accordingly. After computing the comparison matrix $\overline{B_{ij}}$, we convert it to a comparison vector $\overline{\mathbf{b}_{ij}}$ as shown in (16), where each vector $\overline{\mathbf{b}_{ij}}$ corresponds to an N_{Bi}-dimensional row vector. Each comparison vector is used as a measure of how closely matched each frame, in bag B_i, is to the most similar frame in bag B_j

$$\overline{\mathbf{b}_{ij}}[t] = \sum_{n=0}^{N_{Bj}} \overline{B_{ij}}[t, n]. \tag{16}$$

Each element $\overline{\mathbf{b}_{ij}}[t]$ represents the sum of the similarity metrics in the tth column of the comparison matrix $\overline{B_{ij}}$. Since the tth column in $\overline{B_{ij}}$ contains, at most, one similarity metric, then the element of the comparison vector $\overline{\mathbf{b}_{ij}}[t]$ corresponds to the same similarity metric. We also define a transposed comparison vector, as shown in (17), where each element $\overline{\mathbf{b}_{ij}}'[t]$ of the N_{Bj}-dimensional column vector represents the sum of the similarity metrics in the tth row of the comparison matrix. Each transposed comparison vector is used as a measure of which frames in B_j has the most similarities with the bag B_i

$$\overline{\mathbf{b}_{ij}}'[t] = \sum_{n=0}^{N_{Bi}} \overline{B_{ij}}[n, t]. \tag{17}$$

Each comparison vector $\overline{\mathbf{b}_{ij}}$ represents a comparison between the ith and jth bags. The comparison vector does not have any representation of how the ith and jth bags interact with all the other bags in the set of bags. In order to build a full comparison model between each pair of bags (B_i, B_j), we must not only understand how B_i and B_j interact with each other but also how the pair of bags (B_i, B_j) interact with all the other bags in the set of bags B^+. In order to do this, we first build a model of B_i compared to all other bags, followed by a model of B_j compared to all other bags. We then use both these models to understand how B_i and B_j interact. We compare B_i to all the other bags by summing all the comparison vectors $\overline{\mathbf{b}_{ik}}, \forall k \in B^+$ [see (18)]. We then compare B_j to all the other models by summing all the transposed comparison vectors $\overline{\mathbf{b}_{kj}}', \forall k \in B^+$ [see (18)]

$$\mathbf{b}_i = \sum_{k=0}^{N_B} \overline{\mathbf{b}_{ik}} \quad \mathbf{b}_j' = \sum_{k=0}^{N_B} \overline{\mathbf{b}_{kj}}' \tag{18}$$

where $\mathbf{b}_i$ corresponds to an N_{Bi}-dimensional row vector and $\mathbf{b}_j'$ corresponds to an N_{Bj}-dimensional column vector. Fig. 6 shows the spatiotemporal gesture comparison matrices $\overline{G_{ij}}$ for a sample set of five positive bags for the target sign "Alot". Each matrix, at coordinates (G_i, G_j), corresponds to a visual representation of comparison matrices $\overline{G_{ij}}$, where the dark pixels represent a high similarity between the gesture frames. The hand-labeled time segments in each bag represent the frames of each video which were hand labeled as being a target sign. Fig. 6 also shows a visualization of the summed comparison vectors $\mathbf{g}_1$ and $\mathbf{g}_5'$ being computed from the corresponding set

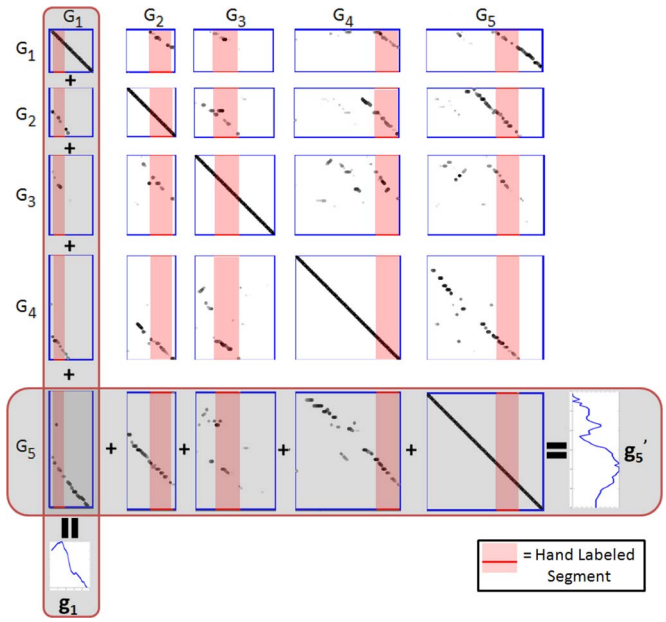


Fig. 6. Visualization of comparison matrices $\overline{G_{ij}}$ and calculation of summed comparison vectors $\mathbf{g}_1$ and $\mathbf{g}_5'$.

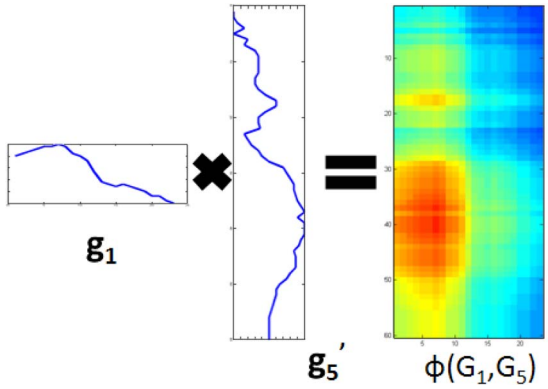


Fig. 7. Visualization of density matrix $\Phi(G_1, G_5)$.

of comparison matrices. It can be seen from the comparison matrices that there exist high areas of similar frames where the hand-labeled sections occur, illustrating the power of our technique for measuring the areas of similarity.

Following the calculation of the summed comparison vectors, we then compute the density matrix $\Phi(B_i, B_j)$, which measures the interaction between the pair of bags B_i and B_j, by multiplying the summed comparison vectors to calculate an $N_{Bj} \times N_{Bi}$ matrix

$$\Phi(B_i, B_j) = \mathbf{b}_i \mathbf{b}_j'. \tag{19}$$

Fig. 7 shows a visualization of the computation of the density matrix $\Phi(G_1, G_5)$ computed from the summed comparison vectors shown in Fig. 6.

We then compute a density vector $\Psi(B_i)$ for bag B_i using the set of density matrices $\{\Phi(B_i, B_1), \ldots, \Phi(B_i, B_{N_B})\}$. Each element $\Psi(B_i)[t]$ of the density vector is calculated by averaging the tth column of each of the density matrices $\Phi(B_i, B_j)$

$$\Psi(B_i)[t] = \frac{1}{N_B} \sum_{j=0}^{N_B} \sum_{k=0}^{N_{Bj}} \frac{\Phi(B_i, B_j)[t, k]}{N_{Bj}}. \tag{20}$$

An overall density vector Ψ_i is then calculated from the inner product of the hand posture and spatiotemporal gesture density vectors

$$\Psi_i = \Psi(H_i) \cdot \Psi(G_i) \tag{21}$$

where $\Psi(H_i)$ and $\Psi(G_i)$ denote the density vector of the ith bag for the hand posture and spatiotemporal gestures, respectively. We then normalize the density vectors such that all density values have a maximum value of one and a minimum of zero (i.e., $(0 \leq \Psi_i[t] \leq 1) \forall t \in N_{Bi}, \forall i \in B^+$).

B. Automatic Sign Labeling

Given the density vector Ψ_i, we now label the frames in the corresponding video. In order to account for the translation ambiguities, we must detect whether the target sign was actually performed in the video even though the translation information specifies that it does occur. In order to do this, we flag the video sequences which have no frames with a high similarity to the other video sequences in the set of positive bags. A video sequence is classified as a noneligible sentence if the maximum frame density $\Psi_i[t^{\max}]$ falls below a set threshold, where $t^{\max}$ is then defined as the frame with the maximum density

$$t^{\max} = \arg\max_{\hat{t}} \Psi_i[\hat{t}]. \tag{22}$$

From the experiments, we observed that a threshold value of 0.55 performed best when thresholding the maximum frame density. Any sequence which is classified as a noneligible sentence of the target sign is automatically discarded and not considered for the remaining automatic training steps for the current target word. The positive sequences are sequences which have a corresponding maximum frame density which exceeds the threshold. We process all positive sequences using our density labeling algorithm which we now describe in the following sentences. We first apply a 1-D Gaussian filter $g(x)(\sigma = 1.5)$ to the density vector Ψ_i to create a blurred density vector $\widetilde{\Psi}_i = \Psi_i * g$. We define the function $L_{\min}(X, t)$ to be the index of the local minima of distribution X at position t. We then find the local density minima as defined in

$$t^{\min S} = L_{\min}(\widetilde{\Psi}_i, t^{\max} - 1) \tag{23}$$

$$t^{\min E} = L_{\min}(\widetilde{\Psi}_i, t^{\max} + 1). \tag{24}$$

The indices of the local minima are then used to calculate a modified density vector as follows:

$$\widetilde{\Psi}_i^* = \Psi_i - \frac{\Psi_i[t^{\min S}] + \Psi_i[t^{\min E}]}{2}. \tag{25}$$

In order to identify the subsequence of the density vector $\widetilde{\Psi}_i^*$, which corresponds to the sequence of the video in which the target sign is performed, we find the maximum sum contiguous subsequence (MSCS) of $\widetilde{\Psi}_i^*$. The MSCS of $\widetilde{\Psi}_i^*$ corresponds to the subsequence with the maximum value of $\sum_t^T \widetilde{\Psi}_i^*[t]$. The start and end frames t_i^s and t_i^e of the target word within bag i are then defined as the indices in which the MSCS begins and ends. The spatiotemporal gesture and hand posture sequences which correspond to the target word are then defined as $\widehat{G_i} = \{G_i[t_i^s], \ldots, G_i[t_i^e]\}$ and $\widehat{H_i} = \{H_i[t_i^s], \ldots, H_i[t_i^e]\}$, respectively. Fig. 8 illustrates the density vectors of the target sign Alot in five different sequences as well as the frames which were hand labeled as containing the target sign and the frames which our automatic sign labeling technique labeled. The density vectors of the five sequences shown in Fig. 8 were calculated using the same spatiotemporal gesture density matrices shown in Fig. 6 along with the hand shape density matrices.

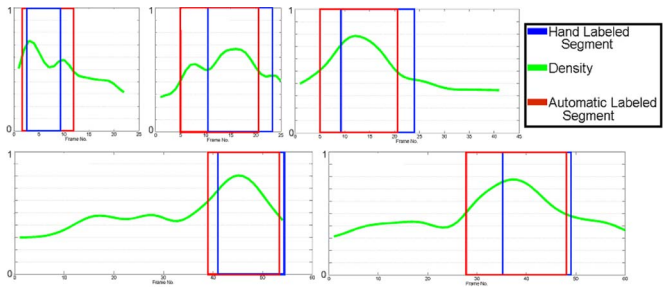


Fig. 8. Vector $\widetilde{\Psi}_i$ and automatically labeled target signs Alot.

V. Training and Classification

For a given sign, we recognize both the spatiotemporal gestures and hand postures independent of each other and then combine the recognition results to make an overall sign classification. Our spatiotemporal gesture spotting system is based on an HMM framework which models the spatiotemporal pattern of the hand movements over time. The hand posture recognition framework is built on a support vector machine (SVM) framework which is automatically trained to classify the key hand postures within the signs.

A. Spatiotemporal Gesture HMM Framework

We propose an HMM-based spatiotemporal gesture spotting framework which detects the movement epenthesis and classifies the spatiotemporal gestures as one of a number of pretrained signs. We use the compact notation $\lambda = \{A, B, \pi\}$ to indicate the complete parameter set of an HMM, where A is a matrix storing transition probability and a_{ij} denotes the probability of making a transition between the states s_i and s_j. B is a matrix storing output probability for each state, and π is a vector storing initial state probability.

1) HMM Threshold Model: Lee and Kim [25] proposed a single-channel HMM threshold model using discrete observations to recognize a set of distinct gestures. Kelly *et al.* [26] expand on the work of Lee and Kim to develop an HMM threshold model system which models the continuous multidimensional sign language observations within a parallel HMM network to recognize the two hand signs and identify the movement epenthesis. A specific HMM, called a threshold model, is created to model the movement epenthesis by calculating the likelihood threshold of an input gesture and to provide a confirmation mechanism for provisionally matched gesture patterns.

For a network of HMMs $\Lambda = \{\lambda^1, \lambda^2, \ldots, \lambda^W\}$, where λ^w is a dedicated gesture HMM used to calculate the likelihood that the input gesture belongs to gesture class w, a single threshold model $\overline{\lambda}$ is created to calculate the likelihood threshold for each of the dedicated gesture HMMs. Each dedicated gesture HMM is trained to model a specific combination of gesture subpatterns while the threshold HMM is created to model an arbitrary combination of gesture subpatterns. It is not in the scope of this paper to describe the threshold model in detail, and readers should consult the works of Lee and Kim [25] and Kelly *et al.* [26] for a more detailed discussion on the HMM threshold model technique.

2) Parallel HMM Training: When recognizing a two-handed spatiotemporal gesture, a parallel HMM is required to model the left and right hands [27]. We implement a parallel HMM threshold model system which initializes and trains a dedicated parallel HMM denoted as λ^w. We denote the parallel HMM as $\lambda^w = \{\lambda_L^w, \lambda_R^w\}$, where λ_L^w and λ_R^w are HMMs which model the left and right hand gestures, respectively.

The parallel HMMs are trained using an automated HMM initialization and training technique, utilizing iterative clustering and the Baum–Welch and Viterbi realignment processes, proposed by Kelly *et al.* [26]. Each HMM is trained on data from all spatiotemporal gesture subsequences $\widehat{G_k}^{Lw}$ and $\widehat{G_k}^{Rw}$ $(1 \leq k \leq K)$ extracted using our automatic sign labeling technique. We define k as the index for the kth training example for target word w, K as the total number of training examples for target word w, and $\widehat{G_k}^{Lw}$ and $\widehat{G_k}^{Rw}$ as the left and right hand observation sequences, respectively. Each of the observation subsequence $\widehat{G_k}^{Lw}$ and $\widehat{G_k}^{Rw}$ contains the same set of features used to calculate the spatiotemporal gesture similarity described in Section III-A; therefore, the kth gesture subsequence of the right hand for word w is defined as $\widehat{G_k}^{Rw} = \{\widehat{f}_{R1}^w, \ldots, \widehat{f}_{RT_{kw}}^w\}$, and T_{kw} is the length of the kth subsequence for word w.

A weighting of ω_{Lw} and ω_{Rw} is applied to the left and right hand HMMs, respectively, to account for the variations in information held in each of the hands for a particular sign. The weighting applied in the system is based on a variance measure of the observation sequences. The variances of the left and right hand observations are computed by calculating the variance of each observation dimension $\sigma_{Lw}^2[i]$ and $\sigma_{Rw}^2[i]$, where $0 \leq i \leq D$ and D are the dimension of the observation vectors. The left HMM weight ω_{Lw} and the right HMM weight ω_{Rw} are then calculated using

$$\omega_{Lw} = \sum_{i=0}^{D} \frac{\sigma_{Lw}^2[i]}{(\sigma_{Lw}^2[i] + \sigma_{Rw}^2[i]) \times D} \tag{26}$$

$$\omega_{Rw} = \sum_{i=0}^{D} \frac{\sigma_{Rw}^2[i]}{(\sigma_{Lw}^2[i] + \sigma_{Rw}^2[i]) \times D}. \tag{27}$$

A parallel HMM threshold model $\overline{\lambda^v} = \{\overline{\lambda_L}, \overline{\lambda_R}\}$ is then created using the network of trained parallel HMMs λ_w $(w \in W)$.

3) Parallel HMM Gesture Classification: To classify the parallel observations $\widehat{G} = \{\widehat{G}_L, \widehat{G}_R\}$, the Viterbi algorithm is run on each model given the unknown observation sequences $\widehat{G}_L$ and $\widehat{G}_R$, calculating the most likely state paths through each model w. The likelihoods of each state path, which we denote as $P(\widehat{G}_L|\lambda_L^w)$ and $P(\widehat{G}_R|\lambda_R^w)$, are also calculated. We calculate the overall likelihoods of a dedicated gesture and a movement epenthesis with the equations defined in

$$P(\widehat{G}|\lambda^w) = P\left(\widehat{G}_L|\lambda_L^w\right)\omega_{Lw} + \left(\widehat{G}_R|\lambda_R^w\right)\omega_{Rw} \tag{28}$$

$$P(\widehat{G}|\overline{\lambda}) = \frac{P\left(\widehat{G}_L|\overline{\lambda_L}\right)\Gamma_{Lw} + P\left(\widehat{G}_R|\overline{\lambda_R}\right)\Gamma_{Rw}}{2} \tag{29}$$

where Γ_{Lw} and Γ_{Rw} are constant scalar values used to tune the sensitivity of the system to the movement epenthesis. The sequence of observations can then be classified as w if $P(\widehat{G}|\lambda^w) \geq P(\widehat{G}|\overline{\lambda})$ evaluates to be true.

B. Hand Posture Training

While the spatiotemporal gesture spotting framework can be directly trained on data extracted from our automatic sign extraction framework described in Section IV and IV-B, the hand posture spotting system must apply further processing to the extracted hand posture subsequences $\widehat{H_i}$. This additional extraction process is required due to the variation in the possible hand postures which can occur within a particular sign sequence. For a particular sign, there are usually only a small number of frames in which the key hand postures are performed. The remaining hand postures performed in a sign are transitional postures which do not contribute to identifying the meaning of the sign.

1) Hand Posture Clustering: The goal of our hand posture clustering process is, given a set of hand posture subsequences $\widehat{H}$, to automatically extract the clusters that contain the hand postures which best represent the key postures of that sign. We now describe the key posture clustering algorithm in the following paragraph.

We define the hand posture density subsequence as $\widehat{\Psi}(H_i) = \{\Psi(H_i[t_i^s]), \ldots, \Psi(H_i[t_i^e])\}$ and calculate the mean μ and standard deviation σ over all density sequences $i \in \widehat{H^+}$. The probability $P(\widehat{\Psi}(H_i)[t])$ of frame t of the ith bag being a key hand posture is then calculated using the cumulative distribution function

$$P\left(\widehat{\Psi}(H_i)[t]\right) = \frac{1}{2}\left(1 + \operatorname{erf}\left(\frac{\widehat{\Psi}(H_i)[t] - \mu}{\sigma\sqrt{2}}\right)\right). \tag{30}$$

When training an automatic sign language spotting framework, the system will be trained on a set of target words w $(1 \leq w \leq W)$. For a target word w, we calculate the start and end frames of the spatiotemporal gesture and hand posture sequences $\widehat{G_i}$ and $\widehat{H_i}$ for all positive bags i. For each word w, we then extract the hand postures $H_i[t]$, where $P(\widehat{\Psi}(H_i)[t]) > 0.5$, and construct an initial hand posture cluster $\xi_w = \{H_i[1], \ldots, H_i[L(\xi_w)]\}$, where $L(\xi_w)$ corresponds to the total number of hand postures $H_i[t]$ with a corresponding probability $P(\widehat{\Psi}(H_i)[t]) > 0.5$. The initial clusters are constructed for all words $w \in W$, resulting in an initial set of hand

posture clusters $\Xi = \{\xi_1, \ldots, \xi_W\}$. Using this set of initial clusters, the aim is to extract the subsets of each cluster which contain only the hand postures that represent the key hand postures useful for discriminating between signs. In order to achieve this, we develop an iterative validation and cluster trimming algorithm to remove non-key hand postures from each cluster. We automatically interpret the set of clusters by analyzing the dissimilarity between each cluster pair using a cluster silhouette [28] representation. We define S_w in (31)–(36) as the measure of dissimilarity for cluster ξ_w, where w is the index of the word and $L(\xi_w)$ defines the number of hand postures in the posture cluster for word w

$$S_w = \frac{1}{L(\xi_w)} \sum_{t}^{L(\xi_w)} s_w[t] \tag{31}$$

$$s_w[t] = \frac{b_w[t] - a_w[t]}{\max\{b_w[t], a_w[t]\}} \tag{32}$$

$$b_w[t] = \chi_{\widehat{l}}^{w}[t] \tag{33}$$

$$\widehat{l} = \underset{l:(l \neq w)}{\arg\min}\, \chi_l^w[t] \tag{34}$$

$$a_w[t] = \chi_w^w[t] \tag{35}$$

$$\chi_l^w[t] = \frac{1}{L(\xi_l)} \sum_{j=0}^{L(\xi_l)} D^H\left(\xi_w[t], \xi_l[j]\right) \tag{36}$$

where $\chi_l^w[t]$ defines the average distance between the tth hand posture in the posture cluster of word w and all hand postures in the posture cluster of word l.

A value of S_w close to one means that the hand postures are appropriately clustered. If S_w is close to -1, then the hand postures are clustered poorly, therefore containing hand postures which occur in other clusters, and, thus, are not useful for discriminating between signs. For each iteration of our validation and cluster trimming algorithm, we calculate all dissimilarity measures S_w and s_w for each word w. We then remove the poorly clustered hand postures based on the dissimilarity measures. If $S_w < 0$, then we perform a cluster trimming procedure where we remove hand postures H, which have the lowest dissimilarity $s_w[t]$, from cluster ξ_w. Removing elements from cluster ξ_w in the current iteration will affect the dissimilarity values for all clusters in the next iteration; therefore, to avoid overfitting, we limit the number of hand postures which can be removed for each iteration to a fraction of the total number of hand postures in the cluster. For the experiments we conduct, we remove, at most, 10% of the hand postures at each iteration. This algorithm will repeat until one of two stopping criteria occurs for all hand posture clusters. The first stopping criteria for our algorithm specifies that no further postures be removed from the cluster if the dissimilarity measure $S_w > 0$. The second stopping criteria specifies that the number of postures in a cluster must not go below a predefined threshold. We set this predefined threshold to be a proportion of the total number of positive bags $N_{B_w^+}$ for each word w. We use the heuristic that a key hand posture occurs for at least 250 ms in each video; thus, each key posture will appear in at least six frames per video. We then set the minimum number of postures $T_{\min}$ per cluster to be $T_{\min} = N_{B_w^+} \times 6$. Algorithm 1 details our iterative validation and trimming procedure, and Fig. 9 shows a visualization of the algorithm being applied to a set of three initial clusters.

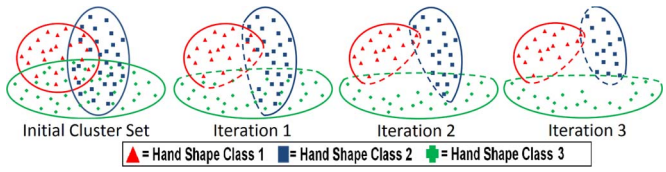


Fig. 9. Visualization of cluster validation and trimming algorithm.

Input: Set of Initial Posture Clusters Ξ
Output: Set of Key Posture Clusters Ξ'
Converged = False
while *!Converged* **do**
  Converged = True
  foreach $w \in W$ **do**
    Calculate Dissimilarity S_w and $s_w[t]$ for all t's
    if $S_w < 0 \cap L(\xi_w) > T_{\min}$ **then**
      MaxRemove = $L(\xi_i) \times 0.1$
      $Sort(s_w, \xi_w)$ in Increasing Order of s_w
      for $j < MaxRemove$ **do**
        if $s_w[j] < 0$ **then**
          Remove $\xi_w[j]$ from ξ_w
          Converged = False
        end
      end
    end
  end
end
Algorithm 1: Cluster Validation and Trimming Algorithm

C. Hand Posture SVM Framework

A set of SVMs [29] is trained on data, using the discussed Hu moments and eigenspace size functions, extracted from the automatically created hand posture cluster set Ξ. Given an unknown hand image, the relevant features are extracted, and the SVMs use the data to estimate the most probable hand posture classification. To classify a hand contour C representing an unknown hand posture, it must be assigned to one of the W possible words. The set of posture classes can be defined as $\alpha = \{\alpha_1, \ldots, \alpha_W\}$. The proposed recognition framework uses two distinct measurement vectors to represent a hand posture. For each posture class α_w, a set of two SVMs $\{SVM_w^{\text{sf}}, SVM_w^{\text{hu}}\}$ is used to calculate $P(\alpha_w | I(C), \zeta(C))$, the probability that posture C belongs to class α_w given the measurement vectors $I(C)$ and $\zeta(C)$, where $I(C)$ is the set of Hu moments and $\zeta(C)$ is the weighted eigenspace size function computed from the hand contour C.

1) SVMs: SVMs are a set of supervised learning methods used in classification and regression. A one-against-all SVM model is used in this paper, and the training of the SVM consists of providing the SVM with data for two classes. The data for each class consist of a set of n-dimensional vectors. The SVM will construct a hyperplane in the n-dimensional

space, attempting to maximize the margin between the two input classes.

The SVM type used in this paper is the classification SVM type 1 using a nonlinear classifier by means of the kernel trick as proposed by Aizerman *et al.* [30]. The kernel used is a radial basis function as defined by $k(x,x') = exp(-\gamma\|x-x'\|^2)$. The SVM is extended to obtain class probability estimates by computing the pairwise class probabilities $r_{ij} \approx p(y=i|y=i \text{ or } j, X)$ using Lin *et al.*'s [31] improved implementation of Platt's method [32] which defines $r_{ij} \approx 1/(1+e^{A\hat{f}+B})$, where A and B are estimated by minimizing the negative log-likelihood function using the known training data and their decision values $\hat{f}$. We then use the second approach proposed by Wu *et al.* [33] to compute p_i from all r_{ij}'s by solving the optimization problem $\min_p (1/2)\sum_{i=1}^{C}\sum_{j:j\neq i}(r_{ij}p_i - r_{ij}p_j)^2$ subject to $\sum_{i=1}^{C} p_i = 1, p_i \geq 0, \forall_i$.

2) SVM Training: The SVMs are trained on the data extracted from the automatically created hand posture cluster set Ξ. The weighted eigenspace size function data and Hu moment data are then extracted from the training clusters to create the sets $Hu_w = \{I_w[1], I_w[2], \ldots, I_w[L(\xi_w)]\}$ and $\zeta_w = \{\zeta_w[1], \zeta_w[2], \ldots, \zeta_w[L(\xi_w)]\}$, where Hu_w is the set of Hu moments and ζ_w is the set of weighted eigenspace size function [defined in (8)] for the hand posture cluster ξ_w.

To train each SVM_w^{sf}, the set ζ_w is used as the positive labeled training data, and $\overline{\zeta_w} := \{\zeta_k\}\forall_{k\neq w}$ is used as the negative labeled training data. Similarly, each SVM_w^{hu} is trained using Hu_w as the positive labeled data and $\overline{Hu_w} := \{Hu_k\}\forall_{k\neq w}$ as the negative labeled data. The SVMs SVM_w^{sf} and SVM_w^{hu} are then trained to maximize the hyperplane margin between their respective classes $(\zeta_w, \overline{\zeta_w})$ and $(Hu_w, \overline{Hu_w})$.

3) Posture Classification: To classify a hand contour C, each SVM_w^{sf} and SVM_w^{hu} will calculate $P(\alpha_w|\zeta(C))$ and $P(\alpha_w|I(C))$; the probabilities $\zeta(C)$ and $I(C)$ belong to class α_w, using the method outlined in Section V-C1, where $I(C)$ and $\zeta(C)$ are the Hu moments and weighted eigenspace size function extracted from the hand contour C, respectively. The classifier weights used to determine the overall probability are defined in (37), where cv_w^{sf} and cv_w^{hu} are the cross validation accuracies achieved during the training of each SVM_w^{sf} and SVM_w^{hu}, respectively

$$\mu_w^{\text{sf}} = \frac{cv_w^{\text{sf}}}{cv_w^{\text{sf}} + cv_w^{\text{hu}}} \quad \mu_w^{\text{hu}} = \frac{cv_w^{\text{hu}}}{cv_w^{\text{sf}} + cv_w^{\text{hu}}}. \tag{37}$$

A weighted combination of the probabilities is then calculated to generate the overall probability $P(\alpha_w|I(C),\zeta(C))$ according to

$$P(\alpha_w|I(C),\zeta(C)) = \left(P(\alpha_w|\zeta(C)) \times \mu_w^{\text{sf}}\right) + \left(P(\alpha_w|I(C)) \times \mu_w^{\text{hu}}\right). \tag{38}$$

VI. Continuous Recognition

Thus far, we have described our techniques developed to automatically train and classify spatiotemporal gestures and hand postures. We will now describe in the following paragraphs how we expand on our automatically trained spatiotemporal gesture and hand posture classifiers in order to spot and classify the signs within continuous sequences of natural sign language.

To perform the continuous gesture spotting, we utilize the automatically trained spatiotemporal gesture HMM framework, discussed in Section V-A, and the automatically trained hand posture SVM system, discussed in Section V-B and C.

The first step in our sign spotting algorithm is the spatiotemporal gesture end point detection using the automatically trained spatiotemporal gesture HMM framework. We detect the gesture end points from continuous streams of spatiotemporal gesture observations $\widehat{G} = \{\widehat{f}_1, \widehat{f}_2, \ldots, \widehat{f}_T\}$ and calculate the HMM model likelihoods of observation windows $\widehat{G}^* = \{\widehat{f}_{T-L}, \widehat{f}_{T-L-1}, \ldots, \widehat{f}_T\}$, where $\widehat{G}^*$ is a subset of $\widehat{G}$ and L defines the length of the observation subset used. In this paper, we set L as the average length of the automatically extracted subsequences used to train the system.

A candidate gesture κ with end point $\kappa_e = T$ is flagged when $\exists w : P(\widehat{G}^*|\lambda^w) \geq P(\widehat{G}^*|\overline{\lambda})$, where λ^w is the spatiotemporal gesture HMM model for word w and $\overline{\lambda}$ is the threshold HMM.

For each candidate end point, we calculate a corresponding spatiotemporal gesture start point κ_s. Different candidate start points are evaluated using the measurement shown in (39), where $\beta_w(\widehat{G})$ is a normalized metric (between zero and one) which measures the strength of gesture w relative to the movement epenthesis likelihood of the given observations $\widehat{G}$

$$\beta_w(\widehat{G}) = \frac{P(\widehat{G}|\lambda^w)}{P(\widehat{G}|\lambda^w)) + P(\widehat{G}|\overline{\lambda})}. \tag{39}$$

To find a candidate start point, the metric $\beta_w(\widehat{G}_{s\kappa_e})$ is calculated over different values of s, where $\widehat{G}_{s\kappa_e} = \{\widehat{f}_s, \widehat{f}_{s+1}, \ldots, \widehat{f}_{\kappa_e}\}$ and $(\kappa_e - L^2) \leq s < \kappa_e$. The candidate gesture start point κ_s is then found using

$$\kappa_s = \arg\max_s \quad \beta_w(\widehat{G}_{s\kappa_e}). \tag{40}$$

A. Candidate Selection

The start and end point detection algorithm may flag candidate gestures which overlap, and for this reason, we expand on our continuous sign spotting algorithm with a candidate selection algorithm. The purpose of the candidate selection algorithm is to remove the overlapping candidate gestures such that the single most likely gesture is the only remaining gesture for a particular time frame.

Each set of overlapping candidates represents a set of gestures with similar spatiotemporal properties. In order to discriminate between the gestures with similar spatiotemporal properties, we incorporate the hand shape measure. We consider all the gesture candidates which are flagged using our technique described earlier in Section VI. We calculate the overall probability of a particular candidate $P(\kappa|\widehat{G},\widehat{H})$ by combining the spatiotemporal and hand posture probabilities as described in

$$P(\kappa|\widehat{G},\widehat{H}) = \beta_w(\widehat{G}_{\kappa_s\kappa_e}) \times P(\alpha_w|\kappa,\widehat{H}) \tag{41}$$

$$P(\alpha_w|\kappa,\widehat{H}) = \max_{\kappa_s<i<\kappa_e} P\left(\alpha_w|I\left(\widehat{H}[i]\right), \zeta\left(\widehat{H}[i]\right)\right) \tag{42}$$

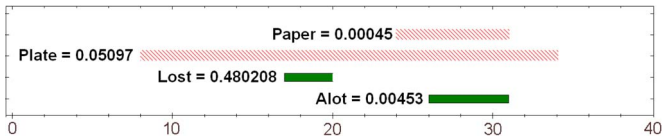


Fig. 10. Candidate gestures Υ. Candidates marked in (dashed) red denote gestures which are removed by the second candidate selection step. Candidates in (solid) green denote the final spotted gestures.

where α_w is the hand posture SVM model for word w. The hand posture probability of candidate $P(\alpha_w|\kappa, \widehat{H})$ is defined as the probability of the hand shape which best fits the gesture w. $I(\widehat{H}[i])$ is the set of Hu moments, and $\zeta(\widehat{H}[i])$ is the weighted eigenspace size function computed from the hand contour $\widehat{H}[i]$ in frame i.

The first step in the candidate selection algorithm is to cluster the overlapping gestures which have the same gesture classification. We remove all but one candidate gesture from this cluster, leaving only the candidate gesture κ^B with the highest $P(\kappa|\widehat{G}, \widehat{H})$ value. We repeat this step for each cluster to produce a set of candidate gestures $\Upsilon = \{\kappa^{B1}, \kappa^{B2}, \ldots, \kappa^{BK}\}$, where K is the total number of clusters created from clustering the overlapping gestures which have the same gesture classification.

The second candidate selection step finds the sets of overlapping candidates and removes the least probable candidates such that a maximum of only one candidate is detected for any given time frame. Fig. 10 shows the time segments and gesture probabilities of the recognized gestures after the first and second candidate selection steps where the signs "Lost" and Alot are correctly spotted from a sample sign language sentence "I Lost Alot of Books."

VII. Experiments

A. Classifier Experiments

Before carrying out a full evaluation of our automatic training and sign language spotting framework, we first perform preliminary experiments to evaluate the performance of the gesture classification components. Kelly *et al.* [21] perform an evaluation of size functions as a method for hand posture classification in sign language and show that the eigenspace size function performs well at discriminating between different hand postures. We now carry out an evaluation of the spatiotemporal gesture classifier described in Section V-A in the following paragraphs.

1) Isolated Spatiotemporal Gesture Classification: The goal of the spatiotemporal gesture experiments was to evaluate the performance of our HMM threshold framework model when recognizing spatiotemporal gestures and identifying epentheses which occur in sign language. We first evaluate our framework on a data set which consists of eight different isolated signs which were manually extracted from videos of a fluent signer performing natural sign language sentences. To evaluate the spatiotemporal gesture framework, the eight signs were chosen such that each sign could be distinguished by position and movement alone and did not require hand posture information. A set of ten training signs and a set of ten test signs were recorded for each sign in the vocabulary (a total of 160 gesture samples). An additional set of gestures, which represents a collection of the movement epenthesis, was also extracted from the video sequences to test the performance of the threshold model. For each sign, ten movement epentheses that occurred before and after the valid sign in different sign language sentences were recorded. An additional set of 20 random movement epentheses was also recorded, resulting in a test set of 100 epenthesis samples to evaluate the models on.

TABLE I
Continuous Spatiotemporal Gesture Recognition: AUC Measurements for Different Models

Gesture	#Correct	$\#Del^{\dagger}$	$\#Ins^{\ddagger}$	$\#Sub^{\dagger\dagger}$	Detection	Reliability
Total HMM	**153**	**0**	**3**	**4**	**0.956**	**0.938**
Total LDCRF	**133**	**2**	**6**	**19**	**0.831**	**0.801**

† *Number of Deletion Errors*, ‡*Number of Insertion Errors*, ††*Number of Substitution Errors*

Morency *et al.* [34] propose a latent-dynamic conditional random field (LDCRF) [34] to combine the strengths of the conditional random fields (CRF) [35] and the hidden conditional random fields (HCRF) [36] for the recognition of human gestures by capturing both the extrinsic dynamics and intrinsic substructure of human gestures. Kelly *et al.* [37] perform a full evaluation of a threshold HMM and different CRF models when recognizing the gestures in sign language. The results showed that the HMM threshold model and the LDCRF model performed best when compared to traditional HMMs, CRFs, and HCRFs.We implemented the LDCRF model [34] and our HMM threshold model and tested and trained both models on the same training and test set. To evaluate the performance of the models, we perform a receiver-operating-characteristic analysis on the different models and calculate the area under the curve (AUC) for each model. The results of the experiment showed that the standard HMM achieved an AUC of 0.902, the LDCRF achieved an AUC of 0.942, and the HMM threshold model achieved an AUC of 0.976. While we evaluated the LDCRF model on different numbers of states, we report only the best performing LDCRF which was an LDCRF model with eight states per sign.

2) Continuous Spatiotemporal Gesture Classification: We perform a second experiment to evaluate the performance of the models when recognizing continuous unconstrained sign language sentences. The models were trained on the same set of eight hand gestures used in the isolated experiments in Section VII-A1. We use the best performing models trained during the isolated recognition experiments, discussed in Section VII-A1, to evaluate the continuous sign language spotting performance of the HMM threshold framework and LDCRF model. Thus, we use an LDCRF model with eight hidden states per label for the spatiotemporal gesture spotting.

Table I shows the performance of our HMM framework and the LDCRF model when spotting and classifying gestures within 160 continuous sign language sentences. The 160 test sentences contain a large amount of signs outside of the eight chosen signs as well as the epenthesis which occurs between the signs. The experiment shows an overall detection rate of 95.6% and an overall reliability of 93.8% for our HMM framework and an overall detection rate of 83.1% and an overall reliability of 80.1% for the LDCRF model when spotting and classifying gestures in continuous sign language sentences.

B. Automatic Training Experiments

A description of the experiments conducted to evaluate the overall performance of our automatic training and sign language spotting system is presented in this section.

1) Data Collection: Two fluent Irish sign language (ISL) signers were recorded while they performed a total of 844 natural sign language sentences. The signers were given no instruction other than to sign to the camera and to sign sentences while trying to incorporate certain key words into the sentences. While recording the sentences, a certified ISL interpreter translated each sentence through a microphone connected to the video camera. The videos were captured at 25 fps with a frame size of 640 $\times$ 480.

From the set of 844 sentences, a lexicon of 30 signs was decided on based on the signs which occurred frequently within the 844 sentences. The 844 sentences contain a large amount of signs outside of the 30 chosen signs. The 30 key signs were then labeled within the 844 sentences. Each of the 30 signs occurred, on average, 44 times within the 844 sentences with a total of 1344 key signs occurring in the data set. The labeling process involved marking the start and end points of each sign within each video. It is important to note that this labeling process is carried out for ground truth data only and none of these labels are used in the training of the system.

2) Sign Labeling: The goal of the automatic sign extraction framework is to accurately detect target signs within unsegmented sentences and label them at a frame level. Due to the ambiguities in the sign translation, the system must also be able to automatically identify and discard noneligible sentences by detecting whether a target sign was actually performed in a given video even though the translation information specifies that it does occur.

When performing the automatic sign extraction, we construct a set of bags B^+ which contains video sequences where the target sign is said to occur based on the translation data. Using our MIL density matrix algorithm, the set of bags is then used to find the similarities in the video sequences in order to label the target sign. The automatic sign labeling algorithm is then used to classify the positive sentences (sentences in which the target word occurs) and noneligible sentences (sentences in which the target word does not occur in the video but does occur in the text translation). For positive signs, the labeling algorithm then detects the start and end points of the target sign while noneligible sentences will be automatically discarded.

Since our automatic sign extraction technique is based on a comparison of the other sentences in the set of bags, the number of sentences and the number of possible noneligible sentences in the set of bags can affect the labeling of all the sentences in the set of bags. We first investigate the effect that the noneligible sentences have on the automatic sign extraction framework by varying the number of videos in a bag and by also varying the percentage of the bag which is made up of noneligible sentences. We vary the number of elements in a positive bag from 5–20 videos and vary the percentage of noneligible sentences, for the current target word, from 0%–50%. To evaluate our automatic sign extraction framework, we compare the results of the automatic extraction to that of the ground truth data. In this experiment, we use true positives (TPs), false negatives (FNs), true negatives (TNs), and false positives (FPs) to quantify the performance of the system. A TP refers to when the classifier correctly classifies a positive sign as one that occurred in the sentence and also correctly labels the start and end points such that the classified sign overlaps with the ground truth sign by at least 50%. A FP refers to a sentence which is incorrectly classified as a positive sentence or a sign in which the start and end points are flagged such that it does not overlap with the ground truth sign. A TN refers to when a sentence is correctly labeled as a noneligible sentence, and a FN refers to a sentence which is incorrectly labeled as a noneligible sentence. For the different percentages of noneligible sentences (0%–50%), we calculate the total precision, recall, and F-measure for all 30 signs when labeled from the bags which contained 5, 10, 15, and 20 sentences.

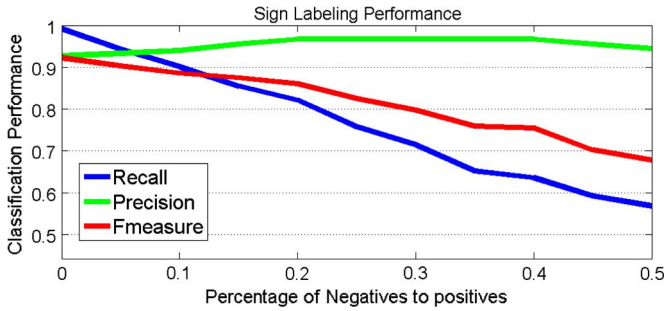


Fig. 11. Performance of automatic sign labeling and effect of translation ambiguities.

Fig. 11 shows the precision, recall, and F-measure for different percentages of noneligible sentences present in the set of bags B^+. The results show an F-measure of 0.92 when there exist no noneligible sentences and an F-measure of 0.67 when 50% of the bag is made up of noneligible sentences. In our data set of 844 sentences, 12.2% of the sentences contained noneligible sentences. Thus, a good indication of how our sign labeling system would perform in reality is to evaluate the system on the bags which approximately contain 12.2% noneligible sentences. In our experiment, the results show an F-measure of 0.874 when labeling signs in a bag made up of 15% noneligible sentences. This can be interpreted as a promising result. By looking at the corresponding precision and recall values for the bag made up of 15% noneligible sentences, we can see that our technique achieves a precision of 0.942. This means that 94.2% of the data that will be used to train the spatiotemporal gesture and hand posture models is correct. The recall rate achieved was 0.833, meaning that only 16.7% of the valid training data were incorrectly discarded. An important observation to make from the results of this experiment is that, as the percentage of noneligible sentences increases, the precision rate does not drop. A vital part of the classification of the sentences is to reduce the number of FPs since any FPs will then be used in the training of the spatiotemporal and hand posture classifiers. The consistent precision rate achieved during this experiment demonstrates that our system performs well at reducing the number of FPs.

3) Start and End Point Detection: The second experiment we conduct on our automatic sign extraction framework is an

evaluation of the performance of the system when detecting the start and end points of positively labeled sequences. In all experiments, from now on, we use a set of bags which contains 15 video sequences of which an average of 12.2% are noneligible sentences which contain translation ambiguities of the target word. It is important to note that, although we manually control the number of noneligible sentences in a bag during the previous experiment, in this experiment and all experiments which follow, the percentage of noneligible sentences are controlled only by the content of our data set and not by the supervised labeling of the positive and noneligible sentences.

The sign labeling algorithm computes the density vectors for each video and automatically identifies and discards any noneligible sentences in the set of bags. An overall F-measure of 0.88 was achieved from the automatic classification of positive and noneligible sentences in each of the 30 bags for each sign. For the positive videos, the start and end points are then detected. To evaluate the performance of the start and end point detection, we compare the automatically detected start and end points with the ground truth start and end points and compute the mean error for each sign. The results of the start and end point detection experiment show that our system detects the occurrence of a target word within an average of 9.4 and 8.6 frames of the ground truth data. This can be interpreted as a promising result as this result means that our technique is able to detect target sign start and end points within 0.376 and 0.344 ms when compared to a human interpreter.

4) Continuous Recognition: The overall goal of this paper is to automatically train models to recognize natural sign language from unconstrained sign language sentences. We now describe the experiments which were carried out to evaluate the performance of the overall sign language spotting system in the following sentences. For each of the 30 target words, a set of target word subsequences was automatically extracted using the techniques we describe in Section IV-B. For the experiments we describe, the subsequences were calculated from the same set of bags used to evaluate the start and end point labeling in Section VII-B3. Each bag contained 15 video sequences of which an average of 12.2% are noneligible sentences which contained the translation ambiguities of the target word. The noneligible sentences were automatically detected and discarded by our system with a precision of 0.931 and a recall rate of 0.856. All the bags contained video sequences from only one of the two signers. A set of parallel HMMs and a parallel HMM threshold model were then automatically trained on the subsequences using the techniques we discuss in Section V-A2. The key hand postures for each target word were automatically extracted, and a set of SVMs was then trained to recognize the key hand postures using the hand posture framework we describe in Section V-B and C. The samples of the automatically extracted spatiotemporal subsequences and key hand postures, used in the actual training, are made available on a video as a supplement to this paper.

Given an unknown sign language sentence, we then apply our sign spotting framework described in Section VI to spot and classify the signs in each sentence. To evaluate the performance of our sign language spotting framework, we test the system on the remaining set of sentences, as performed by both signers, which were not used in the training process.

TABLE II
CONTINUOUS SPOTTER AND CLASSIFIER PERFORMANCE

Sign	#Correct	$\#D^{\dagger}$	$\#S^{\ddagger}$	$\#I^{\dagger\dagger}$	Del^{*}	Rel'	End Error	Start Error
rain	16	0	0	0	1	1	3.8	10.6
book	32	0	11	4	0.68	0.627	7.4	5.1
teacher	28	0	5	1	0.823	0.8	6	2.21
ball	22	0	5	0	0.814	0.814	3.95	2.9
shop	27	0	13	1	0.65	0.64	4.6	3.1
sun	28	0	1	0	0.965	0.965	5.3	2.71
eat	24	0	6	0	0.8	0.8	2.4	6.6
backpack	20	0	3	0	0.86	0.86	3.65	6.8
school	38	0	13	1	0.73	0.71	4.9	3.3
pay	21	0	1	0	0.954	0.954	6	2.8
read	38	0	0	0	1	1	3.6	17
newspaper	19	0	2	0	0.9	0.9	4	5.3
hotel	22	0	1	0	0.956	0.956	3.7	4.6
cruise	16	0	4	0	0.8	0.8	3.8	7.4
airport	26	0	2	0	0.928	0.928	2.3	8.3
car	25	0	1	0	0.96	0.96	4.8	7.9
bus	15	0	6	0	0.71	0.71	3	3.3
country	17	0	5	1	0.77	0.73	7.17	2.17
language	15	0	4	0	0.78	0.78	2.7	4.2
like	81	0	31	2	0.71	0.69	4.8	3.5
play	30	0	7	0	0.81	0.81	4.5	5
friend	35	0	10	0	0.77	0.77	2.6	4.5
brother	30	0	1	0	0.96	0.96	4.7	3.1
sister	21	0	11	0	0.65	0.65	2.9	4.9
bike	31	0	0	0	1	1	3	7.7
alot	23	0	3	0	0.88	0.88	2.7	1.9
allgone	16	0	3	2	0.76	0.69	2.68	3.75
understand	38	0	2	0	0.95	0.95	3.89	3.67
cold	21	0	0	0	1	1	4.7	4.8
warm	17	0	8	0	0.68	0.68	1.23	9.4
Total	**792**	**0**	**159**	**12**	**0.823**	**0.812**	**4.17**	**5.07**
Total Signer 1	**581**	**0**	**94**	**7**	**0.86**	**0.851**	**4.05**	**4.98**
Total Signer 2	**211**	**0**	**65**	**5**	**0.764**	**0.750**	**4.31**	**5.45**
Total No HandShape	**689**	**0**	**267**	**7**	**0.715**	**0.711**	**4.02**	**5.28**

† *Number of Deletion Errors*, ‡*Number of Insertion Errors*
††*Number of Substitution Errors*, ∗*Detection Ratio*, *Reliability*

Table II details the performance of the sign spotting system when tested on the remaining sentences. The results show that the system performs well with an overall detection ratio of 0.823 and an overall reliability of 0.812. The user-dependent results, for Signer 1, show that the system performs with a detection ratio of 0.86 while the user-independent results, for Signer 2, show a detection ratio of 0.764. In the previous works which have used a small number of signers in the training set, the results of the user-independent recognition evaluations have seen large decreases when compared to the user-dependent recognition results [2]. For example, in [38], the accuracy for training on one signer and testing on another was 51.9% compared to 92% when the same signer supplied both the training and test data. Other works [2] have shown that an increased number of signers in the training set can greatly improve the overall performance of a user-independent sign recognition system. Therefore, while the results achieved in this experiment show a decrease in the signer-independent detection rate, the fact that only one signer was represented in the training set means that a decrease of only 9.6% can be interpreted as a promising result.

We conduct a second experiment in which we evaluate the impact that our automatic key hand posture extraction and hand posture recognition components have on the overall detection of the signs. We carry out the same sign spotting experiment as before, but this time, no automatic key hand posture extraction was carried out, and no hand shape probability was included in the sign spotting framework. In the absence of our hand shape

techniques, the results show that the overall detection rate drops by 10.8%. This shows that our automatic hand posture clustering method is an important step in developing a full weakly supervised sign training framework. It also demonstrates the importance of our hand shape classification for the verification of detected spatiotemporal signs. A video showing the software implementation of our system performing sign spotting on a number of sample sentences is made available as a supplement to this paper.

The main disadvantage of our proposed framework is the computational complexity of the continuous sign language spotting framework. During continuous experiments on the vocabulary of 30 signs, the continuous sign spotting system took, on average, two times the length of the video to carry out the gesture spotting algorithm. Performing the Viterbi algorithm on the HMM threshold models is the main cause of the high computation time. Reducing the number of states in the threshold model would decrease the overall computational complexity, and Lee and Kim [25] have proposed methods to reduce the number of threshold model states by half using relative entropy.

VIII. Conclusion

Previous research in sign language recognition has typically required the manual labeling of sign language videos in order to extract isolated examples of particular signs to train recognition systems. In order to advance the research of sign language to the same level as speech recognition, sign vocabularies must be expanded to the same size as the speech recognition vocabularies. Expanding these manually generated sign vocabularies is a very difficult time-consuming expensive procedure [12]. Therefore, advancing the sign language recognition research requires robust automatic training algorithms. In this paper, we present a novel system of automatically training models for the recognition of natural sign language. The proposed system is capable of learning sign language from unsegmented sign language videos using the weak and noisy supervision of text translations. Full sign language sentences are automatically segmented, and the isolated samples of the target words are extracted from the videos. An HMM-based spatiotemporal gesture spotting system is trained to recognize the signs in the vocabulary and to detect the movement epenthesis. Moreover, an SVM-based hand posture recognition system is trained on automatically detected key hand postures using our novel eigenspace size function along with the Hu moments. The spatiotemporal and hand posture recognition systems are then combined in a continuous sign spotting framework to detect the signs from continuous sentences. The experiments demonstrate that our automatic sign labeling algorithm performed well when classifying positive signs and noneligible sentences, with an F-measure of 0.874 when labeling the signs in a bag made up of 15% noneligible sentences. The results also showed the sign labeling algorithm detected start and end points of target words within an average 9.4 and 8.6 frames, respectively. An evaluation of the performance of our the sign spotting system, which was automatically trained on a vocabulary of 30 signs, was carried out. The system was tested on 962 signs which occurred within continuous sentences. The results indicate that our system can detect signs from continuous sentences with a detection rate of 82.3%. The results of the experiments are very promising. The results of our system are comparable with the results achieved from the state-of-the-art recognition systems trained on manual data, such as the work of Yang *et al.* [10]. By way of comparison, their system was manually trained on a vocabulary of 48 signs and could detect signs with an 87% detection rate when tested on 480 signs. The contributions of this paper are the following: 1) We propose a hand posture classification model which robustly recognizes the hand postures independent of the person performing the gesture; 2) we propose a spatiotemporal gesture spotting model which classifies spatiotemporal gestures and detects the movement epenthesis without being explicitly trained on the movement epenthesis; 3) we have combined these posture and spatiotemporal models into our framework which can automatically learn natural signs from the weak and noisy supervision of text translations using our MIL density matrix algorithm. These are important contributions to the area of natural sign language recognition as they introduce a robust framework for training a recognition system without the need for manual labeling. While we evaluate our system on a vocabulary of 30 signs, extending this vocabulary requires only that a larger video data set, with associated text translations, be automatically processed by our automatic training system. Freely available sign language videos and corresponding translations from television broadcasts could be utilized as an expansive training set in which our system could be trained on.

References

[1] S. K. Liddell and R. E. Johnson, "American sign language: The phonological base," *Sign Lang. Stud.*, vol. 64, no. 6, pp. 195–278, 1989.

[2] S. C. W. Ong and S. Ranganath, "Automatic sign language analysis: A survey and the future beyond lexical meaning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 6, pp. 873–891, Jun. 2005.

[3] T. Starner, A. Pentland, and J. Weaver, "Real-time American sign language recognition using desk and wearable computer based video," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 12, pp. 1371–1375, Dec. 1998.

[4] B. Bauer and K.-F. Kraiss, "Towards an automatic sign language recognition system using subunits," in *Proc. Revised Papers Int. Gesture Workshop Gesture Sign Lang. Human-Comput. Interaction GW*, London, U.K., 2002, pp. 64–75.

[5] C. Wang, S. Shan, and W. Gao, "An approach based on phonemes to large vocabulary Chinese sign language recognition," in *Proc. IEEE FG*, Washington, DC, 2002, p. 411.

[6] M. Assan and K. Grobel, "Video-based sign language recognition using hidden Markov models," in *Proc. Int. Gesture Workshop Gesture Sign Lang. Human-Comput. Interaction*, London, U.K., 1998, pp. 97–109.

[7] W. Gao, G. Fang, D. Zhao, and Y. Chen, "Transition movement models for large vocabulary continuous sign language recognition," in *Proc. IEEE FG*, May 2004, pp. 553–558.

[8] C. Vogler and D. Metaxas, "A framework for recognizing the simultaneous aspects of American sign language," *Comput. Vis. Image Underst.*, vol. 81, no. 3, pp. 358–384, Mar. 2001.

[9] R. Yang, S. Sarkar, and B. Loeding, "Enhanced level building algorithm for the movement epenthesis problem in sign language recognition," in *Proc. CVPR*, 2007, pp. 1–8.

[10] H. D. Yang, S. Sclaroff, and S. W. Lee, "Sign language spotting with a threshold model based on conditional random fields," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 7, pp. 1264–1277, Jul. 2009.

[11] J. Triesch and C. von der Malsburg, "Classification of hand postures against complex backgrounds using elastic graph matching," *Image Vis. Comput.*, vol. 20, no. 13/14, pp. 937–943, Dec. 2002.

[12] P. Buehler, A. Zisserman, and M. Everingham, "Learning sign language by watching TV (using weakly aligned subtitles)," in *Proc. IEEE Comput. Soc. Conf. CVPR Workshops*, Jun. 2009, pp. 2961–2968.
[13] H. Cooper and R. Bowden, "Learning signs from subtitles: A weakly supervised approach to sign language recognition," in *Proc. IEEE Conf. CVPR*, 2009, pp. 2568–2574.
[14] S. Nayak, S. Sarkar, and B. Loeding, "Automated extraction of signs from continuous sign language sentences using iterated conditional modes," in *Proc. CVPR*, 2009, pp. 2583–2590.
[15] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2000, vol. 2, pp. 142–149.
[16] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. IEEE CVPR*, 2001, vol. 1, pp. 511–518.
[17] L. A.-C. M. Castrillon-Santana, O. Deniz-Suarez, and J. Lorenzo-Navarro, "Performance evaluation of public domain HAAR detectors for face and facial feature detection," in *Proc. VISAPP*, 2008, pp. 179–187.
[18] M.-K. Hu, "Visual pattern recognition by moment invariants," *IEEE Trans. Inf. Theory*, vol. IT-8, no. 2, pp. 179–187, Feb. 1962.
[19] Intel-Corporation, Open Source Computer Vision Library: Reference Manual, pp. 975–10052000.
[20] C. Uras and A. Verri, "Sign language recognition: An application of the theory of size functions," in *Proc. 6th Brit. Mach. Vis. Conf.*, 1995, pp. 711–720.
[21] D. Kelly, J. McDonald, and C. Markham (2010, Aug.). A person independent system for recognition of hand postures used in sign language. *Pattern Recognit. Lett.* [Online]. *31(11)*, pp. 1359–1368. Available: http://www.sciencedirect.com/science/article/B6V15-4YDC3G4-3/2/7f7269fd 77821c3d2b1d92c52f13910a
[22] J. Stokoe and C. William, "Sign language structure: An outline of the visual communication systems of the American deaf," *J. Deaf Studies Deaf Educ.*, vol. 10, no. 1, pp. 3–37, Winter 2005.
[23] M. Handouyahia, D. Ziou, and S. Wang, "Sign language recognition using moment-based size functions," in *Proc. Int. Conf. Vis. Interface*, 1999, pp. 210–216.
[24] O. Maron, T. Lozano-Pérez, and T. A. L. p Erez, "A framework for multiple-instance learning," in *Advances in Neural Information Processing Systems*. Cambridge, MA: MIT Press, 1998, pp. 570–576.
[25] H. K. Lee and J. H. Kim (1999, Oct.). An HMM-based threshold model approach for gesture recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* [Online]. *21(10)*, pp. 961–973. Available: http://dx.doi.org/10.1109/34.799904
[26] D. Kelly, J. M. Donald, and C. Markham, "Continuous recognition of motion based gestures in sign language," in *Proc. IEEE Int. Workshop Tracking Humans Eval. Motion Image Sequences ICCV*, 2009, pp. 1073–1080.
[27] C. Vogler and D. Metaxas, "Parallel hidden Markov models for American sign language recognition," in *Proc. ICCV*, 1999, pp. 116–122.
[28] P. Rousseeuw (1987, Nov.). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.* [Online]. *20(1)*, pp. 53–65. Available: http://dx.doi.org/10.1016/0377-0427(87)90125-7
[29] C.-C. Chang and C.-J. Lin, LIBSVM: A library for support vector machines, 2001. [Online]. Available: http://www.csie.ntu.edu.tw/cjlin/libsvm
[30] A. Aizerman, E. M. Braverman, and L. I. Rozoner, "Theoretical foundations of the potential function method in pattern recognition learning," *Autom. Remote Control*, vol. 25, pp. 821–837, 1964.
[31] H.-T. Lin, C.-J. Lin, and R. C. Weng, "A note on Platt's probabilistic outputs for support vector machines," *Mach. Learn.*, vol. 68, no. 3, pp. 267–276, Oct. 2007.
[32] J. C. Platt and J. C. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," in *Advances in Large Margin Classifiers*. Cambridge, MA: MIT Press, 1999.
[33] T. fan Wu, C. jen Lin, and R. C. Weng, "Probability estimates for multiclass classification by pairwise coupling," *J. Mach. Learn. Res.*, vol. 5, pp. 975–1005, 2004.
[34] L.-P. Morency, A. Quattoni, and T. Darrell, "Latent-dynamic discriminative models for continuous gesture recognition," in *Proc. IEEE Conf. CVPR*, Jun. 2007, pp. 1–8.
[35] A. Quattoni, M. Collins, and T. Darrell, "Conditional random fields for object recognition," in *Advances in Neural Information Processing Systems*, L. K. Saul, Y. Weiss, and L. Bottou, Eds. Cambridge, MA: MIT Press, 2005, pp. 1097–1104.
[36] S. B. Wang, A. Quattoni, L.-P. Morency, D. Demirdjian, and T. Darrell, "Hidden conditional random fields for gesture recognition," in *Proc. IEEE Comput. Soc. Conf. CVPR*, 2006, vol. 2, pp. 1521–1527.
[37] D. Kelly, J. M. Donald, and C. Markham, "Evaluation of threshold model HMMS and conditional random fields for recognition of spatiotemporal gestures in sign language," in *Proc. IEEE Int. Workshop Mach. Learn. Vision-Based Motion Anal. ICCV*, 2009, pp. 490–497.
[38] M. Assan and K. Grobel, "Video-based sign language recognition using hidden Markov models," in *Proc. Int. Gesture Workshop Gesture Sign Language Human-Comput. Interaction*, London, U.K., 1998, pp. 97–109.

Daniel Kelly received the B.Sc. degree in computer science and software engineering from the National University of Ireland (NUI) Maynooth, Maynooth, Ireland, in 2006, and the Ph.D. degree, with a thesis on the topic of machine learning methods for automated methods of sign language analysis, from the Department of Computer Science, NUI Maynooth, in 2010.

He is currently working as a research fellow in the Clanty center for sensor web technologies in University College Dublin. His research interests include gesture recognition, human motion recognition, computer vision, and machine learning.

John Mc Donald (M'97) received the B.Sc. degree (double honors) in computer science and mathematical physics from the National University of Ireland (NUI) Maynooth, Maynooth, Ireland, in 1996.

Since 1997, he has been with the Department of Computer Science, NUI Maynooth, where he has been a Full-Time Lecturer since 2001. He was a Visiting Researcher at the Department of Electrical and Computer Engineering, University of Connecticut, Storrs, in 2002, and at the National Center for Geocomputation, NUI Maynooth, in 2009. In 2010, he was a Visiting Scientist at the Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge. His research interests include computer vision and pattern recognition, visual simultaneous localization and mapping, place recognition, human face and gesture analysis, and digital holography.

He was the Chair of the International Machine Vision and Image Processing Conference 2007.

Charles Markham received the degree in applied physics and the Ph.D. degree, with a thesis on element-specific imaging in computerized tomography, from Dublin City University (formerly, the National Institute for Higher Education, Dublin), Dublin, Ireland, in 1988 and 1993, respectively.

He is currently a Lecturer with the Computer Science Department, National University of Ireland (NUI) Maynooth, Maynooth, Ireland. In 1998, he was with Dublin City University, Dublin, Ireland, where he specialized in medical imaging in the area of element specific imaging using computed tomography. His research interests include optical brain–computer interfacing, gesture interfaces, and mobile computer vision.