Sign Align – A System to Track and Translate Sign Language

Daniel Nichol

Department of Computer Science

University of Oxford

A thesis submitted for the degree of

Master of Mathematics and Computer Science

I would like to dedicate this thesis to my loving parents ...

Acknowledgements

And I would like to acknowledge Harish Bhanderi for providing this LaTeXclass.

Abstract

This is where you write your abstract ...

Contents

List of Figures

Chapter 1

Introduction

British Sign Language is the preferred first language of over 125,000 in the United Kingdom, in addition to an estimated 20,000 children and thousands of hearing friends, relatives and interpreters. There are many more people worldwide communicating in an estimated 200 different signed languages, with a huge variation in grammar and pronounciation. As such, a system which can recognise signed languages and convert them into text in real time would be a valuable tool for many people.

The aim of this project is to implement a system to recognise gestures of a signed language from a continuous data stream provided from the Microsoft Kinect camera. The problem of continuous gesture recognition has been studied for over 20 years and has seen solutions which often involve specific gloves or expensive hardware and which have been particularly sensitive to lighting conditions and camera placement. Using the Kinect sensor, which is available to purchase off the shelf and is designed to be used without gloves and in a range of conditions, we aim to build a robust sign recognition system which does not suffer from these limitations.

1.1 Sign Language

Signed languages natural languages which have evolved independently of the spoken languages of the areas in which they are used and often independently of one

another. For example, British Sign Language (BSL) is not simply oral English transcribed but rather a distinct language with its own sentence structure which differs significantly from English. Further despite the regions sharing a common language American Sign Language (ASL) is a seperate language from BSL.

Despite the large variation between regional sign languages the means of communication remain the same across most signed languages. The main component of the sign is the movement of the body and each sign is determined by the movement and location of the hands and arms as well as the hand shape and palm direction. In addition to the manual component of the signs the signer will often use posture, facial expressions, mouth shape or eye movements to convey meaning.

The non-manual components of sign language will be ignored for the purposes of this project. It should be noted that eye tracking and facial expression are essentially independent problems from that of gesture recognition and hence if solutions to the facial expression problem exist they may be combined with the work of this project to produce a more substantial sign recognition system.

We can further break the manual part of a sign in to two components. The first is the movement of the hand, arms and body over time and the second is the orientation of the hand throughout this movement. This is a sensible distinction to make as it helps reduce the number of distinct signs to differentiate between. As the same hand shape might be used with two different arm movements to produce different words, we can reduce the number of distinct gestures we wish to detect by detecting hand movement gestures and hand shapes and then combining the two to identify a sign.

In the next section we will see there is a limitation in our choice of hardware that makes hand shape detection a problem which is outside the scope of this project. However the above observation shows that the work in this project to detect arm and body gestures can be combined with future work to produce a system which will detect a signed language. Furthermore, a major task of this project is to implement a library of Hidden Markov Model sequence classifiers to detect gestures and this library can be used to detect hand shape gestures just as we use it here to detect body gestures.

1.2 Hardware

The Kinect is a motion sensing camera designed by Microsoft and released in November 2010. The Kinect combines an RGB camera and an infrared depth sensor to detect objects in 3D space. The raw images of these sensors are combined using proprietary software to detect a human body as a skeleton, in particular the Kinect is capable of detecting the positions of the hands, elbows, shoulders and head as a point in 3D space.

Microsoft released the Kinect software development kit (SDK) for public use on January 16, 2011. This SDK allows developers build applications which interact with the proprietary skeleton tracking software using C#, C++ or Visual Basic. In this project we will use this SDK to build the gesture recognition framework.

The Kinect camera was originially designed with the ability to detect hand and finger positions. In fact the original patent claims the device would be able to detect American Sign Language LATTA et al. (2010). However this functionality was removed from the original release of the Kinect sensor. The SDK was updated to recognise open and closed hands on March 18th, 2013 Microsoft (2013) and it is believed that the next iteration of the Kinect hardware will be able to fully detect hand gestures.

Third party hand gesture recognition frameworks do exist, however the most successful of these do not make use of the Kinect SDK and so do not make use of its features at all I. Oikonomidis (2013). We aim to build a framework that will be easily extended when the capabilities of the Kinect are improved and for this reason we choose to only use the tools provided within the Kinect SDK.

1.3 Outline

Chapter 2

Background and Model Selection

2.1 Hidden Markov Models

A hidden Markov Model (HMM) is (following the definitions in Rabiner (1989)) a doubly stochastic process which consists of an underlying discrete Markov chain with state set $S = \{S_1, \ldots, S_n\}$ and stochastic matrix $A = [a_{i,j}]_{N \times N}$ where

$$a_{i,j} = \mathbb{P}(q_{t+1} = S_i | q_t = S_j) \text{ for each } 1 \leq i, j \leq N$$

which is hidden from an observer in the respect that one cannot directly observe the current state of the Markov chain. Instead we have a collection of M observable symbols, say $V = \{v_1, \ldots, v_M\}$, which may be observed with probability dependent on the state underlying Markov chain.

We encode these so-called *emission probabilities* in a matrix $B = [b_j(k)]_{N \times M}$ where

$$b_j(k) = \mathbb{P}[v_k \text{ occurs at time } t | q_t = S_j]$$

Now if we take an initial state distribution $\boldsymbol{\pi} = [\pi_1, \ldots, \pi_N]$ for our Markov chain with

$$\pi_i = \mathbb{P}[q_1 = S_i]$$

Then the HMM generates a sequence of observations

$$\boldsymbol{O} = O_1, O_2, \ldots, O_T$$

by the following process:

1. Choose an initial state $q_1 = S_i$ stochastically from the initial state distribution $\boldsymbol{\pi}$
2. For $t = 1$ to T
    i. Choose $O_t = v_k$ according to the distribution $b_i(k)$ of the current state S_i
    ii. Stochastically transition to a new state S_j from S_i according to A

Note now that a hidden Markov Model is entirely determined by the transition matrix A, the emissions matrix B and the initial distribution $\boldsymbol{\pi}$ (noting that the dimensions N and M are encoded in the dimensions of the matrices) hence for convenience we may denote a HMM by

$$\lambda = (A, B, \boldsymbol{\pi})$$

Hidden Markov Models were first introduced by in a series of papers by L.E Baum and others in the late 1960s (Baum and Petrie, 1966; Baum et al., 1970) and have been since used to study handwriting recognition (Bunke et al., 1995), speech recognition (Jelinek, 1998; Juang and Rabiner, 1991), natural language modelling (Jurafsky et al., 2002; Manning and Schütze, 1999) and biological processes (Durbin et al., 1998; Krogh et al., 1994; Liò and Goldman, 1998).

Given these defintions there exist three basic problems which form the basis of using HMMs as a tool for machine learning

1. Given an observation sequence $\boldsymbol{O} = O_1, O_2, \ldots, O_T$ and a HMM λ, what is $\mathbb{P}(\boldsymbol{O}|\lambda)$ - the probability that the observation sequence $\boldsymbol{O}$ was produced by λ?
2. Given a HMM λ and an observation sequence $\boldsymbol{O} = O_1, O_2, \ldots, O_T$ what is the state sequence of the underlying Markov chain in λ which is most likely to have generated $\boldsymbol{O}$?

3. Given an observation sequence $\boldsymbol{O}$, how to do we choose the parameters of $\lambda = (A, B, \boldsymbol{\pi})$ which best optimise $\mathbb{P}(\boldsymbol{O}|\lambda)$?

In fact the problem of sign language gesture recognition can be seen as instance of problems 3 and 1. First we take for each gesture g a set of training data which are recordings of the joints in 3D space. This training data forms a collection of observation sequences $\boldsymbol{O_1}, \ldots, \boldsymbol{O_n}$ which are used in a solution to problem 3 to parameterise a hidden Markov Model λ_g to model the gesture.

Then given a fresh observation sequence $\boldsymbol{O}$ we can use a solution to problem 1 to compute $\mathbb{P}[\boldsymbol{O}|\lambda_g]$ for each g. We can then take the gesture g for which this probability is maximised and, provided the probability exceeds some threshold, conclude that the gesture g corresponds to the observation sequence $\boldsymbol{O}$.

2.1.1 The Forward and Backward Variables

Suppose we are given an observation sequence $\boldsymbol{O} = O_1, O_2, \ldots, O_T$ and a HMM λ and we wish to solve the evaluation problem (problem 1). Rabiner (1989) notes that a direct computation of $\mathbb{P}[\boldsymbol{O}|\lambda]$ will take time order $\mathcal{O}(2TN^T)$ which is infeasible even for moderate values of N and T. Instead we use a dynamic programming approach, the forward algorithm, introduced in Baum and Sell (1968) and Baum et al. (1970).

Define the *forward variables* $\alpha_t(i)$ for each $1 \leq t \leq T$ and $1 \leq i \leq N$ by

$$\alpha_t(i) = \mathbb{P}[O_1, \ldots, O_t, q_t = S_i|\lambda]$$

Then note these can be inductively computed by the following procedure

$$\begin{aligned}
&\alpha_1(i) = \pi_i b_i(O_1) \\
&\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1}) \qquad \text{for } 1 \leq t \leq T-1 \text{ and } 1 \leq j \leq N
\end{aligned}$$

Finally we have

$$\mathbb{P}[\boldsymbol{O}|\lambda] = \sum_{i=1}^{N} \mathbb{P}[\boldsymbol{O}, q_T = S_i|\lambda] = \sum_{i=1}^{N} \alpha_T(i)$$

The set of forward variables can be computed by dynamic programming in $\mathcal{O}(N^2T)$ time and hence we can compute $\mathbb{P}[\boldsymbol{O}|\lambda]$ in this time - a significant improvement over direct computation. This algorithm is the *forward algorithm* for evaluation in HMMs.

Symmetrically to the forward variables we can define a collection of *backward variables* by

$$\beta_t(i) = \mathbf{P}[O_{t+1}, O_{t+2}, \ldots, O_T | q_t = S_i, \lambda]$$

Which gives the probability of a partial observation sequence from a time t given the state of the HMM λ at time t. These too can be computed iteratively as

$$\begin{aligned} &\beta_T(i) = 1 && \text{for each } 1 \leq i \leq N \\ &\beta_t(i) = \sum_{j=1}^{N} a_{i,j} b_j(O_{t+1}) \beta_{t+1}(j) && \text{for } t = T-1, T-2, \ldots, 1 \text{ and } 1 \leq i \leq N \end{aligned}$$

and these too can be computed by dynamic programming in $\mathcal{O}(N^2T)$ time. The backward variables are not needed in the solution to the evalutation problem but are used in the following section to re-parameterise hidden Markov Models.

2.1.2 Forward-Backward Algorithm

The problem of parameterising a hidden Markov Model λ is considerably more difficult than the other two problems of HMMs. In fact it is known that there is no analytic solution which provides a model λ to maximise the probability of some observation sequence $\boldsymbol{O}$. Instead we must use local optimisation methods which given an initial parameterisation λ_0 iteratively improve it until $\mathbb{P}[\boldsymbol{O}|\lambda]$ reaches a local maximum.

We will use a modified version of the *Baum-Welch algorithm* introduced by Baum et al. (1970) called the *forward-backward algorithm* (Rabiner, 1989) which is reproduced below. This algorithm is an instance of an expectation-maximization algorithm Moon (1996) which is a general technique used to determine maximum likelihood estimators in a number of machine learning models Bishop et al. (2006).

For $t \in \{1, \ldots, T-1\}$ and $i, j \in \{1, \ldots, N\}$ define

$$\gamma_t(i,j) = \mathbb{P}[q_t = S_i, q_{t+1} = S_j | \boldsymbol{O}, \lambda]$$

such that $\gamma_t(i,j)$ is the probability of being in state S_i at time t and transitioning to S_j at the next time step given the HMM λ and the observation sequence $\boldsymbol{O}$. The by definiton of the foward and backward variables we have

$$\gamma_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta(j)}{\mathbb{P}[\boldsymbol{O}|\lambda]}$$

We can define for each $1 \leq t \leq T-1$ and each $1 \leq i \leq N$ the probability of being in state i at time t by

$$\gamma_t(i) = \mathbb{P}[q_t = S_n | \boldsymbol{O}, \lambda] = \frac{\alpha_t(i)\beta_t(i)}{\mathbb{P}[\boldsymbol{O}|\lambda]}$$

and then we have

$$\gamma_t(i) = \sum_{j=1}^{N} \gamma_t(i,j)$$

Now using these equations we can compute

$$\sum_{t=1}^{T-1} \gamma_t(i) = \text{The expected number of transitions from } S_i$$

$$\sum_{t=1}^{T-1} \gamma_t(i,j) = \text{The expected number of transitions from } S_i \text{ to } S_j$$

which can be used to reparameterise the model $\lambda = (A, B, \boldsymbol{\pi})$ as follows, set

$$\begin{aligned}
\overline{\pi} &= \text{the expected number of times in state } S_i \text{ at time } 1 = \gamma_1(i) \\
\overline{a_{ij}} &= \frac{\text{the expected number if transitions from } S_i \text{ to } S_j}{\text{expected number of transitions from state } S_i} \\
&= \frac{\sum_{t=1}^{T-1} \gamma_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \\
\overline{b}_j(k) &= \frac{\text{the expected number of times in state } S_j \text{ observing symbol } v_k}{\text{the expected number of times in state } S_j} \\
&= \frac{\sum_{\substack{t=1 \\ O_t = v_k}}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}
\end{aligned}$$

Then if denote $\overline{\lambda} = (\overline{A}, \overline{B}, \overline{\boldsymbol{\pi}})$ then it has been proven (Baum and Sell, 1968; Levinson et al., 1983) that either

1. $\mathbb{P}[\boldsymbol{O}|\overline{\lambda}] > \mathbb{P}[\boldsymbol{O}|\lambda]$ or
2. λ is locally maximized with respect to $\mathbb{P}[\boldsymbol{O}|\lambda]$ and $\overline{\lambda} = \lambda$

It follows that given an initial HMM λ we may improve it to a locally optimal model for some observation sequence $\boldsymbol{O}$ via the following local search:

1. Whilst $\mathbb{P}[\boldsymbol{O}|\lambda]$ increases:
   i. Compute the $\alpha_t i$, $\beta_t(I)$, $\gamma_t(i,j)$ and $\gamma_t(i)$
   ii. Compute the new parameters $\overline{A} = [\overline{a}_{ij}]$, $\overline{B} = [\overline{b}_j(k)]$ and $\overline{\boldsymbol{\pi}} = [\overline{\pi_1}, \ldots, \overline{\pi_N}]$
   iii. Set $\lambda := \overline{\lambda}$
2. return λ

Note that this algorithm may not actually terminate. In practice we threshold the increase of $\mathbb{P}[\boldsymbol{O}|\lambda]$ to ensure termination in a reasonable time. Further, in practice we will not have single observation sequence but rather a collection, perhaps from a variety of signers of different heights, genders, ages or dialects. In the implementation of our HMM framework we will modify this algorithm to work for multiple observation sequences. This will again increase the time complexity of the algorithm, however this problem is not so significant as in practice we will use this algorithm once per sign and save the parameterisations.

2.1.3 Limitations of HMMs

A significant limitation of hidden Markov Models is that they can have only a finite set V of possible observations. In practice this can prevent us using a HMM to model a specific gesture exactly. Take for example the problem of detecting the gesture of a circle drawn on a 2D plane. We might create a hidden Markov Model in which the states of the Markov chain represent some specific points on the plane and want our matrix B to be such that

$$b_j(\boldsymbol{p}) = \mathbb{P}[\text{the pen is at point } \boldsymbol{p} \text{ at time } t \mid q_t = A_j] \qquad \text{for each point } \boldsymbol{p} \in \mathbb{R}^2$$

However as the plane is continuous and V is finite this is not possible. One solution is the discretise the plane and have only a finite (but large) set of observation symbols. This method has been used with some success to distuingish between gestures with large variations, for example different tennis strokes (Yamato et al., 1992). However without sufficiently fine grained discretisation, which will severely impact the efficiency of our algorithms, our observation sequences will suffer from signal degradation. As we plan to model signed languages, in which certain subtle changes to a gesture can change the meaning [GIVEN AN EXAMPLE], it is likely signal degradation will impact on the accuracy of our system.

2.2 Continuous Distribution HMMs

Chapter 3

Building a Hidden Markov Model for Isolated Signs

Chapter 4

Determining a Sign from the Possibilities

Chapter 5

My Conclusions ...

Here I put my conclusions ...

Appdx A

and here I put a bit of postamble ...

Appdx B

and here I put some more postamble ...

References

Leonard E Baum and Ted Petrie. Statistical inference for probabilistic functions of finite state markov chains. *The Annals of Mathematical Statistics*, 37(6): 1554–1563, 1966. 5

Leonard E Baum and George R Sell. Growth transformations for functions on manifolds. *Pacific Journal of Mathematics*, 27(2):211–227, 1968. 6, 9

Leonard E Baum, Ted Petrie, George Soules, and Norman Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *The annals of mathematical statistics*, pages 164–171, 1970. 5, 6, 7

Christopher M Bishop et al. *Pattern recognition and machine learning*, volume 4. springer New York, 2006. 7

Horst Bunke, Markus Roth, and Ernst Günter Schukat-Talamazzini. Off-line cursive handwriting recognition using hidden markov models. *Pattern recognition*, 28(9):1399–1413, 1995. 5

Richard Durbin, Sean R Eddy, Anders Krogh, and Graeme Mitchison. *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge university press, 1998. 5

A. A. Argyros I. Oikonomidis, N. Kyriazis. Kinect 3d Hand Tracking, March 2013. URL `http://cvrlcode.ics.forth.gr/handtracking/`. 3

Frederick Jelinek. *Statistical methods for speech recognition*. MIT press, 1998. 5

Biing-Hwang Juang and Lawrence R Rabiner. Hidden markov models for speech recognition. *Technometrics*, 33(3):251–272, 1991. 5

Dan Jurafsky, James H Martin, and Andrew Kehler. *Speech and language processing: an introduction to natural language processing, computational linguistics and speech recognition*, volume 2. MIT Press, 2002. 5

Anders Krogh, Michael Brown, I Saira Mian, Kimmen Sjolander, and David Haussler. Hidden markov models in computational biology: Applications to protein modeling. *Journal of molecular biology*, 235(5):1501–1531, 1994. 5

Stephen G. LATTA, Kudo TSUNODA, Kevin GEISNER, Relja MARKOVIC, Darren Alexander BENNETT, and Kathryn Stone PEREZ. Gesture keyboarding, 08 2010. URL http://www.patentlens.net/patentlens/patent/US_2010_0199228_A1/en/. 3

Stephen E Levinson, Lawrence R Rabiner, and Man Mohan Sondhi. An introduction to the application of the theory of probabilistic functions of a markov process to automatic speech recognition. *Bell Syst. Tech. J*, 62(4):1035–1074, 1983. 9

Pietro Liò and Nick Goldman. Models of molecular evolution and phylogeny. *Genome research*, 8(12):1233–1244, 1998. 5

Christopher D Manning and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT press, 1999. 5

Microsoft. What's New - Kinect SDK 1.7, March 2013. URL http://www.microsoft.com/en-us/kinectforwindows/Develop/New.aspx. 3

Todd K Moon. The expectation-maximization algorithm. *Signal Processing Magazine, IEEE*, 13(6):47–60, 1996. 7

Lawrence R Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989. 4, 6, 7

Junji Yamato, Jun Ohya, and Kenichiro Ishii. Recognizing human action in time-sequential images using hidden markov model. In *Computer Vision and Pattern Recognition, 1992. Proceedings CVPR'92., 1992 IEEE Computer Society Conference on*, pages 379–385. IEEE, 1992. 10