CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

A Swedish Sign Language Chatbot

Building sign dataset, sign language recognition network and domain-specific chatbot.

Master's thesis in Computer science and engineering

Joakim Burman

Martin Hagmar

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2023

Master's thesis 2023

A Swedish Sign Language Chatbot

Joakim Burman, Martin Hagmar

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2023

A Swedish Sign Language Chatbot
Building sign dataset, sign language recognition network and domain-specific chatbot.
Joakim Burman, Martin Hagmar

Supervisor: Carl-Johan Seger, Department of Computer Science and Engineering
Examiner: Jean-Philippe Bernardy, Department of Computer Science and Engineering

Master's Thesis 2023
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2023

A Swedish Sign Language Chatbot
Building sign dataset, sign language recognition network and domain-specific chatbot.
Joakim Burman
Martin Hagmar
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Being unable to hear is a huge obstacle to smooth and natural communication, especially in interactions with non-sign language speakers. This communication barrier makes hearing-impaired people more vulnerable in various emergency situations where the ability to be understood quickly can be the difference between life and death.

A significant amount of the previous sign language recognition research has been focused on the translation from signs to words, through sensory gloves and cameras, focused on the signer. This previous research can be seen as circumventing the core problem, that non-hearing impaired people lack the ability to understand sign language. The thesis aims to expand this research field through the development of a sign language chatbot, a tool that hopefully allows for faster learning of sign language.

This study presents the development of a sign recognition neural network that predicts sign language made by a person through a webcamera. This sign recognition network achieved a 99% accuracy in recognizing signs from a test dataset, demonstrating the effectiveness of the chosen neural network architecture and training methodology and keypoint estimation. However, the real-world evaluation showed less optimal results, but still sufficiently high, due to a lower generalization of unseen data and novice signers as test subjects. No proper evaluation was conducted to see which of the two reasons contributed the most to the lower real-world performance.

This sign recognition network has been paired with a chatbot to enable sign-language conversations between a user and a computer, potentially enabling the system to be used as a sign-language training tool.

Keywords: Computer science, sign language, machine translation, chatbot, sign language recognition, keypoint estimation, machine learning, thesis, project

Acknowledgements

We would like to thank our supervisor, Carl-Johan Seger, for not only being a guide and giving feedback but also acting as a great source of ideas for both solutions and future developments. We also thank Stockholm University Institution of Linguistics, especially Thomas Björkstrand and Calle Börstell, for answering any questions that arose around sign language and for helping to find appropriate sources. We would also like to thank our testing users, who had to redo the testing on short notice after an improvement to the system was made. Finally, we thank Göteborgs Dövas Föreing (Gothenburg Association for the Deaf), for answering our questions during the thesis proposal, regarding the necessity of such a product.

Joakim Burman, Martin Hagmar, Gothenburg 2023-05-23

Contents

List of Figures

List of Tables

1

Introduction

Not being able to hear sounds is a huge obstacle to smooth and natural communication. This obstacle is especially problematic for the hearing impaired in interactions with non-sign language speakers. These communication barriers makes hearing-impaired people more vulnerable in various emergency situations where the ability to be understood quickly can be the difference between life and death.

A significant amount of research has been done on tools that help the communication between deaf people and non-sign-language speakers. This translation research has been done through the usage of sensory gloves [1], 3D depth sensor cameras [2] and regular cameras with keypoint estimation [3]. This research has been successful in developing different versions of sign language to text translation with high accuracy. However, a problem with this research is that the translation only works one way, from sign to text, requiring the non-sign language speaker to communicate back through text. This inability to communicate smoothly without barriers, that non-hearing impaired people lack the ability to understand sign language, is a core problem in the research that has not been solved and has instead been circumvented through translation.

The aim of the thesis is to bridge this gap of barriers, developing a tool that hopefully allows for faster learning of sign language, a sign language chatbot.

1.1 Objective

The aim of the thesis is to build a sign language chatbot system able to interpret sign language and respond with sign language videos back to the user. This would enable a user to train the usage of sign language through conversations without needing a proficient sign language signer to train with. The system is intended to be able to be used at home, in a setting similar to a home office, sitting in a chair in front of a computer with a webcam.

Our proposed solution to this problem can be divided into two areas, a conversational chatbot and machine learning translation of sign language.

1.2 Sign Language

Sign language uses visual signs to convey meaning instead of spoken words. The number of sign languages worldwide is not precisely known, as each country has their own version of sign language, and some more than one, each with its own unique grammar and lexicon [4]. Although some sign languages share some similarities, there are generally large variances between sign languages.

In Swedish sign language (SSL), the label of signs is called *glosor* (Hereafter, referred as to the English translation glosses). In the case of sign language, glosses are generally written in capital letters and allow for easier talking, reading and writing about SSL. For an example of translations between a Swedish sentence into SSL glosses, see Table 1.1.

Igår var roligt. Min familj, inkluderande min man, son och dotter gick till sjön och simmade hela dagen.
IGÅR ROLIG. FAMILJ MAN(MAKE) SON DOTTER KOMMA-TILL SJÖ SIMMA HELA-DAGEN

Table 1.1: Translation of a Swedish sentence into SSL with the corresponding glosses. Translation given by Stockholm University Institution of Linguistics.

The purpose of this example is to illustrate that although SSL is based on the Swedish language, it is not as similar to Swedish as one might think. SSL lack many of the linking words such as *var*, *min* and *och* (The English translation being *was, mine, and*). Although the glossary translation might seem similar to the Swedish sentence at first glance, it is important to note that glosses do not indicate the meaning, movement, place of articulation, or grammar of a sign. One must know sign language to understand the glossed form, otherwise, it is simply a label.

The signs that correspond to each gloss have a distinct full motion, from a resting position to the articulation of the sign, and back to the resting position. However, when signs are combined into a sentence, the motions are often reduced, with transport and articulation movements changed depending on the context of surrounding signs in the utterance, as well as the style of signing [5]. Advanced signers articulate signs in a quick, cohesive manner, making it more difficult for beginner signers to distinguish the boundaries of individual signs.

Advanced signers also incorporate more parts of the body other than just the hands to convey the meaning and grammatics of the sign or to differentiate between two similar-looking signs. These other body parts, linguistically significant elements that are not expressed by the hands, such as head movement, eyebrow movement, facial expression, gaze and mouth, are called “nonmanual markers” or “nonmanuals” [6]. It has been shown that advanced signers, while communicating, primarily focus on each others faces to read the nonmanual behaviour, rather than the hands, to understand the essential grammatical information [7][8].

1.3 Sign language training in Sweden

In Sweden, if disregarding any self learning method, there are a few different courses available in SSL. An elective introductory course offered by Hermods VUX is called "Svenskt teckenspråk för hörande 1" (Swedish Sign Language for the Hearing 1), an adult course for either in-person learning or remote [9]. The course material consists of learning the basics of SSL, such as the alphabet, numbers, weekdays, and history, among others. If you are a parent of a deaf child, further learning can be done through the follow-up education, *TUFF*, *Teckenspråksutbildning för föräldrar* (Sign language education for parents), a 240-hour course [10]. As the name suggests, the education is meant for parents of deaf or hearing-impaired children, where parents of younger children are prioritized for the application. This course is mostly held on location, and due to it not being held in all cities, might require parents to commute long distances or stay away for some time away from their children.

After an email conversation with Göteborgs Dövas Förening (Gothenburg Association for the Deaf), it became clear that the existence of a sign language chatbot would help immensely in the learning of sign language and "being worth gold for parents". This training tool would thus allow parents to train efficiently during and after these courses to further improve, given the limited time these courses offer, and spend more time with their children.

1.4 Sign Language Dataset

Due to the intended use case mentioned in Chapter 1.1, the machine learning algorithm must learn on a sign language dataset recorded from the front. As the intention is for users to learn the complete motion without shortening or grammatical inflection, the dataset has to thereby also contain only complete signs without shortening. Currently, there exists no public SSL dataset suitable for this thesis with complete motion signs filmed from the front with a large number of examples, but two options exist in the Swedish Sign Language Lexicon, discussed in Chapter 1.4.1, and Swedish Sign Language Corpus, discussed in Chapter 1.4.2.

1.4.1 Swedish Sign Language Lexicon

The closest suitable option for a dataset is an SSL dictionary database, called Svenskt Teckenspråkslexikon 2023 (Swedish Sign Language Lexicon 2023), containing 20186 published signs, created by Stockholm University Institution of Linguistics [11]. These published signs are filmed on a blue screen background; see Figure 1.1 for an example image of how these videos look.

Figure 1.1: Frame from the video of the word “HEJ” (*hello*) [12].

The signs in this SSL Lexicon are complete in motion and filmed from the front, however, are limited in the number of examples per sign, often no more than one. Thereby, these videos are not suitable for a machine learning dataset due to too few training examples, and can only be used as guidance for a later dataset recording, mentioned in Chapter 5.1.

1.4.2 Swedish Sign Language Corpus

There does exist an SSL corpus called Svensk Teckenspråkskorpus, also created by Stockholm University Institution of Linguistics, containing long-form videos of conversations in sign language [13][14]. These long videos are time-annotated with the signs shown, but filmed from the side of the signer and not in front; see Figure 1.2 for an example. These long-format videos in the corpus are also not suitable for a machine learning dataset, at least not for the purposes of this thesis. As mentioned in Chapter 1.1, the intended use case is on signs seen from the front, from the perspective of a webcam, and not from the sides as seen in the corpus. Adding to this, the signs need to be clear, complete in motion, and not shortened, as would be the case in a conversation.

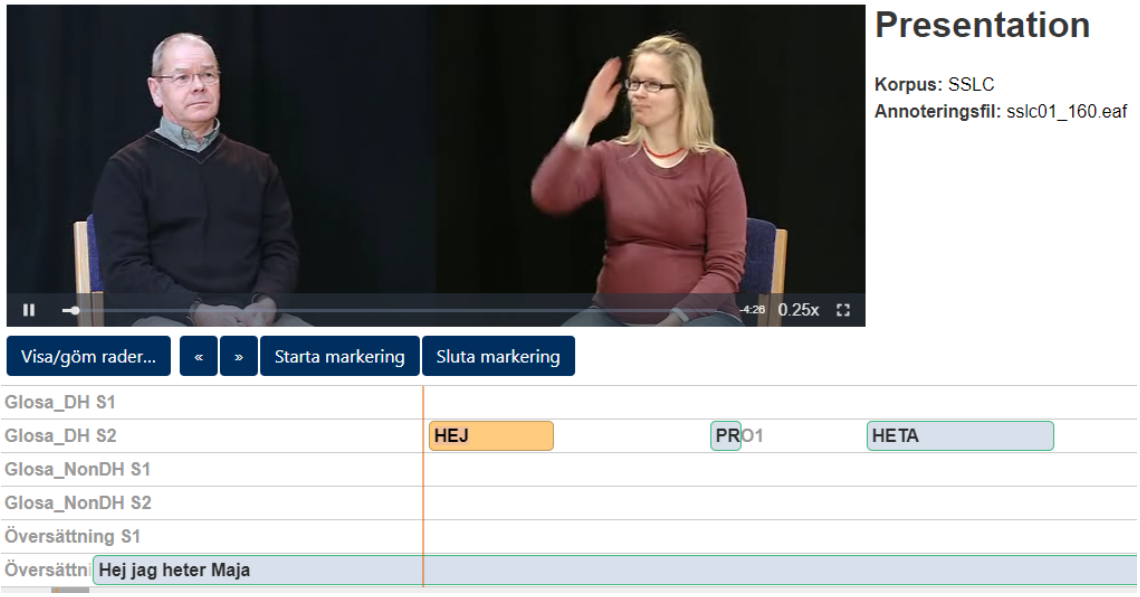


Figure 1.2: Frame from the time annotation of the word "HEJ" (*hello*) in the SSL corpus [15]. The person to the right is making the sign.

2

Related Work

Due to the work being split into two parts, sign language recognition and a chatbot, this section of related work is split up into these two parts. The section for chatbots has been extended with related work of chatbots being used as a tool for learning.

2.1 Sign Language Translation

In 2005 by Oz et al. developed a sign language translation system using a glove with sensors called a Cyberglove [1]. The glove contained 18 sensors measuring the angle data of the fingers. This data was used to define the shape of the hand in the different gestures. Together with an alphabet recognition network, they were able to generate words and had an accuracy of 96%. A drawback of this research was that it is limited to predicting only the alphabet and not whole words. Although the system achieved high accuracy, it was not considered a complete success because of the high costs of the gloves and the difficulty of using them outside laboratories. Solutions that use computer vision techniques are seen as more cost-effective and portable due of the use of a single camera.

In 2015, Dong et al., continued the research of sign language recognition through a vision-based system using a Microsoft Kinect camera [2], which has both an RGB camera and two 3D depth sensors. To generate the training data, a multi-coloured latex glove was used to help segment the joint data. The system achieved a mean accuracy of 92%, however, this research was still limited to the alphabet, consisting of a dataset containing 24 static alphabet signs.

In 2019, Ko et al. developed a neural network model using keypoint estimation for the translation of sign language videos into natural language sentences [3]. The keypoint estimation was done through OpenPose, an open-source toolkit for real-time multi-person keypoint detection. The estimated coordinates of 124 keypoints were used in the system, 12 for the body, 21 for each of the hands, and 70 for the face, however due to the ability of OpenPose to only track the x and y-axis for each keypoint, the total number of parameters was 248. The neural network was based on gated recurrent units (GRU), and achieved over 93% translation accuracy on the validation set. The system was compared to a convolutional neural network (CNN), and it was found that an approach based on keypoint estimation had significantly better results and led to less overfitting, especially with a small number of training examples.

In a 2020 paper by Camgöz et al., a novel, transformer-based architecture for sign language recognition and translation was introduced [16]. The system was evaluated on a so-called PHOENIX14T dataset, which features sign language interpretation recorded from news and weather forecast broadcasts from the German public TV station PHOENIX [17]. From this evaluation, the transformer-based architecture showed state-of-the-art recognition and translation results, and in some cases doubled the performance of previous approaches.

In 2020, Google released the possibility of tracking hands through their real-time keypoint estimation software Mediapipe, called the Mediapipe hand solution [18]. Through the usage of this new keypoint estimation software, Halder and Tayade iterated on the concept of keypoint estimation for sign language in 2021, pairing it with a Support Vector Machine [19]. The sign recognition system was for the alphabet and not full signs, and the system was able to interpret and translate the alphabet signs with an average accuracy of 99%. The utilized MediaPipe solution could only track the hands, allowing a maximum of 42 keypoints combined of both hands with none for either the pose or the face. Due to the ability of MediaPipe to track in three dimensions, a total of 126 parameters were used.

2.2 Chatbot

The first implementation of a chatbot was developed by Joseph Weizenbaum in 1966, called ELIZA, which purpose was to act as a psychotherapist [20]. The chatbot used a simple form of pattern matching through keywords. The conversational ability of ELIZA was limited, and the pattern-matching rules were not flexible enough to be easily applied to new domains [21]. An new dialogue system called GUS, the Genial Understander System, was developed by Daniel G. Bobrow et al. in 1977, originally designed to take on the role of a travel agent [22]. The system was the first to incorporate the frame-based approach, where each frame represents a collection of information that has been gathered by the system. The dialogue started with an empty frame structure that contained slots for each bit of information that could be used for planning. The next large step in the evolution of chatbots came in 1995, with the creation of the chatbot ALICE (Artificial Intelligent Internet Computer Entity), developed by Richard S. Wallace [23]. ALICE was based on the Artificial Intelligence Markup Language (AIML), a derivative of Extensible Markup Language (XML). This was done to allow the possibility of adding to the knowledge base of ALICE, a significant improvement over previous pattern matching systems [21]. AIML consists of data objects that are units of topics and categories. Each category is a rule for the matching of the inputs to the conversion into an output.

With the development of new deep machine learning algorithms, there is less need to pattern match the user input, allowing a more nuanced understanding of conversations such as user sentiment, semantic analysis and chat history [21]. Due to this, chatbots have been expanded to new areas of application. One such new area is that of smart personal assistants, such as Apple Siri (2011) [24], Microsoft Cortana (2014) [25], Amazon Alexa (2014) [26] and Google Assistant (2016) [27]. These new personal assistant chatbots can communicate with the user through both voice and text and

can be found in smartphones, smartwatches, cars, and dedicated speakers.

The most recent advancement in chatbot technology has come from the invention of the transformer network in 2017 [28]. In 2020, the company OpenAI announced the third version of their pre-trained transformer language model called GPT-3 (Generative Pre-trained Transformer 3), the largest language model constructed to date. This new model has 175 billion parameters in a 96 layer network, trained on a corpus of 499 billion tokens of web content [29]. In 2022, this massive network was modified and trained on a large dataset of conversational text and incorporated into a chatbot. This new chatbot, named ChatGPT, is able to answer appropriate responses in context-specific scenarios and maintain a coherent conversation that mimics humans.

2.2.1 Chatbots as a Tool for Learning

Chatbots have been used as a tool to teach languages, and are not a new concept. The concept was studied in 2006 when Fryer and Carpenter argued that the potential value of chatbots as a language learning tool is immeasurable [30]. They argued that one of the strengths of such chatbots is convenience, that the chatbots are ready to chat when and wherever students are. The study also proposed six potential advantages of chatbots for language learning: (1) Students tend to feel more relaxed talking to a computer compared to a human, leading to less anxiety; (2) Chatbots can repeat the same material endlessly as they do not lose patience; (3) Chatbots may allow students to practice skills in both reading and listening; (4) Chatbots are novel and interesting to students and therefore may create new or renewed interest in language learning, improving motivation; (5) Chatbots allow for the opportunity for students to use new vocabulary that they normally would not use; (6) Chatbots can potentially provide quicker feedback to a student's grammar and spelling. Although Fryer and Carpenter conducted the study when the capabilities of chatbots are less than that of today, it is interesting that already then the benefits of chatbots as a learning tool could be seen.

In 2013, Wang and Petrina created and studied the chatbot Lucy, a digital language tutor capable of extensive conversations with users as they speak into their computer microphones [31]. Lucy is trained to play five characters, a travel agency assistant, a hotel assistant, a tour guide, a waitress, and a call centre assistant. In the study, Wang and Petrina studied the effectiveness of Lucy in teaching English to non-native speakers. The findings of this suggest that Lucy is effective in improving the English language proficiency of learners, which was measured in standardized tests.

A 2019 study by Fryer et al. suggests that an advantage mentioned in Fryers earlier work from 2006, point (4), which is of the novelty to the students, is not as helpful as once thought. After students become more familiar with the technology, the "novelty effect" - the newness of the technology to the students - wears off, and the improved engagement might thereby only be short-term [32]. This study showed that the student's interest in conversing with a chatbot dropped significantly between conversations one and two, whereas it remained consistent when conversing with a human partner. Fryer thereby argues that spacing out interactions with a chatbot

as a learning tool with regular learning to avoid the novelty decreasing too much. This understanding of the novelty effect is supported by a 2018 study by Gallacher et al. arguing that students view chatbot as a learning tool as a novelty rather than a lasting partner in daily language practices [33].

In 2022, Huang et al. conducted a systematic review of chatbots for language learning [34]. This study concluded three technological advantages of such chatbots: (1) Timeliness, as students gain the opportunity to learn a language at any time as well as receive immediate responses; (2) Ease of use, as many chatbots now are available instantly on websites and mobile applications, where students are able to understand what to do without needing instructions; (3) Personalization, as even given a common topic to discuss with the chatbot, students can communicate with the chatbot differently with different inputs. However, the study also identified two big technological challenges, the first being the novelty effect mentioned above. The other is the limitations of chatbot technology itself, as the most frequently reported issue was how unnatural the computer-generated voice was perceived. Other complaints were those of failed communications stemming from both the users and the chatbot, and finally, the lack of emotion and visible cues from interactions with chatbots. The study concluded by saying that despite two decades of demonstrating the potential of using educational chatbots for language learning, the effects of chatbots on students' learning are still underresearched.

3

Limitations

There are a number of limitations on this thesis, partly due to hardware limitations and the fact that no large enough dataset of sign language videos exists to train the network on. This absence of a public dataset necessitated the creation of one by the authors. Many of the limitations of the sign recognition network mentioned in Chapter 3.2 are limitations that are rooted in the creation of the dataset, mentioned in Chapter 3.1. However, these two chapters are split up due to clarification of where the issue lies. If the limitation affects the end user, it was decided that the issue is placed in Chapter 3.2.

3.1 Limitation on the dataset

Since there is no currently existing dataset for SSL, a new, custom-made dataset had to be created. Therefore, only a small amount of words were chosen, mainly due to the time-intensive part of data collection, and due to the thesis being more of a proof of concept rather than a full market product. Since the authors not being advanced speakers of SSL, nonmanual signals such as facial expression, mentioned in Chapter 1.2, were ignored in this dataset. Adding to the fact that the dataset was recorded by novice sign language signers, there might be some errors that occur in the sign, however, this has been mitigated through in-depth analysis of the signs seen in the SSL lexicon, mentioned in Chapter 1.4.1.

Due to the fact that the dataset was only recorded on two people, the videos needed to be augmented to create greater availability and variability of the data to prevent overfitting and reducing the labor of recording an entire dataset. However, an augmented dataset inherits the same biases as the original data [35].

3.2 Limitation on the Sign Recognition Network

One major limitation of the sign recognition network is that since every word of the dataset is recorded one at a time and not as a part of a sentence, the user has to sign each word separately with a pause. Thereby, it is impossible for the user to sign words in a flowing manner as part of a sentence, as would more experienced signers, as mentioned in Chapter 1.2. However, since the intended use case is for novice signers, the system can still achieve the objective of being used for sign language training.

Another limitation of the sign recognition network rooted in the dataset is that of it only being able to predict signs made with the right hand as the main hand. Both hands can be used, but only if two hands are required for the sign. If only one hand is needed for a sign, this was always recorded with the right hand for the dataset, thereby not allowing any left-handed signs.

As mentioned in Chapter 1.1, the intended use case is for seated signing in front of a web camera, thereby the signs were recorded as such for the dataset. This does however impact the performance of the model when the user is standing, so being seated is best for optimal performance.

As mentioned in Chapter 5.1.1, the system does not record detailed keypoints of the face, and the system is thereby unable to detect any nonmanual from this body part, and can therefore not understand any grammatical changes that the user is signing. As mentioned before, as the intended use case is for novice signers, it is sufficient to only learn the signs and not the nonmanual signals.

3.3 Limitation on the chatbot

The chatbot implemented is an information retrieval chatbot with topics limited to only being relevant to the chosen signs. Due to the user being limited to 15 signs, there are a limited number of topics of conversation that the user can create. Thereby, an information retrieval-based chatbot is reasonable since there is no point in developing a complex chatbot for discussion topics that the user is unable to communicate.

4

Theory

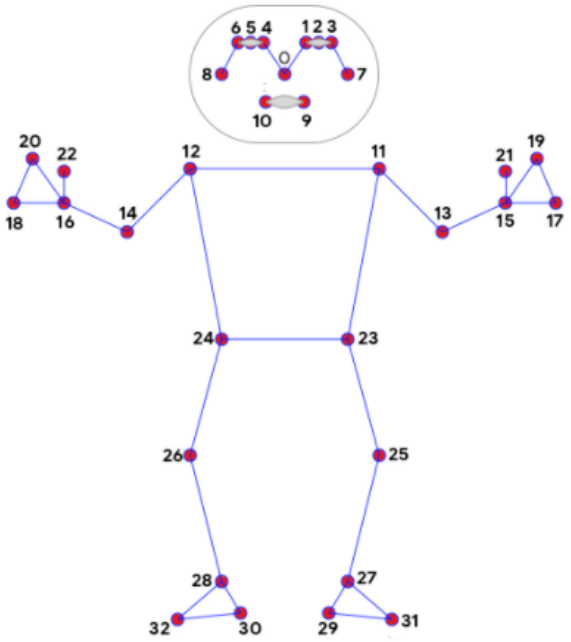
To make this thesis possible, MediaPipe was used for the tracking of keypoints for the body and hands [36], a gated recurrent network was used for the prediction of the signs being done, and the chatbot was created using an information retrieval model. These methods are explained further in the sections below.

4.1 Google MediaPipe

Keypoints are specific points of interest used for tracking the motion of objects, on humans they often more easily identifiable points such as joints in the body. MediaPipe is an open-source, cross-platform framework for the tracking of these keypoints, developed by Google [36]. The framework is designed to be lightweight, allowing it to run in real-time on both desktop and mobile platforms.

MediaPipe has many different solutions for various types of tracking. The solution used in the thesis is called Mediapipe Holistic, which allows for the simultaneous tracking of a person's hands, pose, and face [37]. The tracking from this Holistic solution outputs x, y and z coordinates for up to 543 keypoints (33 for pose, 21 per-hand and 468 for the face) [38][39]. The pose estimation can also output a fourth parameter, visibility, for indicating the likelihood of the keypoint being visible. In total, if all keypoints are utilized with all of the corresponding value parameters of each keypoint, a single video frame outputs 1662 parameters.

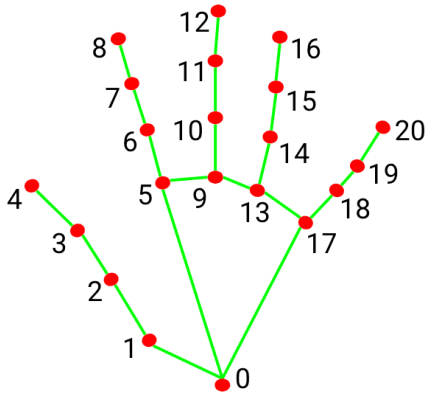
The pose estimation and its keypoints which can be seen in Figure 4.1 can also be divided into two separate implementations, *pose_world_landmarks* and *pose_landmarks*. The former outputs the coordinates in meters, normalized from the origin of the centre between the hips. The latter instead outputs the x and y coordinates, normalized to the height and width of the screen, (0,0) being the upper left corner and (1,1) being the lower right corner. The z coordinate is an estimation of the distance the object is from the camera. The smaller the value of z, the closer the keypoint is to the camera. However, this value of z contains a disclaimer of not being entirely correct, as the model is not fully trained to predict this depth.



0. nose
1. left_eye_inner
2. left_eye
3. left_eye_outer
4. right_eye_inner
5. right_eye
6. right_eye_outer
7. left_ear
8. right_ear
9. mouth_left
10. mouth_right
11. left_shoulder
12. right_shoulder
13. left_elbow
14. right_elbow
15. left_wrist
16. right_wrist
17. left_pinky
18. right_pinky
19. left_index
20. right_index
21. left_thumb
22. right_thumb
23. left_hip
24. right_hip
25. left_knee
26. right_knee
27. left_ankle
28. right_ankle
29. left_heel
30. right_heel
31. left_foot_index
32. right_foot_index

Figure 4.1: Keypoints of the pose model of the MediaPipe Pose solution [40].

For the keypoint estimation of the hands, there is only one implementation. This implementation follows the same normalization to the width and height of the screen as *pose_landmarks*. The z-coordinate follows a similar depth estimation as before, however, contains no such disclaimer of not being entirely correct. The depth estimation of the hand is rooted in the wrist, keypoint 0, seen in Figure 4.2, where every other keypoint has a value of depth relative to 0. The hand tracking can track both hands simultaneously and has two separate outputs for each hand.



0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

Figure 4.2: Keypoints of the hand model of the MediaPipe Hands solution [41].

Similarly, the face keypoint detection also has only one implementation, which also follows the same normalization to the width and height of the screen. The value of z is rooted in the centre of the head, but is otherwise the same as with the other values of z.

The values of the x and y coordinates of all three tracking solutions can be compared as they all follow the same normalization to the screen, with the exception of *pose_world_landmarks*. The value of the depth, z, between the three solutions, can not be compared, without some modification, as they are rooted in different points, i.e. the wrists, the centre of the hip and the face.

4.2 Neural Networks

To make this chapter a bit easier to understand, this is an introductory section of general terms and information regarding neural networks that will be expanded upon later.

Neural networks are layers of nodes, one input layer, one output layer, and a varying amount of hidden layers in between. Each node of each layer is connected to nodes from the previous layer with weights [42]. During the training process, these weights are gradually learned through a process of backpropagation and gradient descent. Backpropagation is an iterative process which goes backwards through the network, updating the weights based on the difference between the predicted output and the true value [43]. Gradient descent is an optimization method that indicates how and by how much the weights should be updated [44].

When a neural network has completed one pass of the training dataset, it is called an epoch. The number of epochs is a hyperparameter that determines the total number of passes the neural network should work on [45]. A batch is another hyperparameter, which defines the number of training samples required before updating the weights and gradient. When the batch is set to more than one, but less than the total size of the training dataset, this is called a mini-batch [45]. Small mini-batches during training, ranging between size *m=2* and *m=32*, have been shown to provide improved generalization performance for the neural network [46].

Supervised learning is a category of machine learning defined by the use of a dataset with labeled data, predicting unseen data based on these labels [47]. In terms of this thesis, the dataset consists of videos of sign language where each video is labeled as the corresponding gloss.

4.2.1 Recurrent Neural Network

A recurrent neural network (RNN) is a type of neural network specialized in processing sequential data. First developed by John Hopfield in 1982, RNNs are different from a normal feedforward neural network in that they have a memory, stored information from previous inputs that influence the current input [48]. This allows RNNs to extract temporal dependencies in sequences of data, where the order of the data is important [49].

RNNs use a backpropagation through time algorithm (BPTT) to calculate the gradients of the network [50]. During this backpropagation, a problem known as vanishing gradients can occur, where the gradient becomes smaller and smaller, leading to a stoppage in learning [51]. This vanishing gradient problem can lead to

a short memory of the network, not properly processing long-term dependencies in longer sequences of data.

4.2.2 Gated Recurrent Unit

A Gated Recurrent Unit, (GRU), is a variant of an RNN network created by Cho et al. in 2014 [52]. A GRU addresses the vanishing gradient problem, and thereby the short memory of regular RNN networks mentioned in Chapter 4.2.1.

A GRU has internal mechanisms, a so-called hidden state and two gates, a reset gate and an update gate. These mechanisms allow for the control of the amount, and of which information the network should retain. The two gates, reset and update, parse information through a sigmoid activation, which together controls the flow of information through the network [52].

Another variant of an RNN is the long short-term memory, (LSTM), developed by Hochreiter and Schmidhuber in 1997 [53]. This LSTM has been compared to a GRU by Yang et al., in which it was concluded that GRUs are faster compared to LSTMs in terms of model training speed as well that, for small datasets, GRUs have a higher accuracy [54].

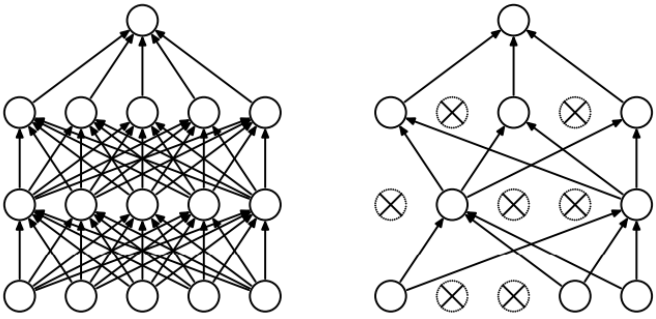
4.2.3 Overfitting and Regularization

Overfitting is a fundamental problem in machine learning and neural networks, where the model starts to learn too well on the noise in the training data. If overfitting has occurred, and the model is unable to generalize, it is possible for it to be unable to perform on unseen, new data. In neural networks, overfitting can occur when there are too many parameters and the network is too complex in relation to the amount of data available, or when the model is trained for too long [55].

To prevent the issue of overfitting, regularization can be applied to the model. The two primary regularization techniques used in this thesis are dropout and data augmentation. Dropout regularization is a technique in which neurons and their connections in the network are randomly dropped during training, creating a much thinner network with a smaller number of weights, seen in Figure 4.3. This thinning of the network reduces the risk of individual neurons of becoming overfitted to training data, improving generalization [56]. The probability p of units dropping is set manually for each layer in the network.

The main idea of data augmentation is to increase the available data in the training dataset, as an expanded dataset can improve prediction accuracy [55]. This data augmentation technique is commonly used in image processing, where operations such as image shifting, mirroring, scaling and rotating can easily be implemented to increase the dataset [57]. Data augmentation can be argued as not being a regularization technique due to it not being used to calibrate a machine learning model. However, in the 2016 book Deep Learning, Goodfellow et al. propose a broader definition, “Regularization is any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training

error" [58]. Data augmentation, through increasing the training data also decreases the model's variance, ultimately decreasing generalization error. As such, data augmentation can be seen as a regularization technique.



(a) Standard Neural Net. (b) After applying dropout.

Figure 4.3: Dropout Regularization. (a): A standard neural network with 2 hidden layers. (b): An example of a thinned net produced by applying dropout to the network on the left. Crossed units have been dropped. Figures and caption taken from [56].

4.3 Chatbots

A chatbot is an intelligent conversational computer system designed to mimic human conversation. A chatbot processes the user input, usually natural language text, and produces an output, most relevant to the user input sentence [21]. There are a few different models of chatbots that will be discussed, such as rule-based in Chapter 4.3.1, deep learning-based in Chapter 4.3.2, and information retrieval-based in Chapter 4.3.3.

4.3.1 Rule Based Chatbots

Older chatbots, like ELIZA from 1966 utilize a rule-based system, pattern matching the user's input to topics and categories, and inferring an answer from this [20]. Rule-based models are usually easier to design and implement, but have some drawbacks in terms of their capabilities, such as having difficulty answering complex queries or inputs that do not fit any known pattern [21].

4.3.2 Deep Learning Based Chatbots

More modern chatbots utilize various machine learning algorithms, allowing them to learn from an existing database of a large number of human conversations. Machine learning allows chatbots to require no need for predefined rules of pattern matching and make them not contained in specific domains of knowledge. This leads these modern chatbots to be more flexible and automatically generate answers in various

situations and complex queries. The category of machine learning chatbots can be further divided into generative models and information retrieval-based models.

Generative models generate responses based on the input from the user, going through it word by word, making it very flexible and requiring no pre-defined response. However, to achieve this, the model needs to be trained on large datasets of natural language sentences to learn sentence structure and syntax. This learning is often based on deep learning, and the training is costly in terms of time and infrastructure, and even after training the outputs can sometimes lack quality or consistency [21].

4.3.3 Information Retrieval Based Chatbots

A middle ground between rule-based models and deep learning models is the information retrieval model. Information retrieval models include a knowledge database consisting of question-answer pairs. A chat index is constructed from this database, containing all possible responses based on the message that prompted them. When presented with an input, the system queries the database and matches the input with the most suitable answer. The main advantage of this model is that it ensures that the responses are of high quality. This is due to the fact that the answers are not automatically generated, and instead manually written, similar to a rule-based chatbot. Despite the benefit of quality compared to generative models, information retrieval models require the time-consuming creation of a knowledge database, manually linking answers to questions [21]. Due to the pre-generated answers, information retrieval models are therefore less suitable for social chatbots, as they lack the ability to develop personality, which is an important aspect of such chatbots [59].

4.3.4 Chat Input Processing

Tokenizing is the process of segmenting a text into smaller and more meaningful inputs such as paragraphs, sentences, words, and letters, called tokens. The most common type of tokenization is that of spaces, the segmentation of text into each of the individual words at each space [60]. The text string *Hello my name is John Smith* is tokenized into the following six word tokens: [*Hello, my, name, is, John, Smith*].

Lemmatization is the process of converting words into the lemma form, the intended meaning of the word. This intended meaning of a word depends on the context of the word in the sentence [61]. The words *break, broke, broken* and *breaks* are forms of the same lexeme, the underlying meaning of the word, and are categorized by *break* as the lemma, the base or dictionary form of the words.

Natural Language Toolkit, NLTK, is a Python platform for language processing and contains tools for tokenization and lemmatization [62].

5

Methods

The method of creating this system can be seen as three separate parts, data gathering, the sign recognition network (SRN), and the chatbot. Data gathering is the collection of data, to be used to train the network for sign recognition, while the SRN and chatbot are connected together to create the system as a whole, used by a user.

5.1 Data Gathering

As discussed in Chapter 1.4, there exists no dataset suitable for the intended use case. Due to this, it was decided early on that the dataset was to be recorded internally. With the help of the SSL dictionary as a sign guide, mentioned in Chapter 1.4.1, the signs were manually performed and recorded, with a total of 10 versions of each sign from each author. The duration of each video was chosen to be 25 frames, which on a regular phone camera is 833 milliseconds recording at 30 frames per second. The number of 25 was chosen due to it being the best option between 20 and 30. A duration of 20 frames was too short with it cutting long signs before completion, and 30 being too long, leading short signs to be stationary for too long after completion.

The words chosen to be recorded for this thesis' dataset need to be able to create complete sentences for later use in the chatbot. To allow for some variety in the chatbot two different conversation topics, one regarding the user being injured and the other being about food. These two topics were chosen due to being quite common and useful topics. From these two topics, 15 words to build sentences out of were chosen. The chosen words were: *JAG, MIG, HAR, SKADA, HUVUD, ÄR, JA, NEJ, HUNGRIG, TACK, HEJ, ARM, PIZZA, HAMBURGARE* and *VILL* (*I, Me, Have, Hurt, Head, Is, Yes, No, Hungry, Thank you, Hello, Arm, Pizza, Hamburger* and *Want*). Each recording was started in the resting position, with the hands clasped over the stomach, and ended with the hands static in the end position of the sign.

Two additional signs were selected, the first called *EOS*, end of sign. This EOS sign is used to send the sentence to the chatbot and is a made-up sign, not corresponding to any real-world sign. The movement for this sign was chosen to be the one with the least in common with the other 15 signs to make it easier for the system to guess correctly. The other additional sign was called *BLANK*. This blank sign is an amalgamation of many different signs that the system shall not guess, those being still videos of the end positions of the other signs and the movements from the end

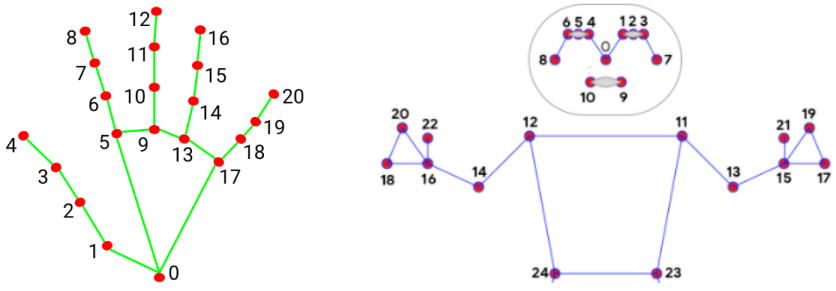
positions back to the resting position. These blank signs were recorded to lower the probability that the network would guess a sign on still hand positions, and stop guessing when the sign is completed. Two still videos of each previous sign was recorded for this blank sign, as well as a neutral position of the person being in the resting position.

These videos were labeled after recording, to be later used as a supervised dataset for the neural network to learn from.

5.1.1 Keypoint Extraction

As mentioned in Chapter 4.1, the total number of keypoints that MediaPipe can track is 543, including all dimensions per keypoint it becomes 1662 parameters per frame, a number too large and not required for the purpose of the thesis. Thereby, a decision was made regarding which keypoints and dimensions to cut. Of the 33 possible keypoints for the pose estimation, keypoints 0 to 24 were chosen, as can be seen in Figure 5.1 (b). As the intended use case is sitting in front of a web camera, the legs would not be visible, and thereby the lower 8 keypoints are not needed. Of the four dimensions of each keypoint of the pose, the visibility parameter was eliminated as only the coordinates were necessary. Thereby, the total number of parameters for the pose estimation ended up being 75. For the hands, no changes were made, ending up in three dimensions for 21 keypoints, 63 parameters each, seen in Figure 5.1 (a).

All of the face keypoints were discarded, as although the face is a major part of the nonmanual part of sign language, facial features are more beneficial in determining the grammatics and intention of a sign rather than the sign itself, as mentioned in Chapter 1.2, eliminating 1389 parameters. However, as seen in Figure 5.1 (b), the pose solution already tracks eleven points for the face, meaning that the network gets some information about head placement, rotation, and angle. In the end, of the total of 1662 parameters that MediaPipe can track, only 201 parameters were used.



(a) Hand keypoints (b) Pose keypoints

Figure 5.1: The chosen keypoints to track. The same keypoints for the other hand.

To these 201, 13 new ones were added, 4 for the pose, 4 for each of the hands, and one for the relative difference in height between the two hands, seen in Table 5.1. From the data obtained from the keypoints indicated in the Table, new distances and angles could be calculated which then determined the values of the new parameters.

These new calculations used both the x and y coordinates of the keypoints, unless stated otherwise. Why these new keypoints were added is discussed in Chapter 7.1.

Hand Parameters		
Type	**Keypoints**	**Value**
Closed Hand (x2)	(0, 9, 12)	-10, 10
Palm towards Camera (x2)	(5, 9)	-10, 10
Horizontal Direction (x2)	(0, 9)	-10, 10
Vertical Direction (x2)	(0, 9)	-10, 10
Height over other hand (x1)	(9.y)	-10, 0, 10

(a) Added parameters through hand keypoints, (x2) for both hands. Both x and y coordinates are used unless otherwise specified.

Pose Parameters		
Type	**Keypoints**	**Value**
Right Armpit Angle	(12, 14, 24)	(0- 180)
Left Armpit Angle	(11, 13, 23)	(0- 180)
Right Elbow Angle	(12, 14, 16)	(0- 180)
Left Elbow Angle	(11, 13, 15)	(0- 180)

(b) Added parameters through pose keypoints. Both x and y coordinates are used unless otherwise specified.

Table 5.1: Added parameters

As mentioned in Chapter 4.1, the pose tracking could be done through two implementations, *pose_landmarks* and *pose_world_landmarks*. The decision was made to utilize the former, as this implementation uses the same scale for the x and y coordinates as the hand tracking, making it easy to correlate and modify the data between the two.

The keypoint data extracted by MediaPipe contains coordinates to an eight-point decimal place. The data was thereby rounded to a five-point decimal, allowing the network the learn the general place of keypoints instead of being constrained to too specific coordinates.

5.1.2 Coordinate Simplification

As stated in Chapter 4.1, the values of the x and y coordinates of the pose estimation implementation *pose_landmarks* and hand estimation were normalized to the width and size of the screen. By taking the values of the right shoulder, keypoint 12 seen in Figure 4.1, and normalizing all other keypoints to these values, a new origin, (0, 0), was created. To this origin, all other x and y coordinates were relative. A positive value of x meant to the right of the shoulder, and a positive value of y meant above the shoulder. This created four distinct “zones”, dividing up signs into left/right and above/below the shoulder, allowing the network to more easily read where the sign had been created. An additional benefit of normalizing every keypoint to the right

shoulder is that it no longer matters where the user is placed in the video, due to the centre point now following the user.

5.1.3 Data Augmentation

It is infeasible to obtain a sufficiently large enough dataset of sign language videos for training, validation and testing of the sign language recognition network through recording of people. Thereby, a small dataset has to be made larger through data augmentation. For this data augmentation, the dataset was increased in two steps. The first pass was to scale all of the keypoints by a random factor relative to the centre point of the chest, simulating different sizes of people but of the same proportions but also different distances away from the camera. This first pass increases the size of the dataset by a factor of 5. The next pass was to on these new and original videos, scale only the hands and arms by three random factors, one in either axis, relative to the centre point of the chest. This data augmentation allowed for the creation of new joint angles and proportions of the person doing the signs. This second pass increases the size of the dataset by a factor of 10.

5.2 The Neural Network

As discussed in Chapter 4.2.2, the RNN variant GRUs are well suited for training on small datasets and were thereby chosen as the first three layers of the neural network, seen in Table 5.2.

Dropout regularization was utilized with probability p of 0.5 for the first five layers of the model. The dropout regularization can however only be seen in the model summary after the dense layers, as they are hidden within the GRU layers.

The mini-batch size was decided on being set to $m = 32$.

The dataset was split into a train-test split with an 80/20 distribution. This determines the percentage of data being split up, with the training dataset containing 80% of the data and the test dataset containing the other 20%. No validation set was used.

Layer (type)	Output Shape	Param #
GRU	(None, 25, 64)	52800
GRU	(None, 25, 128)	74496
GRU	(None, 64)	36248
Dense	(None, 64)	4160
Dropout	(None, 64)	0
Dense	(None, 32)	2080
Dropout	(None, 32)	0
Dense	(None, 16)	528
Total params:	171,312	
Trainable params:	171,312	
Non-trainable params:	0	

Table 5.2: Sign Recognition Network Model Summary.

5.3 The Chatbot

As mentioned in Chapter 4.3, there are three different types of chatbot that can be developed, but the decision was made to create an information retrieval-based chatbot mentioned in Chapter 4.3.3. This decision was made due to the increased flexibility compared to rule-based chatbots, while still being suitable for the smaller domain that the thesis covers. Due to information retrieval-based chatbots having a smaller amount of pre-generated answers, it is possible to have pre-recorded sign language videos that the chatbot plays as these answers. A generative chatbot that could answer in many different ways would require the recording and storage of equally as many videos as answers.

The information retrieval-based chatbot is developed around a structure of conversational intents, created in a JSON file, on which a simple neural network is trained. The intents follow the structure seen in Figure 5.3, with four categories for tag, pattern, response, and context, with an example in regards to the frame of the user being hurt. The tag is the name of the overall intent, the patterns are what inputs a user could say, and the responses are randomly chosen in response to the user if the user's input is matched to the aforementioned patterns. The context is later used to determine the direction of the conversation. An example of such contextual direction is that, in case the user answers yes to a question, it is in the context of the user being hurt and not whether the user is hungry.

Intent	
Tag	Hurt
Pattern	"I have hurt myself", "I am hurt", "I am wounded"
Responses	"Where are you hurt?", "Is it serious?"
Context	Conversation Hurt

Table 5.3: Structure of the conversational intents, saved as a JSON file.

This JSON file is later processed through both tokenization and lemmatization mentioned in Chapter 4.3.4. This lemmatization also removes duplicate and synonym words from the list, as the important part is the intended meaning and not all of the different grammatic inflexions. To help with this processing, the Python library NLTK was used. A simple RNN is then trained on this processed list to learn what intent in the JSON file a user's input belongs to.

5.4 The System

Each sign has to start from the resting position, and after the user has made a sign, the end position of that sign has to be held in the same static position for 10 frames. Although it is not very true to real-world signing, it was a decision made to indicate to the system that a sign has been completed, not requiring it to predict a sign on every frame, making it less "eager" to guess, but also reducing computation time to not always be required to guess. The system predicts signs on a sliding scale of 30 frames, the first 25 (oldest) being for the actual sign and the last 10 (newest) being to detect this static position of a sign being complete. This means that 5 of the 10 static frames are for the prediction of the sign. If a sign has been completed, but the system is unable to decide which sign was made over a certainty of 90%, then nothing is shown.

After the user has completed a sentence and indicated to the system that it is complete by signing the *EOS* sign, the sentence is tokenized and lemmatized before being sent to the chatbot. The chatbot then returns with a random answer from the most fitting category, discussed in Chapter 5.3. Due to the system being able to understand the context, it will correctly respond to the user if the input is yes if the context is *Conversation Hurt*, and it will not assume that the user is hungry as would be the case if the context were *Conversation Hungry*.

The response from the chatbot is parsed into each of the individual words, shown to the user as a series of sequential, but individual videos, of each sign. If the system is unable to provide a fitting answer over an 80% threshold, then the system will respond with the message *FÖRSÖK IGEN* (*try again*).

6

Results

This section starts with an overview of the system and its interface and a short step-by-step summary of how the system works. Then there will be two evaluations of the developed system, one for the theoretical performance and the other for the real-world performance. The theoretical performance is evaluated with data provided from the test dataset, while the real-world performance is evaluated through the experience of four people using the system, two being authors of this thesis and the other two being people which the system has not seen.

6.1 The System and its Interface

When the system is started, a live video feed is shown to the user, seen in Figure 6.1.

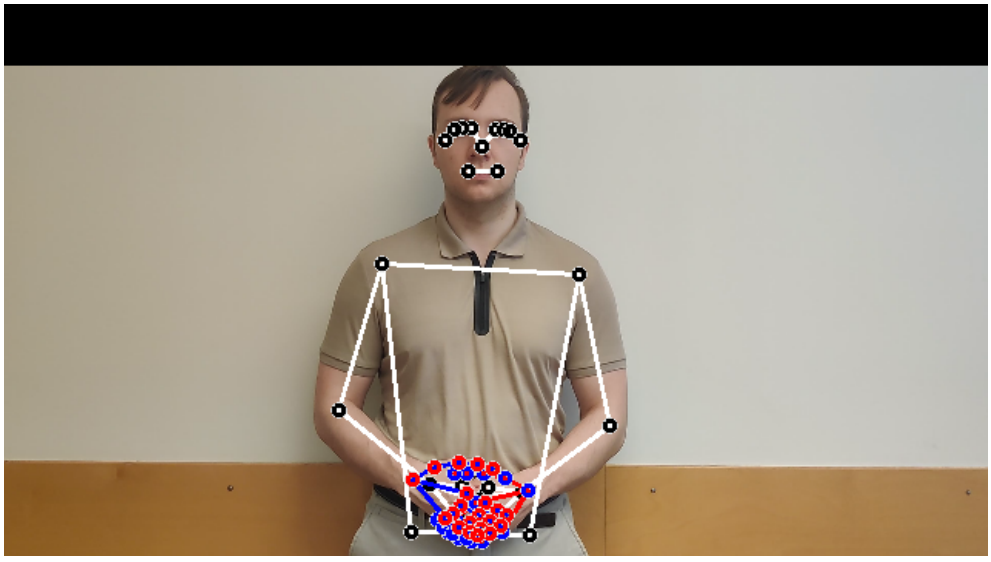


Figure 6.1: Live video feed shown to the user.

After a sign has been predicted, the last five signs the user has made will be shown in the top black bar. In Figure 6.2, the user has completed the signs *JAG*, *HAR*, *SKADA*, *MIG*, (*I*, *have*, *hurt*, *myself*), in that order.

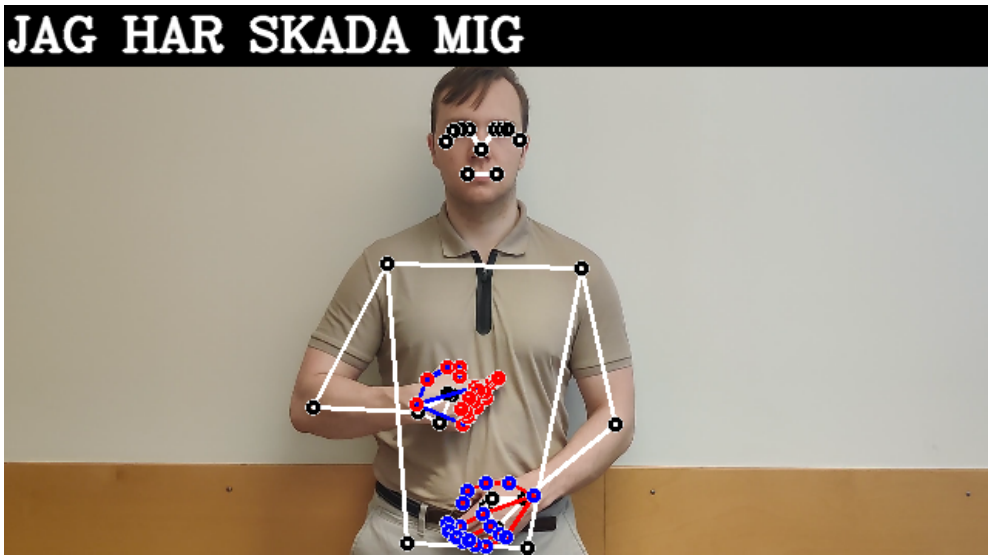


Figure 6.2: The system showing predicted signs.

Pressing *q* on the keyboard stops the program while *s* removes the last word from the sentence in the black bar. Hiding both hands, done by either placing them off screen or behind the back, removes all words in the black bar.

After the user has sent the sentence with the *EOS* sign, the chatbot responds with a series of pre-recorded videos as discussed in Chapter 5.4. In Figure 6.3, the chatbot has responded to the previous input with the response *VAR HA DU SKADA DIG?* (*Where have you hurt yourself?*). The last sign is shown on top, with the thus far completed signs building a sentence below.

The response is not a grammatically correct Swedish sentence as the response is a construction of different base signs with no nonmanual markers. As such, each word is written in its gloss label.

Figure 6.3: The system responding with a series of pre-recorded videos.

As the system does not give any feedback, the new users do not know if the sign was performed incorrectly or if the recognition software is not working unless directed by a proficient sign language human observer.

6.2 Theoretical Performance Evaluation

There are a few metrics available to evaluate the SRN. After training for 100 epochs, a total accuracy of 99% was achieved. As seen in the confusion matrix in Figure 6.4, the system achieve an overall good accuracy with only signs *HUVUD, NEJ, PIZZA* and *BLANK* (*head, no, pizza* and *blank*) not achieving 100% accuracy. The categorical accuracy and the epoch loss from training can be seen in Figure 6.5.

However, these results do not guarantee that the system will work well for real users, and thus the real-world evaluation will play a more important role.

The result of this theoretical performance evaluation will be discussed in Chapter 7.1.1.

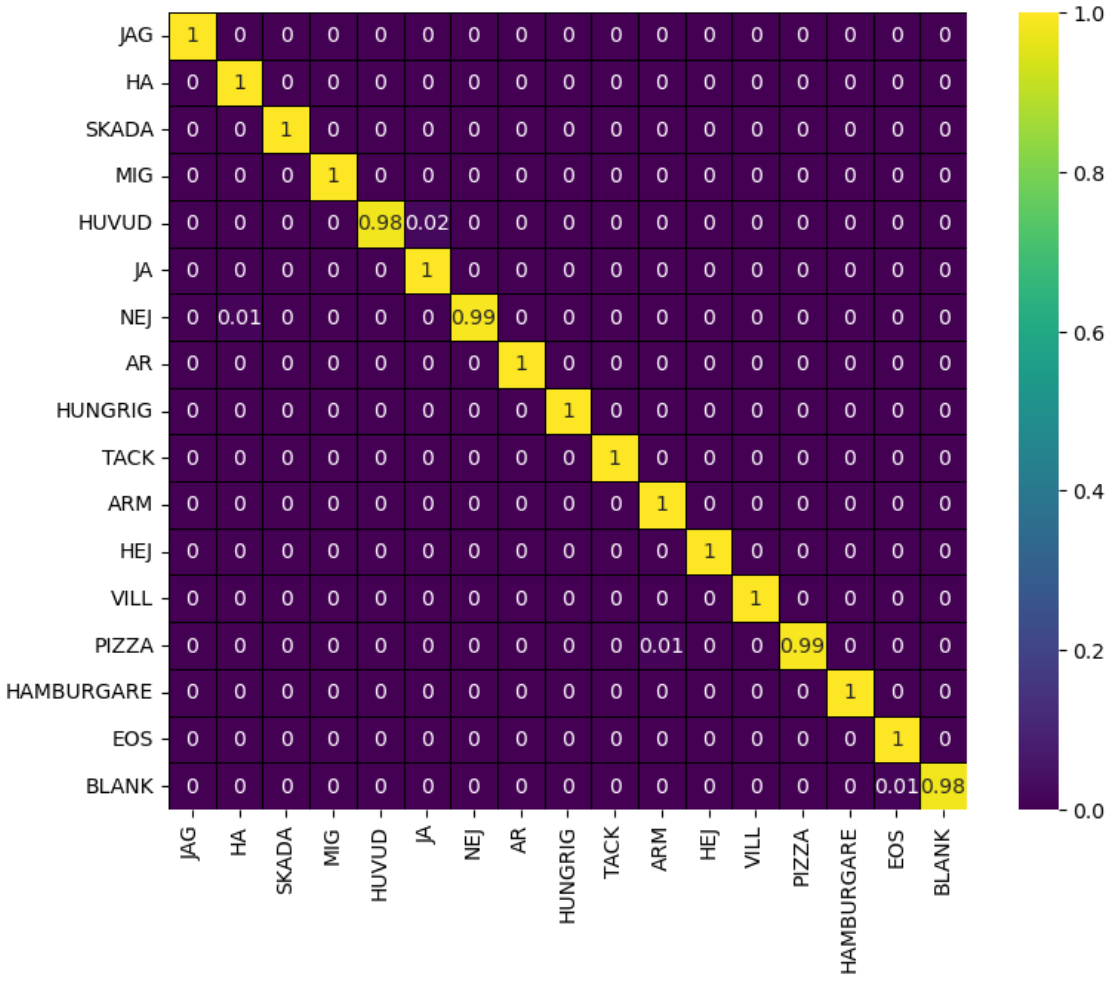
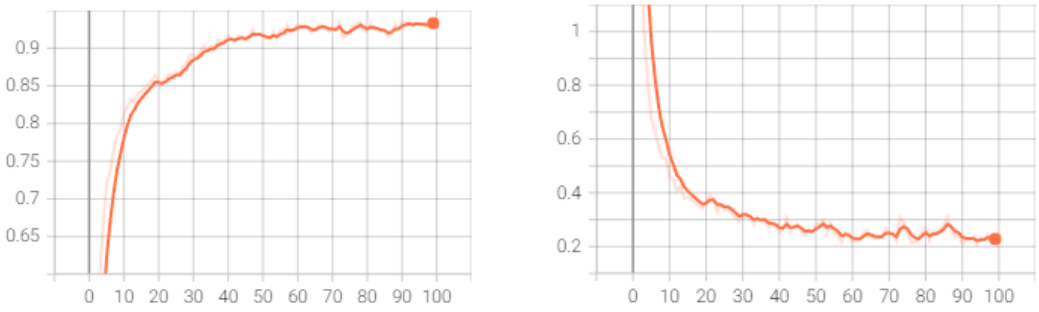


Figure 6.4: Confusion matrix. Left: True Label, Bottom: Predicted Label.



(a) Categorical Accuracy

(b) Epoch Loss

Figure 6.5: Graph showing categorical accuracy and epoch loss of the SRN training. Smoothing set to 0,6.

6.3 Real-World Performance Evaluation

The real-world performance of the system is high, it can predict signs by the authors very well due to the training data being recordings of the authors. The prediction performance on users the system has not been trained on is less high with a few more errors, although sufficiently high for a proof of concept. Signs that are dissimilar to the others, i.e. the *EOS* sign mentioned in Chapter 5.1, are almost always predicted correctly. Other signs that are more similar, such as *JAG* (*I*) and *MIG* (*me*), are more troublesome, as the sign *MIG* is an extension of the sign *JAG*. Another such troublesome extension is the *HEJ* (*hello*) and *HUVUD* (*head*), with how *HEJ* starts very similarly to how the sign *HUVUD* is made. This can be seen in Table 6.1, with how *HEJ* and *MIG* have worse performance.

Sign	**Author**	**Test User**
JAG (I)	10/10	10/10
HA (Have)	10/10	8/10
SKADA (Hurt)	10/10	10/10
HUNGRIG (Hungry)	10/10	10/10
HEJ (Hello)	9/10	8/10
MIG (Me)	9/10	10/10

Table 6.1: Real-World test of six signs. Number of correct guesses / Total number of guesses. 2 authors and 2 test users, 5 tries per person.

The system also more frequently made a “no guess” on these test users, indicating that no sign achieved a probability higher than 90%. However, this can probably be explained through user errors, as the test users were unfamiliar with the requirement of staying still for 10 frames after a completed sign and often reverted back to the standard position immediately as well as not being familiar with the movements. As such, any time the system made no guess, it was not counted as an error, and the sign was simply tested again.

Due to the limited evaluation performed, no real metrics can be shown in the real-world performance evaluation of the system, however, for the purposes of the thesis with a limitation to create more of a proof-of-concept product, the performance was sufficiently high.

The result of this real-world performance evaluation will be discussed in Chapter 7.1.2.

7

Conclusion

This chapter will contain a general discussion about the thesis, why some decisions were made, and different challenges that emerged, and how they were solved or unsolved. The chapter will end with a final conclusion and a section on any future work that would improve the product if more time were available.

7.1 Discussion

There are a few discussion points that need to be brought up. The first is why it was decided to create a network from the ground up, instead of utilizing pre-existing networks and then using transfer learning, training that network on the sign language data. It was decided very early on to initially create a bespoke network as a proof-of-concept. When this network then surpassed expectations, it was decided to continue and improve on this bespoke network rather than using any publicly available ones.

Another question that needs to be answered is why a supervised learning method was utilized instead of a semi-supervised method. As the data was recorded by ourselves, some sort of labelling had to be done, so it was decided to label all data anyway. A semi-supervised learning approach might be better when the amount of signs and data is much larger than that of this thesis.

In Chapter 5.1.1 it was discussed how extra parameters were added. Although little evaluation was performed on these individually, the system performed much better with these as a whole, albeit some might be more important than others. It is worth noting that some of these parameters might be seen as superfluous, as their data can be seen in the keypoint extraction data itself. An example of that is the extra parameter *Height over other hand* which says which hand is over the other, or being 0 if they are of the same height. This parameter was added to enforce the importance of the height difference and to help the network understand the importance of it rather than inferring it from the many other relevant keypoints.

Two parameters that were tested but not used were that of the angle of the hand using the x and y-axis. This parameter is similar to that of *Horizontal Direction* and *Vertical Direction*, but rather than a simple "up/down" value, it was a 360-degree rotation. This parameter for each hand increased the performance of guessing signs for some signs but drastically lowered it for others, so it was never implemented.

In Chapter 5.1.1, it was mentioned how the precision of coordinates was rounded to

a five-point decimal, allowing the network to learn the general place of keypoints. In hindsight, this might not allow for the intended behaviour, and instead, it could have been better to keep the precision high but introduce noise instead.

Another topic that is important to convey is why an information retrieval-based chatbot was implemented rather than a generative deep learning-based one. The decision of this lies very much in the scale of the thesis and the available data. As it was decided to keep the scope smaller, with only 15 signs to build sentences from, the variety of the input to the chatbot is quite small. Adding to the fact that the importance of the thesis is in the sign recognition part of the system and not the chatbot, the chatbot could be simplified and more suitable for the small amount of input. As the chatbot responds in sentences built up of individual videos of sign language signs, a deep learning chatbot would require a much larger database of sign videos to generate all of the outputs, whereas, with an information retrieval chatbot, the output could be kept small, limiting the number of videos to record and store. Therefore, it seemed unnecessary to create a deep learning-based chatbot. However, a thesis with a larger scope, consisting of more signs, could be better served by a generative deep learning-based chatbot. See Chapter 7.5 for more discussions about possible future work.

7.1.1 Discussion: Theoretical Results

As stated in Chapter 6.2, the system achieved an accuracy total of 99% on the test dataset, with only a few signs not achieving full accuracy. This is comparably very high to the other works mentioned in related work in Chapter 2.1 with the accuracy of 92, 93 and 96%. Only the other MediaPipe sign language translation paper mentioned achieved a similarly high accuracy rate, indicating a few possibilities. It can be either the immense power of the tracking ability of MediaPipe or that the paper and this thesis had a similar problem, that being a lack of a large and varied set of data to both train and test on.

Another possibility of high accuracy could be due to the augmentation occurring before the train/test split. This leads the testing dataset to be more likely injected with small variations of the training dataset. Due to this, there is a risk of introducing unwanted correlations between the training and test dataset, making the testing overestimate the accuracy, and giving the test accuracy score the illusion of a higher score than what it might actually be. The reason why the augmentation occurred before the train/test split is that we did not know this was an issue until after the project was complete.

Early stopping is a regularization technique that can be applied, one that was considered in the thesis. However, for this technique to be properly applied, a validation set needs to exist, which in turn requires more available data. Thus, the problem loops back to the lack of available data which has been a large problem in the thesis. Despite this, through testing of how long the model usually trained, a semi-early stop was conducted to try to have some of the beneficial regularization effects, however, this is not able to be properly measured.

7.1.2 Discussion: Real-World Results

As stated in the real-world evaluation in Chapter 6.3, the very high theoretical accuracy score did not transfer as well to the real-world. Other than the inflated score mentioned in Chapter 7.1.1, the current theory of the lower performance on the testing users is due to three factors. The first factor is the lack of diverse training data, which can lead to low generalization of the system for unseen data. Adding to this, due to all bodies being different, the signs are inherently performed differently with different joint angles and placement of the hands. Although this was introduced through augmentation, it was probably not enough, making the network worse in predicting signs of other people.

The second factor for the worse real-world performance, at least on the test users, is that they were novice signers. This means that the test users had difficulty performing the signs properly with all of the intricacies. The performance of the system increased after the test users received feedback on how to perform the signs correctly, although still not as high as with the testing on the authors. It is important to note that the guidance from the authors on how to perform the signs might be seen as introducing bias in the results.

Overfitting is the third factor that is suspected to cause the lower generalization to unseen data. The issue is not foreign and was somewhat expected to be a problem due to the small amount of data available, and was an issue for which mitigation techniques were considered from the very beginning. As mentioned in Chapter 4.2.3, both dropout and data augmentation and regularization techniques were used to address this. The dropout could not be increased further without affecting the testing accuracy and is thereby not an available option for further work. Augmentation however is one such option that can be taken further. As mentioned in Chapter 5.1.3, two passes were made, but this could be increased to more passes and more parameters, going further with the augmentation of joint angles and sizes of different body parts as well as camera depth. However, due to augmenting on data arrays rather than video, it is difficult to see how the augmentation affected the data, as well as not wanting the augmentation to affect the signs to become non-recognizable, a conservative approach was used for the augmentation. As discussed in Chapter 7.1.1, early stopping was unfortunately not able to be done properly so this technique could not help in reducing any overfit. It is also important to note that due to the errors regarding the train/test split, also mentioned in 7.1.1, this could introduce overfitting into the system. If more time was available, some experimentation could be done to eliminate unnecessary parameters, lowering the total amount. This would lower the complexity of the data, which, as discussed in Chapter 4.2.3, could increase generalization. See Chapter 7.5 for more discussions about possible future work.

With the limited tests performed and the lack of time for a proper evaluation study with controlled experiments of both novice and advanced signers, it is unknown what is the major factor of the lower performance.

7.2 Ethical Evaluation

It is a fact that the data collection part of the thesis consisted of recording the faces and bodies of people, and that keypoint information was extracted from these videos. However, as this was recorded live, the only data stored were those of the keypoint coordinates of each frame of a recording, in our case 25 frames. Thereby, no video is stored, and it is impossible to see any identifiable RGB value from this data. As stated in Chapter 5.1.1, the 468 keypoints of the face estimation were removed, keeping only the 11 keypoints from the pose estimation that exist on the face. Thereby, the amount of facial data stored is very small, making it difficult to identify specific people. Adding to this, no labels were created to differentiate between people, so the network can not know which signer is which.

Another ethical aspect is that of learning, as a teacher or, in this case, a computer-based learning tool, where the feedback should be trusted. It is thereby important not to give incorrect feedback and put bad habits in the early stages of learning. This has not been addressed much in the thesis, and the problem is perhaps mitigated due to the inability of the system to give any feedback. However, if the training data was incorrect, or as stated in Chapter 5.1, slightly changed to work with MediaPipe, this can enable users to learn incorrect signs as they try to achieve higher accuracy. This is not seen as a problem currently, as the thesis is more of a proof of concept; however, any future work would have to address this and more, discussed in Chapter 7.5.

7.3 Challenges

Through the work, several challenges have emerged, some of which were solved and others not.

The largest initial challenge was that of the lack of an SSL dataset large enough and suitable for the intended use case of the sign recognition chatbot, mentioned in Chapter 1.4. This required the two authors to record all of the videos themselves, a time-consuming endeavour, especially as the process of recording the keypoints was changed through development, and sometimes re-recording was thereby necessary. The recorded videos were then augmented as mentioned in Chapter 5.1.3, to vastly increase the amount of data available in the dataset, and to introduce variance to inhibit overfitting.

A challenge that was not solved completely is that of how MediaPipe affects a smooth live recording, as a new frame is only shown once all calculations are complete. Adding to the problem, all the extra parameters and other calculations that were added increased the processing time for each frame, before the SRN needs time to guess the sign. All this combined means that the live video recording is not smooth 30 frames per second (fps). Some optimizations were made to address this; however, more can most likely be done to make the video smoother, i.e. run at a higher fps. This issue of a low fps number was extra noticeable when the system was run on a less powerful computer. Due to this, when on a less powerful computer, the 10 frames of

still motion to indicate that a sign has been completed, mentioned in Chapter 5.4, takes an extra long to complete. This takes it from a short stop to a very unnatural stillness.

An unsolved challenge is that of the MediaPipe keypoint tracking software. The software worked as intended for most signs, however, in some edge cases, when the side of a hand was pointed directly at the camera, thus making the hand very thin, the tracking could completely stop. In other cases, the software could not track fingers that were obscured, as would happen if a fist is closed with the palm away from the camera. Other times, if a sign required very fast movements, the tracking of the hands could stop working for a frame or two. For this problem of fast movements, it is unknown if it is due to a problem within MediaPipe or if it needs a camera recording with less motion blur. If it is the latter, it is unlikely to be solved, as the intended use case is with a web camera, not known for its quality.

As the above problems are rooted in the MediaPipe tracking software, they were either not possible to fix, or required workarounds, such as removing or slightly altering problematic signs to ensure that the software can track the hands properly.

Another unsolved challenge with MediaPipe is that of the z-axis distance for the pose estimation mentioned in Chapter 4.1. Although the estimation of the z-axis came with a disclaimer of not being entirely correct, an increase in performance could be seen with it in the data, as although it was not very precise, it gave an extra dimension to the data, which was overall helpful in determining signs. However, due to the lower precision of the dimension, it was decided not to create any new parameters using the z-axis, similarly to what was done in Chapter 5.1.1.

A method that could help with this z-axis estimation is the use of a second camera, viewing the signer from the side. This new camera, also using MediaPipe keypoint estimation, would give a more accurate depth estimation, due to it being this new camera's x-axis. While adding this second camera might increase accuracy, and would perhaps allow for new parameters to be implemented and thus increase performance, this would also be a step away from the intended use case. This is due to the assumption of most home offices not having two web cameras, as such it would thereby not be a reasonable requirement.

7.4 Conclusion

This study presents the development of a sign recognition network trained on MediaPipe keypoint estimation using a supervised machine learning training method. This SRN has been paired with an information retrieval-based chatbot to enable sign language conversations between a user and a computer. The integration of the chatbot adds a unique feature, transforming the system from a sign language translator to a sign language training tool.

The sign recognition network achieved 99% accuracy in recognizing signs of a test dataset, demonstrating the effectiveness of the chosen neural network architecture and training methodology paired with MediaPipe keypoint estimation. Real-world

evaluation, however, showed less optimal results, although still sufficiently high, due to both lower generalization of unseen data and novice signers as test subjects. No proper evaluation was conducted to see which of the two reasons contributed the most to the lower real-world performance.

The chatbot, although limited in vocabulary, is capable of conducting short and various conversation within two specific domains and responding to the user with videos of sign language. The chatbot is, however, unable to provide feedback to the user about how the sign was performed.

Despite the challenges encountered during the real-world evaluation and a lack of feedback from the chatbot, the product of a sign recognition network paired with a chatbot still represents an advancement in the field of sign language recognition as a training tool. This system is a stepping stone to a product facilitating easier sign language training to hearing-impaired and non-hearing impaired people alike, bridging the communication gap between the two.

7.5 Future Work

In Chapter 5.4, it is mentioned how 10 frames of a static hand are required for the detection of a sign. As mentioned, it was made like this to make the system less eager to guess, not constantly displaying guesses before the sign is complete. This solution works great for the intended use case of having each sign be complete in motion and back to a neutral position afterwards. However, as mentioned, it is not very true to how a signer signs in the real world as mentioned in 1.2. To solve this, a solution could be to allow the network to guess constantly, but apply a second neural network to this output, only displaying the guess that is most fitting, however, a better solution might exist.

After starting the project, Google has updated the MediaPipe website, removing and placing the Holistic Solution used in the thesis, under *MediaPipe Legacy Solutions* of a GitHub repository. This shows how fast the area of research is moving, with constantly new and improved ways to track the user being released. These new solutions could mean that many of the problems discussed in Chapter 7.3, those related to the MediaPipe holistic solution, might be addressed. However, this is at the moment of writing unknown. If the issues are addressed, future work could also include the development of more parameters, similar to what was mentioned in Chapter 5.1.1. These new parameters could include the z-axis, allowing for depth angles and more, which could improve accuracy.

Any future work would include having many more signers, diversifying the dataset and not relying on augmentation as much. This more diverse dataset could improve the generalization of the SRN to unseen data. Any new dataset recording could also include more signs, including nonmanual signals. As mentioned in Chapter 7.1, the capability of the chatbot was limited by the size of the dataset. As the dataset grows, allowing more words and grammar to be expressed by the user, the many possible ways the chatbot can respond also grow. At some point, a generative deep learning-based chatbot would be a better fit for the project, however, the issue of

storing the videos of all the words of all the many responses is still an issue, especially if the chatbot allows nonmanual grammar. Perhaps having a generated 3D avatar performing the signs responding to the user could be a solution, removing the need to store videos of all different variations.

Any future work would also include a proper evaluation of the system. As mentioned in Chapter 6.3, only a limited evaluation was performed with no real metrics to be shown. A future evaluation would include a controlled experiment of both expert and non-expert signers, including a diverse set of people to see how the system performs on many different types of unseen data.

An issue that became apparent early on in the development of the system is that of each sign being recorded through 25 frames, as mentioned in Chapter 5.1. Although this did not have any major impact on the results of this thesis, it still required short signs to be either still or slow and long signs to be quick. Future work would have to explore the option of being able to train on a variable length of signs.

Another direction for future work could be to build a more educational chatbot, making it able to give feedback to the user about hand placements, movements, and angles, as well as grammar. Such capabilities would allow the chatbot to say "I think you are trying to sign X, but you place your right hand too high in relation to your left. The sign might be confused with Y". This capability could move the chatbot from a novel system to a more educational system, hopefully avoiding the "novelty effect" mentioned in Chapter 2.2.1.

Bibliography

[1] C. Oz and M. C. Leu, "Recognition of finger spelling of american sign language with artificial neural network using position/orientation sensors and data glove," J. Wang, X.-F. Liao and Z. Yi, Eds., pp. 157–164, 2005.

[2] C. Dong, M. C. Leu and Z. Yin, "American sign language alphabet recognition using microsoft kinect," pp. 44–52, 2015. DOI: 10.1109/CVPRW.2015.7301347.

[3] S.-K. Ko, C. J. Kim, H. Jung and C. Cho, "Neural sign language translation based on human keypoint estimation," *Applied Sciences*, vol. 9, no. 13, 2019, ISSN: 2076-3417. DOI: 10.3390/app9132683. [Online]. Available: %5Curl%7Bhttps://www.mdpi.com/2076-3417/9/13/2683%7D.

[4] T. Reagan, "Sign language and linguistic universals, wendy sandler and diane lillo-martin," *Studies in Second Language Acquisition*, vol. 29, no. 4, pp. 624–625, 2007. DOI: 10.1017/S0272263107070507.

[5] J. Fenlon, T. Denmark, R. Campbell and B. Woll, "Seeing sentence boundaries," *Sign Language & Linguistics*, vol. 10, no. 2, pp. 177–200, 2007.

[6] R. Pfau, J. Quer *et al.*, *Nonmanuals: Their grammatical and prosodic roles*. na, 2010.

[7] P. Siple, "Visual constraints for sign language communication," *Sign Language Studies*, no. 19, pp. 95–110, 1978.

[8] M. V. Swisher, K. Christie and S. L. Miller, "The reception of signs in peripheral vision by deaf persons," *Sign Language Studies*, vol. 63, no. 1, pp. 99–125, 1989.

[9] *Svenskt teckenspråk för hörande 1*, https://www.studentum.se/skola/hermodsvux/svenskt-teckensprak-for-horande-1-442168. (visited on 29th Nov. 2022).

[10] *Teckenspråksutbildning för vissa föräldrar*, https://www.spsm.se/kurser--aktiviteter/teckenspraksutbildning/. (visited on 29th Nov. 2022).

[11] *Svenskt teckenspråkslexicon (2023)*, https://teckenspraksl exikon.su.se/. (visited on 22nd Feb. 2023).

[12] *Svenskt teckenspråkslexikon, 2023*, https://teckenspraksl exikon.su.se/ord/00032. (visited on 28th Apr. 2023).

[13] *Svenskt teckenspråkskorpus*, https://teckensprakskorpus.su.se/s/#/. (visited on 22nd Feb. 2023).

[14] J. Mesch, *Svensk teckenspråkskorpus-dess tillkomst och uppbyggnad*, 2015.

[15] *Svenskt teckenspråkskorpus, 2023*, https://teckensprakskorpus.su.se/s/#/video/sslc01_160.eaf?rowId=19&q=HEJ&t=5.219. (visited on 28th Apr. 2023).

[16] N. C. Camgöz, O. Koller, S. Hadfield and R. Bowden, "Sign language transformers: Joint end-to-end sign language recognition and translation," *CoRR*,

vol. abs/2003.13830, 2020. arXiv: 2003.13830. [Online]. Available: https://arxiv.org/abs/2003.13830.

[17] J. Forster, C. Schmidt, T. Hoyoux *et al.*, "Rwth-phoenix-weather: A large vocabulary sign language recognition and translation corpus," in *International Conference on Language Resources and Evaluation*, 2012.

[18] F. Zhang, V. Bazarevsky, A. Vakunov *et al.*, "Mediapipe hands: On-device real-time hand tracking," *CoRR*, vol. abs/2006.10214, 2020. arXiv: 2006.10214. [Online]. Available: %5Curl%7Bhttps://arxiv.org/abs/2006.10214%7D.

[19] A. Halder and A. Tayade, "Real-time vernacular sign language recognition using mediapipe and machine learning," *Journal homepage: www. ijrpr. com ISSN*, vol. 2582, p. 7421, 2021.

[20] J. Weizenbaum, "Eliza—a computer program for the study of natural language communication between man and machine," *Commun. ACM*, vol. 9, no. 1, pp. 36–45, Jan. 1966, ISSN: 0001-0782. DOI: 10.1145/365153.365168. [Online]. Available: https://doi.org/10.1145/365153.365168.

[21] G. Caldarini, S. Jaf and K. McGarry, "A literature survey of recent advances in chatbots," *Information*, vol. 13, no. 1, 2022, ISSN: 2078-2489. DOI: 10.3390/info13010041. [Online]. Available: https://www.mdpi.com/2078-2489/13/1/41.

[22] D. G. Bobrow, R. M. Kaplan, M. Kay, D. A. Norman, H. Thompson and T. Winograd, "Gus, a frame-driven dialog system," *Artificial Intelligence*, vol. 8, no. 2, pp. 155–173, 1977, ISSN: 0004-3702. DOI: https://doi.org/10.1016/0004-3702(77)90018-2. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0004370277900182.

[23] B. A. Shawar and E. Atwell, "Chatbots: Are they really useful?" *Journal for Language Technology and Computational Linguistics*, vol. 22, no. 1, pp. 29–49, 2007.

[24] *Apple siri*, https://www.apple.com/siri/. (visited on 8th Feb. 2023).

[25] *Microsoft cortana*, https://www.microsoft.com/en-us/cortana. (visited on 8th Feb. 2023).

[26] *Amazon alexa*, https://alexa.amazon.com/. (visited on 8th Feb. 2023).

[27] *Google assistant*, https://assistant.google.com/. (visited on 8th Feb. 2023).

[28] A. Vaswani, N. Shazeer, N. Parmar *et al.*, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[29] R. Dale, "Gpt-3: What's it good for?" *Natural Language Engineering*, vol. 27, no. 1, pp. 113–118, 2021.

[30] L. Fryer and R. Carpenter, "Bots as language learning tools," 2006.

[31] Y. F. Wang and S. Petrina, "Using learning analytics to understand the design of an intelligent language tutor–chatbot lucy," *Editorial Preface*, vol. 4, no. 11, pp. 124–131, 2013.

[32] L. K. Fryer, K. Nakao and A. Thompson, "Chatbot learning partners: Connecting learning experiences, interest and competence," *Computers in human Behavior*, vol. 93, pp. 279–289, 2019.

[33] A. Gallacher, A. Thompson, M. Howarth *et al.*, ""my robot is an idiot!"–students' perceptions of ai in the l2 classroom," *Future-proof CALL: language*

learning as exploration and encounters–short papers from EUROCALL, pp. 70–76, 2018.

[34] W. Huang, K. F. Hew and L. K. Fryer, "Chatbots for language learning—are they really useful? a systematic review of chatbot-supported language learning," *Journal of Computer Assisted Learning*, vol. 38, no. 1, pp. 237–257, 2022.

[35] *Data augmentation guide*, https://www.v7labs.com/blog/data-augmentation-guide#h5, Accessed: 2023-03-06.

[36] *Mediapipe*, https://developers.google.com/mediapipe/solutions. (visited on 7th Apr. 2023).

[37] *Mediapipe holistic*, https://google.github.io/mediapipe/solutions/holistic. (visited on 19th Feb. 2023).

[38] Google, *Mediapipe*, https://github.com/google/mediapipe. (visited on 19th Jan. 2023).

[39] I. Grishchenko and V. Bazarevsky, *Mediapipe holistic — simultaneous face, hand and pose prediction, on device*, https://ai.googleblog.com/2020/12/mediapipe-holistic-simultaneous-face.html. (visited on 11th Feb. 2023).

[40] *Mediapipe pose*, https://google.github.io/mediapipe/solutions/pose. (visited on 23rd Feb. 2023).

[41] *Mediapipe hands*, https://google.github.io/mediapipe/solutions/hands. (visited on 23rd Feb. 2023).

[42] C. M. Bishop, "Neural networks and their applications," *Review of scientific instruments*, vol. 65, no. 6, pp. 1803–1832, 1994.

[43] R. Hecht-Nielsen, "Theory of the backpropagation neural network," in *Neural networks for perception*, Elsevier, 1992, pp. 65–93.

[44] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[45] J. Brownlee, "What is the difference between a batch and an epoch in a neural network," *Machine Learning Mastery*, vol. 20, 2018.

[46] D. Masters and C. Luschi, *Revisiting small batch training for deep neural networks*, 2018. DOI: 10.48550/ARXIV.1804.07612. [Online]. Available: https://arxiv.org/abs/1804.07612.

[47] P. Cunningham, M. Cord and S. J. Delany, "Supervised learning," *Machine learning techniques for multimedia: case studies on organization and retrieval*, pp. 21–49, 2008.

[48] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities.," *Proceedings of the national academy of sciences*, vol. 79, no. 8, pp. 2554–2558, 1982.

[49] *What are recurrent neural networks?* https://www.ibm.com/topics/recurrent-neural-networks. (visited on 25th Feb. 2023).

[50] A. Sherstinsky, "Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network," *Physica D: Nonlinear Phenomena*, vol. 404, p. 132 306, 2020.

[51] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 6, no. 02, pp. 107–116, 1998.

[52] K. Cho, B. van Merrienboer, D. Bahdanau and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *CoRR*, vol. abs/1409.1259, 2014. arXiv: 1409.1259. [Online]. Available: http://arxiv.org/abs/1409.1259.

[53] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. DOI: 10.1162/neco.1997.9.8.1735.

[54] S. Yang, X. Yu and Y. Zhou, "Lstm and gru neural network performance comparison study: Taking yelp review dataset as an example," in *2020 International workshop on electronic communication and artificial intelligence (IWECAI)*, IEEE, 2020, pp. 98–101.

[55] X. Ying, "An overview of overfitting and its solutions," in *Journal of physics: Conference series*, IOP Publishing, vol. 1168, 2019, p. 022 022.

[56] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[57] I. Nusrat and S.-B. Jang, "A comparison of regularization techniques in deep neural networks," *Symmetry*, vol. 10, no. 11, 2018, ISSN: 2073-8994. DOI: 10.3390/sym10110648. [Online]. Available: https://www.mdpi.com/2073-8994/10/11/648.

[58] I. Goodfellow, Y. Bengio and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[59] H.-Y. Shum, X.-d. He and D. Li, "From eliza to xiaoice: Challenges and opportunities with social chatbots," *Frontiers of Information Technology & Electronic Engineering*, vol. 19, pp. 10–26, 2018.

[60] N. Bhirud, S. Tataale, S. Randive and S. Nahar, "A literature review on chatbots in healthcare domain," *International journal of scientific & technology research*, vol. 8, no. 7, pp. 225–231, 2019.

[61] V. Balakrishnan and E. Lloyd-Yemoh, "Stemming and lemmatization: A comparison of retrieval performances," 2014.

[62] *Natural language toolkit*, https://www.nltk.org/. (visited on 14th Mar. 2023).