Exploitation of noisy automatic data annotation for CNN training and its application to hand posture classification

Georgios Lydakis

Thesis submitted in partial fulfillment of the requirements for the

Masters' of Science degree in Computer Science and Engineering

University of Crete
School of Sciences and Engineering
Computer Science Department
Voutes University Campus, 700 13 Heraklion, Crete, Greece

Thesis Advisor: Professor *Antonis Argyros*

This work has been performed at the University of Crete, School of Sciences and Engineering, Computer Science Department.

The work has been supported by the Foundation for Research and Technology - Hellas (FORTH), Institute of Computer Science (ICS).

This thesis has been supported by the European Regional Development Fund of the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH – CREATE - INNOVATE (project code: T1EΔK-01299-HealthSign).

This thesis has been supported by the Hellenic Foundation for Research and Innovation (HFRI) and the General Secretariat for Research and Technology (GSRT), under the 1st Call for HFRI Research Projects to support Postdoctoral Researchers (GA. no. 188-SoCoLa).

University of Crete
Computer Science Department

Exploitation of noisy automatic data annotation for CNN training and its application to hand posture classification

Thesis submitted by
Georgios Lydakis
in partial fulfillment of the requirements for the
Masters' of Science degree in Computer Science

THESIS APPROVAL

Author:

Georgios Lydakis

Digitally signed by Georgios Lydakis
Date: 2021.07.09 13:16:45 +03'00'

Georgios Lydakis

Committee approvals:

Digitally signed by ANTONIOS ARGYROS
Date: 2021.07.09 14:21:45 +03'00'

Antonis Argyros
Professor, Thesis Supervisor

Panagiotis Tsakalidis

Digitally signed by Panagiotis Tsakalidis
Date: 2021.07.09 18:19:19 +03'00'

Panagiotis Tsakalides
Professor, Committee Member

ANASTASIOS ROUSSOS

Digitally signed by ANASTASIOS ROUSSOS
Date: 2021.07.09 16:12:48 +03'00'

Anastasios (Tassos) Roussos
Principal Researcher, Committee Member

Departmental approval:

Polyvios Pratikakis

Digitally signed by Polyvios Pratikakis
Date: 2021.07.12 10:15:34 +03'00'

Polyvios Pratikakis
Assistant Professor, Director of Graduate Studies

Heraklion, July 2021

Exploitation of noisy automatic data annotation for CNN training and its application to hand posture classification

Abstract

In recent years, advances in deep learning have resulted in a large-scale revolution in the field of Artificial Intelligence. Deep learning methods have been successfully applied to a variety of research topics, ranging from natural language processing and bioinformatics to speech recognition and computer vision, with the common goal of inferring a function which maps an input domain to a target one. However, the success of deep models at inferring such a function usually relies on the existence of a large amount of annotated training data, that is, input samples for which the output is specified. Due to the requirement for this kind of large training datasets, significant research is being conducted on methods for reducing the human effort that is necessary for their annotation.

Semi-supervised approaches, methods for synthetic data generation, and techniques for generating and handling automatic annotation are receiving increasing attention. In this work, we investigate a technique for utilizing automatically annotated data in classification problems. Using a small number of manually annotated samples, and a large set of data that feature automatically created, noisy labels, our approach trains a Convolutional Neural Network (CNN) in an iterative manner. The automatic annotations are combined with the predictions of the network in order to gradually expand the training set. This expansion attempts to select automatically annotated samples for which the label is deemed to be correct.

The proposed approach is generic and can be applied to any classification problem. In order to evaluate its performance, we apply it to the problem of hand posture recognition from RGB images. In general, the observation and interpretation of the human hand is highly useful in several application areas, so significant research has been carried out in the topics of 3-D hand tracking, observation of interaction of hands with objects as well as hand posture and gesture recognition. Sign Language Recognition is one area where hand posture recognition is especially useful, as the postures of a signer's hands are essential features in the translation of a sign language.

Motivated by the usefulness and impact of Sign Language Recognition, we develop a method for automatically annotating images or videos of hand postures, and apply it to the problem of classifying 19 postures that are common in the Greek Sign Language. The manual annotation of such data is a time-consuming process. Their automatic annotation is based on associating 3D joint configurations of hands with classes (hand posture labels), and yields noisy labels which can be used to train a Convolutional Neural Network. These characteristics of the problem make it a suitable candidate for applying our proposed method for automatic expansion of training datasets. We compare the results of training a CNN classifier with and without the use of our technique. Our method yields a significant increase in

average classification accuracy, and also decreases the deviation in class accuracies, thus indicating the validity and the usefulness of the proposed approach.

Εκμετάλλευση αυτόματης επισημείωσης δεδομένων για εκπαίδευση Συνελικτικών Νευρωνικών Δικτύων και εφαρμογή στην κατηγοριοποίηση χειρομορφών

Περίληψη

Τα τελευταία χρόνια, η πρόοδος σε τεχνικές βαθιάς μάθησης έχει επιφέρει μια επανάσταση μεγάλης κλίμακας στο πεδίο της Τεχνητής Νοημοσύνης. Μέθοδοι βαθιάς μάθησης έχουν εφαρμοστεί επιτυχώς σε μια πληθώρα ερευνητικών τομέων, από επεξεργασία φυσικών γλωσσών και βιοπληροφορική μέχρι αναγνώριση ομιλίας και υπολογιστική όραση, με κοινό στόχο την αυτόματη εκτίμηση μιας συνάρτησης που απεικονίζει ένα πεδίο εισόδου σε ένα πεδίο επιθυμητού αποτελέσματος. Ωστόσο, η επιτυχία των μεθόδων αυτών στο να εξάγουν μια τέτοια συνάρτηση συνήθως εξαρτάται από την ύπαρξη ενός μεγάλου όγκου επισημειωμένων δεδομένων εκπαίδευσης, δηλαδή δειγμάτων εισόδου για τα οποία η έξοδος είναι καθορισμένη. Λόγω της απαίτησης για μεγάλα σύνολα τέτοιων δεδομένων, σημαντική έρευνα διεξάγεται πάνω σε μεθόδους που μειώνουν το κόστος σε ανθρώπινη προσπάθεια που απαιτείται για την επισημείωση αυτών των δεδομένων.

Προσεγγίσεις ημιεπίβλεψης, μέθοδοι δημιουργίας συνθετικών δεδομένων και τεχνικές για δημιουργία και χειρισμό αυτόματης επισημείωσης συγκεντρώνουν αυξανόμενο ενδιαφέρον. Σε αυτή την εργασία, διερευνούμε μια τεχνική για αξιοποίηση αυτόματα επισημειωμένων δεδομένων σε προβλήματα κατηγοριοποίησης. Χρησιμοποιώντας έναν μικρό αριθμό δεδομένων που έχουν επισημειωθεί από κάποιο ειδικό, και ένα μεγάλο σύνολο δεδομένων που χαρακτηρίζονται από αυτόματα εκτιμημένες, θορυβώδεις ετικέτες, η προσέγγισή μας εκπαιδεύει ένα Συνελικτικό Νευρωνικό Δίκτυο (CNN) με επαναληπτικό τρόπο. Οι αυτόματες επισημειώσεις συνδυάζονται με τις προβλέψεις του δικτύου ώστε να επεκταθεί σταδιακά το σύνολο δεδομένων εκπαίδευσης. Αυτή η επέκταση επιχειρεί να επιλέξει αυτόματα επισημειωμένα δείγματα των οποίων η ετικέτα κρίνεται σωστή.

Η προτεινόμενη προσέγγιση είναι γενική και μπορεί να εφαρμοστεί σε οποιοδήποτε πρόβλημα κατηγοριοποίησης. Προκειμένου να αποτιμήσουμε την απόδοσή της, την εφαρμόζουμε στο πρόβλημα της αναγνώρισης χειρομορφών από έγχρωμες (RGB) εικόνες εισόδου. Γενικά, η παρατήρηση και η ερμηνεία του ανθρώπινου χεριού είναι πολύ χρήσιμη σε ποικίλες εφαρμογές, οπότε και έχει αναπτυχθεί σημαντική έρευνα στα θέματα της τριδιάστατης παρακολούθησης του χεριού, της παρατήρησης των αλληλεπιδράσεων του χεριού με αντικείμενα, καθώς και της αναγνώρισης χειρομορφών και χειρονομιών. Η Αναγνώριση Νοηματικής Γλώσσας είναι μια περιοχή όπου η αναγνώριση χειρομορφών είναι ιδιαζόντως χρήσιμη, επειδή οι μορφές των χεριών ενός νοηματιστή είναι κρίσιμα χαρακτηριστικά για την μετάφραση μιας νοηματικής γλώσσας.

Παρακινημένοι από την χρησιμότητα και τον αντίκτυπο της Αναγνώρισης Νοηματικής Γλώσσας, αναπτύσσουμε μια μέθοδο για αυτόματη επισημείωση εικόνων η βίντεο

χειρομορφών, και την εφαρμόζουμε στο πρόβλημα της κατηγοριοποίησης 19 χειρομορφών κοινών στην Ελληνική Νοηματική Γλώσσα. Η χειροκίνητη επισημείωση τέτοιων δεδομένων είναι μια χρονοβόρα διαδικασία. Η αυτόματη επισημείωσή τους βασίζεται στο συσχετισμό τριδιάστατων αναπαραστάσεων των χεριών με τις κλάσεις (ετικέτες χειρομορφών), και δημιουργεί θορυβώδεις ετικέτες που μπορούν να χρησιμοποιηθούν για την εκπαίδευση Συνελικτικών Νευρωνικών Δικτύων. Αυτά τα χαρακτηριστικά του προβλήματος το καθιστούν υποψήφιο για την εφαρμογή της μεθόδου που προτείνουμε για την αυτόματη επέκταση συνόλων δεδομένων εκπαίδευσης. Συγκρίνουμε τα αποτελέσματα εκπαίδευσης ενός CNN με και χωρίς τη χρήση της τεχνικής μας. Η μέθοδός μας επιφέρει σημαντική αύξηση στην μέση ακρίβεια κατηγοριοποίησης, και επιπλέον μειώνει την απόκλιση της ακρίβειας ανά κλάση, καταδεικνύοντας έτσι την εγκυρότητα και τη χρησιμότητα της προτεινόμενης προσέγγισης.

Acknowledgements

First and foremost, I would like to thank my supervisor, Prof. Antonis Argyros, for his guidance and support since the beginning of my Master's studies. Our frequent discussions were invaluable both for defining the direction of this thesis, as well as instrumenting its execution in practice. Also, I would like to thank Dr. Iason Oikonomidis, for his advice and insights, and our many discussions of ideas that failed or succeeded. Without their contributions, the completion of this work would not have been possible.

I am grateful to my family, for supporting me and encouraging me during my studies. I would especially like to thank my parents, for teaching me the value of perseverance. I am also thankful to my friends, and to Eirini, for making my life better during these studies. Finally, I would like to thank everyone who provided hand posture data for this project: Alexandros, Dimitra, Dionysis, Eleutheria, Kyriakos, Manos, Myron, Nikos, Olia, Spyros, Alexandros, Zina, this work would have been much harder without your help.

στους γονείς μου

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In the past few years, deep learning methods have revolutionized the field of Artificial Intelligence (AI), achieving previously unattainable performance on a plethora of challenging tasks. Examples include image recognition [34], object detection in images [27], scene understanding [88], generation of novel images of a particular domain [72], natural language processing and machine translation [18] and speech recognition [30].

Part of the success of deep learning methods is attributed to the fact that their architectures allow the models to automatically extract useful features from their training data [50]. This allows them to discover how to efficiently solve a problem without the need for complex feature extraction techniques developed by humans. However, in order for deep models to successfully extract features, a large amount of training data must be available. This is due to the fact that such models comprise a large number of trainable parameters, often in the order of millions. Therefore, if only a limited number of samples is available, deep models tend to overfit on them, essentially memorizing each sample. This leads to poor performance on data that the model has not been trained on.

Machine learning refers to the problem of inferring a function that maps input of one domain to output of another. Among many approaches to it, supervised learning is a commonly employed one. In this approach, learning the function is performed by training on a set of samples, where each sample is an input-output pair. Deep models adopting this approach have been quite successful in many tasks. For example, image classification is a supervised learning task, where the input is an image and the expected output is a label that describes the content of the image, chosen from a predefined set of labels. The traditional approach for training a deep model on a supervised learning task is to manually annotate a set of input samples with their corresponding ground truth. Manual annotation involves human effort, since each sample must be examined prior to providing the associated ground truth information.

Another common approach is semi-supervised learning, in which the available data is usually a combination of a small number of manually annotated samples

and a large number of samples for which the labels are unknown. In most cases, semi-supervised approaches assume that both labeled and unlabeled data follow a common distribution. For example, points that are near each other in the feature space are considered more likely to share the same label. In the general case, semi-supervised learning attempts to devise methods which require less manually annotated data, and as such, less human effort. Unsupervised learning is a third approach to machine learning, in which all samples are unlabeled. Unsupervised learning algorithms aim to discover underlying patterns in the data, by employing techniques such as clustering.

In the case of supervised learning, annotating large amounts of data can quickly become very expensive in terms of human effort, especially so if the annotation procedure is itself difficult to perform, for example, creating pixel-level segmentation masks. For this reason, besides semi-supervised and unsupervised learning, research is also highly active in the field of reducing the annotation effort required for training deep models in a supervised manner. In this work, we propose a technique for utilizing a large number of samples that have been automatically annotated with labels for a classification task. Given that the annotation is automatic, it is possible that the extracted labels may be noisy.

The method we develop is generic in nature, and can be tailored to address any classification problem. It assumes the existence of a small number of manually annotated samples, as well as a large set of automatically annotated ones, whose labels might be noisy. We begin by training a Convolutional Neural Network (CNN) on the manually annotated data, resulting in a classifier that might not generalize well, given the small number of samples.

Then, we compare the predictions of the network for all automatically annotated samples with the noisy ground truth labels. We incorporate in the training set a subset of those, for which the predictions of the classifier agrees with the labels. The intuition is that the agreement of the two predictors (classifier, noisy automatic annotation) is probably not coincidental. The network is then trained again on the new dataset, and the procedure is iteratively repeated until there is no improvement in validation accuracy. We experiment with two variations of the method, based on how conservatively/aggressively we expand the training dataset with automatically annotated data.

In order to evaluate our technique, we apply it to the problem of hand posture recognition from RGB images. In general, the ability of a computer system to understand and monitor the structure of a human hand can be useful in a variety of applications. For example, Jang et al. [39] develop a system that allows the user to interact with Virtual Reality objects by using their hands, and Markussen et al. [56] present a virtual mid-air keyboard, the use of which is based on tracking the hands in 3-D space. Another important area of application is Sign Language Recognition. The posture and motion of hands play an important role in conveying information in sign languages, when combined with other non-manual features. Additionally, in the case of finger spelling, each letter of the alphabet is represented with a specific hand posture [65], and thus entire words can be communicated from a deaf user to

a computer system solely via hand posture recognition.

Motivated by the domain of Sign Language Recognition, we formulate a classification problem for hand postures used in Greek Sign Language. More specifically, we aim to develop a lightweight yet robust hand posture classifier, in the context of the HealthSign project [63]. This project aspires to create a system for the automatic translation of Greek Sign Language in the context of healthcare services. No definitive formal documentation exists for Greek Sign Language [2], however, 54 postures are estimated to be in use today, 24 of which are also used to represent the letters of the Greek alphabet [64]. Our posture recognition problem is defined on a subset of 19 commonly used postures. We present a method that processes unlabeled videos of subjects signing, and automatically assigns ground truth labels to the frames. This is based on using a 3D hand pose estimation tool, in our case, Google's MediaPipe [97], for estimating the posture in each frame, and comparing the 3D configurations of joints to those corresponding to the problem's classes. Precisely because the annotation produced in this manner features noise, this is a suitable problem on which to apply the techniques we present for handling noisy ground truth data. Our experiments indicate that our method shows promising results when used in a problem that features a small number of manually annotated samples and a large number of noisy, automatically annotated samples. Its use yields significant improvements over training on the manual data only, and over treating all automatic annotations as correct.

The rest of this thesis is organized as follows. Chapter 2 provides an overview of research work in the three broad areas relevant to our topic: observation and interpretation of the structure and motion of hands, sign language recognition and automatic annotation of data as well as methods for reducing annotation effort. Chapter 3 gives a detailed description of the proposed method for handling noisy ground truth, as well as the techniques developed specifically for the problem of hand posture recognition. An evaluation of our methods is available in Chapter 4, and chapter 5 summarizes our conclusions and proposes possible future work.

Chapter 2

Related work

The work presented in this thesis is relevant to three highly active fields of research. Firstly, the main element being studied is a method for utilization of automatic ground truth, which is in fact applicable to classification problems in general. As such, research in the field of annotation effort reduction and automatic ground truth utilization is highly relevant. Secondly, since we tackle the problem of hand posture recognition, and since we employ the use of a 3-D hand posture estimation method, our work is related to the general area of hand observation and interpretation. This area includes the subdomains of hand tracking and posture or gesture recognition. Thirdly, because we develop an application that recognizes postures important in Greek Sign Language, an overview of work done in the field of Sign Language Recognition is also pertinent.

Therefore, in the following sections we proceed to present an overview of each of these three research fields.

2.1 Annotation effort reduction

The successful application of a deep learning method on a problem requires a large amount of training data, generally much larger than the amount required for traditional machine learning techniques. This is due to the fact that deep learning models can have a very large number of trainable parameters, and thus can easily overfit on a small number of samples. Annotating large amounts of data, that is, specifying the ground truth associated with each sample, is a process that requires significant human effort. Therefore, it is clear that techniques which reduce this effort either by automating a part of it or by decreasing the amount of necessary data are of great interest in the general area of deep learning.

Techniques for reducing annotation effort can either be general or problem-specific in nature. A general technique which has proven invaluable is data augmentation [81], i.e. the application of randomized transformations on already annotated data. This process yields new data samples, for which, however, the transformations ensure that the original ground truth is still applicable. One example of a

domain-specific technique is the work of Voigtlaender et al. [89], in the problem of semi-supervised video object segmentation. This refers to the task of segmenting objects in all frames of a video, given their ground truth pixel masks in the first frame in which each object appears. Clearly, manually creating pixel masks requires a significant amount of human effort, and is a task that does not scale well when a large number of annotated frames is desirable. The authors attempt to reduce this effort by creating a CNN that can extract pixel-level pseudo-labels given only bounding box annotations. This method can significantly reduce the necessary human effort, since it is much easier to annotate object segmentation at the coarser level of bounding boxes.

Reducing annotation effort is also desirable in the field of active learning, i.e. the field of machine learning where an algorithm repeatedly queries a user, known as the oracle, for labeling new data. Sun et al. [84] argue that even though several automatic or semi-automatic annotation methods can reduce the number of instances that need to be labeled, the queries to the oracle which offer the most to the learning algorithm remain the most difficult cases to label. The authors attempt to alleviate this issue by leveraging available metadata that can give the oracle "hints" for the labeling process, by clustering data points with similar metadata attributes.

The work of Mueller et al. [60], an overview of which is available in section 2.2, is also an example of automatic annotation, in the problem of 3-D hand pose estimation. It is based on generating a plethora of synthetic hand images for which the 3-D pose is known, and used in the generation process. These images are crucial for the successful training of the paper's 3-D hand tracker. In the same field, Tang et al. [86] present a semi-supervised method for RGB-D based 3-D hand pose estimation. Semi-supervision refers to a family of machine learning techniques which operate on a small set of manually labeled data combined with a larger set of unlabeled data. The method presented in this paper uses transductive transfer learning [66], a technique where a model is trained on one domain, in order to gain knowledge useful in a different, related domain. In this case, the authors train a Regression Forest on synthetic images of hands, which are, by design, fully labeled, as well as on a small number of manually labeled real images. Furthermore, by leveraging transduction, the training process also exploits a larger set of unlabeled real images. This is made possible by knowledge acquired from the synthetic data, which feature a plethora of different poses, assisted by some domain knowledge acquired from the small set of manually annotated real images.

The general approach of semi-supervised learning is also closely related to the problem of reducing annotation effort. The term semi-supervised learning refers to methods that are trained on a combination of a small amount of manually annotated data and a large amount of unlabeled data. In essence, semi-supervised approaches attempt to leverage any existing ground truth, while also benefiting from a collection of data for which annotation would be impractical. The work of Tang et al. [86], which has been previously analyzed, features elements of semi-supervised learning: a large set of unlabeled real images of hands is used as training

data in combination with a small, labeled set of real images and a larger, also labeled set of synthetic images.

Honari et al. [36] develop two techniques for landmark localization based on partially annotated datasets. Landmark localization refers to detecting precise locations of specific parts in an image, for example, joints that comprise the human hand or skeleton, or a set of points that describe important facial features. Their approach does not depend on the specific nature of these landmarks. Essentially, the authors leverage the limited samples where landmark annotation exists, as well as a more abundant set of samples for which only a more general, high-level label is available. This label can be related either to a classification or regression task, and serves as an auxiliary guide towards localization of landmarks on the unlabeled data.

In a work closely related to the ones presented in section 2.2, Wan et al. [91] present a semi-supervised approach for 3-D hand posture estimation from single depth images. The approach is based on creating two generative models which share a feature space, such that any point in this space can be mapped to a unique depth image, and a unique 3-D hand pose. Then, estimating the hand pose present in a depth image is performed with the use of a model that estimates posterior probabilities of poses given depth maps. The nature of the architecture allows the exploitation of unlabeled depth images: the authors develop a generative model that maps between the feature space and the hand pose space. Furthermore, an unlabeled depth image is, by definition, a valid point in the depth map space, and therefore the corresponding ground truth pose can be recovered.

One of the oldest techniques in semi-supervised learning is self-learning, or self-training [17], which is based on the repeated use of a supervised method. Initially, the supervised algorithm is trained on labeled data only, and on each step some subset of the unlabeled data is labeled by using the trained model. The procedure is repeated, and the training set gradually expands to feature data labeled by the algorithm itself.

Also relevant to the topic of reduced annotation effort is the concept of learning from data where the ground truth is noisy. The proximity of the two topics arises from the fact that automatically created ground truth has a greater chance of featuring noise. Therefore, techniques that tackle this issue also facilitate the use of automatic annotation methods. Jiang et al. [40] develop a strategy for training on noisy ground truth based on Curriculum Learning. Curriculum Learning, developed by Bengio et al. [9], is a technique for guiding the optimization of a neural network by presenting the training examples in an order which encourages it to progressively learn more complex features. Jiang et al. [40] apply this concept by simultaneously training two networks, one which learns the actual task featuring the noisy ground truth, and another which learns how to guide the first by presenting samples that are deemed correct. Note that, since the second network undergoes a training process, the curriculum adapts to the data at hand.

Hacohen et al. [31] investigate Curriculum Learning with two strategies. The first strategy involves a second, "teacher" network, which transfers knowledge it

has accumulated from some other dataset. The second strategy is a type of bootstrapping, where the network is first trained on the target dataset without any curriculum. Then, the resulting classifier is used to determine an estimate of how beneficial each sample is in the problem, and retrain the network via the resulting curriculum.

Han et al. [33] also tackle the problem by coining a method known as "Co-teaching". It involves training two networks simultaneously, where each one teaches the other about what data it considers correctly labeled. The intuition behind their method is based on the fact that a network tends to learn correct samples in the first stages of optimization, and memorize the wrong ones at a later point [4]. The reason for this effect is that correct samples are easier to learn, since they are described by meaningful patterns, whereas wrongly labeled ones require explicit memorization by the network.

Mirzasoleiman et al. [57] address the same issues by developing a method that can select subsets of the data that are likely to be free of noise. Their selection is based on inspecting the Jacobian matrix of the loss function being optimized, and choosing medoid data points in the gradient space. A medoid of a set is a point of the set that minimizes dissimilarity to a cluster of data points of this set. By choosing the samples in this fashion, they avoid overfitting on parts of the dataset featuring corrupted ground truth.

2.2 Hand observation and interpretation

The capability to track the motions of a human hand in 3-D space can prove highly valuable for a plethora of systems where humans interact with computers. Recent advances in the field of computer vision have yielded techniques that facilitate this process without the need for expensive wearable sensors or other specialized hardware.

Applications of such estimation methods include Augmented Reality (AR) and Virtual Reality (VR) systems where the primary mode of interaction is hand gestures. For example, Jang et al. [39] present a system that tracks the user's hands from an egocentric viewpoint and allows interactions with VR objects. Markussen et al. [56] develop a mid-air keyboard, with which the user interacts via hand motions tracked in 3-D space. Besides human-computer interaction, other applications include activity recognition, and sign language recognition. An example of use in activity recognition is the work of Rohrbach [74] for recognition of activities related to cooking, where subtle hand movements play an important role in the identification of a particular activity. For sign language recognition, a more extensive overview is available in section 2.3, where the importance of hand-related features in this task is highlighted through multiple relevant research works.

Due to the increased interest in the area, a great number of research works have emerged in recent years, involving hand tracking in situations that have progressively become more challenging. Although older works required depth information

as input, nowadays estimation from RGB images has become more and more common. Zimmermann et al. [100] present one of the first works in the area, which is based on a deep network that learns a 3-D pose prior describing poses that are feasible for an articulated hand. This works in tandem with the detection of 2-D hand joints on the image to produce a pose that is simultaneously feasible as a configuration and consistent with the image itself. The network was trained on synthetic hand data from 3-D models of humans, since the 3-D joint positions were readily available. Iqbal et al. [38] tackle the problem by creating a convolutional neural network (CNN) architecture that learns depth maps and heatmap distributions for potential positions of the joints on an image, and results in a 2.5-D pose representation. This means that joint depths are expressed relative to the wrist joint, and poses are estimated up to a scaling factor, which can be determined if a hand size prior is available.

A general issue for training deep learning models on the task of 3-D pose estimation has always been the need for large amounts of training data, e.g in the form of RGB images with 3-D annotations [3]. Mueller et al. [60] attempt to solve this problem by generating large amounts of synthetic training data with a modification of a CycleGAN [99]. A CycleGAN is a neural network capable of transforming images from one domain to another while preserving their content (e.g. photos to paintings of a particular art style). The authors use a CycleGAN modified with a geometric consistency loss in order to be able to translate images of 3-D rendered hands into synthetic images that approach the appearance of real hands. The generated images maintain the 3-D pose, and therefore a large number of training samples can be produced. The images are used to train a CNN, whose output is combined with a kinematic 3-D hand model in order to produce more natural pose results during tracking.

Zhang et al. [97] develop a pipeline that can compute estimations for 2.5-D hand joints at real-time speeds on mobile devices. They first use a CNN for palm detection, then use its cropped outputs to localize 2.5-D hand joints via another deep model. Their training data combines real images of hands with synthetically generated images. These images are rendered using a hand model of variable finger and palm thickness as well as variable skin tone.

There has also been recent work on even more challenging variations of the problem. For example, Hampali et al. [32] tackle the case of two hands closely interacting with each other and Kwon et al. [49] work on the problem of two hands interacting with objects in a first-person viewpoint. Rudnev et al. [75] propose a method for 3-D hand tracking from event camera feed. Event cameras are specialized sensors that react to events like brightness changes and whose temporal resolutions can vary, depending on the speed of changes which take place in the scene [54, 75].

Besides the general problem of 3-D hand pose estimation, another common problem in the literature is hand posture and gesture recognition. In posture recognition, we are interested in recognizing a specific set of hand postures, i.e. static configurations of hands. In gesture recognition we are interested in recognizing a

particular sequence of postures, i.e. the problem also has a temporal dimension.

Barros et al. [7] present a CNN for hand posture recognition, which works on three different channels of information: grayscale images, and two channels which result from the application of horizontal and vertical Sobel filters for edge detection. Tang et al. [85] propose a method that uses a Kinect sensor to detect and track hands, and a CNN to classify hand postures. Their detection method relies on skin color and depth to track and segment hands, and the images of the isolated hands are used as input to a small convolutional network. Nai et al. [61] present an algorithm for hand posture classification from depth images, which is based on extracting features from randomly positioned line segments. Each segment essentially serves as a sort of "scan line" over the depth image, detecting depth disparities which indicate the presence and orientation of fingers. These extracted low-level features are then used in a Random Forest classifier.

De Smedt et al. [23] present an approach for gesture recognition that is based on extracting a feature representation from skeleton joints sourced from a depth camera. Their feature descriptor is based on the vectors which connect hand joints to each other, and attempts to be invariant to hand rotations and translations. Although this approach tackles gesture recognition, one could use the features extracted for each frame in the closely related problem of posture recognition, which does not have a temporal component. Similar strategies could be used for exploiting other works in gesture recognition.

Asadi-Aghbolaghi et al. [5] present a survey of deep learning methods for gesture recognition, and more generally action recognition in image sequences. Their taxonomy divides approaches in three groups. The first group uses 3-D convolutional filters, i.e. filters on both spatial and temporal dimensions. The second group uses as input to the networks features that are two-dimensional in nature, but relate to motion, such as optical flow. The third group uses 2-D convolutions in combination with some technique for temporal modeling (e.g. Hidden Markov Models).

Evangelidis et al. [25] approach the problem as an assignment of per-frame labels for gestures. The assignment has to obey certain temporal smoothness properties. Their method makes use of articulated posture data as well as color information. The per-frame feature vectors are classified with an SVM [80]. Their classifications are then used in a dynamic programming approach for solving the optimization problem of assigning labels that also obey the temporal smoothness criterion. Neverona et al. [62] approach the same problem via multi-modal deep learning. Specifically, they use different spatial scales to capture information about posture movement of various “grains” (e.g. body, hand movement), and different temporal scales to capture variations in gesture speed. Their combination of modalities is performed with a dropout training strategy for modalities, so as to achieve more robust performance and learn correlations between modalities. Wu et al. [93] describe a method for multimodal gesture recognition, where the modalities are skeleton information and RGB-D images. Their method uses different types of neural networks for the different modalities, tailoring their architecture to the type

of the specific input. The networks are integrated in a Hidden Markov Model (HMM) framework, for tackling the temporal aspects of the problem. Temporal segmentation and recognition of the gestures are thus performed simultaneously.

2.3 Sign language recognition

Sign languages are the main means of communication for deaf individuals, and have been developed in parallel with spoken languages [78]. Although there is no single universal sign language, as a general rule in most sign languages the signs are composed of features related to hand posture, movement and orientation, as well as features related to body posture and facial expressions. A gloss is a description of the manual and non-manual features necessary to convey a specific meaning in sign language [94]. Adaloglou et al. [1], as well as Rastgoo et al. [73], present two comprehensive overviews of the significant body of work on Sign Language Recognition (SLR). The task is highly complicated in nature, due to the fact that some of the features involved can be very subtle (e.g. small movements, or differences in speed may hold meaning), as well as due to the lack of standardization even within the same sign language [1].

Most of the work in the area uses visual input, although exceptions exist, such as haptic sensors [42]. Useful categorizations can be defined over this body of work. The taxonomy presented in that survey [1] yields three categories of tasks related to the general problem:

- Isolated SLR, where the task is classifying video segments that have already been ensured to contain a single gloss. Kadous [42] presents an early example of Australian Sign Language recognition through the use of a haptic glove (Power Glove). The work proposes machine learning techniques applied on sensor data from the glove, in order to recognize 95 glosses, using only hand-related features. Camgoz et al. [11] study the results of using 3D Convolutional Neural Networks for gesture recognition. Although this is not done strictly in the context of SLR, the work presents a technique for recognizing video segments of gestures, through the use of spatiotemporal convolutions. Cooper et al. [19] explore the use of "linguistic sub-units", that is, features related to the location of the hands relevant to the signer, the motion of the hands and their relative positions to each other. The paper experiments with sub-units based on appearance (i.e. RGB data), as well as 2-D and 3-D pose tracking data. The combination of these features, in such a way as to encode temporal information as well, is explored, through the use of Markov Models and a boosting technique. Tests are conducted on datasets featuring examples of German Sign Language signs.

- Sign detection in continuous streams, where the task is the detection of predetermined glosses in a video, without the segmentation assumed in isolated SLR tasks. Therefore, segmentation and recognition must be jointly tackled.

The works mentioned in section 2.2 regarding gesture recognition [25, 62, 93] can be categorized as belonging in this research domain. Although they approach gesture recognition in a context that is not necessarily SLR, their methods could certainly be applied in sign detection in continuous streams.

- Continuous SLR, where the input is a video without segmentation, and the glosses that are present are not known a priori. This is, by definition, the task that is closest to real-life applications. Methods developed for this task must, by necessity, somehow deal with the temporal component of signing.

  To this end, they make use of various techniques that have been developed. Examples include Hidden Markov Models, Dynamic Time Warping (DTW, [76], and a differentiable variation, soft-DTW [22]), an algorithm for detecting similarities in time sequences despite speed differences, Connectionist Temporal Classification (CTC, [29]), a method for training recurrent neural networks with unsegmented sequences. LSTMs [35], a type of recurrent neural network capable of learning sequences, are also used. A different approach is through the use of spatiotemporal convolutional networks, which directly learn features from videos, e.g. the work of Tran et al. [87], which uses 3-D convolutions on video data. Another architecture (I3D) similar to 3-D convolutional networks is presented by Carreira et al. [16], in a way that allows the use of 2-D architectural designs that have been successful in tasks such as ImageNet [24]. Encoder-decoder architectures have also been utilized, starting from their successful application in the field of machine translation [6], since SLR is itself a kind of translation process.

  Koller et al. [45] present a classifier that can learn frame-labels for hand postures from videos with only sequence-level annotations. They achieve this by using a CNN with an iterative training process in the style of Expectation Mazimization [58]. Koller et al. [46] use a CNN in combination with an HMM, in order to be able to train end-to-end on sequences (therefore, recognize glosses). Pu et al. [69] use two data channels, RGB videos of gestures and a feature representation of joint trajectories. The RGB videos are processed with a 3-D CNN to extract gesture variations, while the shape contexts [8] of joint trajectories are fed to another CNN to extract a robust representation. The output of these networks is combined in a feature vector, for the classification of which an SVM is used. Camgoz et al. [12] present an architecture combining CNNs, LSTMs and a CTC loss. Essentially, the CNNs extract image-level features, the LSTMs are used to add a temporal dimension to these spatial features, and the CTC layer allows training on sequences of variable lengths. The work of Cui et al. [20] yields an architecture for SLR which assumes knowledge of the order of appearance of glosses in the test data, but no exact segmentation of the test video. They use recurrent networks coupled with CTC in order to learn sequences end-to-end and extract an approximate temporal segmentation of the video based on

the labels. Then, they tune their features using this information and regularize sequence predictions with a sign detection network. Koller et al. [47] expand their previous work [46] by combining their CNN-HMM model with LSTMs, in order to address the issue of noise in the labels of training videos. They treat them as weak labels, which their method can realign in order to achieve better consistency with the actual frames. Huang et al. [37] use a 3-D CNN that receives as input both full frames as well as frames cropped around hands, and a Hierarchical Attention Network (HAN, extension to LSTM with attention mechanism) that works on a latent feature space. This allows them to avoid explicit temporal segmentation.

Joze et al. [41] use the aforementioned I3D architecture, in a work with emphasis on realistic settings, leading in the creation of a dataset with 200 signers and 1000 signs. Pu et al. [70] present another example of 3-D architecture, by using a 3-D ResNet [34] for feature extraction, and a CTC for learning sequence-to-label mappings. For end-to-end training they use an iterative strategy with intermediate pseudo-labels for the 3-D CNN, which are generated by the CTC from the sequences. These two steps are executed alternately. Koller et al. [44] further explore the idea of weak supervision on multiple input data streams, each handled with a HMM and a CNN-LSTM combination. The input streams consist of full frame RGB data, frames cropped around the hands, and frames cropped around the mouth, in such a way as to discover features which would otherwise be harder to discriminate. Cui et al. [21] extend their work in [20], by using CNNs for feature extraction and recurrent models for sequence learning, with an iterative training strategy where sequence alignment and feature tuning are performed alternately, until no improvement is observed on the validation set. Also, they explore the use of a combination of RGB information and optical flow in SLR. Zhou et al. [98] tweak the idea of CNNs combined with recurrent networks and CTC, by using the I3D architecture and a dynamic programming approach to intermediate pseudo-label generation. Yang et al. [94] propose SF-Net, which attempts to learn features for multiple levels (frame, gloss and sentence level). To this end, they combine 2-D and 3-D CNNs, LSTMs and CTC for extraction of both spatial and temporal features. Another work by Pu et al. [71] combines two modules, a 3-D ResNet and an encoder-decoder. The encoder-decoder module uses an LSTM and a CTC decoder, both of which require soft-DTW during training to quantify the alignment agreement between predicted and ground truth labels. The alignment is then used to tune the 3-D ResNet feature extractor, which is used to tune the sequence learning, leading in an alternating training process.

Another interesting subdomain of SLR is Sign Language Translation, which focuses on the extraction of entire, grammatically structured sentences from videos, instead of simple gloss annotations. Camgoz et al [13] address the problem with an approach that uses an encoder-decoder architecture for

translation. They use a CNN to learn spatial embeddings for glosses, that is, feature representations where glosses with similar meanings are close to each other. Li et al. [53] present a method for translating signing video sequences into text-based natural language. The method is based on a representation of video segments that features multiple temporal granularities, combined with a technique for hierarchical feature learning. This hierarchical learning uses attention mechanisms to detect both local semantic information from segments, as well as context information that can resolve ambiguities in word meanings. Yin et al. [95] develop a transformer-based approach that is trained end-to-end to map sign language videos to sentences. Their method exploits spatial features from multiple cues: faces, hands, and entire body postures. Furthermore, it is capable of learning temporal relationships for features originating from the same or different cues. They compare their end-to-end approach with approaches that use gloss annotation as an intermediate step. Stoll et al. [83] present a work that tackles the problem of generation of signing videos from text sentences. This is essentially the inverse of the problem of Sign Language Translation. They present a two-step method that first translates sentences into pose sequences, and then uses this pose information as input to a GAN that produces photorealistic videos.

Due to the increased interest the area of SLR has garnered, several datasets have also been created. Some of them are:

- Signum [90], a dataset featuring RGB data of German Sign Language in laboratory conditions (20 signers).
- RWTH-Phoenix [26], [13], also an RGB dataset in German Sign Language, extracted from weather forecasts (9 signers).
- The Chinese Sign Language (CSL) datasets [37], [69] for isolated and continuous sign language, which feature RGB-D data for daily conversations in laboratory conditions (50 signers).
- The American Sign Language (ASL) dataset [41], recorded in real-life settings and, as such, featuring greater variability. The videos originate from many online sources and feature 222 signers, with RGB video data where the sign labels where created with OCR from video subtitles.
- BosphorusSign [14], with signs from the domains of health, finance and everyday life in Turkish Sign Language. It features RGB-D data from 10 different signers in laboratory conditions.
- The GSL dataset [1], featuring interactions of signers with public services, such as the police, hospital and citizen service centers. The dataset provides RGB-D data in Greek Sign Language for 7 signers, both for isolated and continuous SLR.

- HealthSign [48], which features scenarios from interactions of signers with mental health professionals. Currently, it features 8 signers, and provides RGB-D data as well as 2-D pose information for the body and 3-D hand pose information, as well as 2-D facial data.
- The WLASL dataset [52], which is a dataset of American Sign Language, featuring 2000 different words, performed by approximately 120 signers. The creators aim to feature significant diversity in signer-specific characteristics by collecting and annotating a plethora of RGB videos from the Internet.

For a more detailed comparison of these datasets see Adaloglou et al. [1] and Kosmopoulos et al. [48].

2.4 Our approach

This work investigates an approach for working with noisy ground truth data that have been automatically annotated. Our approach, presented in section 3.1, exhibits similarities to the approaches presented here, especially to Curriculum Learning and self-training approaches. This is due to the fact that it attempts to select appropriate subsets of the automatically annotated training data. Furthermore, this is done in an iterative manner which involves the predictions of the trained classifier itself. In addition to presenting this approach, we evaluate its use on the problem of hand posture recognition from RGB images, for a set of hand postures which convey meaning in Greek Sign Language.

Chapter 3

Methods

3.1 Methods for exploiting automatic ground truth

This work is primarily concerned with investigating techniques which can be used to efficiently utilize automatically annotated data for training CNNs on classification tasks. Although these methods are studied and evaluated on a particular application, i.e. hand posture classification from RGB images, they are generic in nature.

Assume a classification problem featuring K classes, and a method which is capable of classifying a sample in one of these classes. However, it is assumed that the annotations produced by the method are imperfect: given a set of unlabeled samples, each of which may or may not belong in one of the classes, the method may occasionally err in one of two ways. First, it may label samples of one class as belonging to another. Second, since some of the unlabeled samples may not belong in any of the classes, the method may assign a class label to an out-of-problem sample. Clearly, training a CNN on a dataset where such failure cases are somewhat common will affect its ability to discriminate between the classes of the problem.

Manually annotating a large set of data for training a CNN is a task that requires a significant amount of human effort. Furthermore, it must be performed anew for every different task. On the other hand, methods for automatic annotation often suffer from the imperfections described above, and would have adverse effects on the performance of the resulting classifier. It is therefore useful to investigate techniques that can reduce the necessary human effort by exploiting the noisy automatic labeling in a better way than simply using all the automatically annotated data as a training set.

Among all candidate techniques, the simplest way to utilize automatically annotated data is to consolidate large numbers of such samples, potentially reducing the impact of failure cases on the performance of the classifier. One could also treat those samples as completely unlabeled, combining them with a small number of manually annotated samples and thus adopt a semi-supervised approach. However,

it is our intuition that the utilization of a noisy ground truth label can be beneficial, and assuming we can annotate a small number of samples, a more sophisticated approach can be devised. We now have at our disposal a large dataset, D_{auto} for which automatically produced ground truth labels are available. We also have a small set of samples, D_{manual} which we have annotated manually, and thus their labels are expected to be significantly less noisy.

Observe that in this case, it is likely that training a CNN on D_{manual} will yield a classifier that does not generalize effectively. However, it can still provide a noisy estimate of the likelihood of a sample belonging in a particular class. In common practice, CNNs trained on classification tasks of K classes are designed to output K values. The network is trained so that given a training sample of class i the i-th value is 1 and the remaining $K-1$ values are 0 (one-hot encoding). In practice, these values end up somewhere between 0 and 1, effectively approximating the likelihood of a sample belonging in each of the classes (section 3.5). Furthermore, as previously explained, the automatic annotation method also provides a noisy labeling for the samples.

Therefore, we can train a CNN on D_{manual}, compute its predictions on all samples of D_{auto}, and select only those samples where the prediction agrees with the labeling produced by automatic annotation. This yields $D_{auto,0} \subseteq D_{auto}$, a set of samples which are more likely to be correctly labeled, since we are conservatively selecting only the samples for which two noisy predictors agree. If we were then to train a CNN on $D_{manual} \cup D_{auto,0}$, we expect it to generalize better than the one trained on D_{manual} only, since it has a larger number of samples from which to extract useful classification features. Furthermore, the increased number of samples is more likely to prevent overfitting.

We can then use the predictions of the CNN on $D_{auto} \setminus D_{auto,0}$, and again select the samples where the prediction agrees with the automatic labeling. This yields $D_{auto,1}$, and we can then form a new training set $D_{manual} \cup D_{auto,0} \cup D_{auto,1}$. By continuing this iterative process, increasingly robust classifiers are formed, and we expect a better utilization of D_{auto}, since the combination of the two predictors will filter out some of the noise of the automatic annotation. Provided we have a validation set for estimating classifier accuracy, we can continue this iterative process until the validation accuracy of the classifier trained on the selected data no longer increases, or starts to decrease. The latter could potentially occur if the automatic labeling results in many similar samples which share the same incorrect label. In that case, if the process happens to include even a few of those samples in the training set, the capacity of the CNN to memorize samples could result in many more of them "contaminating" the training set later on.

Henceforth we shall call this method "Greedy Iterative Dataset Expansion Algorithm", or "G-IDEA". The pseudocode implementing it is summarized in algorithm " G-IDEA" (it assumes a classification problem of $Kclasses$).

Note that in practice, on each iteration, m_{new} is selected to be the model which achieves the highest validation accuracy on $D_{validation}$ during training.

A second important observation is that we expect the classifier's predictions

Algorithm G-IDEA Train a CNN on a combination of manually and automatically annotated data, attempting to avoid noisy samples

Input:
D_{manual}, a set of manually annotated samples
D_{auto}, a set of automatically annotated samples
$D_{validation}$, a set of manually annotated samples to be used as a validation set
Output:
m, a CNN model trained on some of the input data
$D_{selected}$, the corresponding training data

$m_{new} \leftarrow$ train a CNN on D_{manual}
$D_{selected} \leftarrow D_{manual}$
$val\_acc \leftarrow$ compute accuracy of m on $D_{validation}$
$prev\_acc \leftarrow -\infty$
$D_0 \leftarrow D_{manual}$
$i \leftarrow 0$
while $val\_acc > prev\_acc$ **do**
  $i \leftarrow i + 1$
  $m \leftarrow m_{new}$
  $D_{selected} \leftarrow D_i$
  $new\_data \leftarrow$ all samples of $D_{auto} \setminus D_{i-1}$ where the class output of m agrees with the label given in D_{auto}
  $D_i \leftarrow D_{i-1} \cup new\_data$
  $m_{new} \leftarrow$ train a CNN on D_i
  $prev\_acc \leftarrow val\_acc$
  $val\_acc \leftarrow$ compute accuracy of m_{new} on $D_{validation}$
return $m, D_{selected}$

to become more trustworthy as the iterations progress, since it has more data available for training. Following this logic, we observe that during the early stages of this algorithm it is easier for incorrect samples to be introduced into the dataset. Furthermore, as previously discussed, if many similar incorrect samples exist in D_{auto}, a CNN trained on a few of them can end up memorizing them. Subsequently, it has a greater probability of introducing an increasing number of similar incorrect samples in its dataset as the iterations progress.

Motivated by these observations, we can modify algorithm G-IDEA so that on each iteration only a portion of the data where the predictions agree with the automatic labeling is included in the training set. To select this portion, we assume that the CNN outputs K numbers representing the likelihoods of a sample belonging in each of the K classes, more specifically log-likelihoods, as detailed in section 3.5. For each of the classes we can then order the predicted samples by decreasing likelihood. Intuitively, we are ordering the samples by a measure of how certain the CNN is of its prediction. We can then select only a conservative percentage of each class' data, and then gradually increase this percentage as the iterations progress. We shall call this modification "Conservative Iterative Dataset Expansion Algorithm", or "C-IDEA", as described in algorithm C-IDEA.

Note that both of these algorithms are generic in nature, and could be applied to any classification problem. In this work, we evaluate their use on the problem of hand posture recognition from RGB images. The following sections of chapter 3 are concerned with the details of devising an automatic annotation scheme for this problem, and with the application of these methods on the resulting data. Chapter 4 is concerned with evaluating and comparing the effectiveness of the algorithms presented in this section.

3.2 Hand posture recognition - problem definition

As mentioned, the algorithms of the previous section are applicable in any classification problem. For the work presented here, we choose to apply and evaluate them on the problem of hand posture recognition from RGB images. Given a single RGB image of a human hand, which is performing one of K different postures, the task here is to output a label indicating which of the K postures appears on the image.

Experimental evaluation of the proposed techniques requires us to specify a set of hand postures which we are interested in recognizing. Motivated by the general problem of Sign Language Recognition, we apply our methods to a set of hand postures that convey semantic information in Greek Sign Language (GSL). Although recognition and translation of GSL cannot be performed with hand posture information only, the configuration of each hand can serve as a useful feature in the general task. Furthermore, for the case of finger spelling, a number of different hand postures represent the letters of the alphabet [65]. Therefore, hand posture recognition can potentially be used as a module of a system which aims to recognize

Algorithm C-IDEA Train a CNN on a combination of manually and automatically annotated data, attempting to avoid noisy samples

Input:
D_{manual}, a set of manually annotated samples
D_{auto}, a set of automatically annotated samples
$D_{validation}$, a set of manually annotated samples to be used as a validation set
$selection\_percent$, initial percentage of training data to be selected per class
$increase\_factor$, factor by which to increase selection percentage on each iteration
Output:
m, a CNN model trained on the input data
$D_{selected}$, the corresponding training data

$m_{new} \leftarrow$ train a CNN on D_{manual}
$D_{selected} \leftarrow D_{manual}$
$val\_acc \leftarrow$ compute accuracy of m on $D_{validation}$
$prev\_acc \leftarrow -\infty$
$D_0 \leftarrow D_{manual}$
$i \leftarrow 0$
while $val\_acc > prev\_acc$ **do**
  $i \leftarrow i + 1$
  $m \leftarrow m_{new}$
  $D_{selected} \leftarrow D_i$
  $new\_data \leftarrow$ all samples of $D_{auto} \setminus D_{i-1}$ where the class output of m agrees with the label given in D_{auto}
  $l \leftarrow$ likelihoods of each sample belonging in the predicted class
  $sel\_data \leftarrow <>$
  for each class c of all K classes **do**
    $class\_data \leftarrow$ all samples of new_data whose class is c
    $l_{class} \leftarrow$ corresponding likelihoods of l for $class\_data$
    $class\_data \leftarrow$ permute $class\_data$ by the descending order of l_{class}
    add the first $\lfloor selection\_percent \cdot size(class\_data) \rfloor$ elements of $class\_data$ to sel_data
  $D_i \leftarrow D_{i-1} \cup sel\_data$
  $m_{new} \leftarrow$ train a CNN on D_i
  $prev\_acc \leftarrow val\_acc$
  $val\_acc \leftarrow$ compute accuracy of m_{new} on $D_{validation}$
  $selection\_percent \leftarrow \min(1, selection\_percent \cdot increase\_factor)$
return $m, D_{selected}$

Figure 3.1: Three example frames from the HealthSign dataset, featuring three different signers.

sign language.

3.3 The HealthSign dataset

The initial basis for determining the postures we would like our system to recognize was the HealthSign dataset, which is detailed in the work of Kosmopoulos et al. [48]. This is a dataset focused on communication of deaf patients with mental health professionals. Namely, it features 21 different scripts in GSL, in which a deaf patient communicates with a health professional about issues related to stress and depression. Eight different speakers (four male and four female) of GSL enact these scenarios. Figure 3.1 shows three example frames from the dataset. The dataset features gloss-level annotation, which means that for every video there exists a mapping $(f_{start}, f_{end}) \rightarrow W$, where f_{start}, f_{end} are the initial and final frames of a segment of each video respectively, and W is the Greek word to which the signing sequence of this segment corresponds.

Although GSL is the primary means of communication of deaf individuals in Greece, there exists no definitive formal documentation of it [2]. It is estimated that there are 54 different hand postures in use today. Better documented is the case of the postures which comprise the fingerspelling alphabet: 24 hand postures are used, one for each of the letters of the Greek alphabet [64]. The frequency of use for these postures varies, i.e. some postures are used in many more glosses than others, as is common in other sign languages [10, 59] (see also Zipf's law [101, 92]).

From an analysis of the 450 most common glosses in the HealthSign dataset (see 3.6.1), we found 38 postures being used at least once. From these 38 postures, we selected 19, for which the process described in 3.6.1 had yielded some initial, manually annotated data more easily. In figure 3.2, one can see the selected postures. All of the work described from this section onward was performed with these postures acting as the classes of the hand posture classification problem. Note, however, that the general methodology being presented can be used on any other hand posture classification problem. The only factor affecting our choice of postures was data availability.

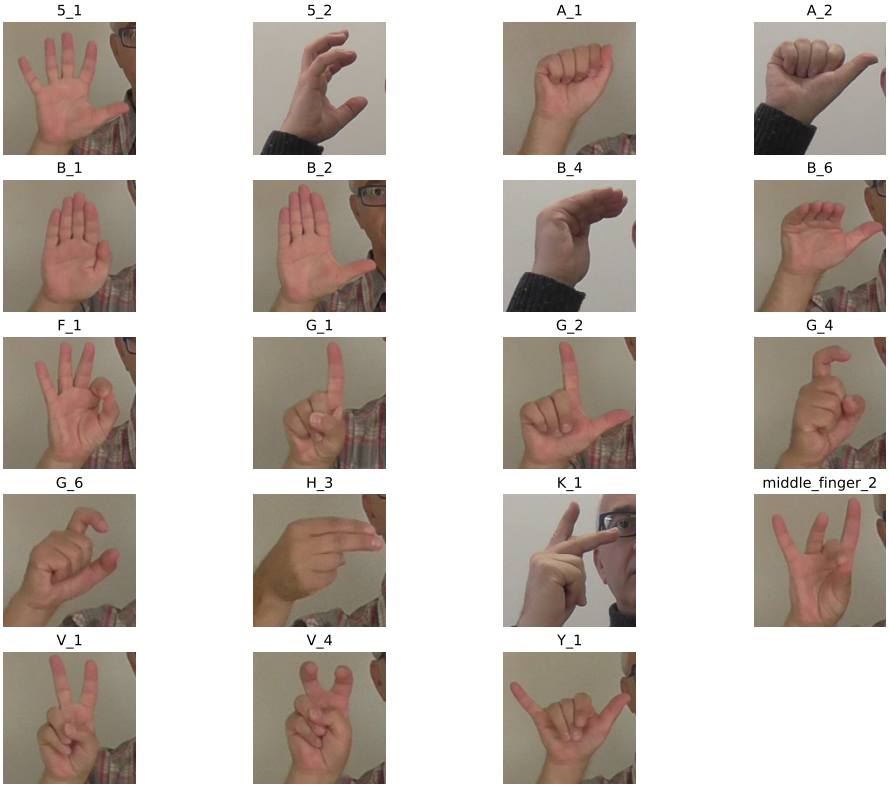

Figure 3.2: illustration of the hand postures employed in the considered hand posture classification problem.

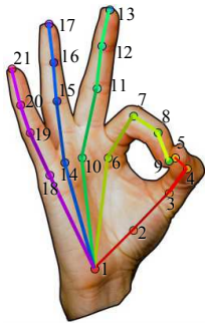

Figure 3.3: Locations of the 21 joints determining the articulated structure of a human hand (figure from [82]).

3.4 Automatic ground truth extraction

The available data is in the form of RGB videos or images. There is no frame-level annotations that indicates the hand postures present on each frame. Producing this information manually would require substantial human effort, and thus it appears imperative to extract this information in an automatic manner. Our approach is based on extracting the 3-D keypoint structure of the hands in every frame, and assigning the frame a ground truth label by matching this structure to one of the postures we are interested in recognizing.

3.4.1 Measuring the distance of hand postures

In order to represent 3-D hand postures, we adopt a hand model commonly used in relevant literature [67], [97], [82]. This model consists of 21 keypoints that map to the joints of the palm and fingers. See figure 3.3 for an illustration of this scheme.

Assuming we are interested in recognizing K different classes (hand postures) and have N available frames, the first step in assigning one of K labels to any subset of the frames is extracting this 3-D structure for all classes and all frames. In our initial investigations, we utilized the method developed by Panteleris et al. [67], for extracting 3-D keypoints from monocular RGB videos. This method uses OpenPose [15] for estimating the 2-D hand joints, and fits a three-dimensional hand model on these estimations through non-linear least-squares minimization. Ideally, for this method to yield accurate results, precise camera calibration information should be available. The estimation results were deemed inaccurate to be relied upon for matching frames to classes, possibly due to the lack of the required calibration information. Therefore, we used MediaPipe Hands [97], a software developed by Google capable of extracting 2.5-D hand landmarks from RGB images. When using the term "2.5-D landmarks", we refer to a representation where the depth coordinate is relative. By a qualitative comparison of the results, this method seemed to perform more accurately than [67] for this task.

Figure 3.4: Greek letters "H", "Z", "Π″ (left to right). An example of postures that our system treats as being in the same class, by design.

Given an input image of dimensions $W \times H$, the scheme [55] utilized by Mediapipe represents each hand posture as a 21-tuple of 3-tuples,

$$P = ((x_1, y_1, z_1), (x_2, y_2, z_2), \ldots, (x_{21}, y_{21}, z_{21})).$$

All x_i, y_i are in the range $[0, 1]$, and $x_i W$, $y_i H$ equal the horizontal and vertical pixel coordinates of landmark i in the image, respectively. Furthermore, z_i represents the relative depth of landmark i: the wrist joint (1, in figure 3.3) by convention is positioned at depth 0, and the depth of the landmarks increases monotonically with their distance from the camera. This is, then, a scheme where the z coordinate is a relative representation of depth, and the magnitude of the x, y coordinates depends on the dimensions of each individual hand, and the overall distance of the hand from the camera. The aforementioned conventions followed by Mediapipe imply that an estimation for a 3-D hand posture P can be located anywhere in three-dimensional space, and be characterized by an arbitrary orientation. At this point, a comparison of two different hands cannot be achieved by directly comparing corresponding joints. Therefore, the first step towards rendering different postures comparable is to translate them to a common point in space, such as the origin, and rotate them to a common orientation.

For this work, we make the simplifying assumption that no two classes can differ from one another solely by a rotation of all their landmarks. This means, for example, that "thumbs-up" and "thumbs-down" are considered identical postures. For the case of Greek Sign Language this is not always true: the Greek letters "H", "Z", "Π″ for example, only differ by an in-plane rotational change (see figure 3.4). However, including such cases would not alter the proposed methodology significantly. In fact, most of the necessary changes would only affect the posture preprocessing described in this section.

We deal with the task of translation and rotation normalization in the following way. Let $P = (j_0, j_1, \ldots, j_{21})$ be a hand posture in an image of dimensions $W \times H$ as described previously, with $j_i = (x_i, y_i, z_i)$ corresponding to the i-th landmark as depicted in figure 3.3. Then, $P^{'} = (j_0 - j_0, j_1 - j_0, \ldots, j_{21} - j_0)$ is the same posture translated so that the wrist lies at the origin. Let $n_1 = \frac{j_{10} \times j_6}{\|j_{10} \times j_6\|}$ be a vector normal

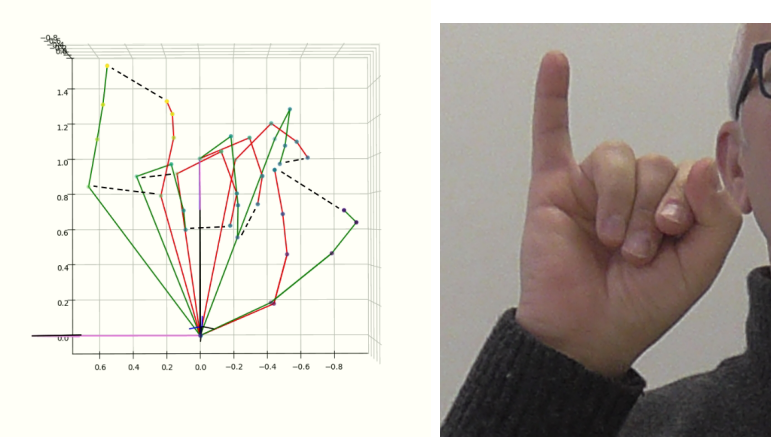

Figure 3.5: Two different hands performing approximately the same posture, shown on the right. The dotted black lines connect corresponding base and tip joints of the hands' fingers.

to the plane defined by the middle and index finger base joints, and $n_2 = \frac{j_{10} \times n_1}{\|j_{10} \times n_1\|}$ be a vector that lies in this plane and is perpendicular to j_{10}. To illustrate, n_2 approximates the thumb when it is extended, and lying on the palm plane. Then, $n_2, \frac{j_{10}}{\|j_{10}\|}, n_1$ can be thought of as the i, j, k unit vectors of a local coordinate system where the middle finger lies on the positive $y^{'}y$ semi-axis, and the normal vector n_1 lies on the positive $z^{'}z$ semi-axis. Let $R \in \mathbf{R^3}$ be the rotation matrix that transforms this coordinate system to the standard basis of $\mathbf{R^3}$, and $P^{''}$ be the hand posture that results from applying R to $P^{'}$. By transforming every posture P to its corresponding $P^{''}$, we ensure that all postures are now at a common point in space, and have the same orientation. It is now meaningful to define the distance of two hand postures, $P_1 = (j_{1,1}, j_{1,2}, \ldots, j_{1,21}), P_2 = (j_{2,1}, j_{2,2}, \ldots, j_{2,21})$ as

$$d(P_1, P_2) = \sum_{k=1}^{21} \|j_{1,k} - j_{2,k}\|, \tag{3.1}$$

that is, as the sum of the Euclidean distances of all corresponding joints. However, the characteristics of each individual hand (e.g. finger lengths) can affect this sum in an undesirable way: two different hands performing the same posture may end up having a distance significantly larger than zero if their anatomical structures are different. An example of this can be seen in figure 3.5: the three-dimensional representations of both the green and red hand are performing approximately the same posture, shown on the right image, and yet differences in their scale will result in the metric of equation 3.1 having a larger value than expected.

Therefore, it is necessary to normalize, as much as possible, the individual hand characteristics. To this end, we chose a predefined hand model, $H_0 = (j_{0,1}, j_{0,2}, \ldots, j_{0,21})$, and applied the following transformation on all postures, in order to match their structure with H_0. First, observe that the the wrist joint and

the base joints of all fingers and thumb can be considered approximately rigid in the human hand. Therefore, information about a posture $P^{''} = (j_1^{''}, j_2^{''}, \ldots, j_{21}^{''})$ is not lost if each finger is transformed so that each base joint is aligned with the corresponding base joint of H_0.

This can be achieved by setting $j_2 = j_{0,2}, j_6 = j_{0,6}, j_{10} = j_{0,10}, j_{14} = j_{0,14}, j_{18} = j_{0,18}$ (changing $P^{''}$'s palm structure to match H_0's), and then translating each finger to its new base joint. Furthermore, we want every "bone" (that is, every segment connecting two joints in the scheme of figure 3.3) to have magnitude equal to its corresponding bone in H_0. For example, the bone connecting joints 10 and 11 has magnitude $\|j_{0,10} - j_{0,11}\|$ in H_0, and $\|j_{10}^{''} - j_{11}^{''}\|$ in $P^{''}$. Again, this does not affect posture information, since it is only the hand's dimensions that are being changed. The entirety of this transformation for the middle finger, comprised of joints 10, 11, 12, 13, is described by the following recurrence relation:

$$\begin{cases} j_k^{'''} = j_{0,k}^{''} & k = 10 \\ j_k^{'''} = j_{k-1}^{'''} + \frac{(j_k^{''} - j_{k-1}^{''})}{\|j_k^{''} - j_{k-1}^{''}\|} \|j_{0,k} - j_{0,k-1}\| & k = 11, 12, 13 \end{cases}, \tag{3.2}$$

effectively transforming the bone lengths to the ones of H_0.

Similar relations can be defined for all other fingers. At this point, we have eliminated, to a large extent, the differences that arise from variability in hand structure and dimensions. Denoting with F the composition of all transformations described in the paragraph, which maps every hand posture P to its corresponding $P^{'''}$, postures $F(P_1), F(P_2)$ can be reliably compared with the metric of equation 3.1, for any P_1, P_2. See figure 3.6 for an illustration of how the transformation renders the postures of figure 3.5 comparable.

With respect to the distance metric that should be used to compare postures, it is worth noting that besides the metric of equation 3.1 we experimented with the alternative metric defined as

$$d^{'}(P_1, P_2) = \max_k \|j_{1,k} - j_{2,k}\|, \tag{3.3}$$

which is based on the infinity norm. However, preliminary experiments similar to those presented in Section 4 indicated that it did not provide an advantage.

3.4.2 Determining the validity and usefulness of each frame

Having developed a method for comparing postures present in each frame to the postures that correspond to the classes of the problem, we must now determine a criterion for determining whether the posture estimated in a frame is actually correct. The need for this arises due to two reasons. Firstly, in the videos of the HealthSign dataset, a signer signing GSL may not necessarily be signing for the entirety of the video. In the case of HealthSign, for example, the signers were following scripts describing the sentences they should sign, and as such there were significant intervals where the signers were still, waiting for the next sentence to be

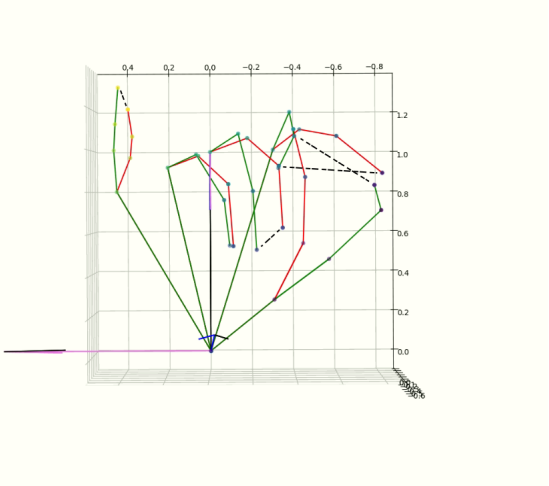

Figure 3.6: Applying all transformations to the postures of figure 3.5. Corresponding palm joints now coincide, and corresponding finger bones have the same length. Again, dotted black lines connect corresponding fingertips.

recited to them. Additionally, in the general case of signing GSL the non-dominant hand is frequently held still, since many signs feature only the dominant hand. Postures estimated for those frames (commonly referred to as neutral postures) likely do not hold meaning in GSL, and should not be taken into account. Secondly, the method that is used for estimating the 3-D landmarks could result in occasional false detections. In the case of our method of choice, MediaPipe [97], the estimator provides a confidence score in the range $[0, 1]$ for every detection it reports, and thus this information can be useful in isolating postures that are more likely to be valid. For this confidence score, a threshold in the range of $[0.8, 0.9]$ was empirically found to be adequate, i.e. it did not discard many true detections, nor did it select many false ones.

For detecting frames where the signer is still, a useful observation is that, in general, sign language tends to be signed relatively close to the face and torso [51], whereas when a signer is idling, their hands are more likely to be closer to their pelvis. This was true for quite a few cases in HealthSign videos, but not all (see figure 3.7). It was therefore found that, in practice, rejecting all postures where the mean of all 2-D landmarks of the hand lied in the lower 20% of the frame is an effective criterion for detecting cases where the signer's hand is still. In order to also detect cases of signers whose stance when still resembles that of the signer on the right of figure 3.7 (i.e. they rest their hands near their torso, an area in which they could well be signing), a slightly more sophisticated approach is needed.

First, let $\overline{l_i} = (\overline{x_{i,k}}, \overline{y_{i,k}})$ be the mean 2-D landmark of a signer's hand (e.g. his right one) in pixel coordinates for the i-th frame of a video (k refers to the 21 landmarks). Then, let $\Delta x_i = \overline{l_{i,1}} - \overline{l_{i-1,1}}, \Delta y_i = \overline{l_{i,2}} - \overline{l_{i-1,2}}$ be the difference of this mean landmark in x, y coordinates from frame $i-1$ to frame i. The quantity

Figure 3.7: Two examples of frames where signers are idle. Observe that the right signer maintains his hands near the area where signing predominantly takes place.

$\Delta l_i = \sqrt{\Delta x_i^2 + \Delta y_i^2}$ is the pixel distance traversed by the mean landmark between the two frames. The hand of frame i is then judged to be still if $\frac{\sum_{k=i-w}^{i+w} \Delta l_k}{2w+1} < T$, i.e. when the mean value of the distance traversed in each frame in a window around frame i is less than some threshold. In practice, values of $w = 0.75 \cdot framerate$, i.e. so that the window corresponds to $0.75 \cdot 2 = 1.5$ seconds, and $T = 0.7$ appear to adequately detect segments where the signers are still adequately.

The logical conjunction of score thresholding and stillness status (each described in the paragraphs above) yields a criterion for determining whether a particular hand in a particular frame can be considered as being in valid posture. This criterion shall be referred to as C in all sections that follow.

3.4.3 Assigning class labels to postures

At this point, we have established a way of comparing every hand posture of every frame with the postures that correspond to the classes of our problem. Furthermore, we can determine which hand postures should not be taken into account for further processing.

To restate the problem, we are given K input images, each corresponding to a particular hand posture (that is, to a class of the problem) and a set of N videos, every one of which features one of S subjects performing these hand postures. The end goal is to select a subset of the frames that can be useful in forming a part of the training dataset of the classifier we are designing. Note that this dataset can also feature manually annotated samples. This classifier should receive as input an RGB image of a hand, and output one of K labels, indicating which hand posture is present in the image. In our particular case, the postures hold meaning in GSL, but the problem could be defined over an arbitrary set of hand postures.

It is desirable to create a classifier that is invariant to human-specific characteristics of the hand, such as skin color. Therefore, it is crucial to populate the classes of the dataset with samples that feature enough diversity in such characteristics. This will allow the classifier to better generalize over images not present in the training set, by only focusing on relevant features.

As such, the frame subset that is chosen should feature hand postures that are similar to the problem's class postures. Also, it should, as much as possible, consist of frames with diversity in all image features irrelevant to the classification problem. To this end, we apply algorithm ASSIGN_LABELS for choosing the frame subset and assigning a class label to every one of its members. The pseudocode assumes that the normalization operation F of section 3.4.1, the filtering criterion C of section 3.4.2, as well as the MediaPipe estimator mentioned in section 3.4.1 are all callable functions. Furthermore, besides returning 3-D postures, MediaPipe also returns a confidence score for each detection, which can be used by C. For presentation clarity, handedness information is not handled in the pseudocode, i.e. we only process the signer's right hand. Also, the pseudocode assumes that the list of videos it operates on feature only one signer. In practice, this algorithm is executed for all signers in the dataset.

In an actual implementation, it is preferable to separate this algorithm in steps which can be performed independently. The first step is the extraction of all frames from the videos, and the estimation of 3-D landmarks via MediaPipe or any other estimator. At this point, the results are saved as arrays containing information about the 3-D points, the corresponding frames, videos and handedness. Then, normalization and filtering can be applied at a later time, potentially using some other filtering method that the one described in 3.4.2, if the problem calls for it. As a third step, the distances of all estimated postures from all reference postures are calculated (with the metric of equation 3.1, or any other), and also saved as arrays. Finally, the results of the filtering process and the distance calculations are combined in order to obtain a sorting of the postures in the classes, and then select a predefined number of postures per class.This information is then used to crop the proper part of each frame that contains the hand, and store the cropped images in the actual dataset, separated by class.

3.5 Classifier architecture

At this point, we are capable of creating a training set of annotated RGB images of hand postures, simply by providing the system we described in 3.4 with videos of various signers performing any desired set of postures. Furthermore, we have in our disposal the methods presented in section 3.1, for utilization of automatically annotated data. We can, therefore, train CNNs for hand posture recognition by using either manually annotated data, automatically annotated data, or a combination of the two. In this section, we examine the architecture we shall be using for our CNN.

Algorithm ASSIGN_LABELS Extract a set of images featuring the problem's specified hand posture classes

Input:
V, a list of videos featuring a uniquely identified signer
ref_images, a list of K images where a signer is signing each one of K reference postures (classes) with their right hand
N, integer denoting the maximum number of images to be selected for every class
Output:
$selected\_images$, a list of tuples of the form (image, posture class)

$ref\_poses\_3d \leftarrow$ MediaPipe(ref_images)
$ref\_poses\_3d \leftarrow F(ref\_poses\_3d)$
$selected\_images \leftarrow <$ empty list $>$
$images\_per\_class \leftarrow <$ empty list $>$
for $i = 1 \ldots K$ **do**
  $images\_per\_class[i] \leftarrow <$ empty list $>$
$estim\_3d \leftarrow$ MediaPipe(V)
$normalized\_3d \leftarrow F(estim\_3d)$
$is\_valid\_frame \leftarrow C(estim\_3d)$
for every index i where $is\_valid\_frame[i] ==$ **true do**
  $nearest\_class \leftarrow \underset{k}{\text{argmin}}(d(normalized\_3d[i], ref\_poses\_3d[k]))$
  $distance \leftarrow \min_{k}(d(normalized\_3d[i], ref\_poses\_3d[k]))$
  $cropped\_img \leftarrow$ crop corresponding frame around 2-D landmarks
  $annotation\_tuple \leftarrow (cropped\_img, distance)$
  append $annotation\_tuple$ to $images\_per\_class[nearest\_class]$
for $i = 1 : K$ **do**
  sort $images\_per\_class[i]$ in ascending order of $distance$
  append first $\min(N, images\_per\_class[i].length)$ elements of $images\_per\_class$ to $selected\_images$
return $selected\_images$

As was mentioned in the introductory chapter, it is desirable for the classifier to be fairly lightweight in terms of the computational power it requires. In the general domain of image recognition, deep convolutional neural networks have proven to exhibit high accuracies and generalization capabilities [50]. Furthermore, they can be trained end-to-end, with their only input being RGB images and their corresponding classes. In particular, the MobileNet v2 network [77] is a classifier of this kind that can be used in real-time conditions on smartphones, while also achieving high accuracy (e.g. on the ImageNet dataset [24]).

Therefore, a MobileNet network is a reasonable choice for a problem like the one tackled here. In particular, we use a MobileNet v2 model that has already been trained on the ImageNet dataset. The intuition behind using a pretrained model is that the low-level image features it has learned to recognize (e.g. edges, corners) will also be useful in our problem. In order to adapt the model to our classification problem, we add a fully connected linear layer at the end of the standard MobileNet architecture. Each of its neurons has as inputs all outputs of the final layer of the MobileNet architecture, and there are as many neurons as the K classes of the problem. Finally, the K outputs $(x_1, x_2, \ldots, x_K)$ are fed through a log-softmax layer which outputs K values as:

$$y_i = \log \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}} \tag{3.4}$$

y_i is a sort of measurement of how "confident" the classifier is of its input image belonging in class i. For the final classification result, the class i which maximizes y_i is chosen as the class of the input image.

During training, the objective function that we attempt to minimize is the negative log likelihood loss function, since this is a classification problem:

$$l(y, actual\_class) = \sum_{n=1}^{N} -y_{n,actual\_class_n}, \tag{3.5}$$

where N is the batch size, y is a $N \times K$ matrix the i-th row of which is equal to the softmax layer output for sample image i, and $actual\_class$ is a vector of size N whose i-th element is the ground truth class index of sample image i.

3.6 The role of automatic ground truth

One way to evaluate the effectiveness of our automatic annotation schemes would be to compare performances of classifiers trained on different datasets, created with manual annotation, automatic annotation, or a combination of the two. At first glance, achieving a better performance than a manually annotated dataset appears unlikely, provided of course that the manually annotated samples are sufficient in number for generalization. After all, human annotation, when performed with care, sets a very high standard for any automatic method.

Note, however, that as the number of postures increases, the effort required to annotate a dataset manually becomes non-trivial. This is especially true because enough sample variability must be ensured for the generalization of the classifier. Therefore, it is likely that an automatic annotation scheme could prove useful, or even necessary, as a tool for augmenting a small, manually annotated dataset. It would provide more samples, automatically annotated, thus improving the generalization of a classifier trained on the dataset.

3.6.1 Manually annotated training data

To establish a baseline for judging automatic annotation schemes, we created a small, manually annotated dataset from the videos of the HealthSign dataset. For the 8 signers featured in this dataset, a target goal of selecting approximately 15 images per hand posture of every signer was set. This would result in approximately 120 images in total for every hand posture. Since per-frame annotation for hand postures was unavailable, even for rather small numbers of images per class like these, the task required a fair amount of human effort. The only available annotation was gloss-level, and it was utilized by making the following observation. For every gloss, the annotation ensures that every signer signs it in the same fashion, i.e. with the same sequence of hand postures signed with the dominant, and possibly non-dominant, hand. Therefore, the frames of a video segment corresponding to gloss X can only feature the postures that comprise S. With this observation, the frame-level annotation effort was performed as follows:

- First, we performed an analysis of the frequency with which each gloss appears in the dataset. We found that approximately the 450 most commonly appearing glosses cover about 90% of all gloss instances. Since most glosses have a rather short duration (a few seconds), we can assume that this also covers about 90% of all frames.

- Then, we map each one of these glosses to its corresponding postures in GSL. This mapping is a time consuming task, regardless if it is done via consulting a sign language interpreter or via inspecting the corresponding video parts and matching the postures with those provided in a compiled reference of GSL postures.

- For every frame present in a video segment featuring a gloss mapped in the previous step, we utilize MediaPipe's hand posture estimations to determine which of the gloss' postures is the closest match for the hand of that frame. Essentially, this is a modified version of the algorithm presented in section 3.4.3, where the number of possible postures a frame can be matched with is limited.

- Finally, we group all cropped hand images by the class posture to which they were assigned, and sort them in ascending order of their posture's distance from its assigned class. Then, human effort is required once again, in order

to inspect some of the selected images, and select 15 images for every class of every signer, ensuring -to the degree possible- that they are not too similar to one another.

From the description of the procedure, it becomes clear that a significant amount of effort is required for manually annotating even a small set of images. An alternative would be the creation of data specifically tailored for this problem, i.e. photographing a number of people signing all different postures, thus knowing a priori the label of each image. This is also a procedure that requires a fair amount of human effort. Furthermore, for the general problem of sign language recognition, this implies the need for additional annotated frames besides the gloss-annotated videos.

Having created the manually annotated dataset, the first experiments to gauge the accuracy achieved by a classifier trained on it were performed with cross validation. Specifically, since the dataset is comprised of eight different signers, we used 8-fold cross validation, where in each fold one of the signers was kept as a validation set, and the other seven as a training set. The classifier managed to achieve an average accuracy of 90% across folds, with no per-class accuracy being lower than 80%.

However, before proceeding to compare with classifiers that use automatically annotated data in any way, we first tested the performance of this particular trained model on the validation set we created (described in 4.1). There, the classifier exhibited a significant decrease in accuracy: the recorded average validation accuracy was approximately 45%, with the lowest per-class accuracy being approximately 15%. Similar accuracies were observed on the test set as well.

Clearly, despite the high accuracies estimated by cross-validation, and despite the data augmentation procedure, this classifier has overfitted on the characteristics of the HealthSign dataset. We speculate that this is due to the fact that all videos were filmed in the same conditions, i.e. in the same room, with the same background and lighting. Although we attempted to alleviate the problem with the augmentations described in 4.2.1, the efforts were not enough to ensure generalization. Therefore, the problem appears to be a good candidate for testing the effects of automatically annotating a potentially much greater amount of data, in the hope of achieving generalization.

3.6.2 Gathering and automatically annotating more data

Due to the apparent need for more training data, we proceeded to gather video data from sources other than HealthSign. Firstly, we enlisted the help of 10 subjects, both male and female, each one of which was asked to record a short video (approximately 10 to 12 minutes). They are asked to perform all of the problem's hand postures in various rotations of the hands, and in an arbitrary environment. Furthermore, they are asked to perform the postures in front of their face and body, as well as in front of the background. It should be noted that these subjects did

Figure 3.8: Example of an error in 3-D pose estimation. Note that the little finger has not been detected as being extended.

not know GSL, but instead performed the postures by following an example video of a person iterating through all of them. Data collection is, therefore, fairly simple in terms of human effort. Secondly, we gathered several videos of signers signing in GSL from YouTube (35 videos of 7 signers). Again, the only human effort involved here is a selection of videos that are deemed appropriate, e.g. have an acceptable image quality.

All of this data is then processed with the system of 3.4, in order to extract a ground truth label for a number of different frames. For the algorithm of 3.4.3, we use a value of $N = 500$, meaning that we extract at most 500 hand images for every class of every signer. Due to the number of different signers, this creates a large pool of data available for training our classifier.

The simplest way to utilize this large dataset is to consolidate it, either in its entirety, or by further limiting the number of images per signer class, with the manually annotated dataset described in 3.6.1, and use the result to train a new model. This allows for much greater variation for each class, and would provide the network with a large number of training examples from which to extract classification features. However, it is at this point that the accuracy of the 3-D hand posture estimation method comes into play. MediaPipe's estimations may be fairly accurate, but they are not always perfect, and the more nuanced the differences between class postures become, the more important estimation accuracy becomes for the creation of an effective training set. This is due to the fact that in such cases, it becomes easier for training samples of one class to be mislabeled as training samples of another. Should this happen to a significant degree, the network may very well have trouble differentiating between the two classes, even if it manages to extract features relevant to classification. This is purely due to the fact that the training set itself does not properly encode class differences.

One example of error that MediaPipe seems to repeat can be seen in figure 3.8.

Specifically, at certain points during tracking, it appears to have difficulty estimating the position of the hand's little finger. Our problem features two classes, "A_2" and "Y_1" in figure 3.2, whose only differences are found in the joints of the little finger. As a result, the automatic annotation of 3.4 may well result in a training set where sample images from one of these classes are labeled as belonging to the other. This and other similar cases, which stem from the fact that quite a few classes in our GSL-defined problem are characterized by rather small differences in hand posture, indicate that MediaPipe's annotations cannot be completely trusted.

Therefore, this is a case that resembles the general task which is analyzed in section 3.1: we have a classification problem for which a small number of manually annotated samples is available. Also available is a large number of automatically annotated samples, for which the produced ground truth contains noise, as described above. We can therefore test the two algorithms presented in section 3.1, and compare the resulting classifiers with a classifier trained without the use of such a technique.

Chapter 4

Experimental evaluation

This chapter presents an evaluation of our proposed methods applied to the problem of hand posture recognition from RGB images. We begin by describing the datasets that are used for model selection (validation set) and performance estimation (test set). We then present the details of the training procedure, with respect to the network's hyperparameters and data augmentation process. Finally, we compare the performance of CNNs trained with and without the use of the algorithms of section 3.1.

4.1 Creation of validation and test sets

To the best of our knowledge, as already outlined in section 2.3, there is no large corpus of annotated images for GSL postures. Therefore, we found it necessary to manually annotate a small dataset for testing our algorithms. It is considered good practice to evaluate a classifier's accuracy on data that have been kept separate from its training set. To this end, the validation set which we created features a male signer not present in any of the training videos, performing all hand postures in a room which the classifier has also never encountered in its training samples. To further attempt to diversify the validation set, the hand postures are performed in front of the signer's body, face and the room in which he is located. Various rotations and viewpoints were also recorded for each posture. Ideally, the validation set would be even more diverse, by featuring data from many different signers. However, the variability that already exists in the aforementioned factors, combined with the disjoint nature of training and validation sets, provides a fairly reliable estimation of generalization capabilities.

More specifically, for each of the 19 postures we have chosen to include in our problem definition, approximately 100 images of a right hand performing them were recorded, in the conditions described above. See figure 4.1 for example images belonging in one class, showcasing some of the variability we have tried to incorporate in our dataset. With respect to the classifier, we chose to train it for right hands only, in order to slightly simplify its task. For this reason, all images in

Figure 4.1: Three examples of images included in the validation set for the posture "Y". Note differences in background, lighting and rotation.

the validation set feature right hands only. If detection of hand posture is desirable for a left hand, the image simply has to be horizontally flipped before being given as input to the classifier. Therefore, in a real-world scenario one must only be aware of whether each image they would like to use the classifier on comes from a left or right hand.

Because we use the accuracy achieved on the validation set as a guideline for choosing the best model during a training session, we run the risk of overestimating the resulting classifier's accuracy. To this end, we also created a test set, which is never observed in any way during training. Accuracies on the test set are only measured after determining the best model based on validation accuracy. The created test set features images predominantly from three different signers, two male and one female. A small number of samples are also sourced from three other signers, one male and two female. This dataset features lower variability in rotations, but higher variability in signer-specific characteristics, e.g. skin tone. Furthermore, it features higher variability in image resolution. Figure 4.2 shows three example images for posture "Y" from the test set, each from a different signer. As with the validation set, approximately 100 right hand images comprise each class.

4.2 Training details

The CNN architecture we employ is described in section 3.5. For feeding images to the network, we use a batch size of 64 during training. For iteratively adjusting the parameters of the network to minimize the negative log-likelihood loss, an optimizing strategy must be chosen. We tried both the AdaDelta [96] and Adam [43] optimizers, with the PyTorch [68] default learning rates 1 and 10^{-3} respectively, and preliminary experiments determined that the Adam optimizer generalized marginally better. The experiments detailed in this chapter were all performed with the Adam optimizer.

In every experiment presented in this chapter, the corresponding CNN was trained for 120 epochs. At the end of each epoch, the accuracy of the classifier on the validation set was measured, and the final model for that experiment was chosen to be the one which achieved maximum validation accuracy.

Figure 4.2: Three examples of images included in the test set for the posture "Y", from three different signers.

4.2.1 Data augmentation

Applying augmentation techniques on the training data has proven to assist convolutional neural networks in generalizing to unseen samples [81]. Data augmentation can range from geometrical transforms of the input (e.g. rotating or scaling images) to transforms applied on the color profile (e.g. varying the contrast or hue) to more application-specific transforms.

In our case, the fact that the classes are rotationally invariant allows us to use a random rotation between 0° and 360° as augmentation. Before applying a rotation augmentation, input images are padded so that they are square, in order for the aspect ratio to be maintained when the image is rotated. Since the images fed to the network are already tightly cropped around the hand, we avoid using random cropping as augmentation, since it could result in obscuring parts of the image that contain useful information for classification. We do, however, use a random translation augmentation of up to 10% of the image's dimensions.

After applying all transforms, each image that is fed to the network is always scaled to be 224×224 pixels in size, and also normalized so that its pixel values have a mean of $\mu = (0.485, 0.456, 0.406)$ and standard deviation of $\sigma = (0.229, 0.224, 0.225)$ (values computed on the ImageNet dataset, and recommended for use with a pretrained MobileNet model). The same scaling to 224×224 and the same normalization is performed for images of the validation set. However, these do not undergo the random transformations applied on the training set, in order to ensure that performance estimation is always applied on exactly the same data.

For the case of training images that originate from the HealthSign dataset, we also apply one custom transformation before applying any of the others described above. Specifically, because the HealthSign videos were recorded in a room with a mostly green background, we can determine an approximate pixel value (r_{bg}, g_{bg}, b_{bg}) for the background color in an offline preprocessing step. Then, during training, we can perform a crude segmentation of the background by selecting those pixels with an RGB value (r, g, b) such that $\|(r, g, b) - (r_{bg}, g_{bg}, b_{bg})\| < T$, where T is a threshold. Assuming that each color value is in the range [0, 255], we found that a value of $T = 63$ serves well enough to select most of the background, while not selecting parts of the foreground.

Figure 4.3: Transforming the background of HealthSign images. On the left, one of the original images. On the right, the same image augmented with a random background

Having selected the background pixels, we then replace them with the corresponding pixels from an image randomly sampled from the Stanford backgrounds dataset [28]. Figure 4.3 shows the results of applying this transformation to an example image. We apply this transformation with a probability of 0.7 on any HealthSign training image that is fed to the network. This serves to provide greater variety to the HealthSign data, reducing the probability of overfitting on irrelevant features, such as learning to expect a green background around some or all hand postures.

4.3 Training on manually annotated data only

As mentioned in section 3.6.1, the first experiments on the validation and test set were performed with a CNN that was trained on manually annotated data, only. Details on the classifier architecture and the augmentation applied on the data can be found in sections 3.5 and 4.2, respectively.

In this case, the resulting CNN achieves an average accuracy of 46.5% on the validation set, and 41.5% on the test set. Figures 4.4 and 4.5 show the confusion matrices for the validation and test set, respectively. For these confusion matrices, as well as for every other confusion matrix presented in this chapter, the value of the i-th row and j-th column is the percentage

$$\frac{\text{number of samples of class i predicted as class j}}{\text{number of samples of class i}} \cdot 100\%$$

On its training set, the network achieves almost 100% accuracy. Therefore, from the average validation (46.5%) and test (41.5%) accuracies, as well as the characteristics of the confusion matrix, we observe that the network overfits: there is a large discrepancy between training and testing accuracies. Furthermore, the

Images per signer class	15	30	60	120	**500**
Validation accuracy	69.06%	71.33%	70.4%	72.3%	**73.38%**

Table 4.1: Validation accuracy achieved when using a variable number of images per signer class.

network tends to prefer some classes over others in many cases. This is especially clear in the case of the class "G_1". We speculate that the network has learned relatively simple features for these classes from the training set, which it can often detect in test images, and as a result it also ends up predicting these classes more frequently.

4.4 Combining manual and automatic data

Motivated by the lack of proper generalization, we begin to experiment with the use of automatically annotated data as a complement for our small set of manually annotated images. In the first experiment of this sort, we use algorithm ASSIGN_LABELS of section 3.4.3 on the data we describe in section 3.6.2. This is a large pool of samples from 17 signers. The aforementioned algorithm requires as a parameter the number of samples to select for every class of every signer. As we increase this number, if the automatic annotation was perfect we'd expect the accuracy to increase, since more training samples become available to the network. However, the fact that the automatic labeling is noisy leads to the conjecture that this increase may not be monotonic: an introduction of a large number of noisy samples could potentially affect prediction accuracy in an adverse manner. The effects of the noise could nullify or perhaps even outweigh the benefits of introducing new correct samples.

We experiment with several values for the number of samples per class for every signer, beginning with 15 images per signer class. By the term signer class we refer to all the images belonging in a particular class which feature the same signer. This number is chosen so that each new signer has a contribution equal to that of every signer from the original, manually annotated set (see section 3.6.1). Then, we experiment with the values of 30, 60, 120 and 500 images per signer class. Table 4.1 summarizes the highest validation accuracies recorded with each configuration.

We observe that, even with 15 images per signer class, there is a significant improvement (25%-30%) over using manually annotated data only. Furthermore, increasing the number of images per signer class is, in general, beneficial to validation accuracy. We record the highest validation accuracy of 73.38% when using 500 images per signer class. However, we also observe that the increase in accuracy is rather small in proportion to the increase in training set size. Considering that a large number of new, correctly labeled samples are introduced, one would expect a larger increase in accuracy. This hints that a significant amount of mislabeled samples are gradually being introduced in the dataset as well.

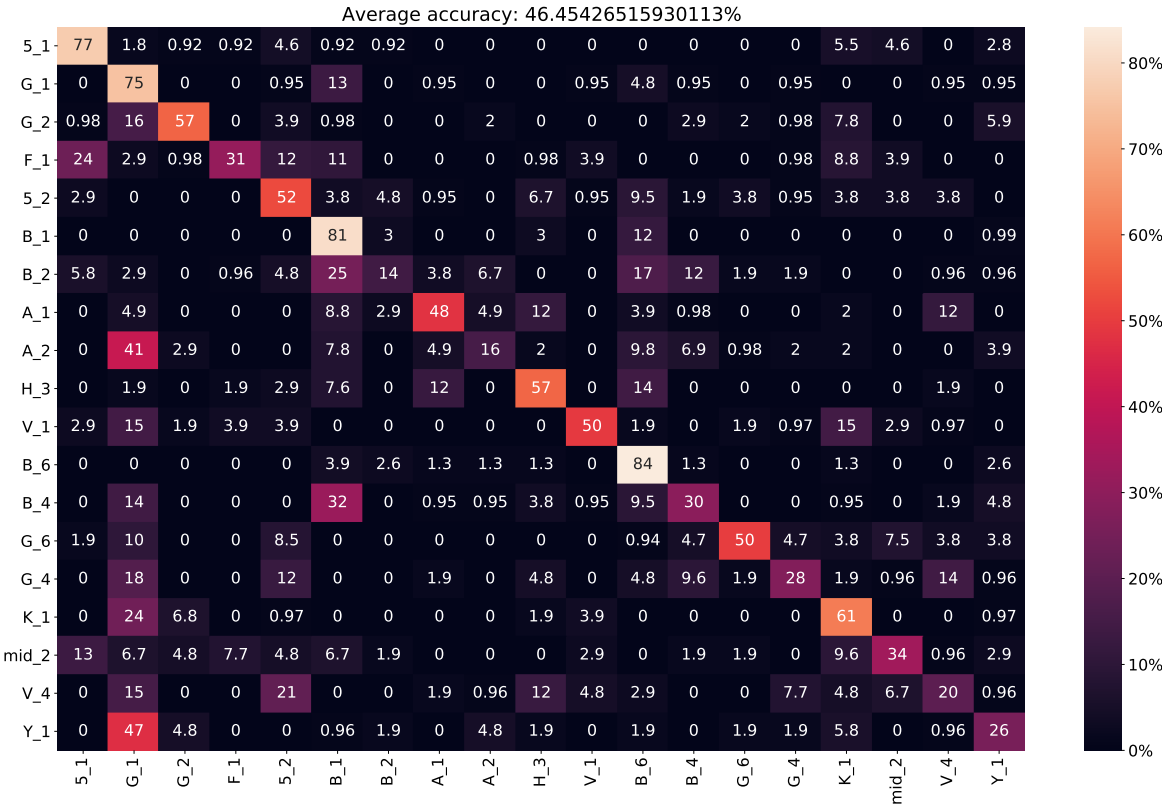

Figure 4.4: Validation set confusion matrix for the CNN trained on manually annotated data only.

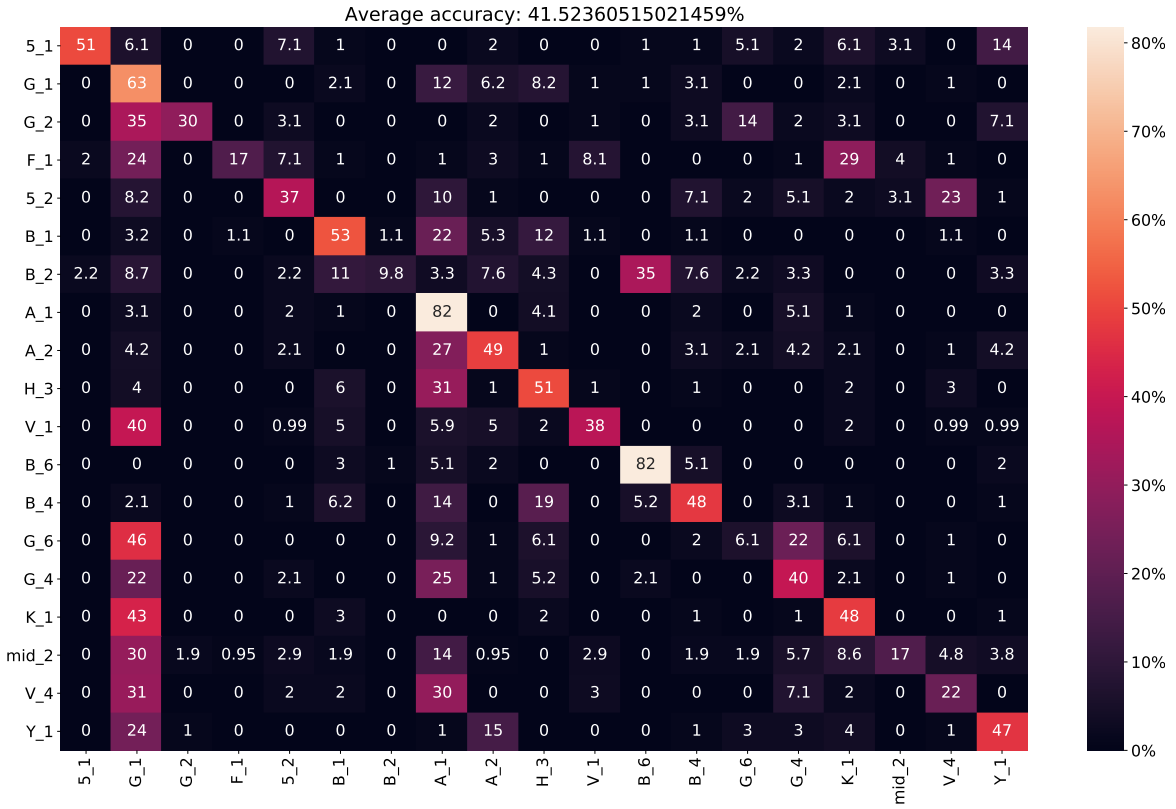

Figure 4.5: Test set confusion matrix for the CNN trained on manually annotated data only.

In figures 4.6 and 4.7, we can see the confusion matrices for the classifier that achieves maximum validation accuracy. Clearly, the addition of automatically annotated data greatly improves generalization. We observe an increase in the order of 30% both in validation and test accuracy. Furthermore, the confusion between classes is now much more localized, and reflects noise patterns present in the automatic labeling, as well as inherent similarities between classes.

As an example, in both the validation and test confusion matrix there is a significant percentage of samples that belong to class "A_2" for which the prediction is "A_1" or "Y_1". As mentioned in section 3.6.2, errors in 3-D tracking can often result in samples of "Y_1" being mislabeled as "A_2", or the opposite. For the case of confusion between "A_1" and "A_2", similar tracking problems can occur for the thumb joints (see figure 3.2). Another example is the pair of classes "G_4" and "G_6", which also differ only in the position of the thumb, and thus tracking errors can easily end up being the cause of mislabeled samples for these classes. Additionally, by visually inspecting some of the images that were automatically annotated, we confirmed that such cases of mislabeling are indeed relatively common.

4.5 Combining data with G-IDEA

Even though the classifier trained on both manually and automatically annotated data achieves a much higher average accuracy, we can still observe specific failure cases for some classes. These can be at least partially attributed to the noisy nature of the automatic ground truth. For this reason, we proceed to apply the techniques presented in section 3.1.

Therefore, we experiment with G-IDEA of section 3.1, which is designed to be used for training CNNs on datasets that feature a small number of manually annotated samples and a large number of automatically annotated samples. In this case, the training samples are iteratively selected by using both the automatic labels and the predictions of a CNN trained on a smaller dataset. For this reason, we expect the method to result in a less noisy dataset. Consequently, we also expect improvements on validation and test accuracy. Figure 4.8 shows the evolution of validation accuracy on each iteration of the algorithm. Maximum accuracy is reached on iteration 2, and the algorithm terminates on iteration 3, since the accuracy no longer increases. Additionally, we verified that if we run the algorithm for two more iterations, validation accuracy continues to decrease very slowly.

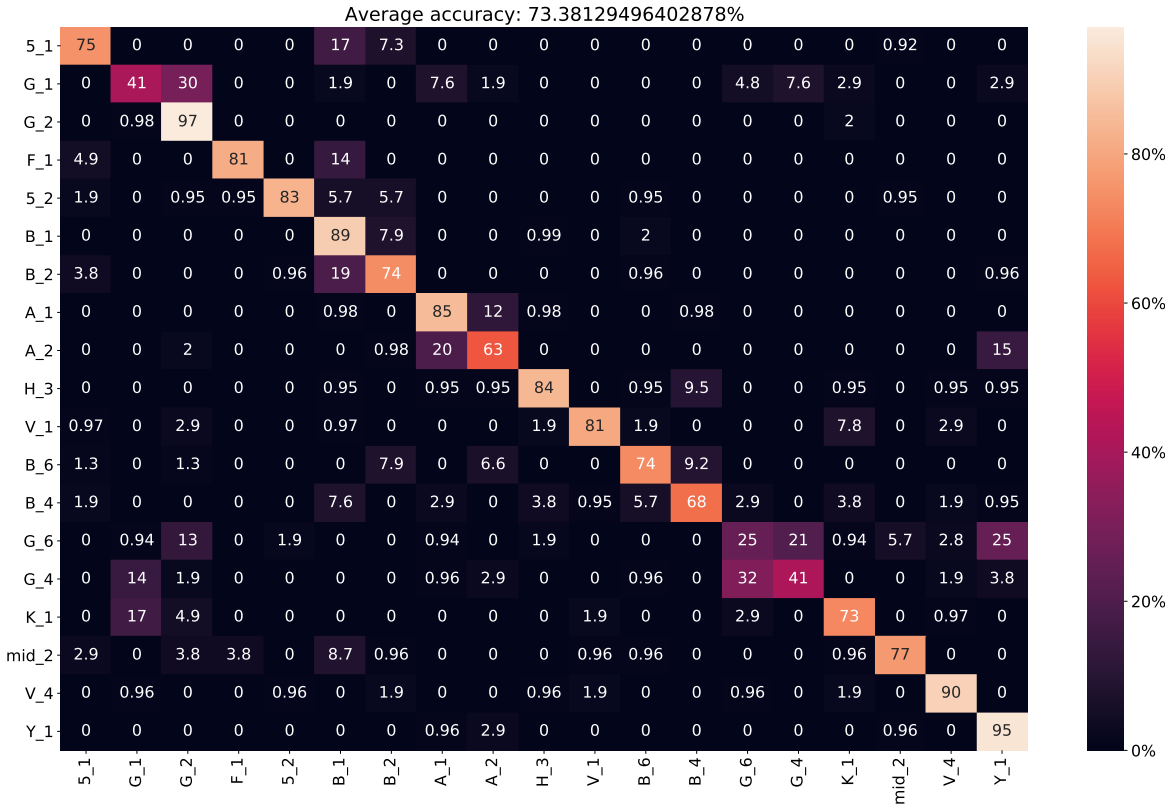

Figure 4.6: Validation set confusion matrix for the CNN trained on both manually and automatically annotated data.

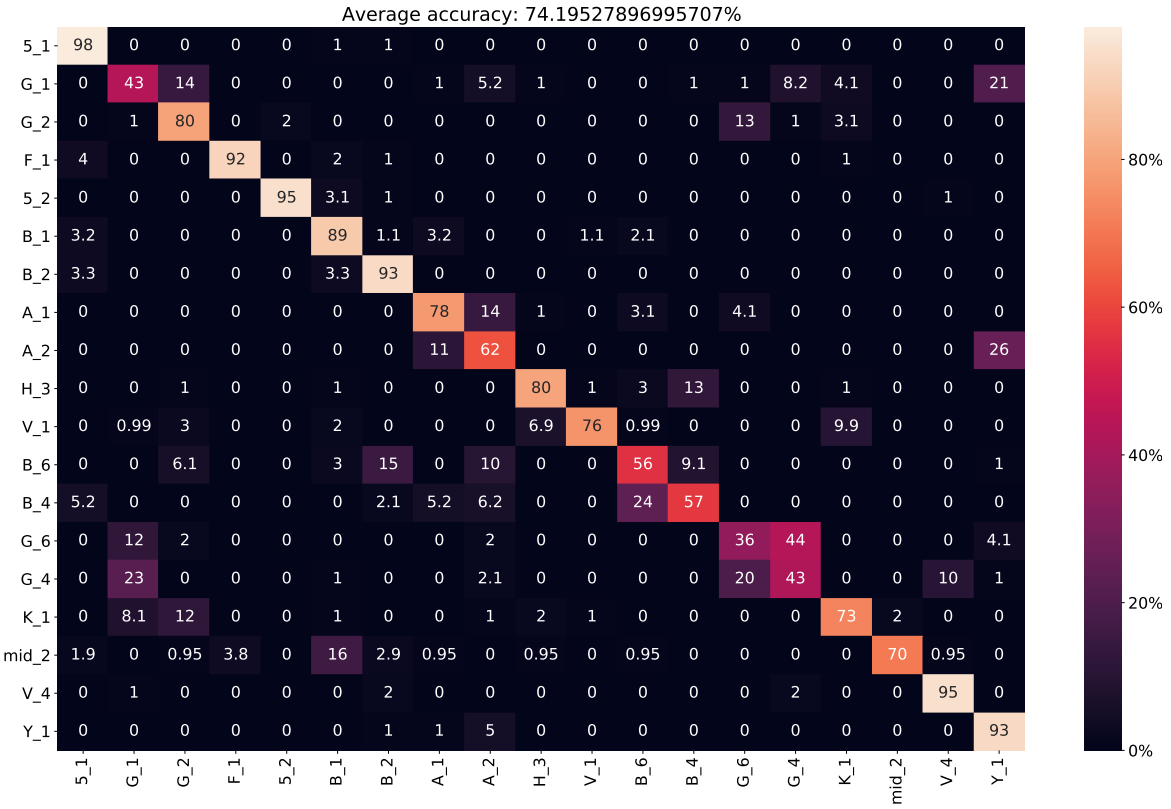

Figure 4.7: Test set confusion matrix for the CNN trained on both manually and automatically annotated data.

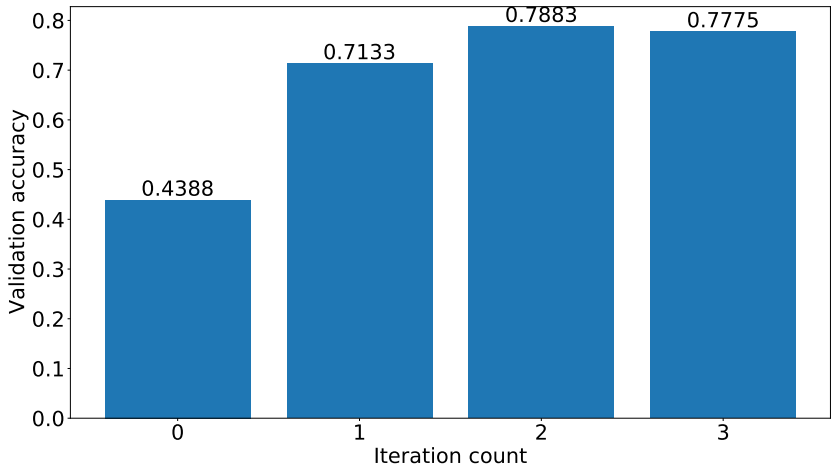

Figure 4.8: Validation accuracy achieved on each iteration when using G-IDEA.

Average accuracy is indeed increased by approximately 5% both on the validation and test set. Furthermore, as we can see in the confusion matrices of figures 4.9 and 4.10, the cases of confusion between specific classes have also decreased. We also note that, without the use of this algorithm the lowest class accuracy recorded on both validation and test sets was 25%, even if the average accuracy was in the order of 70%. When using the algorithm, the lowest class accuracy is 44%. Filtering noisy ground truth samples therefore contributes to the performance of a trained classifier, although there is still potential for further improvement.

4.6 Combining data with C-IDEA

We now proceed to investigate C-IDEA of section 3.1, which also selects new training samples iteratively, but makes its selections in a more conservative manner. We assess whether this more conservative strategy yields improvements. In particular, we use the aforementioned algorithm with arguments $selection\_percent = 0.1$ and $increase\_factor = 0.15$. The evolution of validation accuracy as the iterations progress is shown on figure 4.11. Compared to G-IDEA, accuracy increases at a slower rate, which is to be expected, since the increase of training data is also slower. However, from iteration 5 onward, C-IDEA achieves accuracies higher than the ones recorded in the previous experiment, culminating in a validation accuracy of 83.81%.

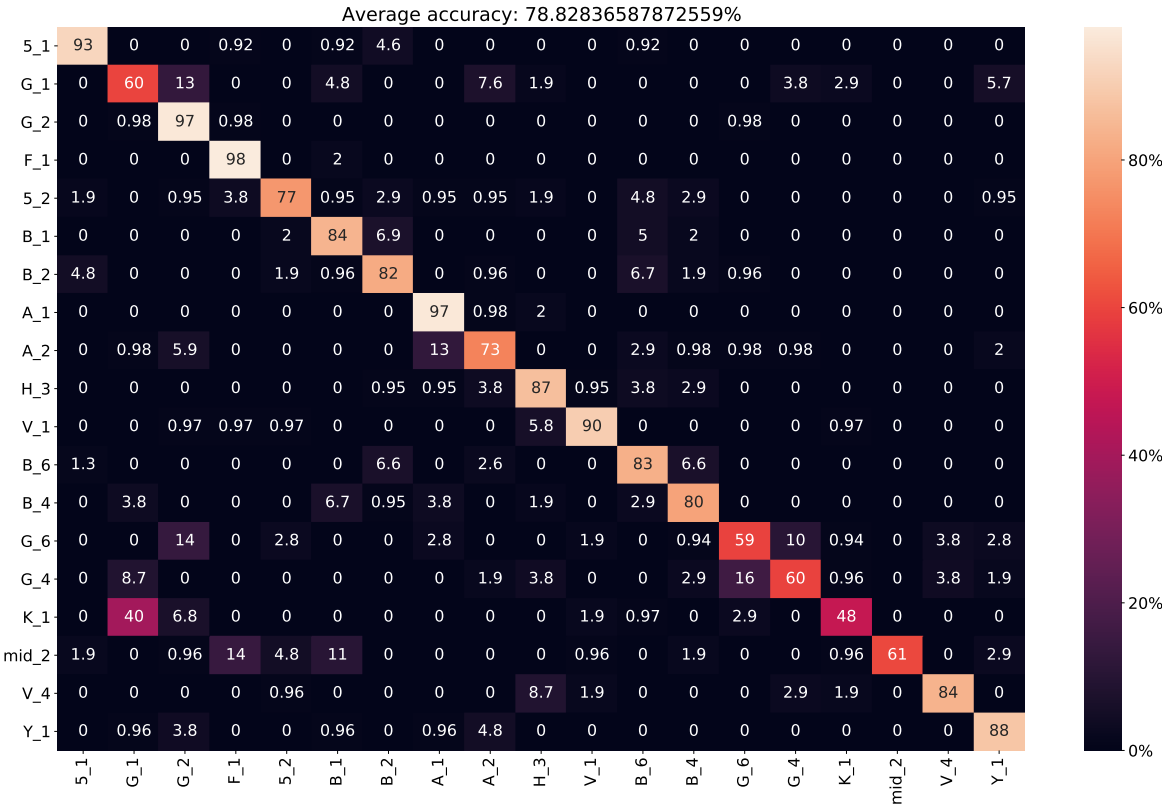

Figure 4.9: Validation set confusion matrix for the CNN trained with G-IDEA

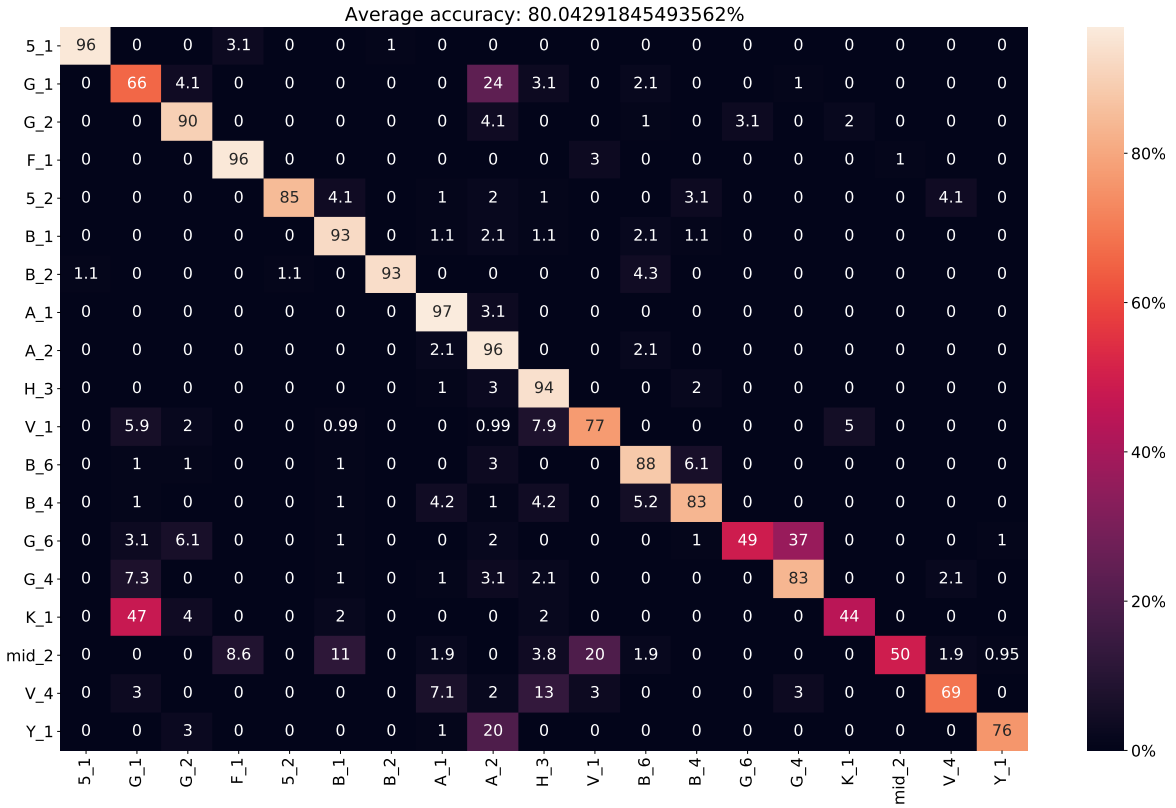

Figure 4.10: Test set confusion matrix for the CNN trained with G-IDEA

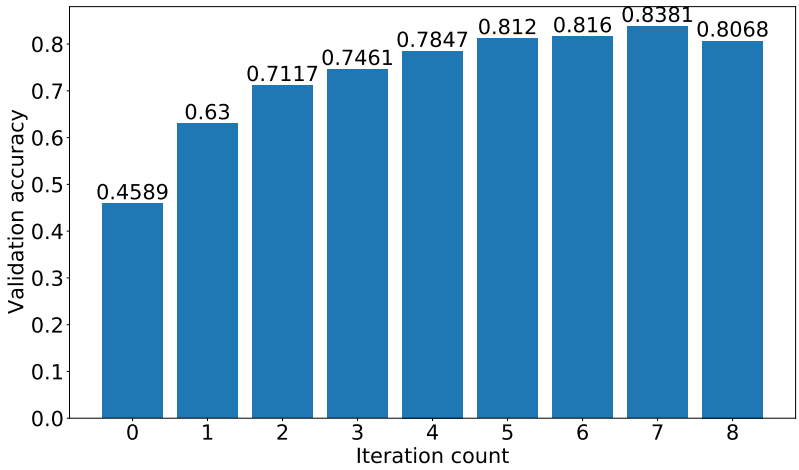

Figure 4.11: Validation accuracy achieved on each iteration when using C-IDEA

Figures 4.12 and 4.13 show the confusion matrices that result from using this method. We achieve an accuracy of 83.81% on the validation set, and 85.72% on the test set, thus improving by approximately 5% in comparison with the experiment of section 4.5. Perhaps more importantly, the lowest class accuracy recorded on both validation and test set is now 65%, and we observe once again a decreased number of cases where one class is significantly confused with another. Therefore, C-IDEA appears capable of selecting mostly correct samples. This is true even in cases where the experiment of section 4.4 indicated a significant amount of confusion between classes in the automatic labeling.

4.7 Results summary

The experimental analysis presented in this section proves that automatically annotated data can be highly beneficial in our problem. The mere inclusion of automatically labeled samples contributes significantly to the generalization of the network, increasing average accuracy on unseen data from the range of 40%-45% to 73%-74%. Furthermore, the cost in human effort for gathering the data is rather small, as described in section 3.6.2. Additionally, the use of the techniques proposed in section 3.1 further increases accuracy to 83%-85%. These results are summarized in table 4.2.

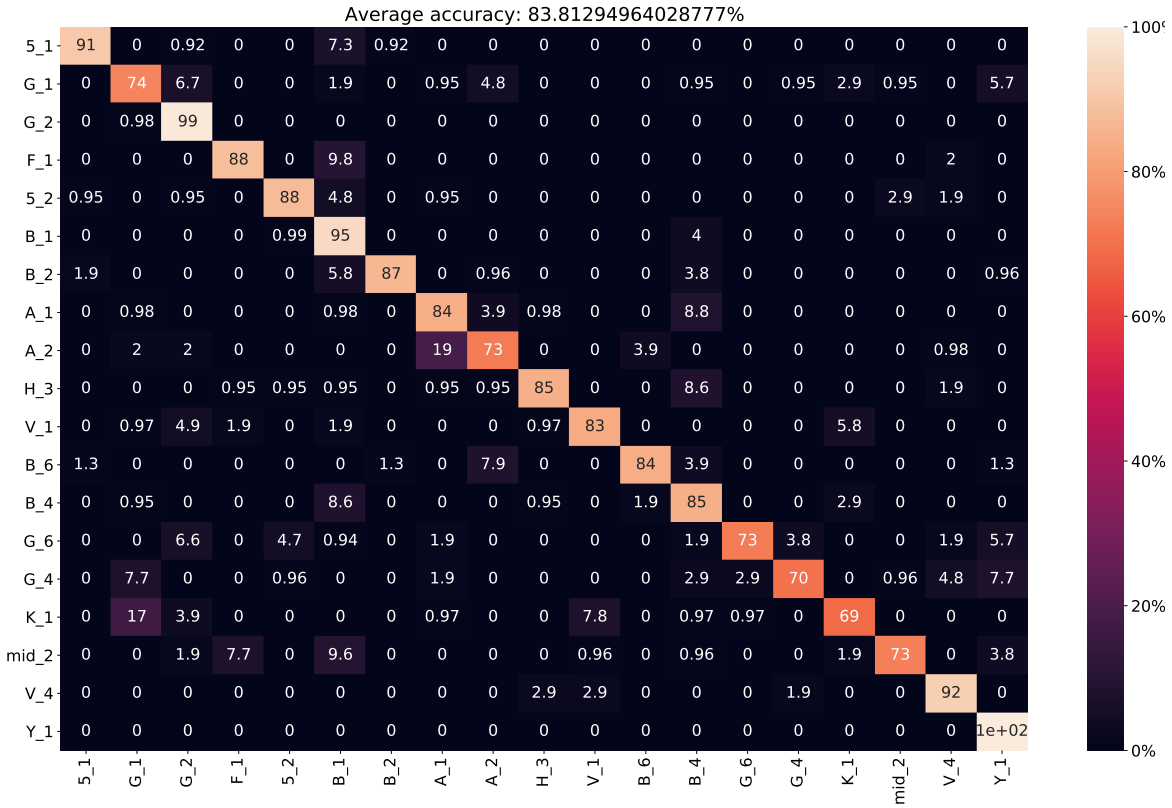

Figure 4.12: Validation set confusion matrix for the CNN trained with C-IDEA

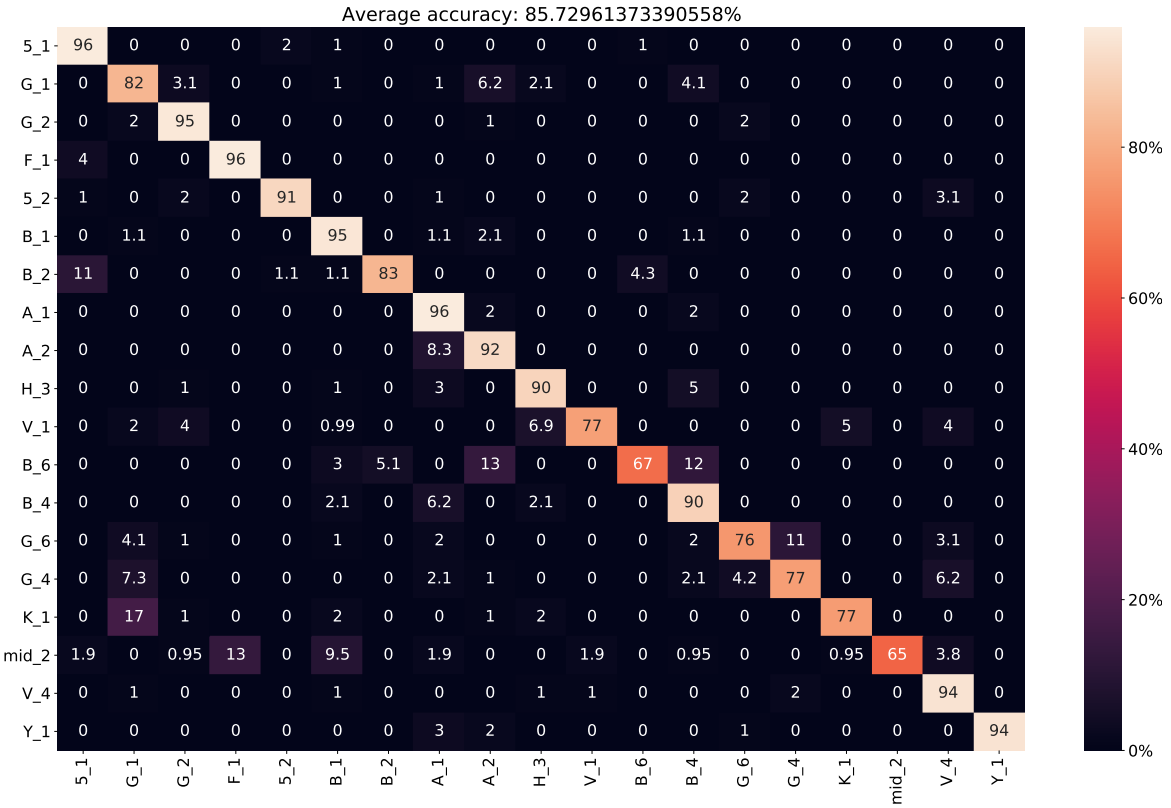

Figure 4.13: Test set confusion matrix for the CNN trained with C-IDEA

Technique	Manual data	Manual & automatic data	G-IDEA	C-IDEA
Validation accuracy	46.45%	73.38%	78.83%	83.81%
Test accuracy	41.52%	74.19%	80.04%	85.73%

Table 4.2: Summary of validation and test accuracies achieved by each method of training.

Of the two algorithms presented in section 3.1, C-IDEA appears to be superior. We attribute the success of the method to the following factors. First, the small corpus of manually annotated data provides indicative examples for each class which are practically guaranteed to be correct. Second, the larger set of automatically annotated samples may be noisy, but also features a much greater variety in individual-specific characteristics of the hand, in lighting conditions and in backgrounds. Third, the automatically annotated samples are not fed blindly to the network, but are subjected to a test which verifies that the initially noisy predictions of the network agree with the noisy automatic labels. Fourth, from the subset of samples where the predictions agree with the labels, only a small percentage for which the network is more confident is selected.

Applying this procedure iteratively allows the network to gradually include more diverse samples in its training set, while simultaneously avoiding a significant percentage of the noisy data. Figure 4.14 provides a qualitative comparison of the distribution of class accuracies for each method. We present a stacked histogram of the class accuracy distribution for all four methods. A smoothed probability distribution function is fitted on the underlying distribution of class accuracy data for each method. Additionally, the height of the bars is scaled for visualization purposes. Visually, as we improve the method of utilization for the automatically annotated data, the curves become steeper and their mode increases. This illustrates the fact that as the method improves, not only does the average accuracy increase, but, perhaps more importantly, the network achieves a more consistent performance across the various classes of the problem.

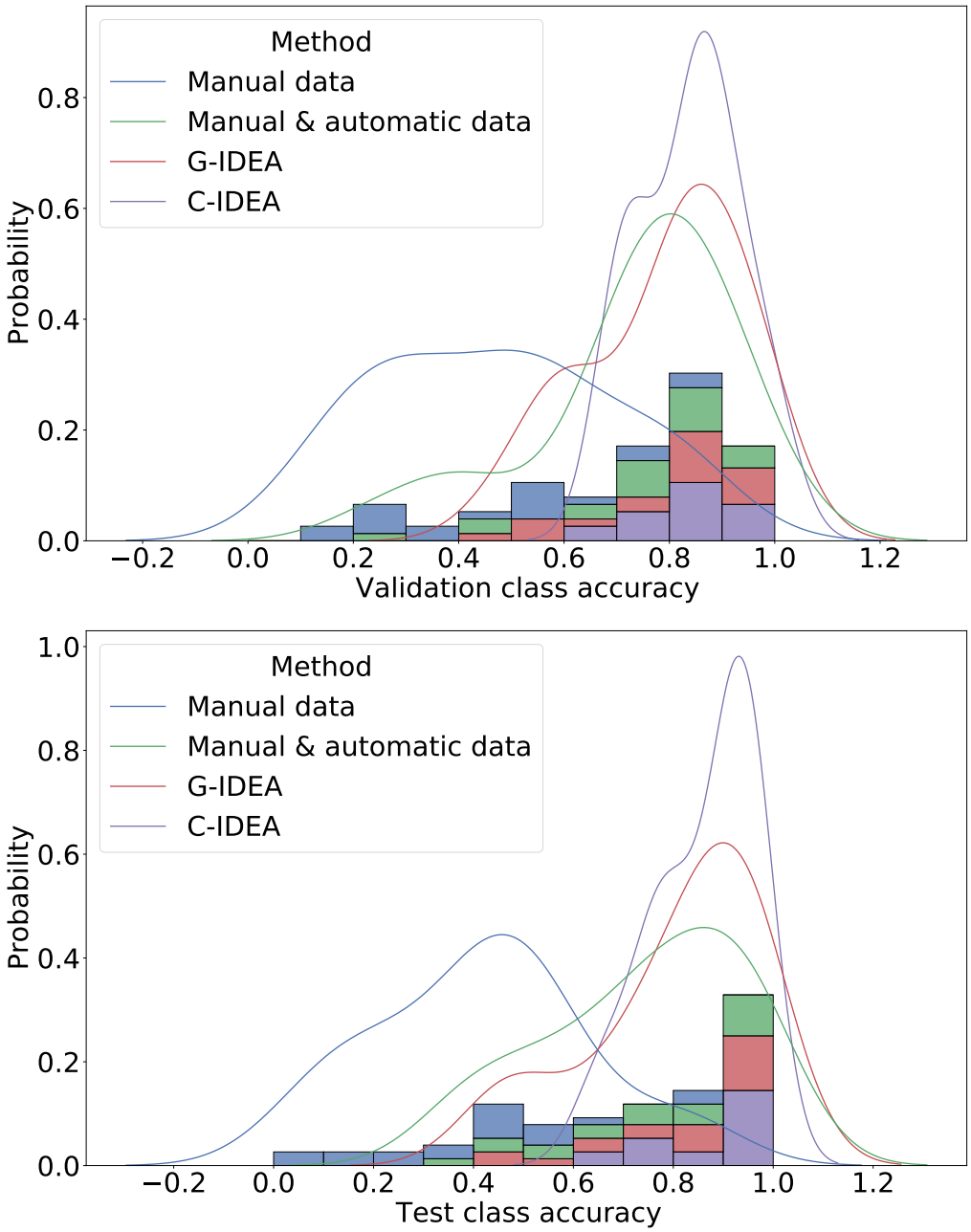

Figure 4.14: Validation (top) and test (bottom) class accuracy histograms for all methods (scaled for visualization).

Chapter 5

Discussion

5.1 Summary

We presented a method for utilizing automatically annotated data in training CNNs on classification problems. The method is based on training the network on a small subset of manually annotated data, and then iteratively selecting subsets of the automatically annotated data for which the prediction of the network agrees with the automatic label. On each iteration, the network is retrained on its new training set, and gradually becomes more robust in its predictions. We propose two variants of the same method, one more conservative than the other with respect to the selection criteria.

We then proceeded to apply these techniques on the problem of hand posture recognition from RGB images. In this context, we also presented a method for extracting automatic ground truth labels from the images, by estimating the corresponding 3-D hand postures and comparing them to the postures that comprise the classes of our problem. Compared to the simple approach of using all automatically annotated samples, the application of our techniques on these labels increases the average test accuracy, and decreases the variance of the per-class accuracies. The results are, therefore, promising, and indicate a significant contribution to the generalization capability of the trained CNN.

5.2 Future work

In this work, we investigated methods for utilization of automatically annotated data in classification problems. We evaluated the use of these methods on the problem of hand posture recognition from RGB images, specifically for a set of hand postures that convey meaning in GSL. The work could, therefore, be extended in a number of directions.

Firstly, with respect to the algorithms themselves, further work is possible in the direction of determining the most appropriate samples for training. The more conservative approach of C-IDEA appears to yield an improvement over G-IDEA.

It is therefore possible that the introduction of some measure of certainty for the automatic labeling itself, and the combination of it with the network's likelihood predictions could yield further improvements.

Secondly, with respect to the problem of sign language recognition, the work presented here utilizes a subset of postures that are meaningful in GSL. Extending this set to feature more postures would both explore the generalization capability of the approach presented here, and result in a system that would be more useful in a practical application. Furthermore, since hand posture information is integral in the understanding of sign language, it would be interesting to study the usefulness of the extracted postures as features of a complete pipeline for sign language recognition.

On a third axis, specifically for the problem of hand posture recognition, we have stressed the importance of the 3-D hand posture estimation method on the quality of the ground truth labeling (section 3.6.2). One could experiment with different 3-D hand trackers, or possibly with the notion of combining results of multiple hand tracking methods, in a boosting [79] approach. This approach could potentially reduce the failure cases of the combined 3-D landmark estimation, and yield a less noisy ground truth labeling.

Lastly, since the algorithms of section 3.1 are generic in nature, they can be used in any classification problem. Therefore, one could evaluate their usefulness in other tasks that feature a small amount of manually annotated data, and a larger amount of automatically annotated data with noisy ground truth labels.

Bibliography

[1] Nikolaos M. Adaloglou, Theocharis Chatzis, Ilias Papastratis, Andreas Stergioulas, Georgios Th Papadopoulos, Vassia Zacharopoulou, George Xydopoulos, Klimis Antzakas, Dimitris Papazachariou, and Petros none Daras. A comprehensive study on deep learning-based methods for sign language recognition. *IEEE Transactions on Multimedia*, page 1–1, 2021.

[2] Klimis Antzakas. The use of negative head movements in greek sign language. *Interrogative and negative constructions in sign languages*, pages 258–269, 2006.

[3] Anil Armagan, Guillermo Garcia-Hernando, Seungryul Baek, Shreyas Hampali, Mahdi Rad, Zhaohui Zhang, Shipeng Xie, MingXiu Chen, Boshen Zhang, Fu Xiong, et al. Measuring generalisation to unseen viewpoints, articulations, shapes and objects for 3d hand pose estimation under hand-object interaction. *arXiv preprint arXiv:2003.13764*, 2020.

[4] Devansh Arpit, Stanisław Jastrzębski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. In *International Conference on Machine Learning*, pages 233–242. PMLR, 2017.

[5] Maryam Asadi-Aghbolaghi, Albert Clapes, Marco Bellantonio, Hugo Jair Escalante, Víctor Ponce-López, Xavier Baró, Isabelle Guyon, Shohreh Kasaei, and Sergio Escalera. A survey on deep learning based approaches for action and gesture recognition in image sequences. In *2017 12th IEEE international conference on automatic face & gesture recognition (FG 2017)*, pages 476–483. IEEE, 2017.

[6] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate, 2016.

[7] Pablo Barros, Sven Magg, Cornelius Weber, and Stefan Wermter. A multichannel convolutional neural network for hand posture recognition. In *International Conference on Artificial Neural Networks*, pages 403–410. Springer, 2014.

[8] Serge Belongie, Greg Mori, and Jitendra Malik. Matching with shape contexts. In *Statistics and Analysis of Shapes*, pages 81–105. Springer, 2006.

[9] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009.

[10] Carl Börstell, Thomas Hörberg, and Robert Östling. Distribution and duration of signs and parts of speech in swedish sign language. *Sign Language & Linguistics*, 19(2):143–196, 2016.

[11] Necati Cihan Camgoz, Simon Hadfield, Oscar Koller, and Richard Bowden. Using convolutional 3d neural networks for user-independent continuous gesture recognition. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 49–54. IEEE, 2016.

[12] Necati Cihan Camgoz, Simon Hadfield, Oscar Koller, and Richard Bowden. Subunets: End-to-end hand shape and continuous sign language recognition. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 3075–3084. IEEE, 2017.

[13] Necati Cihan Camgoz, Simon Hadfield, Oscar Koller, Hermann Ney, and Richard Bowden. Neural sign language translation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7784–7793, 2018.

[14] Necati Cihan Camgöz, Ahmet Alp Kındıroğlu, Serpil Karabüklü, Meltem Kelepir, Ayşe Sumru Özsoy, and Lale Akarun. Bosphorussign: a turkish sign language recognition corpus in health and finance domains. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1383–1388, 2016.

[15] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields, 2017.

[16] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset, 2018.

[17] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542, 2009.

[18] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation, 2014.

[19] HM Cooper, Eng-Jon Ong, Nicolas Pugeault, and Richard Bowden. Sign language recognition using sub-units. *Journal of Machine Learning Research*, 13:2205–2231, 2012.

[20] Runpeng Cui, Hu Liu, and Changshui Zhang. Recurrent convolutional neural networks for continuous sign language recognition by staged optimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7361–7369, 2017.

[21] Runpeng Cui, Hu Liu, and Changshui Zhang. A deep neural framework for continuous sign language recognition by iterative training. *IEEE Transactions on Multimedia*, 21(7):1880–1891, 2019.

[22] Marco Cuturi and Mathieu Blondel. Soft-dtw: a differentiable loss function for time-series. In *International Conference on Machine Learning*, pages 894–903. PMLR, 2017.

[23] Quentin De Smedt, Hazem Wannous, and Jean-Philippe Vandeborre. Skeleton-based dynamic hand gesture recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–9, 2016.

[24] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[25] Georgios D Evangelidis, Gurkirt Singh, and Radu Horaud. Continuous gesture recognition from articulated poses. In *European Conference on Computer Vision*, pages 595–607. Springer, 2014.

[26] Jens Forster, Christoph Schmidt, Oscar Koller, Martin Bellgardt, and Hermann Ney. Extensions of the sign language recognition and translation corpus rwth-phoenix-weather. In *LREC*, pages 1911–1916, 2014.

[27] Ross Girshick. Fast r-cnn, 2015.

[28] Stephen Gould, Richard Fulton, and Daphne Koller. Decomposing a scene into geometric and semantically consistent regions. In *2009 IEEE 12th International Conference on Computer Vision*, pages 1–8, 2009.

[29] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376, 2006.

[30] Alex Graves, Abdel rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks, 2013.

[31] Guy Hacohen and Daphna Weinshall. On the power of curriculum learning in training deep networks. In *International Conference on Machine Learning*, pages 2535–2544. PMLR, 2019.

[32] Shreyas Hampali, Sayan Deb Sarkar, Mahdi Rad, and Vincent Lepetit. Handsformer: Keypoint transformer for monocular 3d pose estimation ofhands and object in interaction, 2021.

[33] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *arXiv preprint arXiv:1804.06872*, 2018.

[34] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[35] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[36] Sina Honari, Pavlo Molchanov, Stephen Tyree, Pascal Vincent, Christopher Pal, and Jan Kautz. Improving landmark localization with semi-supervised learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1546–1555, 2018.

[37] Jie Huang, Wengang Zhou, Qilin Zhang, Houqiang Li, and Weiping Li. Video-based sign language recognition without temporal segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[38] Umar Iqbal, Pavlo Molchanov, Thomas Breuel, Juergen Gall, and Jan Kautz. Hand pose estimation via latent 2.5d heatmap regression, 2018.

[39] Youngkyoon Jang, Seung-Tak Noh, Hyung Jin Chang, Tae-Kyun Kim, and Woontack Woo. 3d finger cape: Clicking action and position estimation under self-occlusions in egocentric viewpoint. *IEEE Transactions on Visualization and Computer Graphics*, 21(4):501–510, 2015.

[40] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International Conference on Machine Learning*, pages 2304–2313. PMLR, 2018.

[41] Hamid Reza Vaezi Joze and Oscar Koller. Ms-asl: A large-scale data set and benchmark for understanding american sign language. *arXiv preprint arXiv:1812.01053*, 2018.

[42] Mohammed Waleed Kadous et al. Machine recognition of auslan signs using powergloves: Towards large-lexicon recognition of sign language. In *Proceedings of the Workshop on the Integration of Gesture in Language and Speech*, volume 165, 1996.

[43] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[44] Oscar Koller, Necati Cihan Camgoz, Hermann Ney, and Richard Bowden. Weakly supervised learning with multi-stream cnn-lstm-hmms to discover sequential parallelism in sign language videos. *IEEE transactions on pattern analysis and machine intelligence*, 42(9):2306–2320, 2019.

[45] Oscar Koller, Hermann Ney, and Richard Bowden. Deep hand: How to train a cnn on 1 million hand images when your data is continuous and weakly labelled. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3793–3802, 2016.

[46] Oscar Koller, O Zargaran, Hermann Ney, and Richard Bowden. Deep sign: Hybrid cnn-hmm for continuous sign language recognition. In *Proceedings of the British Machine Vision Conference 2016*, 2016.

[47] Oscar Koller, Sepehr Zargaran, and Hermann Ney. Re-sign: Re-aligned end-to-end sequence modelling with deep recurrent cnn-hmms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4297–4305, 2017.

[48] Dimitrios Kosmopoulos, Iasonas Oikonomidis, Constantinos Constantinopoulos, Nikolaos Arvanitis, K Antzakas, A Bifis, G Lydakis, A Roussos, and A Argyros. Towards a visual sign language dataset for home care services. In *2020 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020)*, pages 520–524. IEEE, 2020.

[49] Taein Kwon, Bugra Tekin, Jan Stuhmer, Federica Bogo, and Marc Pollefeys. H2o: Two hands manipulating objects for first person interaction recognition, 2021.

[50] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[51] Boris Lenseigne and Patrice Dalle. Using signing space as a representation for sign language processing. In *International Gesture Workshop*, pages 25–36. Springer, 2005.

[52] Dongxu Li, Cristian Rodriguez, Xin Yu, and Hongdong Li. Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 1459–1469, 2020.

[53] Dongxu Li, Chenchen Xu, Xin Yu, Kaihao Zhang, Ben Swift, Hanna Suominen, and Hongdong Li. Tspnet: Hierarchical feature learning via temporal semantic pyramid for sign language translation. *arXiv preprint arXiv:2010.05468*, 2020.

[54] Patrick Lichtsteiner, Christoph Posch, and Tobi Delbruck. A 128×128 120 db 15μs latency asynchronous temporal contrast vision sensor. *IEEE journal of solid-state circuits*, 43(2):566–576, 2008.

[55] Google LLC. Mediapipe hands. `https://google.github.io/mediapipe/solutions/hands.html#output`. Accessed: Nov. 2020.

[56] Anders Markussen, Mikkel Rønne Jakobsen, and Kasper Hornbæk. Vulture: a mid-air word-gesture keyboard. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1073–1082, 2014.

[57] Baharan Mirzasoleiman, Kaidi Cao, and Jure Leskovec. Coresets for robust training of deep neural networks against noisy labels. *Advances in Neural Information Processing Systems*, 33, 2020.

[58] Todd K Moon. The expectation-maximization algorithm. *IEEE Signal processing magazine*, 13(6):47–60, 1996.

[59] Jill P Morford and James MacFarlane. Frequency characteristics of american sign language. *Sign Language Studies*, pages 213–225, 2003.

[60] Franziska Mueller, Florian Bernard, Oleksandr Sotnychenko, Dushyant Mehta, Srinath Sridhar, Dan Casas, and Christian Theobalt. Ganerated hands for real-time 3d hand tracking from monocular rgb, 2017.

[61] Weizhi Nai, Yue Liu, David Rempel, and Yongtian Wang. Fast hand posture classification using depth features extracted from random line segments. *Pattern Recognition*, 65:1–10, 2017.

[62] Natalia Neverova, Christian Wolf, Graham Taylor, and Florian Nebout. Moddrop: adaptive multi-modal gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(8):1692–1706, 2015.

[63] University of Patras. The healthsign project. `http://xanthippi.ceid.upatras.gr/HealthSign/EN/HealthSign.html`. Accessed: Sept. 2020.

[64] Hellenic Pedagogical Institute Department of Special Education. The greek sign language. `http://www.pi-schools.gr/special_education_new/html/gr/8emata/ekp_yliko/kofosi.htm`. Accessed: Jun. 2020.

[65] Carol A Padden and Darline Clark Gunsauls. How the alphabet came to be used in a sign language. *Sign Language Studies*, pages 10–33, 2003.

[66] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[67] Paschalis Panteleris, Iason Oikonomidis, and Antonis A Argyros. Using a single rgb frame for real time 3d hand pose estimation in the wild. In *IEEE*

Winter Conference on Applications of Computer Vision (WACV 2018), also available at CoRR, arXiv, pages 436–445, lake Tahoe, NV, USA, March 2018. IEEE.

[68] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[69] Junfu Pu, Wengang Zhou, and Houqiang Li. Sign language recognition with multi-modal features. In *Pacific Rim Conference on Multimedia*, pages 252–261. Springer, 2016.

[70] Junfu Pu, Wengang Zhou, and Houqiang Li. Dilated convolutional network with iterative optimization for continuous sign language recognition. In *IJCAI*, volume 3, page 7, 2018.

[71] Junfu Pu, Wengang Zhou, and Houqiang Li. Iterative alignment network for continuous sign language recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4165–4174, 2019.

[72] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks, 2016.

[73] Razieh Rastgoo, Kourosh Kiani, and Sergio Escalera. Sign language recognition: A deep survey. *Expert Systems with Applications*, 164:113794, 08 2020.

[74] Marcus Rohrbach, Sikandar Amin, Mykhaylo Andriluka, and Bernt Schiele. A database for fine grained activity detection of cooking activities. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1194–1201. IEEE, 2012.

[75] Viktor Rudnev, Vladislav Golyanik, Jiayi Wang, Hans-Peter Seidel, Franziska Mueller, Mohamed Elgharib, and Christian Theobalt. Eventhands: Real-time neural 3d hand reconstruction from an event stream, 2020.

[76] Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing*, 26(1):43–49, 1978.

[77] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks, 2019.

[78] Wendy Sandler and Diane Lillo-Martin. *Sign language and linguistic universals*. Cambridge University Press, 2006.

[79] Robert E Schapire. The strength of weak learnability. *Machine learning*, 5(2):197–227, 1990.

[80] John Shawe-Taylor and N Cristianini. *An introduction to support vector machines and other kernel-based learning methods*, volume 204. Volume, 2000.

[81] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):1–48, 2019.

[82] Tomas Simon, Hanbyul Joo, Iain Matthews, and Yaser Sheikh. Hand keypoint detection in single images using multiview bootstrapping, 2017.

[83] Stephanie Stoll, Necati Cihan Camgoz, Simon Hadfield, and Richard Bowden. Text2sign: Towards sign language production using neural machine translation and generative adversarial networks. *International Journal of Computer Vision*, 128(4):891–908, 2020.

[84] Yingcheng Sun and Kenneth Loparo. Context aware image annotation in active learning. *arXiv preprint arXiv:2002.02775*, 2020.

[85] Ao Tang, Ke Lu, Yufei Wang, Jie Huang, and Houqiang Li. A real-time hand posture recognition system using deep neural networks. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 6(2):1–23, 2015.

[86] Danhang Tang, Tsz-Ho Yu, and Tae-Kyun Kim. Real-time articulated hand pose estimation using semi-supervised transductive regression forests. In *Proceedings of the IEEE international conference on computer vision*, pages 3224–3231, 2013.

[87] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497, 2015.

[88] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator, 2015.

[89] Paul Voigtlaender, Lishu Luo, Chun Yuan, Yong Jiang, and Bastian Leibe. Reducing the annotation effort for video object segmentation datasets. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3060–3069, 2021.

[90] Ulrich Von Agris, Moritz Knorr, and Karl-Friedrich Kraiss. The significance of facial features for automatic sign language recognition. In *2008 8th IEEE International Conference on Automatic Face & Gesture Recognition*, pages 1–6. IEEE, 2008.

[91] Chengde Wan, Thomas Probst, Luc Van Gool, and Angela Yao. Crossing nets: Dual generative models with a shared latent space for hand pose estimation. In *Conference on Computer Vision and Pattern Recognition*, volume 7, 2017.

[92] Wikipedia. Zipf's law. `https://en.wikipedia.org/wiki/Zipf's_law`. Accessed: Jun. 2020.

[93] Di Wu, Lionel Pigou, Pieter-Jan Kindermans, Nam Do-Hoang Le, Ling Shao, Joni Dambre, and Jean-Marc Odobez. Deep dynamic neural networks for multimodal gesture segmentation and recognition. *IEEE transactions on pattern analysis and machine intelligence*, 38(8):1583–1597, 2016.

[94] Zhaoyang Yang, Zhenmei Shi, Xiaoyong Shen, and Yu-Wing Tai. Sf-net: Structured feature network for continuous sign language recognition. *arXiv preprint arXiv:1908.01341*, 2019.

[95] Kayo Yin and Jesse Read. Better sign language translation with stmc-transformer, 2020.

[96] Matthew D. Zeiler. Adadelta: An adaptive learning rate method, 2012.

[97] Fan Zhang, Valentin Bazarevsky, Andrey Vakunov, Andrei Tkachenka, George Sung, Chuo-Ling Chang, and Matthias Grundmann. Mediapipe hands: On-device real-time hand tracking, 2020.

[98] Hao Zhou, Wengang Zhou, and Houqiang Li. Dynamic pseudo label decoding for continuous sign language recognition. In *2019 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1282–1287. IEEE, 2019.

[99] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks, 2020.

[100] Christian Zimmermann and Thomas Brox. Learning to estimate 3d hand pose from single rgb images, 2017.

[101] George Kingsley Zipf. *Human behavior and the principle of least effort: An introduction to human ecology.* Ravenio Books, 2016.