IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

SIGN LANGUAGE DETECTION SYSTEM

[1]Prarthana Mozes
[1]B. Tech Student, Department of Computer Science and Engineering, Krishna Engineering College, Bhilai, Chhattisgarh, India

Abstract: Sign language, a visual-spatial language modality, serves as a paramount means of communication for individuals with hearing and speaking impairments. The intricate hand movements, facial expressions, and body postures utilized in sign language facilitate the conveyance of meaning and emotions. Communication between a person with normal hearing and speech abilities and an individual who is deaf and mute poses unique challenges. The absence of verbal language and auditory cues hinders direct interaction and understanding. A sign language detection system acts as a crucial bridge to overcome the communication gap between individuals with hearing impairments and those without. By leveraging machine learning techniques and computer vision algorithms, this system is capable of accurately interpreting and understanding sign language gestures in real-time. This project has the potential to significantly improve accessibility and inclusivity for the deaf and dumb community, enabling effective communication and equal participation in various aspects of life.

Index Terms – **Sign Language Detection System, American Sign Language (ASL), Machine learning, Human-Computer interaction, Python, SSD (Single-shot multi-box detector), TensorFlow, OpenCV**

INTRODUCTION

1.1 Introduction of project

Sign language is a complex and rich linguistic system used by deaf, dumb and hard-of-hearing individuals as their primary mode of communication. It is a complete language with its own grammar, syntax, and vocabulary. Sign languages utilize visual and gestural elements as the primary means of communication. Instead of using spoken words, signers use manual gestures, hand shapes, movements, facial expressions, and body language to convey meaning. The grammar of sign languages includes elements such as word order, classifiers, verb agreement, spatial referencing, and non-manual markers.

Sign languages differ across countries and regions due to factors such as historical development, cultural influences, and geographic isolation. Indian Sign Language (ISL), British Sign Language (BSL), French Sign Language (LSF), American Sign Language (ASL), South African Sign Language (SASL), etc. These are just a few examples, and there are many more sign languages used around the world. Each sign language has its own unique features, reflecting the linguistic and cultural diversity of the Deaf and Dumb communities they serve. The most widely known and used sign language all over the world is American Sign Language (ASL).

This sign language detection project work to leverage computer vision and machine learning techniques to develop a system capable of recognizing and interpreting American Sign Language (ASL) gestures. This technology holds tremendous potential for improving communication and inclusivity for the deaf, dumb and hard of hearing community. By enabling real-time translation of sign language into written language, the project seeks to bridge the communication gap between individuals who use sign language and those who do not.

Sign language detection involves utilizing advanced technologies such as machine learning, TensorFlow, OpenCV, and SSD (Single Shot Multi-Box Detector) to enable accurate recognition of sign language gestures.

1.2 Problem Statement

The deaf, dumb and hard of hearing community faces significant challenges due to the limited availability of sign language interpreters, resulting in communication barriers and impeded social interactions. This inadequacy creates a sense of isolation and hampers their access to crucial information and opportunities. While text-based messaging or written notes can be used as communication alternatives, they fail to capture the intricate nuances and expressive nature of sign language, limiting effective communication with non-sign language users. Consequently, there is an urgent and pressing need to develop an automated sign language detection system capable of accurately recognizing and interpreting sign language gestures in real-time. This system should enable immediate translation into written or spoken language, facilitating seamless communication between sign language users and the wider community. By addressing this need, individuals in the deaf, dumb and hard of hearing community can experience enhanced inclusivity, improved social integration, and equal access to information and opportunities.

1.3 Requirements

- **Software Requirements:**

  These are the Software Configurations that are required.
  - Operating System: Linux, Windows, MacOS
  - Language: Python 3.8 with Anaconda3 Environment
  - Libraries: TensorFlow, OpenCV
  - IDE: Jupyter Notebook
  - Software Tools: LabelImg
- **Hardware Requirements:**

  These are the Hardware Configurations that are required.
  - Processor: Any processor above 500 MHz
  - RAM: 4GB and above
  - Hard disk: 40GB and above
  - Input device: Webcam
  - Output device: Monitor

THEORY AND LITRATURE REVIEW

2.1 Theory

The project of sign language detection involves the use of computer vision and machine learning techniques to interpret and understand sign language gestures. The theory behind sign language detection is based on the idea that visual information from sign language gestures can be captured, processed, and analyzed to recognize and interpret the intended meaning.

Sign language detection starts with capturing visual input. The captured data undergoes preprocessing to enhance its quality and remove irrelevant information. Computer vision algorithms are applied to track and recognize the hand movements and gestures. Meaningful features are extracted from the tracked hand movements and gestures. Machine learning algorithms are utilized to learn the mapping between the extracted features and corresponding sign language gestures. During the training phase, the machine learning models learn to recognize and classify different sign language gestures based on the extracted features. Post-processing techniques may be applied to refine the results and improve accuracy.

2.2 Research of Studies

2.2.1 Two-way sign language detection system

In this paper in 2019, author Tanuj Bohra proposed a real-time two-way sign language communication system (sign language to text and text to sign language) built using image processing, deep learning and computer vision, hand detection, skin segmentation, contour detection is performed on images. This paper proposes the use of CNN for the detection.

2.2.2 Sign Language Recognition Based on Indian Sign Language (ISL)

In this paper, Joyeeta Singha and Karen Das proposed a system for Indian Sign Language recognition. Dataset consisted 480 images of 24 signs of ISL signed by 20 people. System was tested on 20 videos and achieved an accuracy of 96.25%.

2.2.3 Sign language detection with LDA

In this paper, Mahesh Kumar proposed a system which can recognize 26 hand gestures of Indian Sign Language based on Linear Discriminant Analysis (LDA). Linear discriminant analysis is used for feature extraction. Each gesture is represented as a column vector in training phase which is then normalized with respect to average gesture.

METHODOLOGY

3.1 Existing System

The existing system for the sign language detection system project employs CNN (Convolutional Neural Network) for classifying sign language gestures. By leveraging convolutional layers, CNNs extract relevant spatial patterns, edges, and textures specific to each sign gesture. The network's objective is to optimize its internal parameters through training, minimizing classification errors and improving the ability to accurately differentiate between different signs.

3.2 Proposed System

In this proposed system for sign language detection, we replace the traditional CNN (Convolutional Neural Network) with the SSD (Single Shot Multi-Box Detector) algorithm. SSD offers significant advantages over CNN for sign language detection tasks. It harnesses the power of deep learning and provides real-time object detection capabilities. By utilizing SSD, we can achieve more accurate and efficient detection and localization of sign language gestures. The algorithm's multi-scale detection and ability to handle class imbalance enhance the system's performance. Compared to CNN, SSD simplifies the training pipeline and offers flexibility for customization. Implementing SSD in the sign language detection project enables us to improve the speed, accuracy, and overall effectiveness of the system, enhancing communication and interaction for individuals using sign language.

3.3 Proposed Techniques

3.3.1 Machine Learning

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine Learning is defined as the study of computer programs that leverage algorithms and statistical models to learn through inference and patterns without being explicitly programmed. Machine Learning field has undergone significant developments in the last decade. Machine learning focuses on the development of computer programs that can access data and use it to learn for themselves. There are also some types of machine learning algorithms that are used in very specific use-cases, but three main methods used today are:

Supervised Machine Learning Algorithm

Unsupervised Machine Learning Algorithm

Reinforcement Machine Learning Algorithm

Among these, the type of machine learning algorithm we used in our system is supervised machine learning algorithm.

- **Supervised Machine Learning**

This can apply what has been learned in the past to new data using labelled examples to predict future events. Starting from the analysis of a known training dataset, the learning algorithm produces an inferred function to make predictions about the output values. The system is able to provide targets for any new input after sufficient training. The learning algorithm can also compare its output with the correct, intended output and find errors in order to modify the model accordingly.

- **Unsupervised Machine Learning**

It holds the advantage of being able to work with unlabeled data. This means that human labor is not required to make the dataset machine-readable, allowing much larger datasets to be worked on by the program. In supervised learning, the labels allow the algorithm to find the exact nature of the relationship between any two data points. However, unsupervised learning does not have labels to work off of, resulting in the creation of hidden structures. Relationships between data points are perceived by the algorithm

in an abstract manner, with no input required from human beings. The creation of these hidden structures is what makes unsupervised learning algorithms versatile. Instead of a defined and set problem statement, unsupervised learning algorithms can adapt to the data by dynamically changing hidden structures. This offers more post deployment development than supervised learning algorithms.

- **Reinforcement Machine Learning**

It directly takes inspiration from how human beings learn from data in their lives. It features an algorithm that improves upon itself and learns from new situations using a trial-and-error method. Favorable outputs are encouraged or 'reinforced', and non-favorable outputs are discouraged or 'punished'. Based on the psychological concept of conditioning, reinforcement learning works by putting the algorithm in a work environment with an interpreter and a reward system. In every iteration of the algorithm, the output result is given to the interpreter, which decides whether the outcome is favorable or not. In case of the program finding the correct solution, the interpreter reinforces the solution by providing a reward to the algorithm. If the outcome is not favorable, the algorithm is forced to reiterate until it finds a better result. In most cases, the reward system is directly tied to the effectiveness of the result. In typical reinforcement learning use-cases, such as finding the shortest route between two points on a map, the solution is not an absolute value. Instead, it takes on a score of effectiveness, expressed in a percentage value. Thus, the program is trained to give the best possible solution for the best possible reward.

3.3.2 Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. It's high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's 18 simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective. It's easy to learn syntax and portability capability makes it popular these days

3.3.3 Jupyter notebook

Jupyter Notebook is an open-source web-based interactive computing environment that allows users to create and share documents containing live code, equations, visualizations, and narrative text. It supports various programming languages, including Python, R, and Julia.

It is a fully open-source product, and users can use every functionality available for free. Jupyter notebooks mostly used in data science workflows such as machine learning experimentations and modeling. A Jupyter notebook has independent executable code cells that users can run in any order.

3.3.4 LabelImg

LabelImg is a free, open-source software program for labelling images using graphs. The software was released by Tzutalin in 2015 and is written in Python; it uses QT for its GI (graphical interface). LabelImg is a straightforward and basic tool to label a few hundred images to create a dataset for computer vision model training.

3.4 LIBRARIES USED

3.4.1 OpenCV (Open-Source Computer Vision Library)

OpenCV (Open-Source Computer Vision Library) is an open-source computer vision and machine learning software library. It provides a comprehensive set of tools and functions that enable developers to process, analyze, and manipulate visual data, such as images and videos.

OpenCV supports a wide range of programming languages, including Python, C++, Java, and more. It has bindings for these languages, allowing developers to utilize its functionality within their preferred programming environment.

OpenCV is widely used in various domains, including robotics, augmented reality, autonomous vehicles, surveillance systems, medical imaging, and more. It offers a vast collection of functions and algorithms, making it a powerful tool for computer vision tasks.

3.4.2 TensorFlow

TensorFlow is a Python library for fast numerical computing created and is maintained by Google and was released under the Apache 2.0 open-source license. The API is nominally for the Python programming language, although there is access to the underlying C++ API.

It is a foundation library that can be used to create Deep Learning models directly or by using wrapper libraries that simplify the process built on top of TensorFlow. Unlike other numerical libraries intended for use in Deep Learning like Theano, TensorFlow was designed for use both in research and development and in production systems. TensorFlow installation comes with a number of Deep Learning models that you can use and experiment with directly.

3.4.2.1 TensorFlow Object Detection API

The TensorFlow Object Detection API is an open-source framework built on top of TensorFlow that makes it easy to construct, train and deploy object detection models. There are already pre-trained models in their framework which are referred to as Model Zoo. It includes a collection of pre-trained models trained on various datasets such as the COCO (Common Objects in Context) dataset, the KITTI dataset, and the Open Images Dataset.

3.5 Flow Chart of System

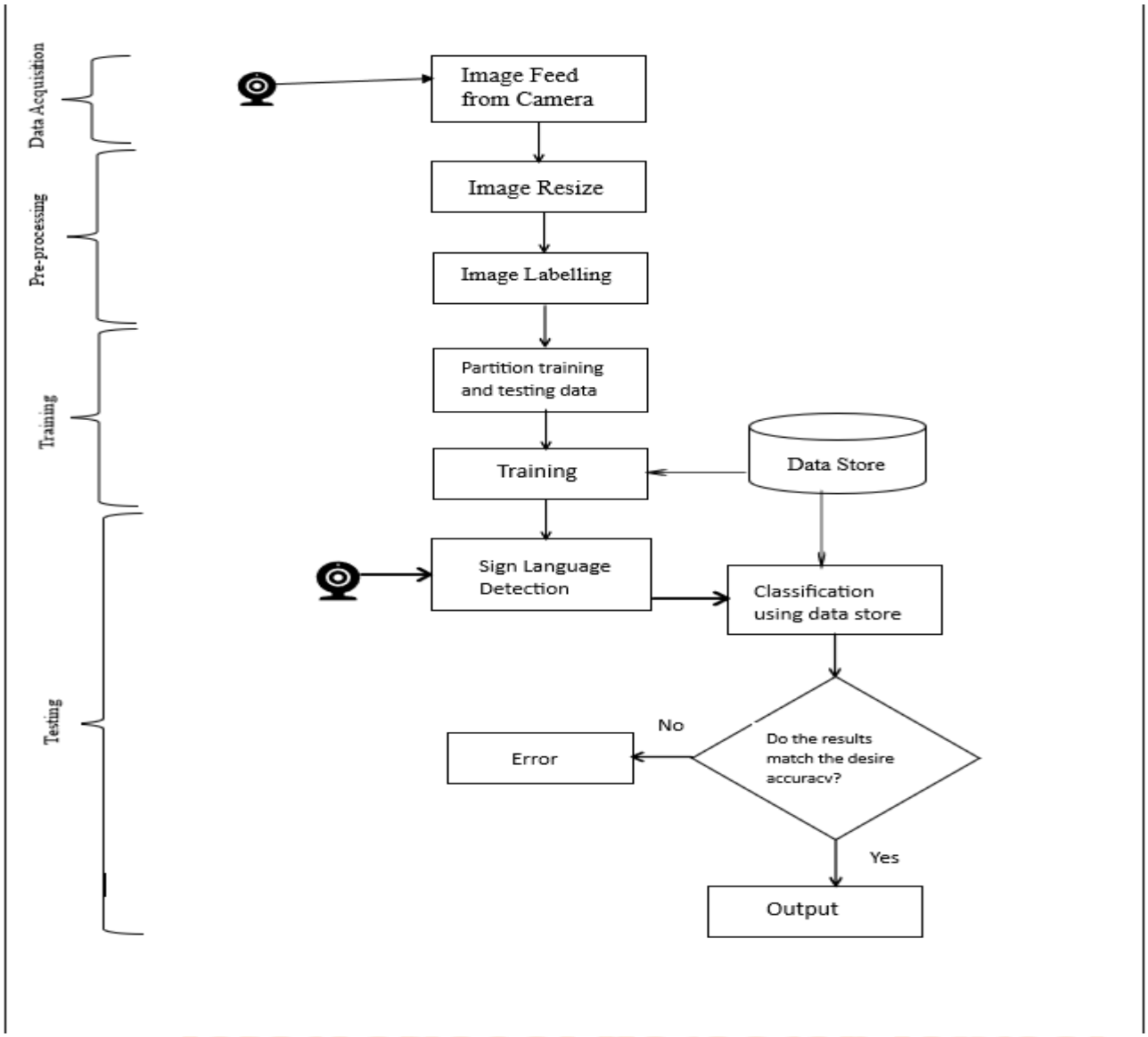

fig. flowchart of sign language detection system

Data acquisition: In this step, the system initializes and opens the camera to capture the feed of the signer's gestures. After the camera is initialized and configured, it continuously captures a series of frames. Each frame represents a snapshot of the signer's hand gestures and expressions.

Pre-processing: The system isolates and extracts the region of interest containing the hand gestures from the whole image. This step helps to focus on the relevant information for further analysis. Subsequently, image labelling is performed, where each cropped sign gesture is assigned a corresponding label or class to indicate the intended meaning. These pre-processing tasks prepare the data by segmenting the gestures and providing labelled.

Training: The labelled data, obtained from the pre-processing step, is divided into two sets: the training data and the testing data. The training data folder contains a portion of the labelled examples, which is used to train the model. The testing data folder holds the remaining labelled examples and is used to evaluate the trained model's performance. The training process utilizes the Single Shot Multi-Box Detector (SSD) algorithm, a popular object detection algorithm, to train the model on the labelled sign language gesture data, enabling it to learn and recognize gestures accurately.

Testing: In this step, the system is ready to detect sign language gestures in real-time scenarios. It starts by activating the camera to capture the live video of the signer's gestures. The deaf or dumb person performs sign language gestures, then system use to check and matches the detected gestures with the labelled data to determine the corresponding meaning. Finally, the system provides an output, which can be in the form of generating a text representation.

RESULT

In this sign language detection system, we have successfully implemented a machine learning-based system using the SSD (Single Shot Multi-Box Detector) model, TensorFlow, TensorFlow and OpenCV. This project has provided valuable insights and knowledge in building a sign language detection system. While the system has several important features, it also has some limitations.

Sign language detection system can provide following features:

Open Camera: The system incorporates the functionality that enables the system to access and utilize the camera feed for sign language detection. This feature allows the system to capture images from a connected camera device, such as a webcam, and process them for sign language recognition.

Image Capture: The image capture feature refers to the functionality that allows the system to capture images of sign language gestures for analysis and recognition. This feature enables users to record their sign language interactions or performances.

Sign Language Detection: By leveraging the SSD model, the system performs real-time detection and recognition of sign language gestures. It can identify and classify various sign language signs or actions accurately and efficiently.

TensorFlow Object Detection API: The system utilizes the TensorFlow Object Detection API, which provides a framework for training and deploying object detection models. This API streamlines the process of building and implementing the SSD model for sign language detection.

OpenCV Integration: OpenCV, a popular computer vision library, is integrated into the system to handle image and video processing tasks. It provides essential functionalities for capturing camera frames, performing image transformations, and facilitating real-time detection and tracking.

LabelImg Annotation: The system employs LabelImg, an annotation tool, to label and annotate sign language datasets. This tool aids in the preparation of training data by allowing users to draw bounding boxes around sign language gestures and assign corresponding labels.

Limitation of our sign language detection system are followings:

Generalization to New Signs: The system's ability to recognize new or unseen sign language gestures may be limited. It may struggle with accurately detecting and interpreting gestures that were not part of the training dataset, requiring additional data collection and model updates.

Environmental Factors: The performance of the system may be affected by environmental factors such as lighting conditions, background noise, or occlusions. These factors can introduce noise into the camera feed and impact the accuracy of gesture recognition.

Hardware Requirements: The sign language detection system may require certain hardware specifications, such as a camera/webcam or dedicated processing unit, to function effectively. The availability and compatibility of such hardware may limit the system's accessibility and usability.

Limited Dataset: Insufficient or unrepresentative training data may lead to suboptimal performance and limited recognition capabilities.

CONCLUSION

The sign language detection project has immense importance in promoting accessibility, inclusivity, and effective communication for individuals with deafness and who are unable to speak. It leverages technology to break down communication barriers and create a more inclusive and accessible society for all.

Inclusive Communication: Sign language is the primary mode of communication for many individuals who are deaf or hard of hearing. By developing a sign language detection system, we aim to bridge the communication gap between the deaf community and the hearing population. This technology enables effective and inclusive communication, fostering understanding and equal opportunities for individuals who rely on sign language.

Real-Time Detection: The ability of the system to detect and interpret sign language gestures in real-time. This is particularly beneficial in scenarios where immediate understanding and response are required.

Independence and Autonomy: Sign language detection technology empowers individuals with deafness to communicate independently, reducing their reliance on interpreters or intermediaries. It enhances their autonomy, allowing them to express their thoughts, needs, and emotions directly to others without barriers.

Technological Advancements: The development of a sign language detection system combines various cutting-edge technologies such as computer vision, machine learning, and real-time processing. This project pushes the boundaries of technological advancements, contributing to research and innovation in these fields.

Empathy and Social Awareness: By raising awareness about the challenges faced by individuals with deafness and the importance of sign language, the project fosters empathy, understanding, and inclusivity within society. It encourages the recognition of diverse communication needs and promotes a more inclusive and accepting community.

In conclusion, the journey of creating a sign language detection project has enriched my understanding of sign language, advanced my technical skills in computer vision and machine learning, and highlighted the significance of inclusivity and empathy in technology development. It has been a transformative experience that has expanded my knowledge and deepened my commitment to creating technologies that positively impact individuals with diverse communication needs.

FUTURE SCOPE

The sign language detection system project has promising future scopes that can further enhance its capabilities and impact. Some potential future directions for the project include:

Expansion of Sign Language Vocabulary: Currently, sign language detection systems typically focus on recognizing a limited set of commonly used sign language gestures. Future scope involves expanding the system to recognize a broader vocabulary of sign language gestures, including complex and specialized signs. This would make the system more versatile and useful for a wider range of applications.

Multi-modal Integration: Integrating multiple modalities, such as incorporating facial expression recognition alongside hand gesture detection, can enhance the system's ability to interpret sign language more comprehensively. The inclusion of facial expressions can significantly improve the accuracy and nuance of sign language interpretation.

Integration with Assistive Technologies: Integrating the sign language detection system with assistive technologies, such as text-to-speech or speech-to-text systems, can facilitate bi-directional communication between sign language users and non-sign language users. This integration would enable seamless translation between sign language and spoken/written language, expanding communication possibilities.

Mobile and Wearable Applications: Optimizing the sign language detection system for mobile devices or wearable technology would enable greater accessibility and portability. Users could utilize the system on their smartphones or other wearable devices, providing on-the-go communication support and accessibility in various environments.

REFERENCE

[1] Bohra, T., Sompura, S., Parekh, K., & Raut, P. (2019, November). Real-Time Two-Way Communication System for Speech and Hearing-Impaired Using Computer Vision and Deep Learning. In 2019 International Conference on Smart Systems and Inventive Technology (ICSSIT) (pp. 734-739). IEEE.

[2] Recognition of Indian Sign Language in Live Video by Joyeeta Singha and Karen Das. Various alphabets of Indian Sign Language are proposed where continuous video sequences of the signs have been considered. International Journal of Computer Applications (0975 – 8887) Volume 70– No.19, May 2013.

[3] Mahesh Kumar proposed a system which can recognize 26 hand gestures of Indian Sign Language based on Linear Discriminant Analysis (LDA). Conversion of Sign Language into Text. International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 9 (2018) pp. 7154-7161.

[4] https://www.anaconda.com/products/individual

[5] https://github.com/protocolbuffers/protobuf/releases

[6] https://github.com/tensorflow/models

[7] https://github.com/tzutalin/labelImg