Realtime Isolated ASL Gesture Sequence Recognition

DGMD E-14: *Wearable Devices and Computer Vision*

Harvard Extension School
HARVARD DIVISION OF CONTINUING EDUCATION

Julia Beltrame ⬢ Wilson Guo ⬢ Brian Sam-Bodden ⬢ Taha Teke

Project Summary & Key Results

Summary

The goal of the project is to facilitate communication for the deaf and hard-of-hearing community by providing a real-time solution to the problem of translating ASL (American Sign Language) into their corresponding English word or words.

The projects main products are several machine learning models used to power two prototype applications:

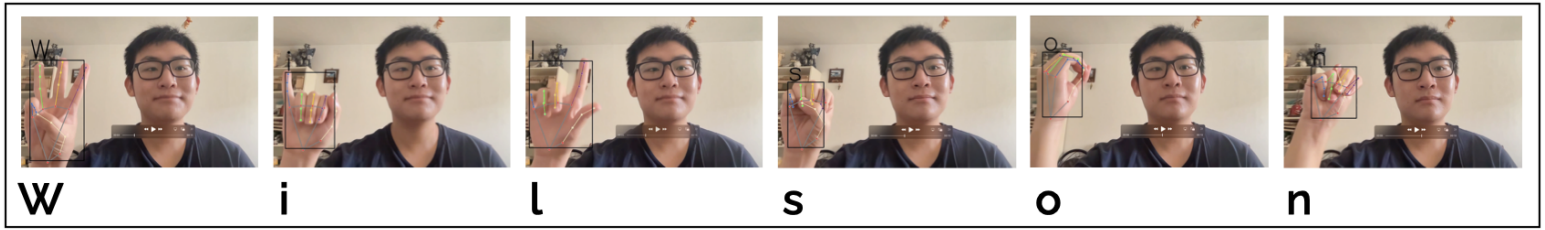
1. **ASL Alphabet Fingerspelling**: Detect and classify the hand shapes that correspond to the letters of the ASL Manual Alphabet
2. **Isolated ASL Signs Classifier:** Process a live video feed and generating an onscreen classification for a sequence of hand, face, and body gestures representing an ASL “gloss”.

From an academic standpoint, this project helped us learn and discover in the field of machine learning models. We have gained substantial expertise in the emerging field of Graph Neural Networks (GNNs), and Random Forest algorithms. These models have been instrumental in the development of our applications, providing the backbone for accurate gesture recognition and translation.

Results

- Several **Random Forest** models where tested for **Fingerspelling** hand configuration **classification**
- The **best performing model** was integrated in the **Fingerspelling demonstration** application
- The fingerspelling model and application were used to inform the development of the ASL gesture recognition
- Several **GNN models**, using different architectures were designed, trained and tested
    - Different techniques for the design of the training graphs were tested with an emphasis in capturing both the **spatial** and **temporal** characteristics of the ASL gesture sequences
- The best performing model with a reduced class population (6) was selected to be integrated with the realtime demo isolated ASL gesture recognition demonstration application
- Extensive work went into the client application for converting **capture landmarks** into an **input graph** for **GNN inference**

Fingerspelling Demonstration Application



- The fingerspelling application using Google's MediaPipe to capture Hand Landmarks
- A Random Forest classification model trained on the ASL alphabet is use to classify the captured landmarks
- A bounding box is drawn with the classifier's label for the highest predicted class

ASL Signs Classifier Demonstration Application

40 Frames Capture ~ 4 seconds

- ASL Signs Classifier Demonstration demo captures **40 frames** of video
- **Selected Landmarks** for **hands**, **face** and **pose** are extracted
- An **input graph** is created by using the **natural anatomical connections** between the landmarks
- The input graph is fed into a **GNN** which produces a **class probability distribution** which is then turned into a **class label** which is displayed on screen

Background
& Previous Works

Background - Relevance

While there are differences between spoken languages, there is a much larger difference between oral languages and signed languages. In order for people in both communities to understand each other, the need to have a proficient human translator is necessary.

With Machine Learning we can provide a basic way for an ASL Signer to be understood by somebody not proficient in ASL. As the ML models and techniques get more efficient, the potential for on-device inferences in real-time become a reality.

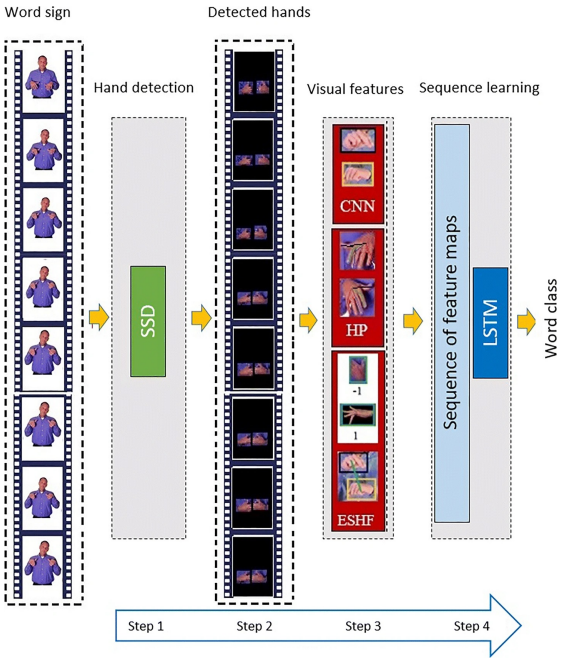
Background - Problem

Sign Language Translation

- Is a complex classification problem
- Involving Seq to Seq Tasks
- Mixing appearance, motion and pose

Typical Architecture for multiframe, seq-2-seq classification learning from “Video-based isolated hand sign language recognition using a deep cascaded model.”

Multimed Tools Appl 79, 22965–22987 (2020).



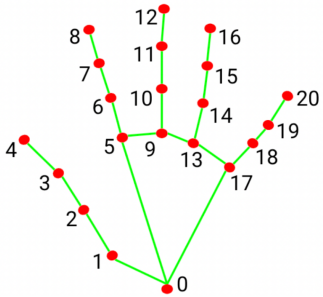
Background - Technology (COMBINED WITH 11)

ASL is a visual language where meaning is conveyed through specific hand configurations, movements, and positions in relation to the face and body. These landmarks serve as the fundamental units of this lexicon. Today, the most accurate/efficient classifiers work on raw pixel data from images. The challenge and opportunity of using landmarks is that they provide a much more compact way to carry the semantics of ASL. Besides the raw provided landmark coordinates, the opportunity to engineer features from the relationship between those landmarks is ripe for exploitation. Also, the “graph nature” of landmark data opens the possibility of different ML architectures to be explored, such as GNNs (Graph Neural Networks).

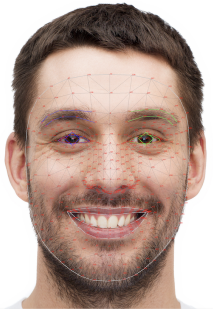
Background - Technology

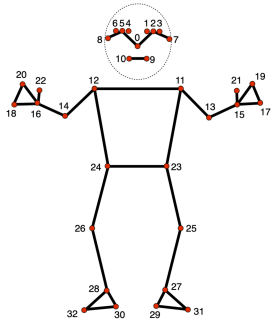
Landmarks are points in 3D space represented by x, y, and z coordinates, as well as a z-value representing the estimated distance from the camera.

For ASL detection, landmarks allow us to express in a compress form the important data about hand position, pose and facial expression. When using multiple landmarks as it is needed for ASL we end up with semantic graph or relationships between hands, faces, and pose, from which other interesting relationships can be extracted.



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





Previous Works

- **Time Series Neural Networks For Real Time Sign Language Translation**:
  - Introduction of neural networks for ASL gloss recognition and English sentence translation”
  - *Kumar, S. S., Wangyal, T., Saboo, V., & Srinath, R. (2018)*
- **Spatial Temporal Graph Convolutional Networks for Skeleton-Based Action Recognition**
  - Spatial Temporal Graph Convolutional Networks for Skeleton-Based Action Recognition
  - *Cao, Z., Hidalgo, G., Simon, T., Wei, S. E., & Sheikh, Y. (2020)*
- **Graph Neural Networks for temporal graphs**
  - Comprehensive overview of GNNs for temporal data analysis
  - *Scarselli, F., Tsoi, A. C., Hagenbuchner, M., & Monfardini, G. (2023)*
- **Two-Stream Adaptive Graph Convolutional Networks for Skeleton-Based Action Recognition**
  - Advanced graph convolutional network approach for action recognition
  - *Shi, L., Zhang, Y., Cheng, J., & Lu, H. (2018)*

Previous Works (cont.)

- **Real-Time Sign Language Detection using Human Pose Estimation**:
  - Focus on real-time sign language detection with pose estimation
  - *Kocaballi, S., Benligiray, B., Topal, C., & Akinlar, C. (2020)*
- **MediaPipe's Landmarks with RNN for Dynamic Sign Language Recognition**
  - Combines MediaPipe with RNN for dynamic sign language recognition
  - *Kishore, R. R., & Kumar, A. (2022)*
- **An integrated mediapipe-optimized GRU model for Indian sign language recognition**
  - Integration of MediaPipe and GRU for sign language recognition.
  - *Kumar, A., Tripathi, S., & Varshney, P. (2022)*
- **Indian Sign Language Recognition using MediaPipe Holistic**
  - MediaPipe Holistic application in Indian Sign Language recognition
  - *Kumar, A., & Kishore, R. R. (2023)*

Technologies & Methods

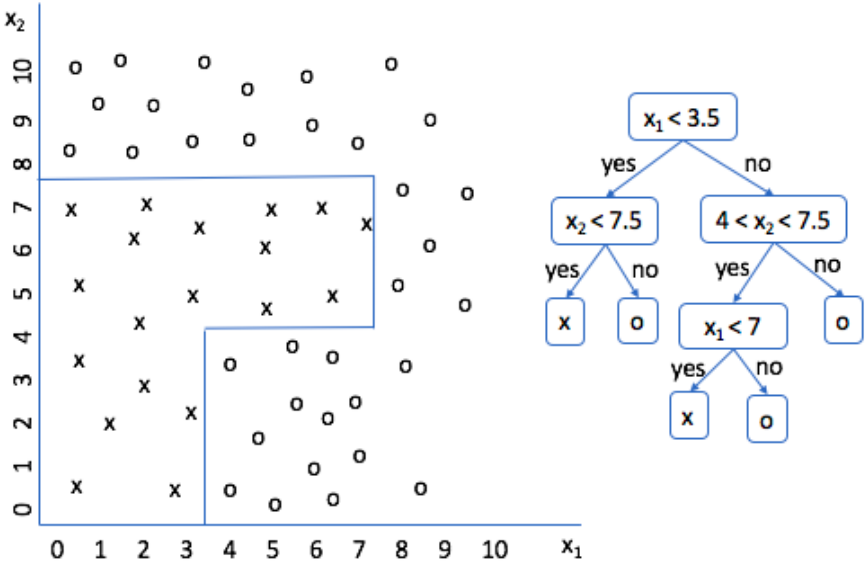
Overall Strategy

- Our methodology involves a two-pronged approach:
  - **Learning** Phase
    - Develop a fingerspelling recognition model using well-known techniques
    - Learn the ins-and-outs of MediaPipe and OpenCV
    - Work with Landmarks Data
  - **Experimentation** Phase
    - Explore the world of GNNs and establish a base model
    - Iterate and experiment with different ways:
      - Build training graphs (topology), number of landmarks used
      - Model architecture: explore different GNN layer types
      - Temporal Data: Find way to incorporate temporal data into the graph's structure
      - Data augmentation: Experiment with different data augmentation techniques
  - **Implementation** Phase
    - Fine-tune as much as possible: hyper-parameters, imputation, etc.
    - Implement a Real-time capture Python desktop application with the GNN model with the highest validation accuracy

Random Forest Models Introduction

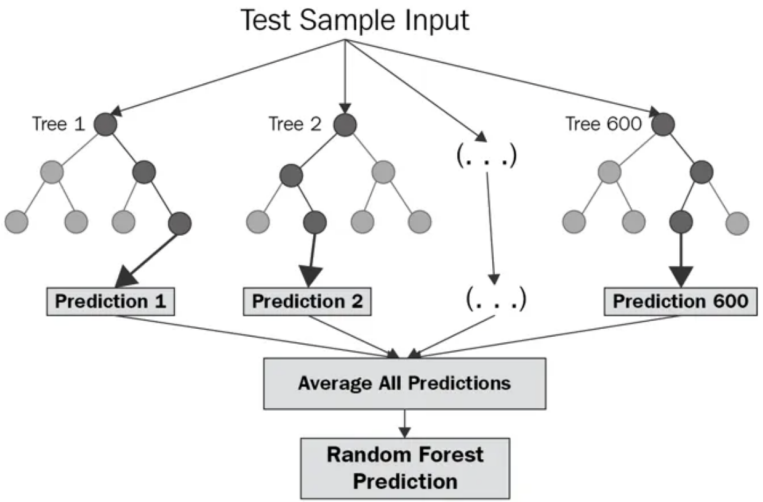
At their heart, Random Forest Models boil down to:

1. Number of Tree Classifiers
2. Depth of trees



Random Forest Models For Classification

- Each tree makes their own prediction on the recorded ASL sign
- The greatest plurality of the predictions is returned by the random forest

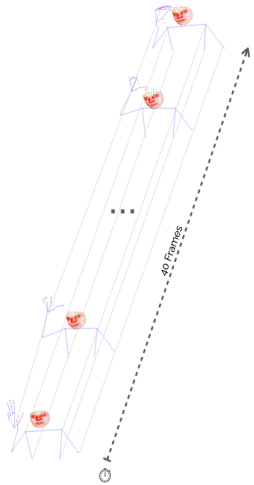


GNN Isolated ASL Gesture Recognition Model

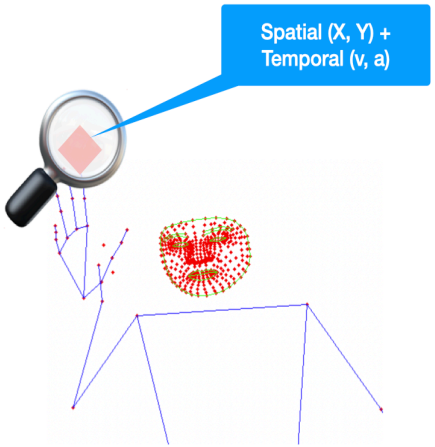
- The architecture of our GNN is a “Spectral Graph Neural Network”, specifically a “Graph Convolution Neural Network” (GCN)
- Well-suited for tasks involving processing of graph-structures where spectral properties and topological structure are essential
- For the task of classifying ASL signs from video sequences, the use of a Spectral GNN like GCN is well-suited for capturing the complex spatial relationships inherent in sign language gestures.

- The architecture features 2 **GCNConv** Layers as it core components, designed to process graph-structured data by applying “graph convolutions”
- The GCNConv layers capture the spatial relationships between nodes
- The first layers expands the input from 6 features (x, y, 2 velocities, 2 accelerations) to 512
- The second layer further expands the network to 1024 features
- Batch Normalization and Layer Normalization are utilize to stability and standardize the data
- Linear layers provide feature transformation and dimensionality reduction
- A Dropout layer is provided to prevent overfitting

```
ASLGraphClassifier(
  (conv1): GCNConv(6, 512)
  (bn1): BatchNorm(512)
  (conv2): GCNConv(512, 1024)
  (bn2): BatchNorm(1024)
  (ln1): LayerNorm(1024, affine=True, mode=graph)
  (lin1): Linear(in_features=1024, out_features=512, bias=True)
  (ln2): LayerNorm(512, affine=True, mode=graph)
  (lin2): Linear(in_features=512, out_features=6, bias=True)
  (dropout): Dropout(p=0.7, inplace=False)
)
```

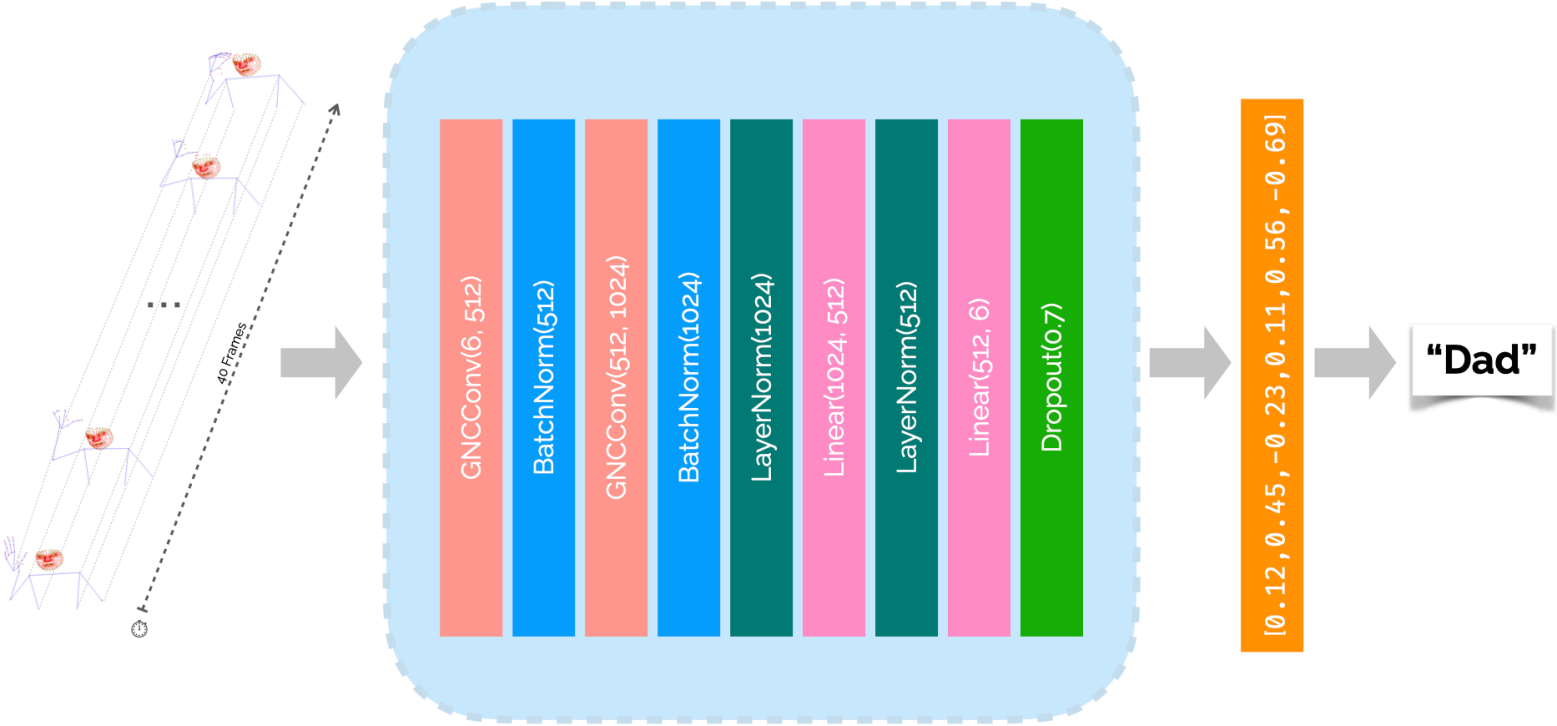


Multi-frame Input Graph



Single-frame Graph

- Each video frame consists of landmarks for hands, face and pose
- A selected subset of those landmarks is kept for building the training graphs
- All videos are interpolated/downsampled to 40 frames
- We drop the z-coordinate, normalize the coordinates and smooth the landmarks
- The we construct a graph using the natural anatomical connections
- Temporal features (velocity and acceleration) are calculated in-between frames
- Adjacent frames are connected landmark-to-landmark to create a training graph for the ASL Example Video



40-Frame-Graph

Graph Convolutional Neural Network

Output

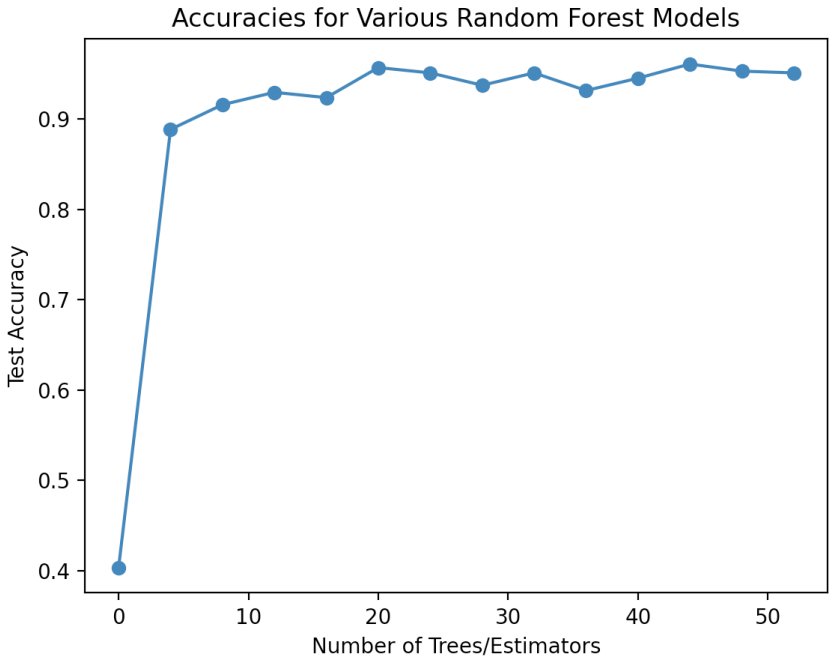
Technologies

- Data
  - **Random Forest**: Custom dataset signing the ASL letters
  - **GNN**: “Google - Isolated Sign Language Recognition” Kaggle Competition: https://www.kaggle.com/competitions/asl-signs .
- Programming Languages + General Libraries
  - Python w/ NumPy, MatplotLib, scikit-learn, seaborn, pickle
  - Heavy use of Jupyter notebooks for experimentation
  - Google Collab Pro+ with A100 GPUs
- AI/ML Frameworks
  - PyTorch, PyTorch Geometric
- ML Model Architectures
  - Random Forest, GNNs (Graphical Neural Networks)
- CV Frameworks
  - MediaPipe, OpenCV

Results, Evaluation & Demo Snippets

Random Forest Results

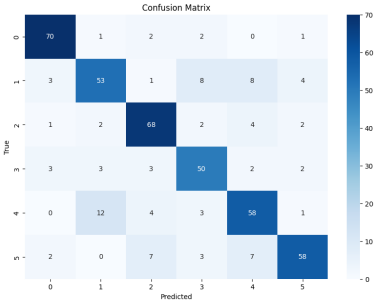
- For our random forest Model, we settled on a depth of five and twenty trees or estimators
- While there were random forest models that had greater accuracies as show in the graph to the right, we wanted to be able to generalize well and not overfit the training data.



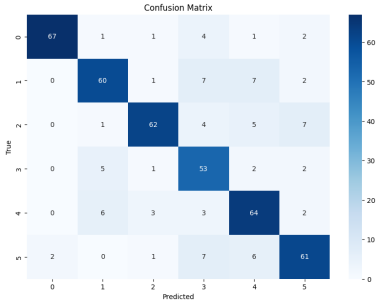
Random Forest

	XS	S	M	L	XL
Final Validation Accuracies	79.3%	81.6%	80.7%	82.0%	80.9%

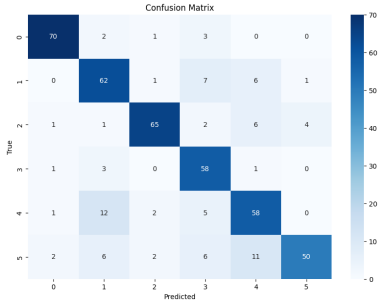
XS



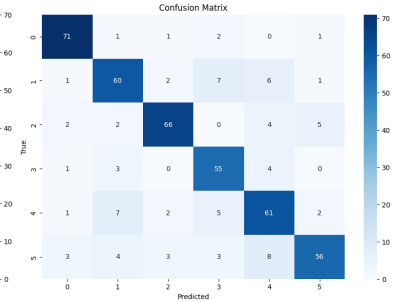
S



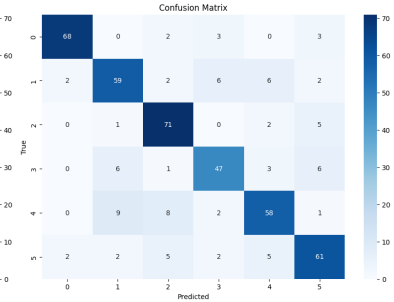
M



L



XL



Graphic Neural Network

Conclusion & Next Steps

Conclusions

- Many different GNN architectures
- Adopted **PyTorch Geometric** for GNN development after maintenance problems with PyTorch Geometric Temporal
- Implemented various **data augmentation** techniques to enhance model training and robustness;
- Explored **multiple methods** for constructing training graphs to accurately represent ASL gestures.
- Experiments in **feature engineering**

Future Work

There are a few areas that we will like to further explore:

- For ASL in particular, analyzing the performance of model with different combinations of hand, pose and face landmarks (which ones carry the most semantic payload in signing)
- Feature engineering; although most architectures nowadays rely on the network to extract features, targeted feature engineering could reduce the dimensionality of the problem (for example, the number of hand configurations is a bounded set)
- Temporal features; we explored the simplest and most obvious of temporal features, velocity and acceleration changes in a landmark from frame-to-frame. Exploring other techniques like optical flow
- Other GNN techniques: 3D Convolutional Neural Networks (3D CNNs), Time-Distributed CNNs, and Temporal Convolutional Networks (TCNs)

Thank you!