Real-time American Sign-Language Recognition via Transfer Learning

Maria F. Harris, Roumen Guha

GitHub

Table of Contents

- Introduction of the topic and its importance/background (Roumen)
- Literature review (Maria)
- Details on the dataset (Roumen)
    - Explaining the two datasets
    - What makes them unique (slight hand gesture rotation)
    - Size of each dataset and overall description
    - Mean subtraction
- Approach (Maria)
    - Transfer learning on two models: VGG16 and ResNet50
    - Retraining the classification layers only
    - Reason for not using fine tuning
    - Multiple experiments using different FC layers, and A-E/A-Z, dropout shizz
- Model Descriptions (Maria)
    - ResNet50 architecture (should be brief)
    - VGG16 architecture (brief)
- Experiments and results of training (Roumen)
    - List all the four experiments, confusion matrix and their plots
    - Describe the plots briefly and results
    - Issues?
- Results for real-time translation: (Maria)
    - Discuss the results of A-E in real-time
    - Best model results
    - Problems while using real-time video
        - Position invariance
        - Misclassification of similar looking signs
        - J and Z motion based not detected
        - Complexion based discrimination
        - Lighting effect?
    - Play a video
    - Improvements? (Roumen)
        - Better dataset
    - Q/A (Maria and Roumen)

Project Motivation

Around one million people use American Sign Language (ASL).

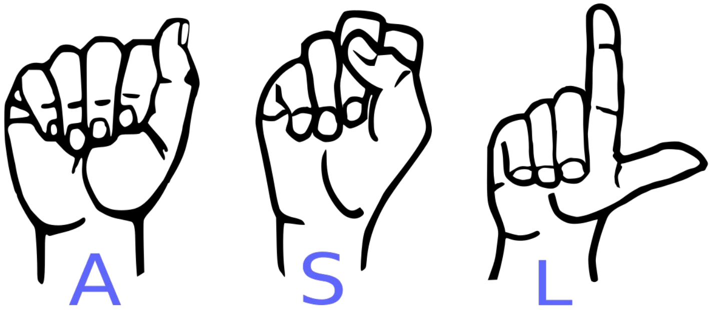
However, it is not easy for users of ASL:

- 98% of deaf people do not receive education in sign language
- 72% of families do not sign with their deaf children
- 70% of deaf people don't work or are underemployed
- 1 in 4 deaf people has left a job due to discrimination
- 1 in 4 deaf women will be sexually assaulted in their lifetimes, compared to 1 in 10 hearing women

Src: https://www.newsweek.com/asl-day-2019-american-sign-language-1394695

Literature Review

- American Sign-Language recognition is not a new computer vision problem.
- Following work has been done on ASL:
    - K-Nearest Neighbors (D. Aryanie and Y. Heryadi)
    - Support Vector Machines (C. Sun, T. Zhang, B. Bao and C. Xu)
    - Hidden Markov Models (T. Starner and A. Pentland)
- Deep Learning Based Solutions:
    - CNN (Gao Q., Ogenyi U.E., Liu J., Ju Z., Liu H. (2020))
    - CNN + RNN (K. Bantupalli and Y. Xie,)



Sourcing Data

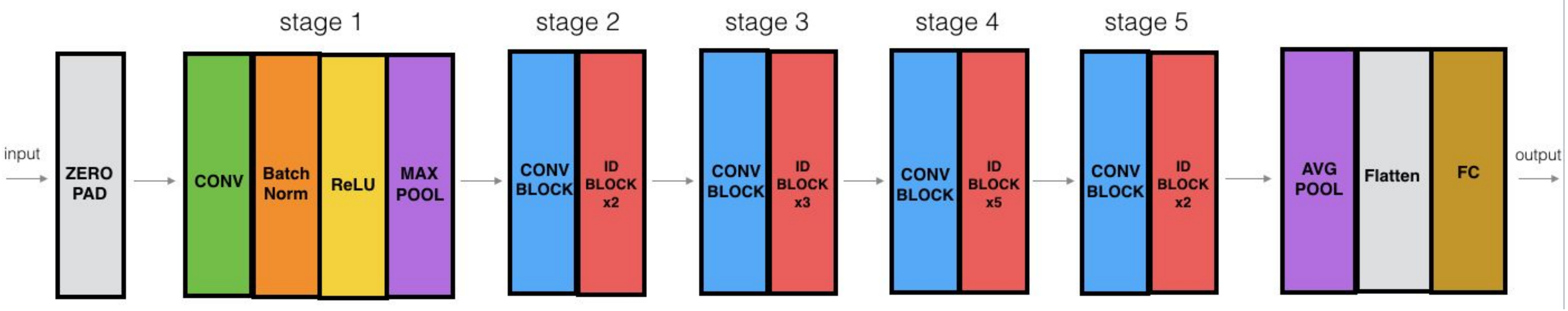
Used two Kaggle datasets with wildly different characteristics.

Project Approach

- Used transfer learning on two popular **pretrained** ConvNet image classification models:
  - ResNet50
  - VGG16
- Used Fixed Feature Extractor based Transfer Learning:
  - Removed the classification layer (FC layers)
  - Treated rest of the network as a fixed feature extractor
  - Trained a new classification layer (FC layers)
- Avoided training on all classes during model-investigation stage
- Finally trained chosen model on dataset of >27K images (1,070 per letter/26 letters).
- Checked performance of trained model:
  - Test set
  - Real-time video from user’s laptop webcam

Model Description - ResNet50

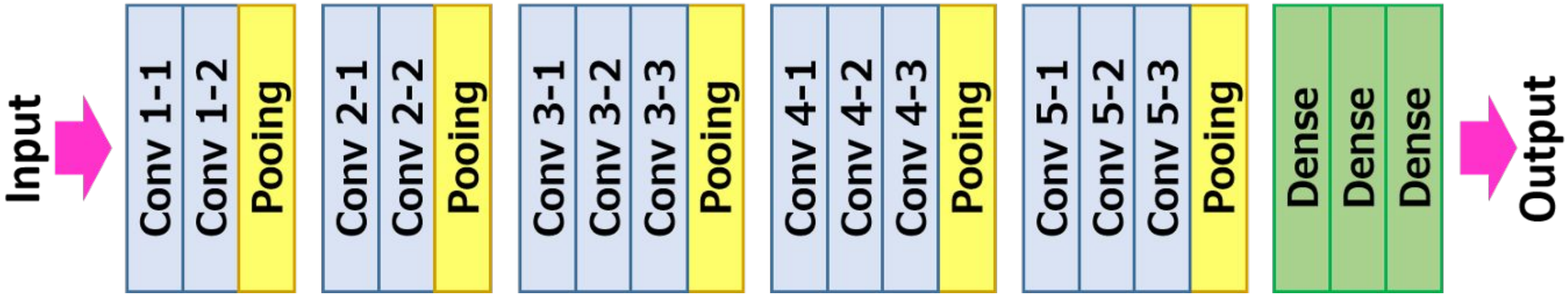
- Trained on ImageNet database (more than 14 million images)..
- Smaller variant of ResNet152
- Consists of 5 stages and 50 layers (1xFC layer)
- Uses skip connections
- Re-trained the FC layer



Picture credits:
https://towardsdatascience.com/understanding-and-coding-Aa-resnet-in-keras-446d7ff84d33

Model Description - VGG16

- Trained on ImageNet database (more than 14 million images) and winner of 2014 ImageNet challenge.
- Replaces large kernel-sized filters with smaller kernel-sized filters.
- Has 16 layers that have weights.
- 3x FC layers were re-trained via transfer learning
- Slow to train



Picture credits:
http://www.renom.jp/notebooks/tutorial/image_processing/neural-style-transfer/notebook.html

Model Training

Both ResNet50 and VGG16 were trained with the following configurations:

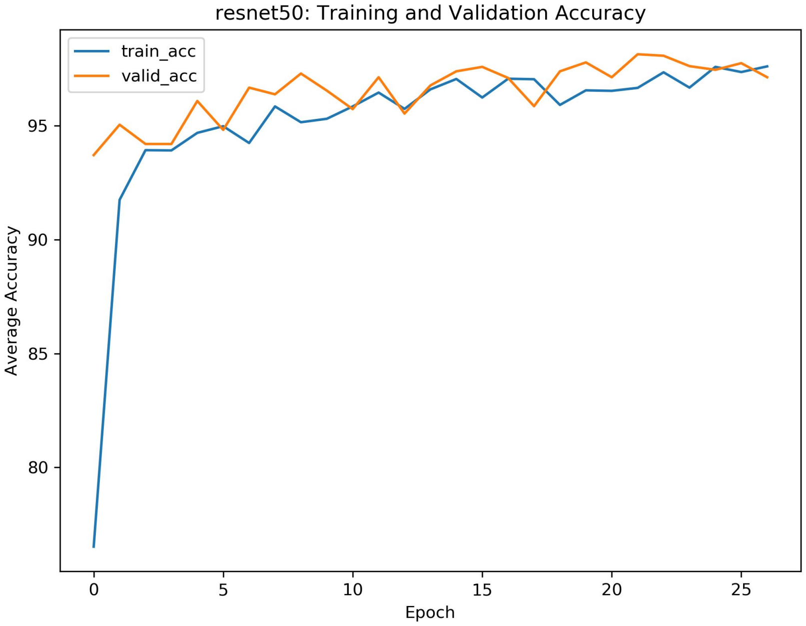
- Loss function: NLL Loss
- Optimizer: ADAM
- Initial Learning Rate = 0.001
- Betas = 0.9, 0.999
- Batch Size = 170
- Dropout = 0.2
- Training, validation, testing set split: 0.6, 0.2, 0.2

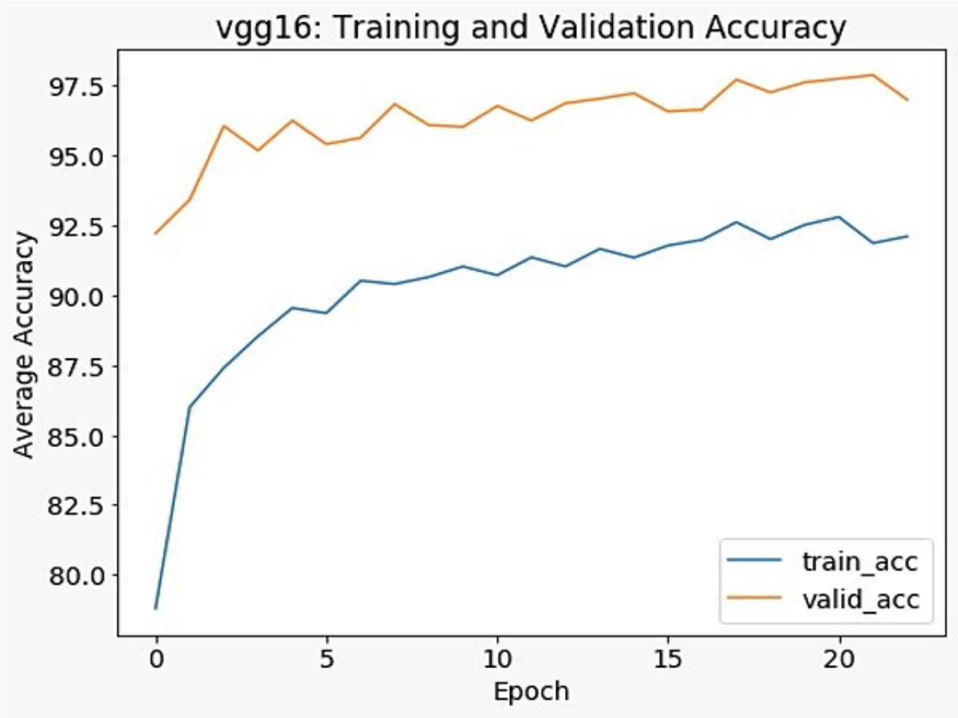
Transformations:

- Augmentation: random crop, random rotation, color jitter, random horizontal flip
- Standardize (match specs of ImageNet): crop to 224x224, and normalize with ImageNet mean and standard deviation

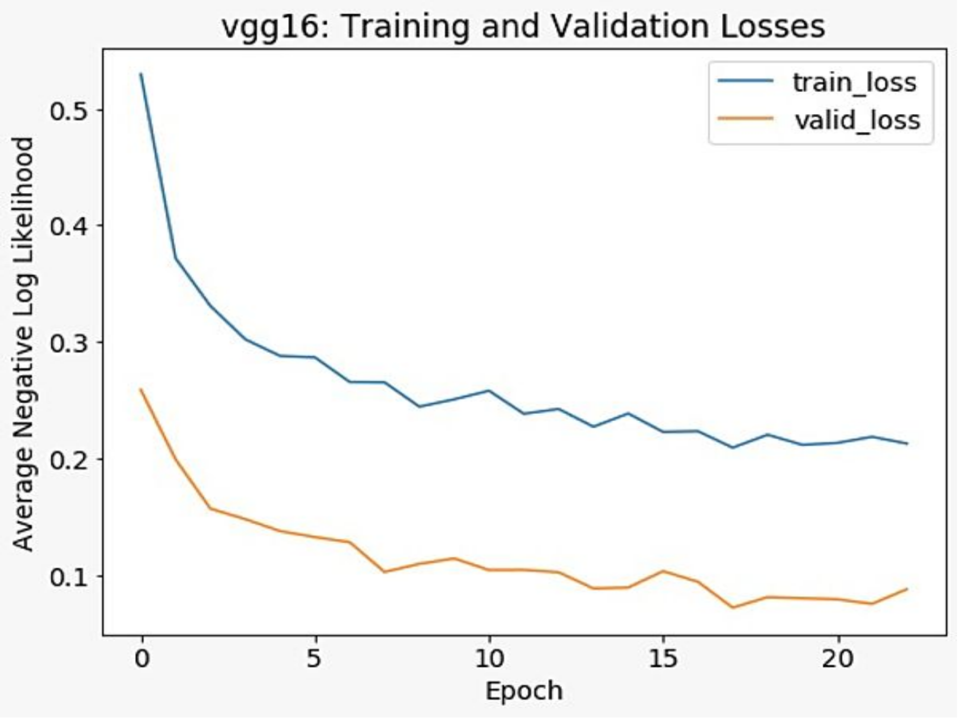
Src: https://mlfromscratch.com/optimizers-explained/

Comparison: A-E, **2**-FC Models: ResNet50 vs VGG16

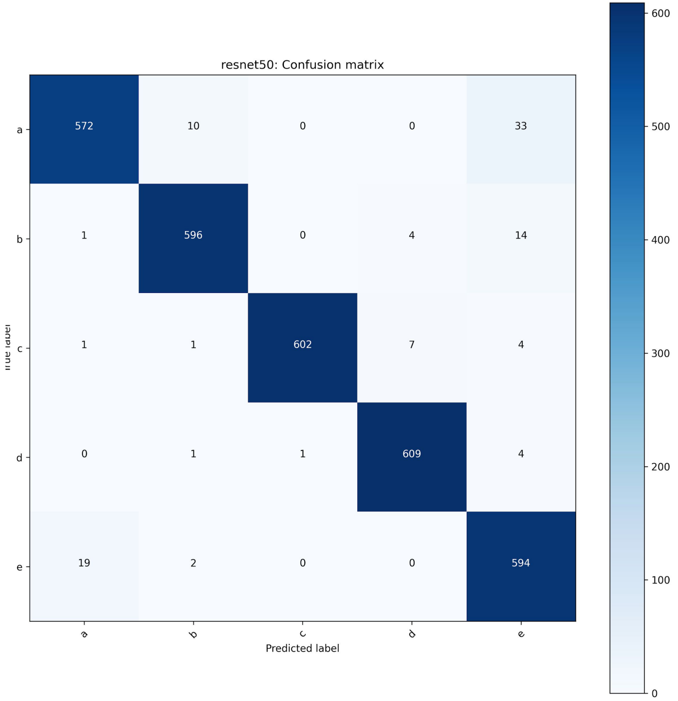


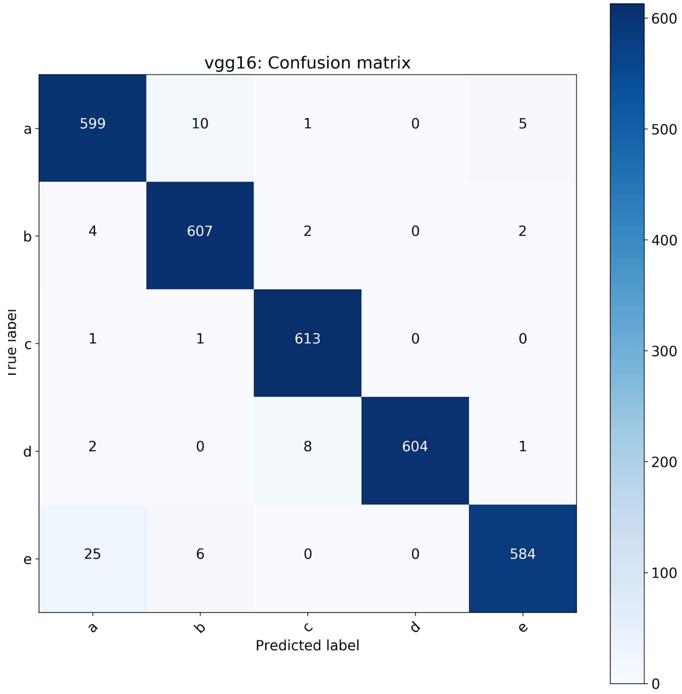


Comparison: A-E, **2**-FC Models: ResNet50 vs VGG16

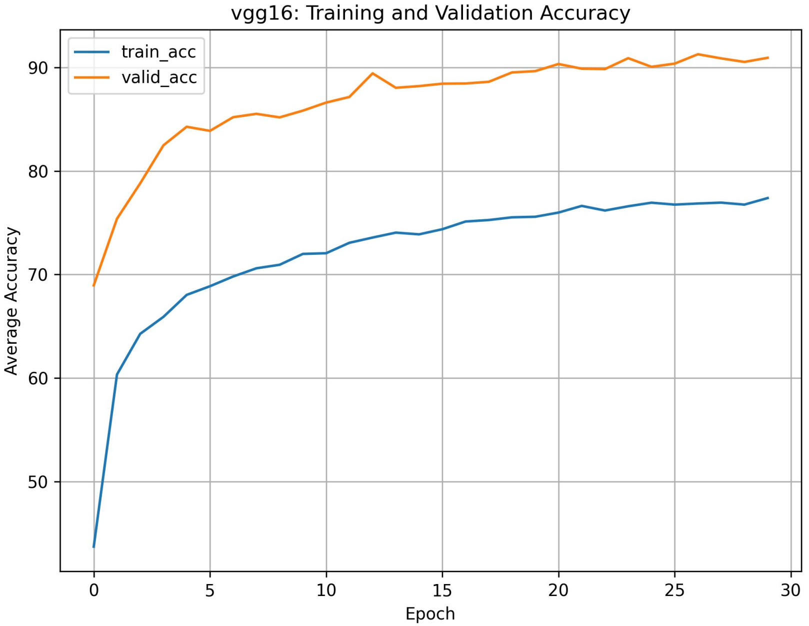


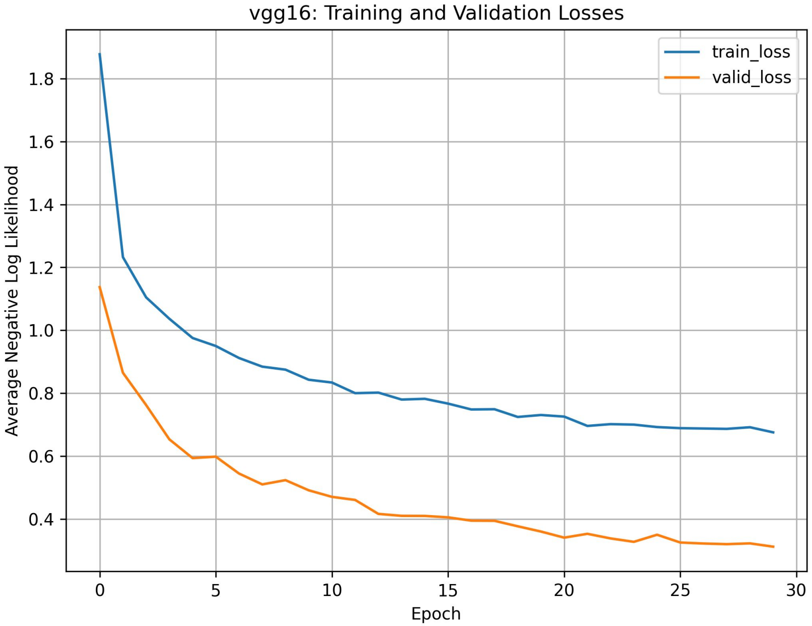
Comparison: A-E, **2**-FC Models: ResNet50 vs VGG16





Best Model's Results: VGG16, A-Z, **2**-FC

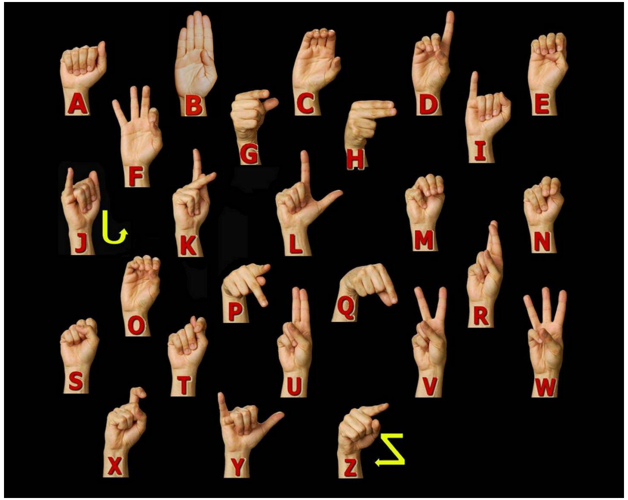




Best Model's Results: VGG16, A-Z, **2**-FC

In order of #misclassifications:

- G as H; U and R; V as U; P as Q; M and N



vgg16: Confusion matrix

True label \ Predicted label	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	199	5	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	3	2	0	0	0	0	1	0
B	0	211	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C	0	0	210	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D	0	1	2	203	0	0	0	0	1	0	2	0	0	0	2	0	0	2	0	0	0	0	0	0	0	0
E	6	7	0	0	191	0	0	2	1	0	0	0	0	1	0	0	0	0	4	0	1	0	0	0	0	0
F	0	0	0	0	0	210	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0
G	0	0	0	0	0	0	184	22	0	1	0	0	0	0	0	0	0	1	1	0	0	0	0	1	2	1
H	0	0	0	0	0	0	0	210	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
I	0	3	0	0	3	0	1	0	189	3	2	5	0	0	0	0	0	0	3	0	1	1	0	2	0	0
J	0	0	0	0	0	0	3	0	0	203	0	0	0	1	0	0	0	0	0	0	0	0	0	1	3	2
K	0	0	0	0	0	0	0	1	0	0	201	0	0	0	0	0	0	2	0	0	0	7	0	1	0	1
L	0	0	2	3	1	1	0	0	0	0	2	201	0	0	1	0	0	0	0	1	0	0	0	0	1	0
M	2	1	0	0	0	0	0	0	0	0	0	0	196	11	0	0	0	0	2	1	0	0	0	0	0	0
N	0	0	0	0	1	1	0	0	0	0	0	0	13	193	0	0	0	0	0	3	1	0	0	0	1	0
O	0	0	3	1	0	0	0	1	0	0	0	0	0	2	203	0	0	0	2	0	1	0	0	0	0	0
P	0	0	0	0	0	0	0	2	1	0	0	0	0	0	0	195	14	1	0	0	0	0	0	0	0	0
Q	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	209	0	0	0	0	0	0	0	0	0
R	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	194	0	0	16	1	0	0	0	0
S	2	1	0	0	3	0	0	0	0	1	0	0	0	1	0	0	0	1	186	4	2	1	0	7	4	0
T	3	0	0	0	0	0	0	0	0	0	0	2	0	0	2	0	0	0	5	189	2	0	0	2	5	3
U	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	17	0	0	193	1	0	1	0	0
V	0	0	0	0	0	0	0	0	0	1	5	2	0	0	0	0	0	4	0	0	14	181	5	0	1	0
W	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	0	1	5	203	0	1	0
X	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	5	13	0	3	0	0	187	2	1
Y	0	0	0	0	0	0	1	2	0	1	0	0	0	0	2	0	0	0	0	5	0	0	0	0	202	0
Z	0	0	0	0	0	0	0	0	0	4	1	0	0	0	1	0	0	7	0	0	1	0	0	2	1	196

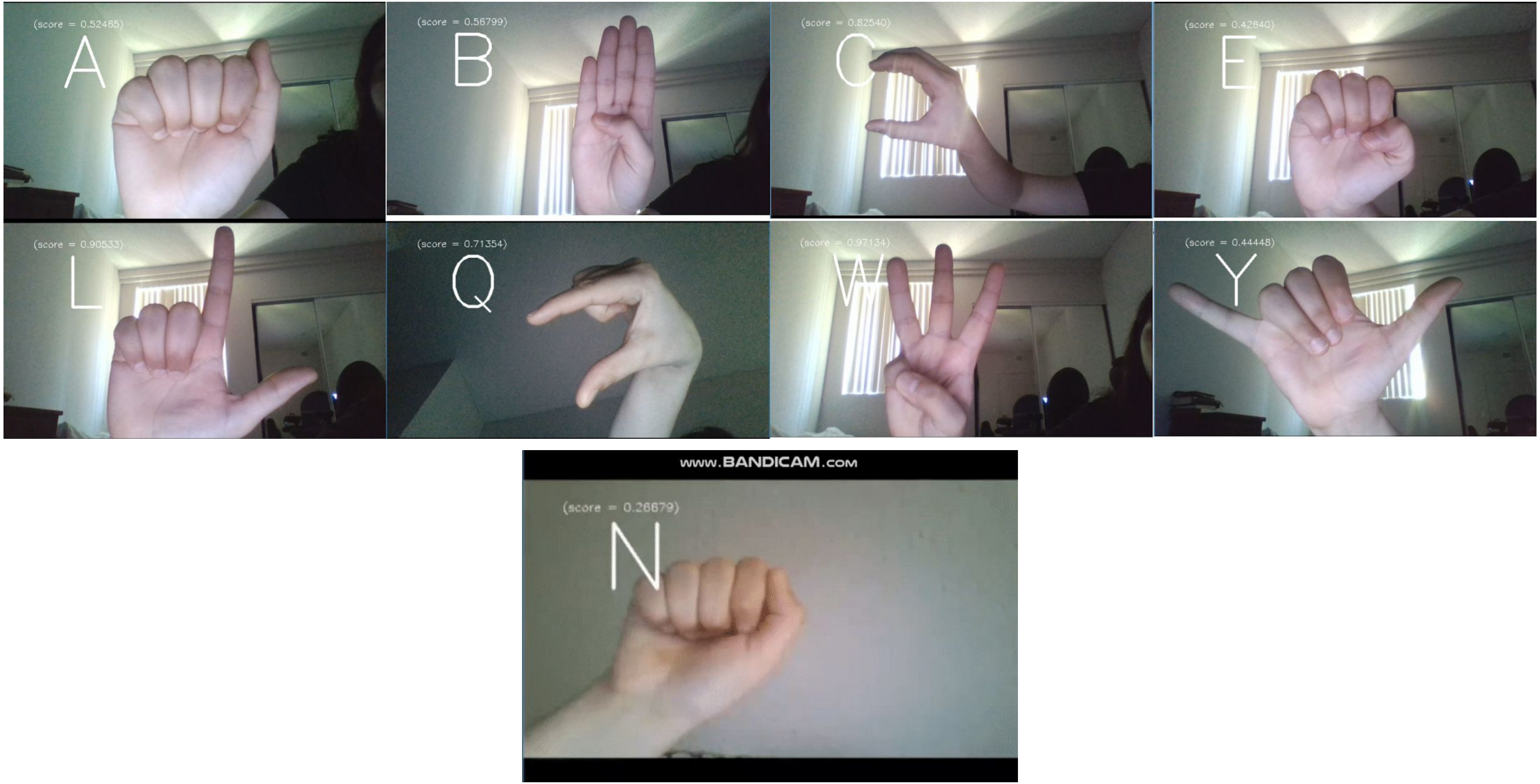
Summarized Experiment Results

Dataset	Network	#Epochs	#Linear	#Dropout	Acc (%)	Loss
A-E	ResNet50	30	2	1	96.9	0.04
A-E	VGG16	30	2	1	96.5	0.06
A-E	ResNet50	16	3	2	96.81	0.07
A-E	VGG16	27	3	2	93.1	0.08
A-Z	ResNet50	-	-	-	-	-
A-Z	VGG16	30	2	1	90.9	0.32
A-Z	ResNet50	-	-	-	-	-
A-Z	VGG16	30	3	2	91.2	0.33

Real-Time Classification

- Fed live video from webcam to model
- Model made predictions frame-by-frame
- VGG16 produced relatively best results in real-time translation (both 2 and 3 FC retrained models).
- ResNet50 produced the worst results.
- Following letters were consistently recognized:
    - A, B, C, E, L, O, Q, W and Y
- Following letters were recognized after significantly adjusting the position of the hand relative to the webcam:
    - R, U and V

Results - Correct Classification



Results - Misclassification

- No perceivable correlation between high accuracy on testing set and performance on real-time videos.
- **All** the experimented models were **not** invariant to the relative orientation of the hand from the camera.
- Potential reasons for misclassification:
  - Dataset did not contain hand poses at drastic angles.
  - Certain alphabets like ‘p’ and ‘q’, and ‘w’ and ‘f’ appear very similar
  - Letters ‘j’ and ‘z’ require motion - hard to classify them in frame-by-frame solution.
  - Lighting effect

Further Work

Possible avenues of investigation and improvement

- Implement an image pre-processing: crop, equalize, edge detection could all help
- Combine with dataset with more human diversity

Thank You!

GitHub