THE APPLICATION OF COMPUTER VISION, MACHINE AND DEEP LEARNING ALGORITHMS UTILIZING MATLAB

Andrea Linda Murphy

Master of Science in Information Technology, Candidate

University of New Hampshire

Spring 2020

Introduction

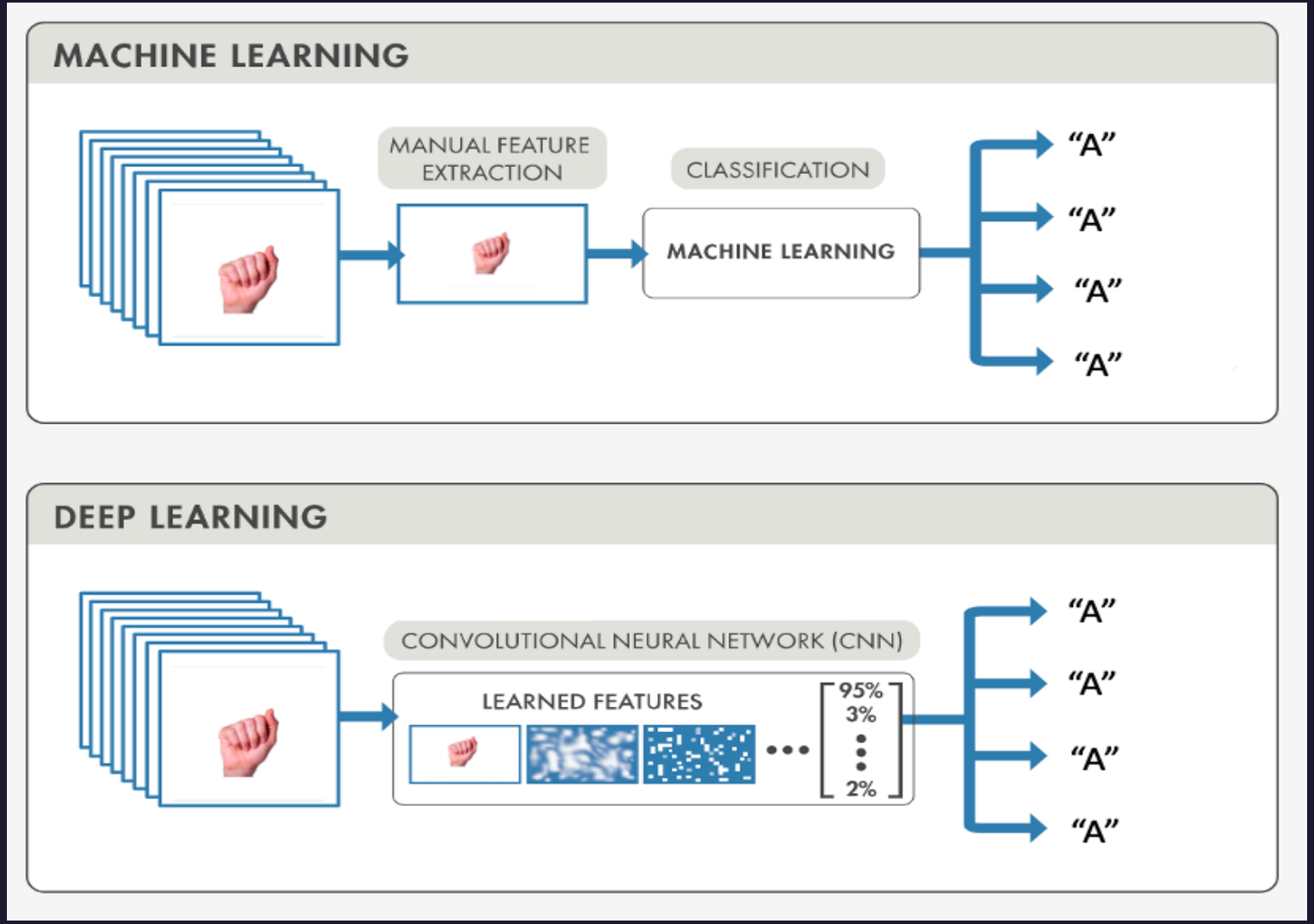
The main objective of my thesis was to research and explore the field of data science. More specifically pertaining to the development of an object recognition application that could be built entirely using MATLAB's IDE, and have a positive social impact on the deaf community. And in doing so, answering the question, could MATLAB be utilized for development of this type of application?

To simultaneously answer this question while addressing my main objectives, I constructed two different object recognition protocols utilizing MATLAB_R2019 and the add-on data science tool packages. I named the protocols ASLtranslate (I) and (II). This allowed me to experiment with all of MATLAB's data science toolboxes while learning the differences, benefits, and disadvantages of using multiple approaches to the same problem.

Methodology and Approach- Design

ASLtranslate (I)

ASLtranslate (II)



Methodology and Approach- Data

ASLtranslate Training Dataset

26 Classes: A: Z

Classes: 3000 images

Total images: 78000

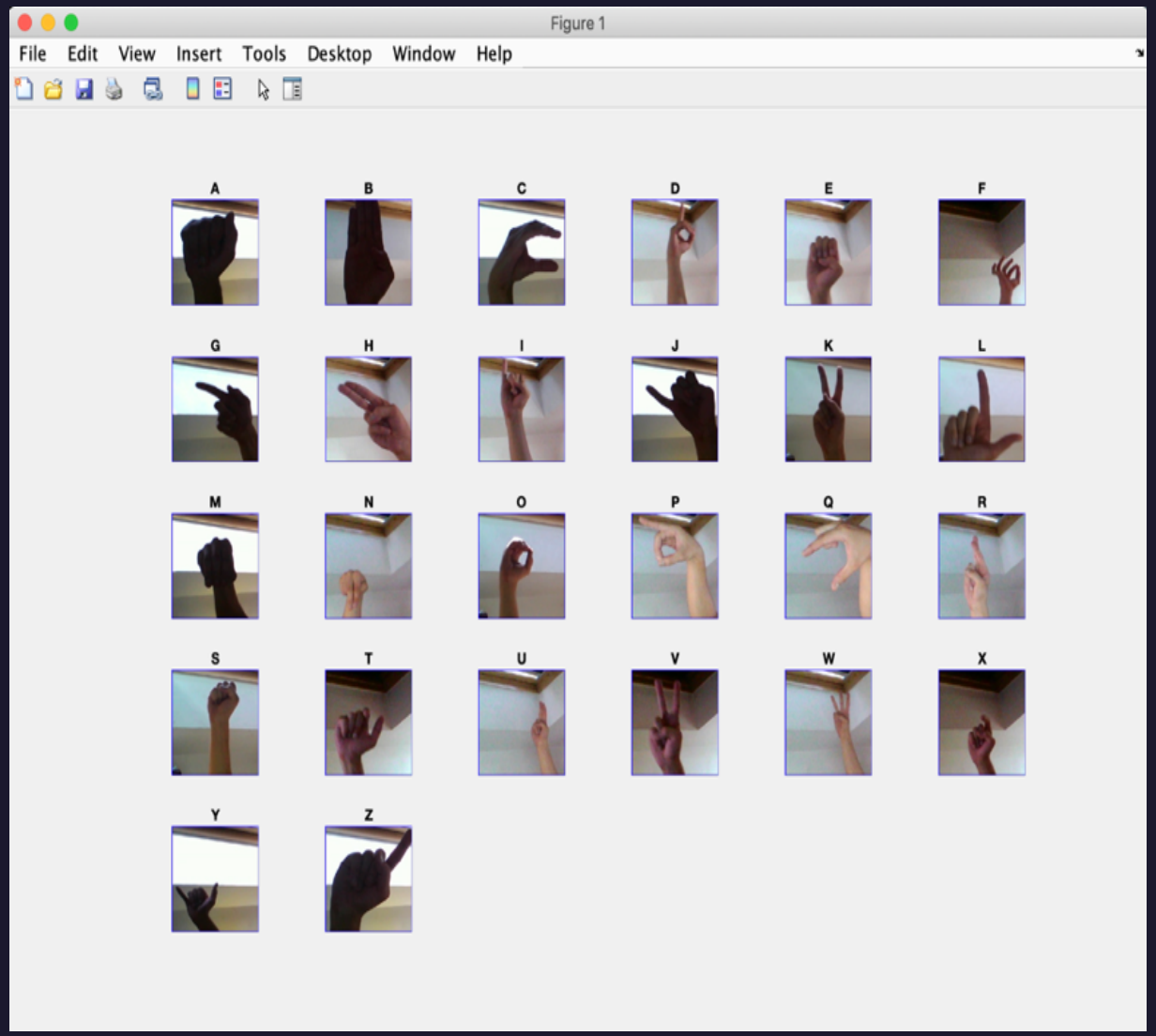
ASLtranslate (I)

```
% Store the output in a temporary folder
outputFolder = fullfile(tempdir, 'aslOutput');

% Load Images into a database
handDatabase = fullfile(outputFolder ,'asl_alphabet_train', 'recursive');

% All the "A" images in the gallery A
%figure;
%montage(handDatabase(1).ImageLocation);
%title('Motage of all the A within our database')

rootFolder = fullfile(outputFolder, 'ALS_ObjectCategories');

%Construct arrays of image sets
imgSets = [ imageSet('asl_alphabet_train/A'), ...
    imageSet('asl_alphabet_train/B'), ...
    imageSet('asl_alphabet_train/C'), ...
    imageSet('asl_alphabet_train/D'),...
    imageSet('asl_alphabet_train/E'), ...
```

ASLtranslate (II)

```
% Importing the training images by creating a datastore
aslDS = imageDatastore('asl_alphabet_train',...
'IncludeSubfolders',true,...
'LabelSource','foldernames');

% Split Data in the datastore for training and validation
[aslTrain,aslValidation] = splitEachLabel(aslDS,0.7,"randomized");
```

Methodology and Approach- ASLtranslate (I)



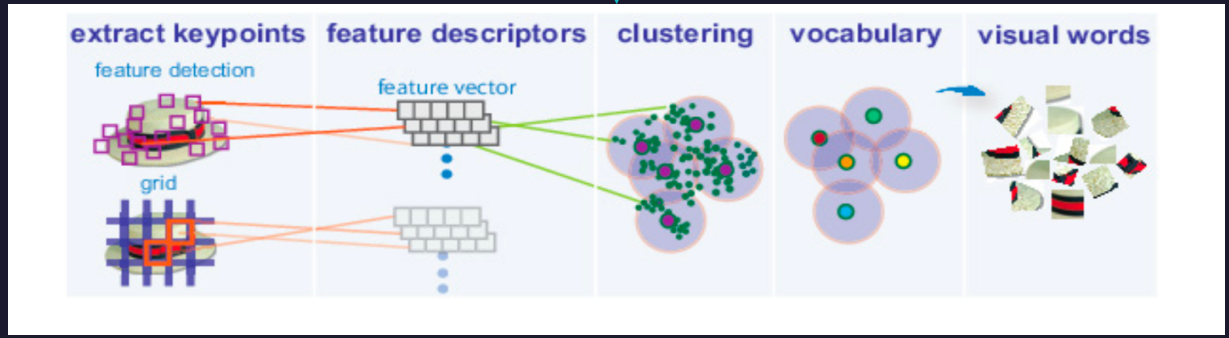

```
>> { imgSets.Description }

ans =

   1×26 cell array

   Columns 1 through 16

{'A'}    {'B'}    {'C'}    {'D'}    {'E'}    {'F'}    {'G'}

{'H'}    {'I'}    {'J'}    {'K'}    {'L'}    {'M'}    {'N'}

{'O'}    {'P'}

Columns 17 through 26

{'Q'}    {'R'}    {'S'}    {'T'}    {'U'}    {'V'}    {'W'}

{'X'}    {'Y'}    {'Z'}
```

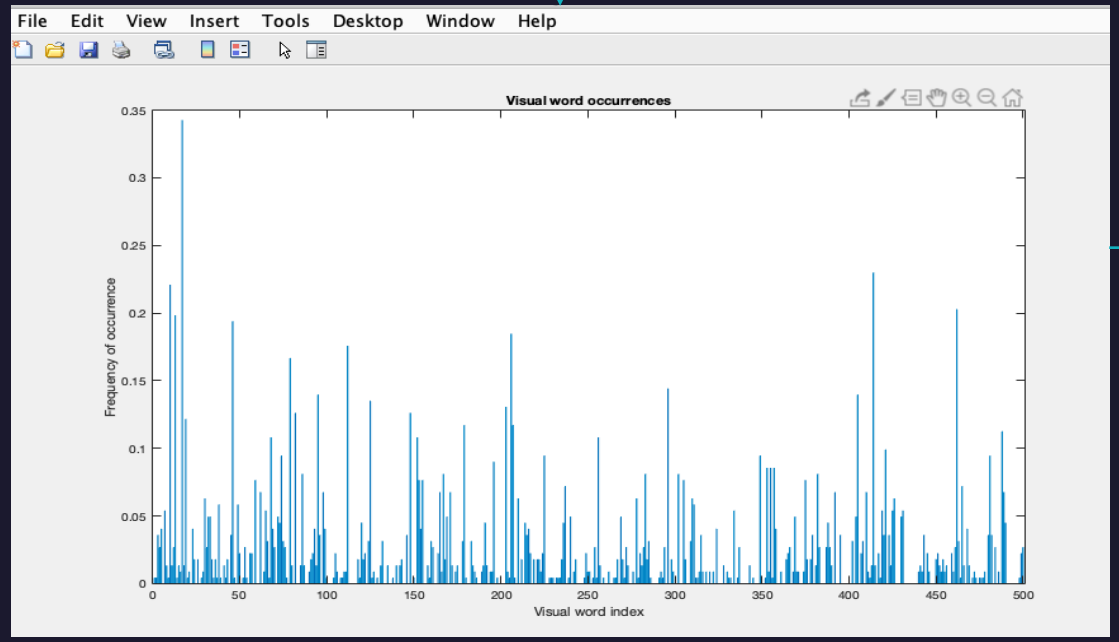
Results after applying the imgSets method to the dataset

Methodology and Approach- bag-of-visual-words (BOVW)

ASLtranslate (I)

```
% Bag-of-Features
bag = bagOfFeatures(trainingSets);
```






```
categoryClassifier = trainImageCategoryClassifier(trainingSets, bag);
```

Results- ASLtranslate (I)

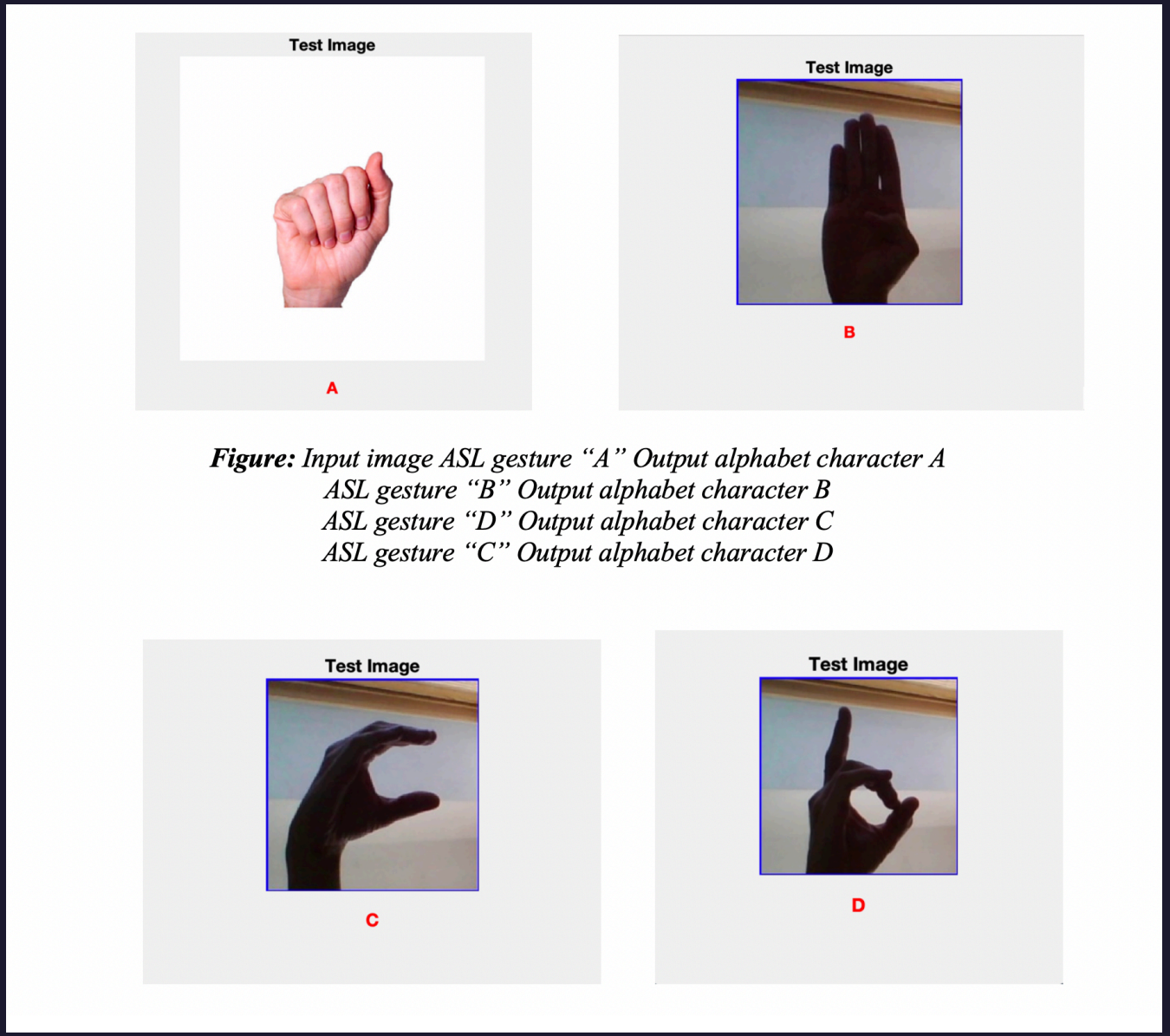
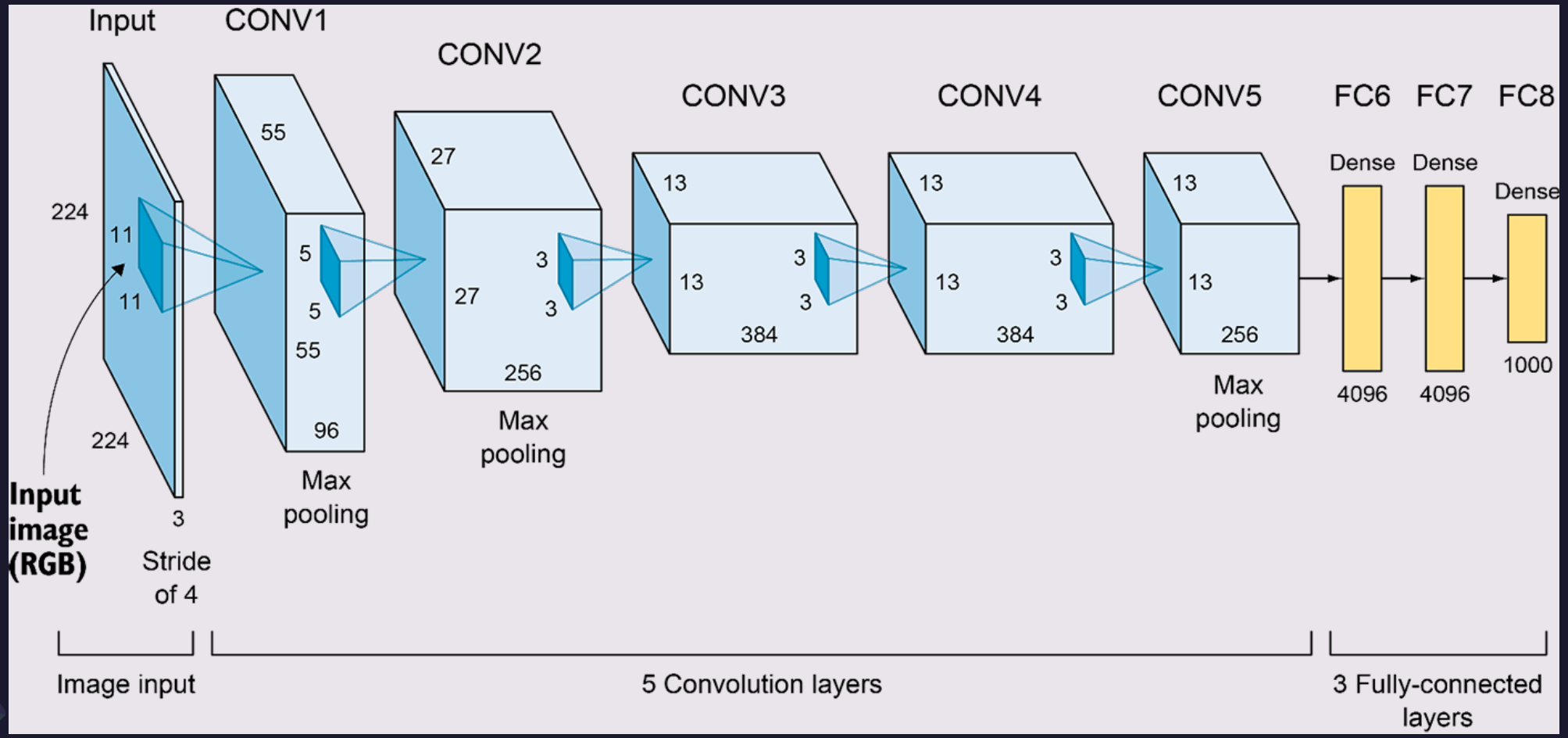


Figure: *Input image ASL gesture "A" Output alphabet character A*
ASL gesture "B" Output alphabet character B
ASL gesture "D" Output alphabet character C
ASL gesture "C" Output alphabet character D

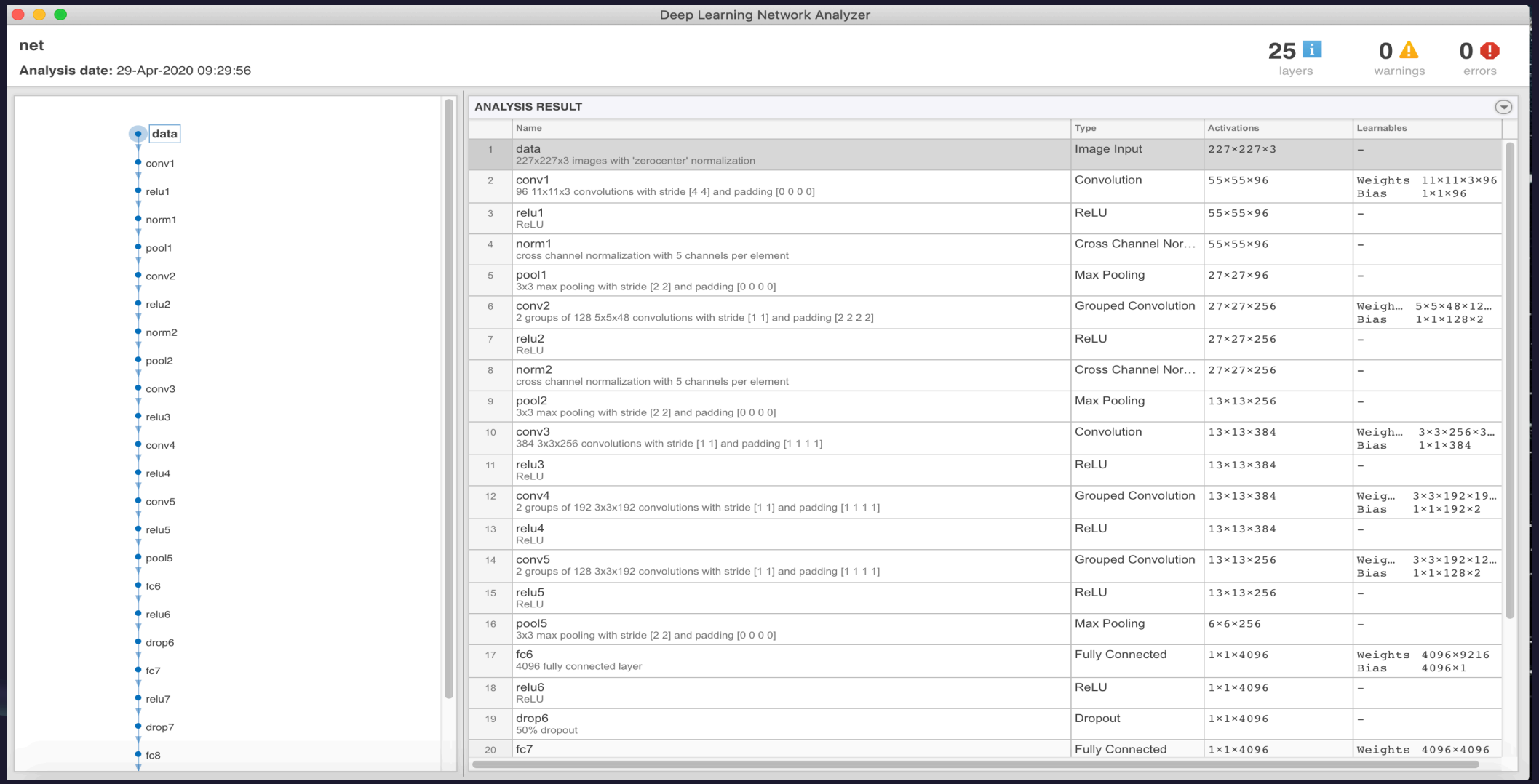
Methodology and Approach- Transfer Learning Using AlexNet

ASLtranslate (II)



Methodology and Approach- Transfer Learning Using AlexNet

ASLtranslate (II)



ANALYSIS RESULT

	Name	Type	Activations	Learnables
1	data 227x227x3 images with 'zerocenter' normalization	Image Input	227×227×3	–
2	conv1 96 11x11x3 convolutions with stride [4 4] and padding [0 0 0 0]	Convolution	55×55×96	Weights 11×11×3×96 Bias 1×1×96
3	relu1 ReLU	ReLU	55×55×96	–
4	norm1 cross channel normalization with 5 channels per element	Cross Channel Nor…	55×55×96	–
5	pool1 3x3 max pooling with stride [2 2] and padding [0 0 0 0]	Max Pooling	27×27×96	–
6	conv2 2 groups of 128 5x5x48 convolutions with stride [1 1] and padding [2 2 2 2]	Grouped Convolution	27×27×256	Weigh… 5×5×48×12… Bias 1×1×128×2
7	relu2 ReLU	ReLU	27×27×256	–
8	norm2 cross channel normalization with 5 channels per element	Cross Channel Nor…	27×27×256	–
9	pool2 3x3 max pooling with stride [2 2] and padding [0 0 0 0]	Max Pooling	13×13×256	–
10	conv3 384 3x3x256 convolutions with stride [1 1] and padding [1 1 1 1]	Convolution	13×13×384	Weigh… 3×3×256×3… Bias 1×1×384
11	relu3 ReLU	ReLU	13×13×384	–
12	conv4 2 groups of 192 3x3x192 convolutions with stride [1 1] and padding [1 1 1 1]	Grouped Convolution	13×13×384	Weig… 3×3×192×19… Bias 1×1×192×2
13	relu4 ReLU	ReLU	13×13×384	–
14	conv5 2 groups of 128 3x3x192 convolutions with stride [1 1] and padding [1 1 1 1]	Grouped Convolution	13×13×256	Weig… 3×3×192×12… Bias 1×1×128×2
15	relu5 ReLU	ReLU	13×13×256	–
16	pool5 3x3 max pooling with stride [2 2] and padding [0 0 0 0]	Max Pooling	6×6×256	–
17	fc6 4096 fully connected layer	Fully Connected	1×1×4096	Weights 4096×9216 Bias 4096×1
18	relu6 ReLU	ReLU	1×1×4096	–
19	drop6 50% dropout	Dropout	1×1×4096	–
20	fc7	Fully Connected	1×1×4096	Weights 4096×4096

Methodology and Approach- Transfer Learning Using AlexNet

ASLtranslate (II)

To perform transfer learning in MATLAB, you need to first create three components

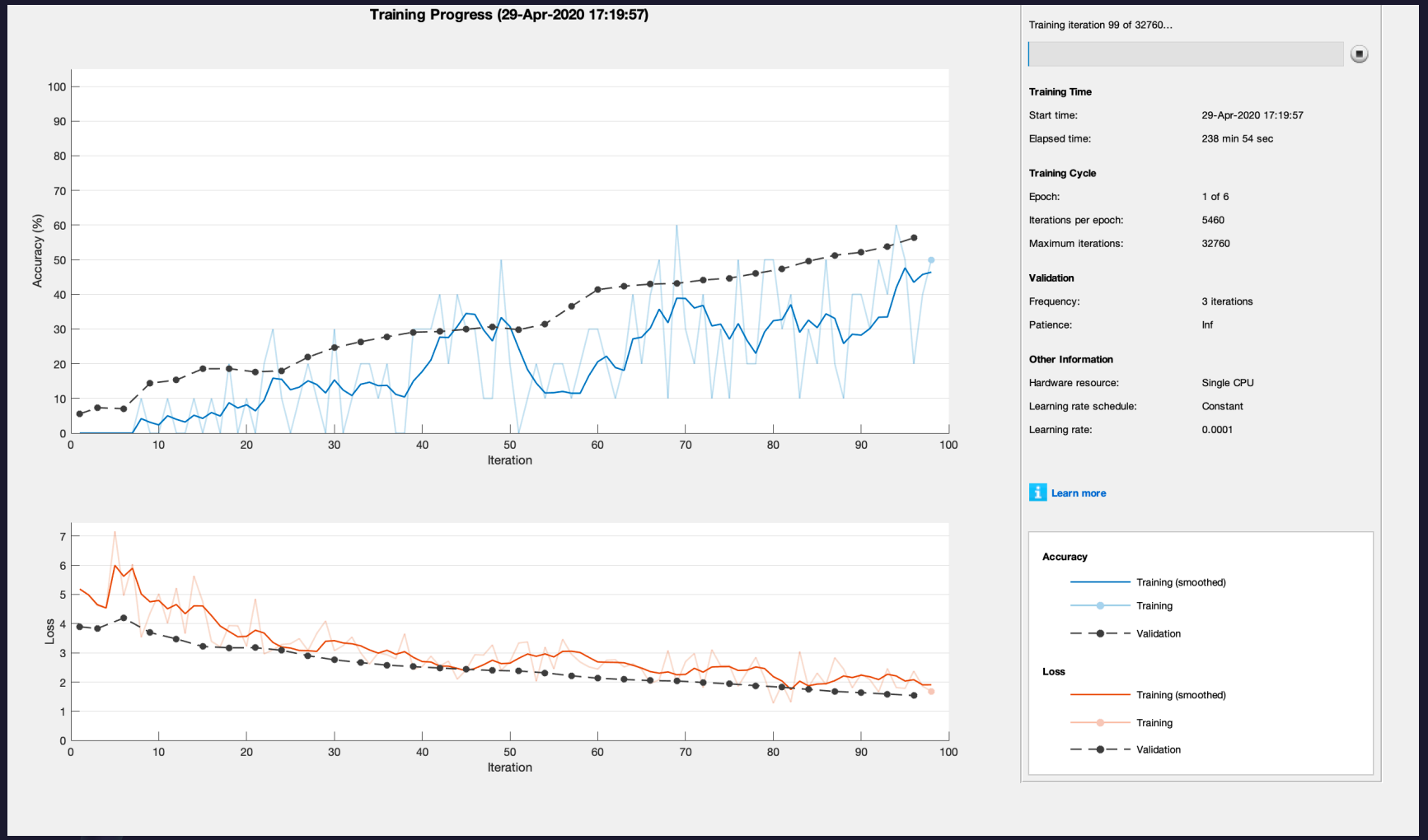
```
% Importing the training images by creating a datastore
aslDS = imageDatastore('asl_alphabet_train',...
'IncludeSubfolders',true,...
'LabelSource','foldernames');

% Split Data in the datastore for training and validation
[aslTrain,aslValidation] = splitEachLabel(aslDS,0.7,"randomized");
```

```
% Replace the final layers of the pretrained network
% fine-tuned last 3 layers for the new classification problem
layersTransfer = net.Layers(1:end-3);

% change the number of classes to match your data
numClasses = numel(categories(aslTrain.Labels));

% Transfer the layers to the new classification task
% by replacing the last three layers with a fully connected layer
% a softmax layer, and a classification output layer
layers = [
    layersTransfer
    fullyConnectedLayer(numClasses,'WeightLearnRateFactor',20,'BiasLearnRateFactor',20)
    softmaxLayer
    classificationLayer];
```

```
% set training parameters
options = trainingOptions('sgdm', ...
    'MiniBatchSize',10, ...
    'MaxEpochs',6, ...
    'InitialLearnRate',1e-4, ...
    'Shuffle','every-epoch', ...
    'ValidationData',augValidation, ...
    'ValidationFrequency',3, ...
    'Verbose',false, ...
    'Plots','training-progress');

% Perform training
netTransfer = trainNetwork(augTrain,layers,options);
```

Methodology and Approach- Training Progress

ASLtranslate (II)



Accuracy and Loss Metrics

Training Accuracy:
Classification accuracy
on each individual mini-batch.

Smoothed Training Accuracy:
Obtained by applying a smoothing algorithm
to the training accuracy.

Validation Accuracy:
Classification accuracy on the entire
validation set.

Training Loss, Smoothed Training Loss, and Validation Loss:
The loss on each mini-batch,
its smoothed version,
and the loss on the validation set.

Results- ASLtranslate (II)

Name ▲	Value
accuracy	0.9972
aslDS	*1x1 ImageDatast...*
aslTest	*1x1 ImageDatast...*
aslTrain	*1x1 ImageDatast...*
augTest	*1x1 augmentedI...*
augTrain	*1x1 augmentedI...*
i	4
I	*200x200x3 uint8*
idx	[11869,21636,19...
imageAugmen...	*1x1 imageDataA...*
info	*1x1 struct*
inputSize	[227,227,3]
label	*1x1 categorical*
layers	*25x1 Layer*
layersTransfer	*22x1 Layer*
net	*1x1 SeriesNetwork*
netTransfer	*1x1 SeriesNetwork*
numClasses	26
numImages	10000
options	*1x1 TrainingOpti...*
perm	*1x20 double*
pixelRange	[-30,30]
scores	*23400x26 single*
translation	*78000x1 categor...*
YPred	*23400x1 categor...*
YValidation	*23400x1 categor...*

```
options: 1x1 nnet.cnn.TrainingOptionsSGDM =

  TrainingOptionsSGDM with properties:

                        Momentum: 0.9000
                InitialLearnRate: 1.0000e-04
    LearnRateScheduleSettings: [1×1 struct]
                L2Regularization: 1.0000e-04
          GradientThresholdMethod: 'l2norm'
                GradientThreshold: Inf
                       MaxEpochs: 6
                   MiniBatchSize: 128
                         Verbose: 0
                VerboseFrequency: 50
                  ValidationData: []
              ValidationFrequency: 50
               ValidationPatience: Inf
                         Shuffle: 'once'
                  CheckpointPath: ''
            ExecutionEnvironment: 'auto'
                      WorkerLoad: []
                       OutputFcn: []
                           Plots: 'training-progress'
                SequenceLength: 'longest'
            SequencePaddingValue: 0
            DispatchInBackground: 0
```

Results- ASLtranslate (II)