UNIVERSITÀ DI PAVIA

Machine Learning Course

Speech Commands Recognition

Andrea Alberti

Department of Computer Engineering - Data Science

University of Pavia, Italy

Email: andrea.alberti01@universitadipavia.it

February 25, 2024

Abstract

This project aimed to develop a highly accurate speech recognition model capable of distinguishing between 35 different spoken words, trained on a dataset comprising 105,829 audio recordings by numerous speakers.
To achieve this objective, various MLP architectures were evaluated, resulting in a test accuracy of approximately 49%. However, the task's complexity, with numerous classes to classify, posed a significant challenge in developing a robust model. Additionally, some words had fewer recordings, which adversely impacted the overall model performance. The model choice was based on the comparison between different model's architectures and normalization techniques. The project provides insights about the significance of addressing dataset imbalances, the importance of feature normalization, and the performance of an MLP model on the speech recognition task.

Contents

1 Introduction

This project is about the implementation of a Neural Network for the classification of audio files. This problem is known as *Speech Command Recognition* and its importance has been raising in the last years thanks to the development of smart devices and the need of a more natural interaction with them. The main library exploited for the implementation is *pvml*.

1.1 Available Data

The data set includes 105829 recordings of 35 words uttered by many different speakers. For the analysis it is divided into three subsets: *train*, *validation* and *test*.

1.2 Goal

The goal is to fit on the data a Multi Layer Perceptron (MLP) so that it can be exploited to classify new unseen data on the 35 different classes.

1.3 Multi Layer Perceptron

A Multi Layer Perceptron (MLP) is a type of Feed Forward Neural Network that consists of zero or more hidden layers of interconnected neurons between an input layer and an output layer. Each neuron in an MLP receives input signals from all neurons in the previous layer, applies an activation function to the weighted sum of its inputs, and outputs the result to the next layer. The parameters of the MLP (weights and biases) are learned optimizing a chosen Loss Function (Cross Entropy) through a process called Backpropagation.

2 Model Building

Prior to start building the model is important to pre-process the data, making them suitable to be passed to the training algorithm. This process is known as *Feature Extraction* and it is a fundamental one in the model creation.

2.1 Features Extraction

Starting from the raw audio data, extraction of features consists in the creation of a spectrogram for each word. This latter has been made uniform in size by padding or truncating the original signals.

2.2 Data Visualization

To better understand the data, it is useful to visualize some of the spectrograms. In figure 1 are reported some examples about some words. The red zones are those in which the frequencies are more present. It is important to underline that the same word can have different spectrograms depending on the speaker.

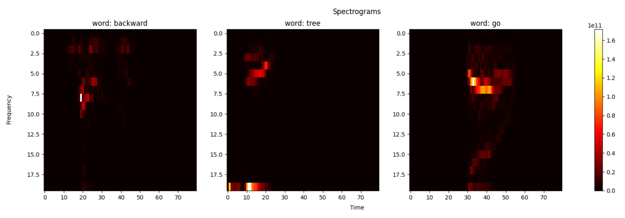


Figure 1: Spectrograms example

2.3 Feature Normalization

Another important step in the pre-processing phase is the normalization of the features. Among the several normalization techniques available, the four used in this project are *Mean-Var*, *Min-Max*, *Whitening* and *Max-Abs* normalization. All these methods are compared in figure 2. It is evident that the best one is the Mean-Variance normalization. This is because it helps to reduce the impact of differences in the volume or intensity of the audio recordings, centering the data around zero and scaling them to have unit variance. These normalized data allow the model to capture underlying patterns.

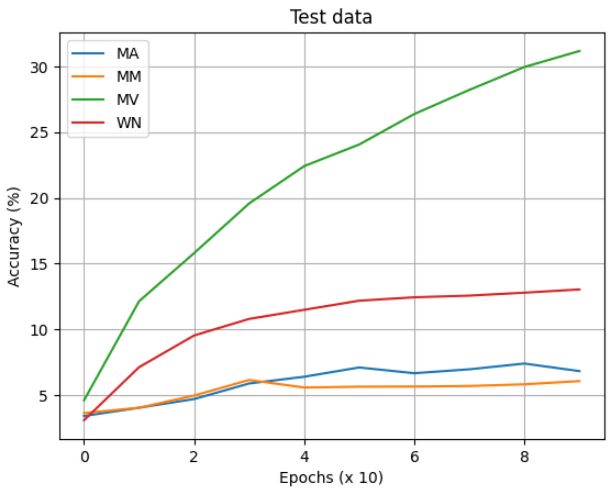


Figure 2: Normalization comparison

2.4 Train Classifier

Batch size
To optimize the training the data are divided into mini-batches. The size of the batch could impact on the model performance, therefore different values have been tested and the result are reported in figure 3. The used model had not hidden layers, therefore its performance is poor, thus all the sizes converges to 18% accuracy on the test set but requiring different number of epochs. The fastest was the 50 batch size and it is chosen as the best one.

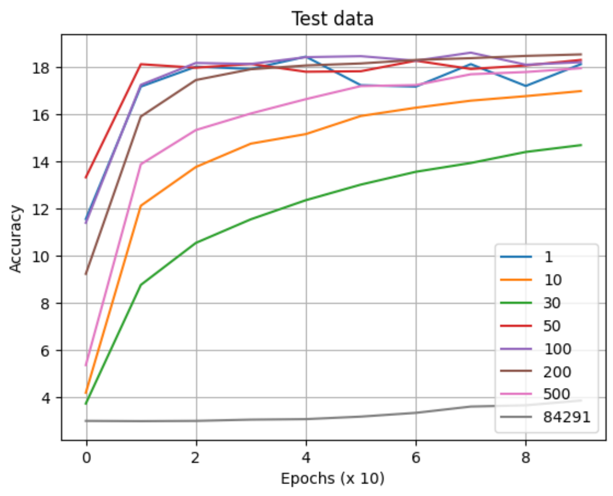


Figure 3: Batch comparison

Architecture
The architecture of the network strongly influences its performance therefore different configurations results are compared in figure 4. For time reasons, the number of epochs is limited to 100 disregarding that some networks have more parameters and could take more time to express their full potential. In this greedy approach the best architecture is that with many hidden layers. This type of architecture was trained with 400 epochs, in figure 5 are shown the results. The blue and orange model reached 80% in training accuracy, however, when tested on the independent test set, their performance was not as good, suggesting that they were overfitting to the training data. To address this problem, several regularization values were tested, and a regularization parameter of $\lambda = 0.001$ was found to be the most effective. Despite this effort, overfitting was only slightly reduced, but not entirely resolved. Ultimately, the red model was selected as the best-performing model and used for further analysis.

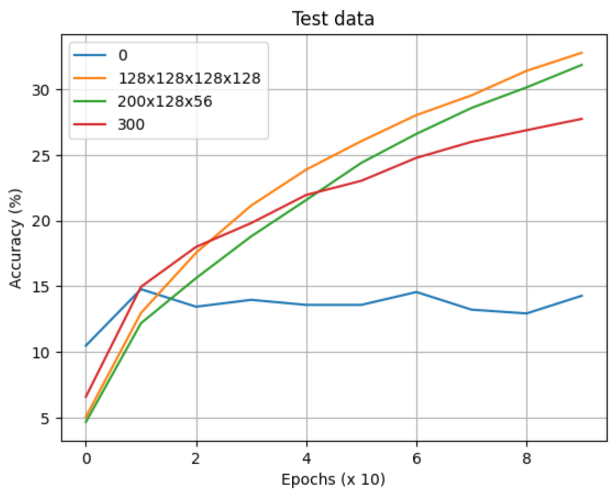


Figure 4: Architecture comparison

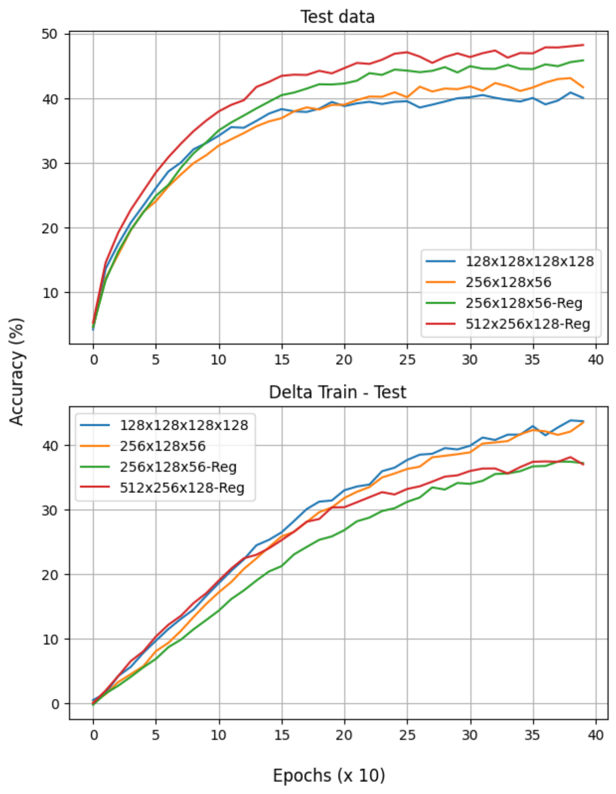


Figure 5: Deep Networks comparison

3 Model Analysis

To analyze the behavior of the chosen model was created a confusion matrix in figure 12 that shows the number of correct and incorrect predictions for each class. The darker the color, the higher the number of samples of class i (row index) classified as class j (column index). As expected, the diagonal is the most populated part of the matrix, meaning that the model correctly classifies a word lots of times.

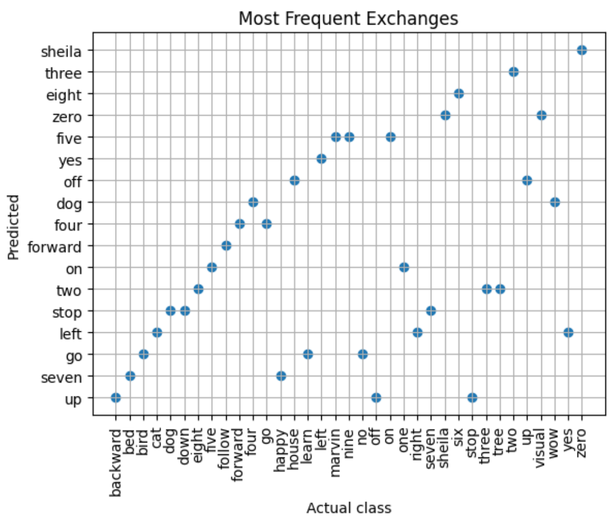


Figure 6: Most exchanged

Most Exchanged Classes
The most exchanged classes are those in which the

model has more difficulties to distinguish between. In figure 6 are reported for each word the class to which the model classifies them (excluded correct classifications) most of times. Lots of exchanges are reasonable like *Forward* classified to *Four* and *Three* to *Two* whose spectrograms are very similar (figure 7).

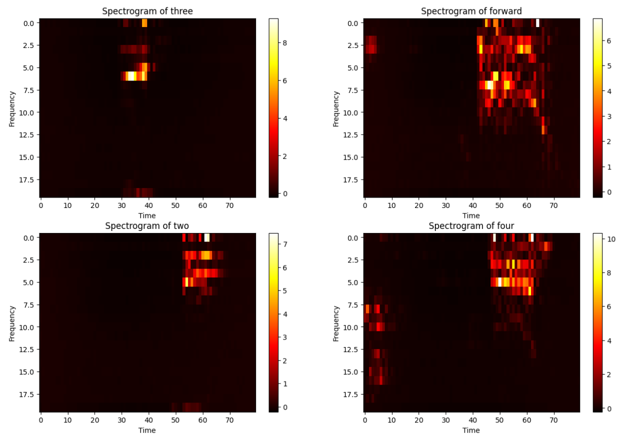


Figure 7: Three VS Two, Forward VS Four

Most Misclassified Samples

Some words are more difficult to be classified than others and in figure 8 are reported the most misclassified samples. *Tree* and *Learn* are the most misclassified words, listening to the audio files of *learn* it is possible to notice that lots of time the word is mispronounced and even a human could have difficulties to understand it. However, the actual reason between the misclassified words is explained by figure 9. It is evident that the most misclassified words are those with less samples, therefore the model has less information to learn and this highly affects its overall performance.

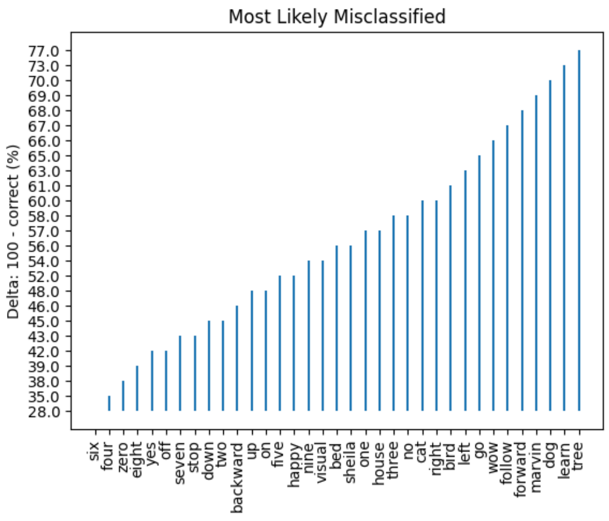


Figure 8: Most misclassified

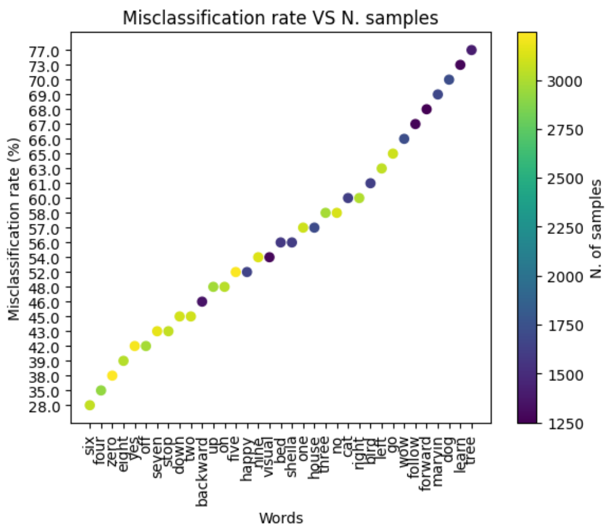


Figure 9: Misclassified VS N. of samples

Weights Visualization

An interesting way to understand how the model works is to visualize the weights of the first layer, shown in figure 11. The red parts votes in favour of the specific class, while the blue parts votes against it. The weights distribution is not uniform and changes from word to word, according to their spectrograms. Remembering that the spectrograms has on y axis the frequency and on x axis the time, it is possible to notice on which frequencies the model focused. For example, the high frequencies strongly penalized words like *Follow* and favours words like *up* and *wow*. In the case of *Forward*, *Four* and *Follow*, due to the similarity of their spectrograms, the model assigns almost the same weights to the same frequencies. Indeed, the model learnt to extract relevant acoustic details, as shown by the overlapping between the assigned weights and the spectrograms of a specific word (figure 10).

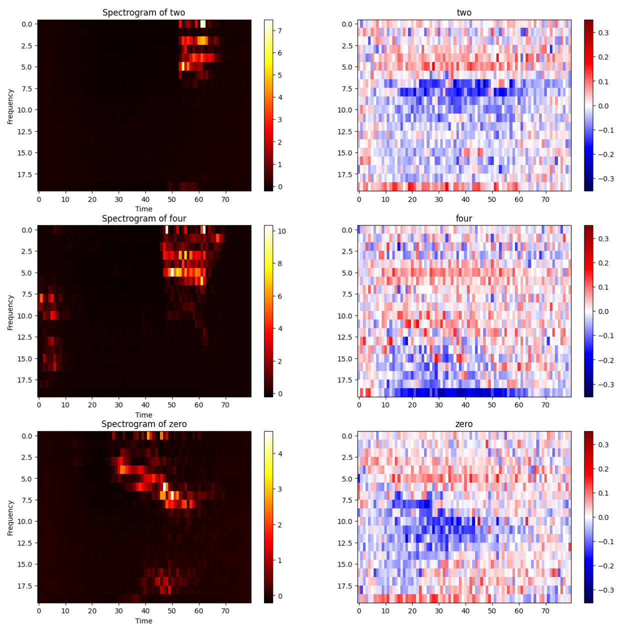


Figure 10: Spectrogram VS Weights

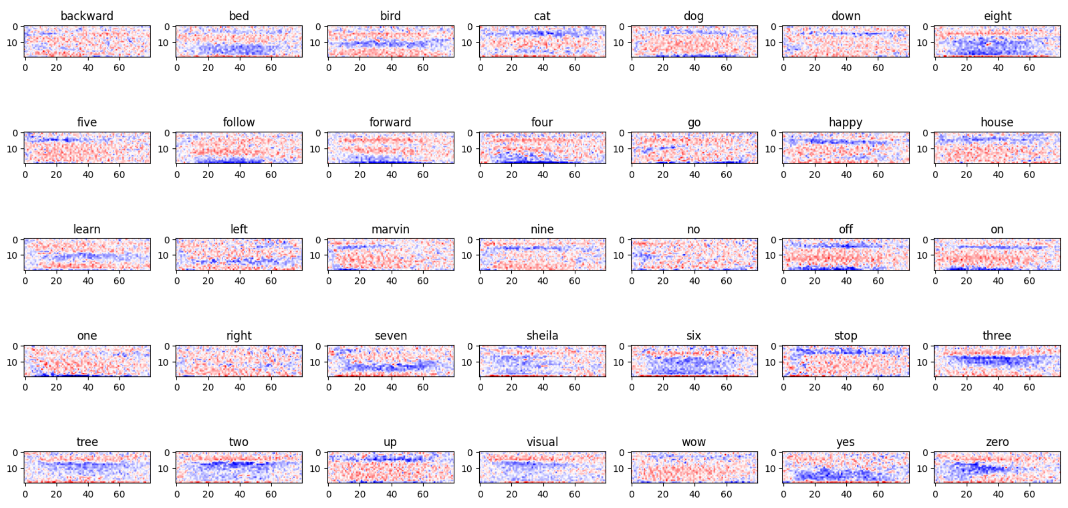
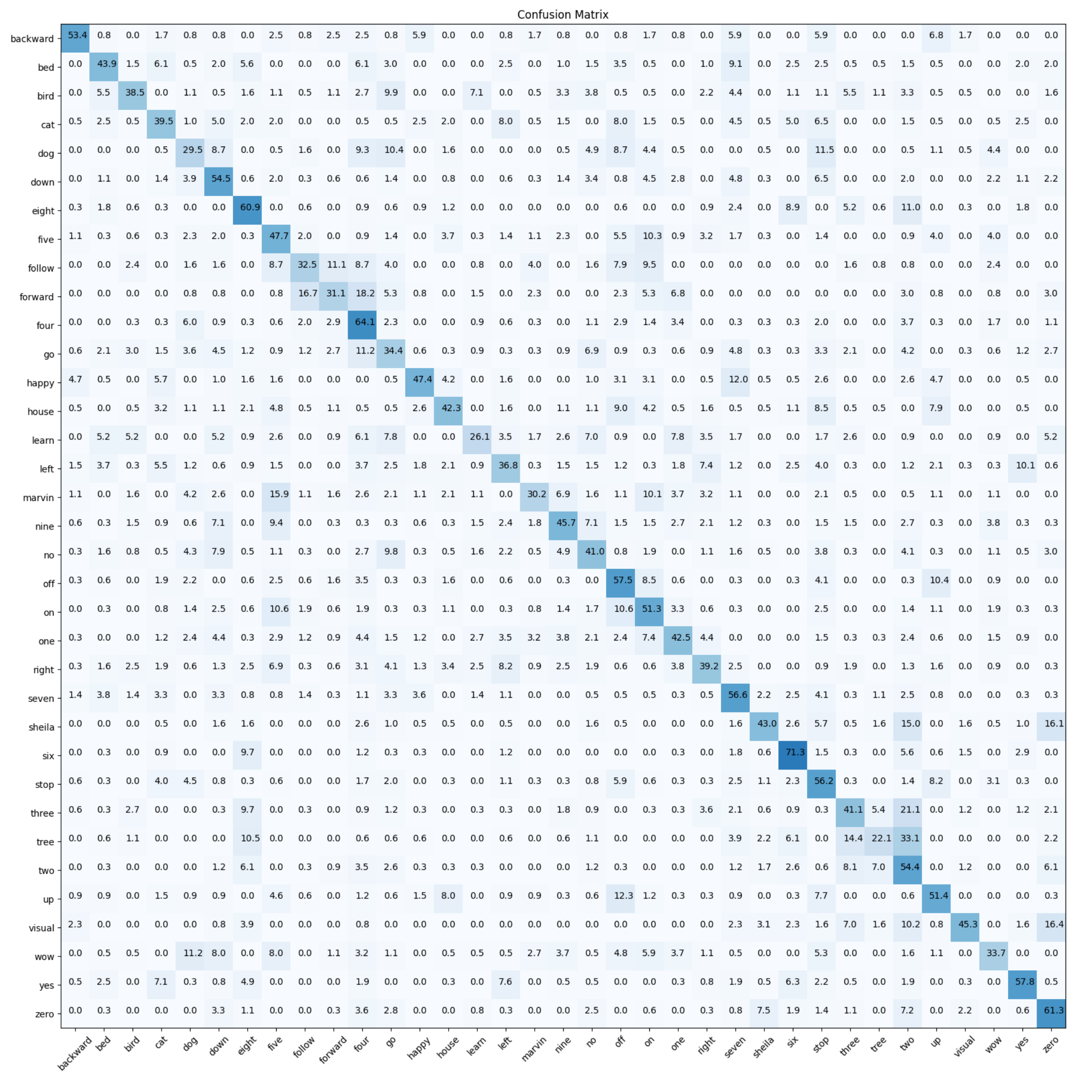


Figure 11: Weights visualization


Confusion Matrix

	backward	bed	bird	cat	dog	down	eight	five	follow	forward	four	go	happy	house	learn	left	marvin	nine	no	off	on	one	right	seven	sheila	six	stop	three	tree	two	up	visual	wow	yes	zero
backward	53.4	0.8	0.0	1.7	0.8	0.8	0.0	2.5	0.8	2.5	2.5	0.8	5.9	0.0	0.0	0.8	1.7	0.8	0.0	0.8	1.7	0.8	0.0	5.9	0.0	0.0	5.9	0.0	0.0	0.0	6.8	1.7	0.0	0.0	0.0
bed	0.0	43.9	1.5	6.1	0.5	2.0	5.6	0.0	0.0	0.0	6.1	3.0	0.0	0.0	0.0	2.5	0.0	1.0	1.5	3.5	0.5	0.0	1.0	9.1	0.0	2.5	2.5	0.5	0.5	1.5	0.5	0.0	0.0	2.0	2.0
bird	0.0	5.5	38.5	0.0	1.1	0.5	1.6	1.1	0.5	1.1	2.7	9.9	0.0	0.0	7.1	0.0	0.5	3.3	3.8	0.5	0.5	0.0	2.2	4.4	0.0	1.1	1.1	5.5	1.1	3.3	0.5	0.5	0.0	0.0	1.6
cat	0.5	2.5	0.5	39.5	1.0	5.0	2.0	2.0	0.0	0.0	0.5	0.5	2.5	2.0	0.0	8.0	0.5	1.5	0.0	8.0	1.5	0.5	0.0	4.5	0.5	5.0	6.5	0.0	0.0	1.5	0.5	0.0	0.5	2.5	0.0
dog	0.0	0.0	0.0	0.5	29.5	8.7	0.0	0.5	1.6	0.0	9.3	10.4	0.0	1.6	0.0	0.0	0.0	0.5	4.9	8.7	4.4	0.5	0.0	0.0	0.5	0.0	11.5	0.0	0.0	0.5	1.1	0.5	4.4	0.0	0.0
down	0.0	1.1	0.0	1.4	3.9	54.5	0.6	2.0	0.3	0.6	0.6	1.4	0.0	0.8	0.0	0.6	0.3	1.4	3.4	0.8	4.5	2.8	0.0	4.8	0.3	0.0	6.5	0.0	0.0	2.0	0.0	0.0	2.2	1.1	2.2
eight	0.3	1.8	0.6	0.3	0.0	0.0	60.9	0.0	0.6	0.0	0.9	0.6	0.9	1.2	0.0	0.0	0.0	0.0	0.0	0.6	0.0	0.0	0.9	2.4	0.0	8.9	0.0	5.2	0.6	11.0	0.0	0.3	0.0	1.8	0.0
five	1.1	0.3	0.6	0.3	2.3	2.0	0.3	47.7	2.0	0.0	0.9	1.4	0.0	3.7	0.3	1.4	1.1	2.3	0.0	5.5	10.3	0.9	3.2	1.7	0.3	0.0	1.4	0.0	0.0	0.9	4.0	0.0	4.0	0.0	0.0
follow	0.0	0.0	2.4	0.0	1.6	1.6	0.0	8.7	32.5	11.1	8.7	4.0	0.0	0.0	0.8	0.0	4.0	0.0	1.6	7.9	9.5	0.0	0.0	0.0	0.0	0.0	0.0	1.6	0.8	0.8	0.0	0.0	2.4	0.0	0.0
forward	0.0	0.0	0.0	0.0	0.8	0.8	0.0	0.8	16.7	31.1	18.2	5.3	0.8	0.0	1.5	0.0	2.3	0.0	0.0	2.3	5.3	6.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.0	0.8	0.0	0.8	0.0	3.0
four	0.0	0.0	0.3	0.3	6.0	0.9	0.3	0.6	2.0	2.9	64.1	2.3	0.0	0.0	0.9	0.6	0.3	0.0	1.1	2.9	1.4	3.4	0.0	0.3	0.3	0.3	2.0	0.0	0.0	3.7	0.3	0.0	1.7	0.0	1.1
go	0.6	2.1	3.0	1.5	3.6	4.5	1.2	0.9	1.2	2.7	11.2	34.4	0.6	0.3	0.9	0.3	0.3	0.9	6.9	0.9	0.3	0.6	0.9	4.8	0.3	0.3	3.3	2.1	0.0	4.2	0.0	0.3	0.6	1.2	2.7
happy	4.7	0.5	0.0	5.7	0.0	1.0	1.6	1.6	0.0	0.0	0.0	0.5	47.4	4.2	0.0	1.6	0.0	0.0	1.0	3.1	3.1	0.0	0.5	12.0	0.5	0.5	2.6	0.0	0.0	2.6	4.7	0.0	0.0	0.5	0.0
house	0.5	0.0	0.5	3.2	1.1	1.1	2.1	4.8	0.5	1.1	0.5	0.5	2.6	42.3	0.0	1.6	0.0	1.1	1.1	9.0	4.2	0.5	1.6	0.5	0.5	1.1	8.5	0.5	0.5	0.0	7.9	0.0	0.0	0.5	0.0
learn	0.0	5.2	5.2	0.0	0.0	5.2	0.9	2.6	0.0	0.9	6.1	7.8	0.0	0.0	26.1	3.5	1.7	2.6	7.0	0.9	0.0	7.8	3.5	1.7	0.0	0.0	1.7	2.6	0.0	0.9	0.0	0.0	0.9	0.0	5.2
left	1.5	3.7	0.3	5.5	1.2	0.6	0.9	1.5	0.0	0.0	3.7	2.5	1.8	2.1	0.9	36.8	0.3	1.5	1.5	1.2	0.3	1.8	7.4	1.2	0.0	2.5	4.0	0.3	0.0	1.2	2.1	0.3	0.3	10.1	0.6
marvin	1.1	0.0	1.6	0.0	4.2	2.6	0.0	15.9	1.1	1.6	2.6	2.1	1.1	2.1	1.1	0.0	30.2	6.9	1.6	1.1	10.1	3.7	3.2	1.1	0.0	0.0	2.1	0.5	0.0	0.5	1.1	0.0	1.1	0.0	0.0
nine	0.6	0.3	1.5	0.9	0.6	7.1	0.0	9.4	0.0	0.3	0.3	0.3	0.6	0.3	1.5	2.4	1.8	45.7	7.1	1.5	1.5	2.7	2.1	1.2	0.3	0.0	1.5	1.5	0.0	2.7	0.3	0.0	3.8	0.3	0.3
no	0.3	1.6	0.8	0.5	4.3	7.9	0.5	1.1	0.3	0.0	2.7	9.8	0.3	0.5	1.6	2.2	0.5	4.9	41.0	0.8	1.9	0.0	1.1	1.6	0.5	0.0	3.8	0.3	0.0	4.1	0.3	0.0	1.1	0.5	3.0
off	0.3	0.6	0.0	1.9	2.2	0.0	0.6	2.5	0.6	1.6	3.5	0.3	0.3	1.6	0.0	0.6	0.0	0.3	0.0	57.5	8.5	0.6	0.0	0.3	0.0	0.3	4.1	0.0	0.0	0.3	10.4	0.0	0.9	0.0	0.0
on	0.0	0.3	0.0	0.8	1.4	2.5	0.6	10.6	1.9	0.6	1.9	0.3	0.3	1.1	0.0	0.3	0.8	1.4	1.7	10.6	51.3	3.3	0.6	0.3	0.0	0.0	2.5	0.0	0.0	1.4	1.1	0.0	1.9	0.3	0.3
one	0.3	0.0	0.0	1.2	2.4	4.4	0.3	2.9	1.2	0.9	4.4	1.5	1.2	0.0	2.7	3.5	3.2	3.8	2.1	2.4	7.4	42.5	4.4	0.0	0.0	0.0	1.5	0.3	0.3	2.4	0.6	0.0	1.5	0.9	0.0
right	0.3	1.6	2.5	1.9	0.6	1.3	2.5	6.9	0.3	0.6	3.1	4.1	1.3	3.4	2.5	8.2	0.9	2.5	1.9	0.6	0.6	3.8	39.2	2.5	0.0	0.0	0.9	1.9	0.0	1.3	1.6	0.0	0.9	0.0	0.3
seven	1.4	3.8	1.4	3.3	0.0	3.3	0.8	0.8	1.4	0.3	1.1	3.3	3.6	0.0	1.4	1.1	0.0	0.0	0.5	0.5	0.5	0.3	0.5	56.6	2.2	2.5	4.1	0.3	1.1	2.5	0.8	0.0	0.0	0.3	0.3
sheila	0.0	0.0	0.0	0.5	0.0	1.6	1.6	0.0	0.0	0.0	2.6	1.0	0.5	0.5	0.0	0.5	0.0	0.0	1.6	0.5	0.0	0.0	0.0	1.6	43.0	2.6	5.7	0.5	1.6	15.0	0.0	1.6	0.5	1.0	16.1
six	0.0	0.3	0.0	0.9	0.0	0.0	9.7	0.0	0.0	0.0	1.2	0.3	0.3	0.0	0.0	1.2	0.0	0.0	0.0	0.0	0.0	0.3	0.0	1.8	0.6	71.3	1.5	0.3	0.0	5.6	0.6	1.5	0.0	2.9	0.0
stop	0.6	0.3	0.0	4.0	4.5	0.8	0.3	0.6	0.0	0.0	1.7	2.0	0.0	0.3	0.0	1.1	0.3	0.3	0.8	5.9	0.6	0.3	0.3	2.5	1.1	2.3	56.2	0.3	0.0	1.4	8.2	0.0	3.1	0.3	0.0
three	0.6	0.3	2.7	0.0	0.0	0.3	9.7	0.0	0.3	0.0	0.9	1.2	0.3	0.0	0.3	0.3	0.0	1.8	0.9	0.0	0.3	0.3	3.6	2.1	0.6	0.9	0.3	41.1	5.4	21.1	0.0	1.2	0.0	1.2	2.1
tree	0.0	0.6	1.1	0.0	0.0	0.0	10.5	0.0	0.0	0.0	0.6	0.6	0.6	0.0	0.0	0.6	0.0	0.6	1.1	0.0	0.0	0.0	0.0	3.9	2.2	6.1	0.0	14.4	22.1	33.1	0.0	0.0	0.0	0.0	2.2
two	0.0	0.3	0.0	0.0	0.0	1.2	6.1	0.0	0.3	0.9	3.5	2.6	0.3	0.3	0.3	0.0	0.0	0.0	1.2	0.3	0.0	0.0	0.0	1.2	1.7	2.6	0.6	8.1	7.0	54.4	0.0	1.2	0.0	0.0	6.1
up	0.9	0.9	0.0	1.5	0.9	0.9	0.0	4.6	0.6	0.0	1.2	0.6	1.5	8.0	0.0	0.9	0.9	0.3	0.6	12.3	1.2	0.3	0.3	0.9	0.0	0.3	7.7	0.0	0.0	0.6	51.4	0.0	0.0	0.0	0.3
visual	2.3	0.0	0.0	0.0	0.0	0.8	3.9	0.0	0.0	0.0	0.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.3	3.1	2.3	1.6	7.0	1.6	10.2	0.8	45.3	0.0	1.6	16.4
wow	0.0	0.5	0.5	0.0	11.2	8.0	0.0	8.0	0.0	1.1	3.2	1.1	0.0	0.5	0.5	0.5	2.7	3.7	0.5	4.8	5.9	3.7	1.1	0.5	0.0	0.0	5.3	0.0	0.0	1.6	1.1	0.0	33.7	0.0	0.0
yes	0.5	2.5	0.0	7.1	0.3	0.8	4.9	0.0	0.0	0.0	1.9	0.0	0.0	0.3	0.0	7.6	0.0	0.5	0.5	0.0	0.0	0.3	0.8	1.9	0.5	6.3	2.2	0.5	0.0	1.9	0.0	0.3	0.0	57.8	0.5
zero	0.0	0.3	0.0	0.0	0.0	3.3	1.1	0.0	0.0	0.3	3.6	2.8	0.0	0.0	0.8	0.3	0.0	0.0	2.5	0.0	0.6	0.0	0.3	0.8	7.5	1.9	1.4	1.1	0.0	7.2	0.0	2.2	0.0	0.6	61.3

Figure 12: Test accuracies comparison