Speech Commands MVA

Haozhe SUN

February 18, 2019

1 Classification of single voice commands

1.1 Parameter tuning

According to some research [1] [2] [3], it turns out that in the context of ASR, typical frame sizes range from 20 ms to 40 ms with 50% (+/ − 10%) overlap between consecutive frames. Popular settings are 25 ms for the frame size and a 10 ms stride (15 ms overlap). So I fix $wlen = 0.025$ and $frate = 100$ for both Mel filterbank and MFCC features.

As for the frequency range that the filters need to span, we all know that the range of hearing which is about 20 Hz to 20 kHz. However it might not be necessary to consider all of them because given the fixed sampling frequency 16000 Hz of the dataset, we should obey Nyquist–Shannon sampling theorem which states that the maximum of the frequency of the signal should not exceed one half of the sampling frequency, otherwise there would be aliasing. Given a fixed sampling frequency, the length of processed signal increases proportionally to the frequency range, which might influence the speed of the calculation of features. It seems that the telephone which transmits only a useful part of the voice frequency band namely 0.3 kHz to 3.4 kHz is enough to make the voice understandable. However, the quality of the voice with this range (300 Hz - 3.4 kHz) is fairly compromised. According to [3], 80 Hz to 8 kHz gives a very good quality, which also achieves the limit of Nyquist–Shannon sampling theorem. Table 1 shows the comparison of accuracy and time consumption on validation set with these two frequency ranges. The accuracy reported in Table 1 are obtained using a basic classifier, they are used to compare the two frequency ranges but does not represent the performance of the best classifier and features.

As for the other parameters of Mel filterbank and MFCC, I follow the classical setting according to [1]: 40 triangular filters in the filterbank, 512-point FFT on each frame to calculate the frequency spectrum, MFCC calculated with 13 cepstral coefficients (typically, only cepstral coefficients 2-13 are retained) with the concatenation of first and second order derivatives, pre-emphasis filter coefficient $\alpha = 0.97$.

All else being equal, this parameter setting provides significantly better results compared to the default setting according to my experiments.

1.2 Feature padding

Given the sampling frequency 16000, some waveform samples of the WAV files provided have a size less than 16000, for example, waveform lengths like 13654, 15702, 15019, 14118 are observed. As a consequence, the Mel filterbank / MFCC features obtained from these WAV files may not have the same size. Let's take Mel filterbank feature as an example, we denote the number of frames by N_{frame}, the number of filters in filterbank by N_{filter}, the size of feature corresponding to a single WAV file by S, we have Equation 1.

Accuracy / Time	Mel filterbank	MFCC
300 Hz - 3.4 kHz	15.7% / 304 s	17.1% / 396 s
80 Hz - 8 kHz	22.7% / 302 s	40.9% / 395 s

Table 1: Comparison of frequency ranges using basic logistic regression. Time consumption in seconds is the sum of the time of feature transformation and the time of training of model. The variance of accuracy is very large, MFCC with 80Hz-8kHz can also encounter a validation accuracy of 10%.

Without CMVN	With CMVN
21.0%	39.1%

Table 2: Comparison of effect of CMVN using basic logistic regression with MFCC 80Hz-8kHz, 9000 training examples

Logistic	Neural 160	Neural 6×90	Neural 6×160	Neural 6×250
38.8% / 9 s	60.2% / 33 s	62.4% / 96 s	66.5% / 132 s	60.8% / 320 s

Table 3: Model selection using 9000 training examples. Validation accuracy / time.

$$S = N_{frame} \times N_{filter} \tag{1}$$

S is thus proportional to the size of the corresponding WAV file because N_{frames} is proportional to the size of the corresponding WAV file. If the size of all WAV files are not equal, then the size of obtained features are not equal. Zero padding is thus used to make features of a given data set have the same size. In my work, I pad only on the right hand side instead of padding on both sides, this is to keep the semantic consistency of features along different instances. As for the minimum length of padded features, I keep 3939 in case of MFCC as $3939 = 101 \times 39$, 101 is the number of frames corresponding to a 10 ms frame stride, 39 is the dimension of MFCC of a single frame in case of 13 cepstral coefficients with concatenation of first and second order derivatives.

1.3 Cepstral Mean and Variance Normalization

As suggested in the instruction, a standard way of improving generalization is to do mean-variance normalization on your data set, which means to compute the mean and variance of each feature dimension on the entire training set, and then use it to normalize train, valid and test set. This procudure can be referred to as *Cepstral mean and variance normalization* (CMVN) and is a common practice in speech recognition [1] [4]. It should be noted that according to [4], the performance of CMVN is known to degrade for short utterances, which is due to insufficient data for parameter estimation and loss of discriminable information as all utterances are forced to have zero mean and unit variance. Table 2 shows the effect of CMVN using a basic classifier, CMVN significantly improves the performance with this data set.

1.4 Model selection

By several experiments, I observe that using 9000 training examples, the validation accuracy of models has large variance. To do a robust model selection, I run multiple times the same model with different random states and then take the average accuracy. By doing this, I try to avoid selecting models from random noises. Table 3 shows the results of some experiments. It is obvious that neural networks are more suitable for this task at the expense of training time. Among the neural networks, it turns out that deep neural networks with large number of parameters usually have better performance. However, if the number of parameters is too large, the validation performance may degrade, which could be explained by the difficulty of training or over-fitting on training set. In the end, I choose to use MLP neural network with 6 hidden layers, each hidden layer has 160 neurons.

1.5 Final classifier for single speech command

In total 64727 audio files are provided in the data set, among which 1000 are used as validation set, 1000 are used as test set. The audio files provided for training are not equally distributed. Among audio files for training, the least frequent label has 1713 occurrences, the most frequent label has 2380 occurrences. If one needs a balanced training data set without using data augmentation techniques, he could use up to 1713 examples per class, i.e. 51390 training examples in total as there are 30 classes. I choose to use 51000 examples as training set, i.e. 1700 examples per class. Larger training set requires more computation time and memory. I don't augment the training data set by adding noises because I am

Validation	Test
69.5%	70.4%

Table 4: Final single speech command results

not faced with the lack of data, I don't even use all the data provided. With 51000 training examples, 1000 validation examples, 1000 test examples, the time used to transform waveform to features is about 9 minutes, the time used to train the MLP model is about 8 minutes. Table 4 shows the results on validation set and test set.

The summary of final setting is as follows: MFCC (nflit = 40, ncep = 13, 80 Hz - 8 kHz, $\alpha = 0.97$, frate = 100, wlen = 0.025, nfft = 512, with first and second order derivatives), Cepstral Mean and Variance Normalization, features padded only on the right hand side with 3939 as minimum length, 51000 training examples without using data augmentation, 6×160 MLP neural network classifier.

2 Classification of segmented voice commands

2.1 Questions

Question 2.1

Word Error Rate is defined as

$$\text{WER} = \frac{\text{substitution } S + \text{insertion } I + \text{deletion } D}{\text{number of words in reference sequence } N} \tag{2}$$

It is not possible that WER < 0 because $S >= 0, I >= 0, D >= 0, N >= 0$. It is possible that WER > 1 because it is possible that $S + I + D > N$. For example, WER $\longrightarrow \infty$ when insertion $I \longrightarrow \infty$ given a fixed finite N. This is valid because I has no constraints besides $I >= 0$, it is not restricted by S, D, N.

Question 2.2

This line of code approximated the prior probability of each word W_i to be equal:

$$\text{nb\_ex\_per\_class} = 1700$$

The training set is then appended only if the number of examples in the corresponding class does not exceed nb_ex_per_class.

Question 2.3

WER is just Levenshtein distance adapted for words. After alignment, one uses Equation 2 to compute the value of WER.

Question 2.4

Laplace smoothing bigram approximation formula of the language model is Equation 3. It should be noted that when doing N-grams with N larger than 1, it is useful to create a symbol for the termination of sentences and a symbol for the beginning of sentences, which will be considered as a word.

$$\mathbb{P}_{\text{Laplace bi-gram}}(w_i|w_{i-1}) = \frac{\sum \mathbb{1}_{\{w_{i-1}w_i\}} + 1}{\sum \mathbb{1}_{\{w_{i-1}\}} + |\text{vocabulary}|} \tag{3}$$

Question 2.5

There are several ways of smoothing to deal with sparsity of the data. Common ones are Laplace (add-one) smoothing, backoff, Jelinek-Mercer interpolation and Kneser-Ney Smoothing. Laplace smoothing is the most simple and the most intuitive, however, if the bi-grams are sparse, Laplace smoothing

	greedy search	beam search	Viterbi algorithm
train WER	48.4%	16.0%	41.3%
test WER	47.9%	13.1%	41.8%
decoding time	0.038 ms	1.695 ms	1.679 ms

Table 5: Segmented voice command classification results. Decoding time means time in milliseconds used for one execution of algorithm.

mean value	variance	minimum value	maximum value
61.2	2626.96	0	206

Table 6: Occurrences of different single speech commands in test set (test_sequence_list).

would assign too much weights to bi-grams that never appeared. Further more, Laplace smoothing assign equal probability to all bi-grams that never appeared in the training set, which might not be the optimal choice. Backoff means to use information of lower order N-grams when the higher order N-grams do not appear. Jelinek-Mercer Interpolation does a convex combination of different order N-grams, however the optimal value of coefficients of convex combination should be determined by EM algorithms. Kneser-Ney smoothing [5] is widely considered the most effective method of smoothing but it is quite complicated.

In the implementation, I use log scale instead of linear scale to avoid numerical instabilities.

Question 2.6

Increasing N gives more robust language models but the complexity grows exponentially.

Question 2.7

Time complexity of beam search algorithm is $O(LBV\log(BV))$, where B is the beam size, L is the length of sentence, V is the size of vocabulary. The term $BV\log(BV)$ comes from the sorting which dominates linear search.

Question 2.8

Let's denote the probability to be in state j at step k by $\mathbb{P}_k(j)$, the probabilities to be in state j' at step $k-1$ by $\mathbb{P}_{k-1}(j')$, $\mathbf{A}$ the transition matrix. We have Equation 4.

$$\mathbb{P}_k(j) = \mathbb{P}_{k-1}(j')\mathbf{A}(k-1,k) \tag{4}$$

The final complexity of Viterbi algorithm is $O(LV^2)$, where L represents the length of sequence, V represents the size of vocabulary.

2.2 Analysis of results of segmented classification task

Table 5 shows results of three decoding strategies. It is confirmed that by using greedy search, the train WER and test WER are approximately the same as the language model is not taken into account in decoding. Computation time for beam search and Viterbi algorithm are approximately the same, while greedy search is much faster. In terms of accuracy, it turns out that beam search has a significantly better performance than the others, especially in terms of test WER which achieves 13.1%. Viterbi algorithm only performs slightly better than the greedy approach, which could be explained by the fact that we approximate $\mathbb{P}$(speech|word) by $\mathbb{P}$(word|speech). In training phase of single speech command classifier, words are equally distributed so we have constant $\mathbb{P}$(word), which is not the case for $\mathbb{P}$(speech). For example, single speech commands in test set of segmented speech sequences are not equally distributed, which is shown in Table 6.

References

[1] Haytham Fayek. Speech processing for machine learning: Filter banks, mel-frequency cepstral coefficients (mfccs) and what's in-between. https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html. accessed February 15, 2019.

[2] Manjunath Pai H. Quora what is the minimum sampling frequency of human speech? https://www.quora.com/What-is-the-minimum-sampling-frequency-of-human-speech. accessed February 15, 2019.

[3] Hilmar. Stack exchange what is the bandwidth of human speech? https://dsp.stackexchange.com/questions/36505/what-is-the-bandwidth-of-human-speech. accessed February 15, 2019.

[4] Wikipedia contributors. Cepstral mean and variance normalization — Wikipedia, the free encyclopedia. accessed February 15, 2019.

[5] Wikipedia contributors. Kneser–ney smoothing — Wikipedia, the free encyclopedia. accessed February 16, 2019.