Speech Recognition Using Convolutional Neural Networks on Small Training Sets

Speech Recognition Using Convolutional Neural Networks on Small Training Sets

A V Poliyev[1] and O N Korsun[1]

[1] State Research Institute of Aviation Systems, Moscow, Russia

E-mail: marmotto@rambler.ru

Received xxxxxx
Accepted for publication xxxxxx
Published xxxxxx

Abstract

The algorithm of separate words automatic recognition based on convolutional neural networks is developed and presented in this paper. Distinctive feature of this algorithm is the training on sets consisting of only hundreds or thousands of samples. Therefore, important problem is the selection of optimal architecture for neural network, which was firstly proposed and tested. After that, four different cases for recognition were researched: speaker-dependent recognition without noise, speaker-independent recognition without noise, speaker-dependent recognition with noise, speaker-independent recognition without noise. Finally, we analyse the experiment results that showed good results for all cases of interest.

Keywords: man-machine interface, speech recognition, convolutional neural networks

1. Introduction

A reliable and rational man-machine interface is an important task of modern engineering [1–3]. In recent years in avionics we witness a considerable growth of attention to audio interfaces and speech recognition [4–9], since their successful implementation may improve the flight safety and reduce the pilots' workload [1, 9]. For example, on Eurofighter Typhoon since 2005 a speaker-dependent system based on the comparing with the patterns is employed.

The principal requirement for speech interface in avionics is high probability of correct word recognition even under strong noises, because an error in aircraft control may be critical for the flight safety. In addition, the time complexity of the algorithm is an important point for onboard software.

In this paper one proposes the algorithm based on convolutional neural networks on small training sets which shows a good performance for recognition of separate words. The description of the method along with proposed structure of used convolutional neural network is presented in theoretical part. The description of employed dataset and th results of the method for several cases are presented in the experimental part.

2. Speech signal parametric portrait in automatic speech recognition

The traditional methods for automatic recognition of speech commands are based on the time-spectral quantisation of an input word record. The transform used in this article obtains a time-spectral parametric portrait and includes the following steps. In the beginning, a speech signal is divided into equal intervals of 20–50 ms. After that, one uses the standard digital signal processing procedures, such as amplification of high-frequency signal components, interval weighing by Hannah window, fast Fourier transform, averaging over frequencies and logarithm of spectral densities [10]. Each of these intervals is projected into 30–40 frequency bands [1]. As a result, for each of the time intervals bands we obtain the estimates of spectral densities logarithms in 30–40 frequency bands [4, 7, 9].

These parameters, presented in a form of a matrix, we call the parametric portrait of the word. The columns of this portrait characterize the spectral decomposition of the speech signal for each time interval.

3. Development of the algorithm of convolution neural network

This section describes the development of algorithms for automatic recognition of speech commands based on convolutional neural networks. When conducting experiments on neural networks, parametric portraits of small sizes were used, and the best results for word recognition were obtained on portraits with 18 frequency bands and 25 time intervals.

To train neural networks of forward propagation, it is common to use the back propagation method. There are batch, stochastic and mixed, i.e. mini-batch implementations of this method. The latest implementation is the most used at the moment and will be applied in this work. Its features include some randomness in the learning process, which leads to slightly different recognition results with the same training parameters. Moreover, in some neural networks regularization is used, which also introduces a small randomness in the learning process. From this we can conclude that all the results are approximate and may not be reproduced exactly at repeated launches.

The simulation of artificial neural networks was carried out in a software package written in the Python programming language using the TensorFlow and TFLearn libraries. TensorFlow is an open source low-level library designed to solve the problems of construction and training neural networks [11]. TFLearn is a high-level library for simplifying work with neural networks; it is a wrapper for the TensorFlow library [12]. Both libraries can perform calculations both on the central processor and on the graphical processor. The use of the latter is in priority, since such processors contain a lot of low-performance cores, which are well suited for the process of training of neural networks, the algorithm of which is efficiently parallelized. To effectively use the graphics processor in the program, the software-hardware architecture of parallel computing CUDA is used, actively using mentioned above plug-in libraries.

In the beginning, there was provided a search for neural network architectures most suitable for use in speech recognition problem. Traditional single and double layer perceptrons showed unsatisfactory recognition results, with the number of errors approximately equal to 30–40%. Further, new architectures of artificial neural networks, mainly recurrent networks and deep learning networks, were tested.

The input array is a two-dimensional parametric portrait. Here, unlike single-layer and multi-layer perceptron, there is no need to convert a two-dimensional parametric portrait into a one-dimensional array. Next comes the convolutional layer with 32 convolutional filters of size 3×3. Then follows a subsample layer with a kernel size of 2, that is, each section of each feature map of size 2×2 is converted to one element in a new feature map. After that, there is a convolutional layer with 64 filters, each of which is applied to all feature cards and at the output of this layer 64 feature cards are obtained. Next is another subsample layer with a kernel size of 2, followed by three fully connected layers with 128, 256 and 20 elements, respectively, where 20 is the number of recognizable words. Thus, the total number of parameters of this neural network is $(3 \times 3 \times 1 + 1) \times 32 + (3 \times 3 \times 32 + 1) \times 64 + (64 \times 3 \times 5 + 1) \times 128 + (128 + 1) \times 256 + (256 + 1) \times 20 = 179'988$.

In convolutional layers, as well as in multi-layer perceptron, the ReLU activation function is used. The first two fully connected layers use the hyperbolic tangent as an activation function, which is given by the equation $f(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$. Also, after both hidden, fully connected layers, regularization is applied to prevent overfitting of the network. For the last fully connected layer, the softmax function is used to normalize the obtained output probabilities. To train this type of neural network, cross entropy is used as a loss function. The learning process uses the Adam optimization for stochastic gradient descent.

The number and sizes of filters shown above, as well as the dimensions of the final fully connected layers are taken from empirical recommendations for image recognition problem solved by convolutional neural networks. A parametric portrait is a some kind of image, so it was decided to use similar parameters.

An important issue is the use of convolutional neural networks for small training sets, the size of which in our case is from 300 to 4900 samples. It is also worth noting that there is only 20 classes for word recognition problem. The number of elements of each class in the training set is not less than 30.

Although initially convolutional neural networks were trained on large sets, recently encouraging results have been obtained on small training sets. For example, [13] used 27000 elements from 121 unbalanced classes for image recognition, while [14] used the CIFAR-10 database with 50000 elements from 10 classes in the training set [15]. At the same time, under certain conditions, convolutional neural networks show good results when training on small sets containing from 100 to 1000 elements, even with the number of trained parameters about several hundred thousand [16]. The number of gradient updates at one iteration is equal to the number of elements in the training set; therefore, it is necessary to pay special attention to the number of iterations and to verify the convergence of the learning process. It is recommended to use the ReLU activation function for faster convergence of the learning process and various regularization methods to prevent overfitting. All these recommendations were applied in the employed convolutional neural network, which gives hope that a good result can be achieved.

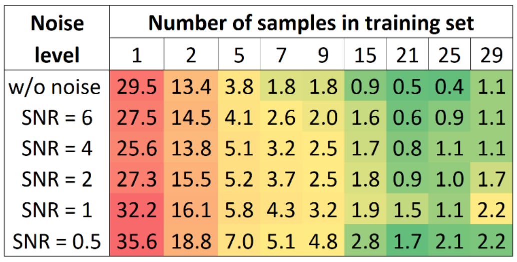
The model parameters and the optimal network architecture was selected on a small training sample when recognizing words outside this sample.

The optimal parameters of the neural network are 10000 for the number of learning iterations, 0.0003 for the learning speed and 600 for the size of the pack.

4. Results

4.1 Own speaker recognition without noise

The following experiment was carried out. Samples of each speaker were divided into 2 parts. In the first part, a convolutional neural network was trained, and the second part was used as a test set. The results of these recognitions are shown in the first line in the Figure 1. The columns show the number of samples for each word in the training set. The rows indicate the noise level in recognized samples which would be described in the following subsections.



Noise level	Number of samples in training set								
	1	2	5	7	9	15	21	25	29
w/o noise	29.5	13.4	3.8	1.8	1.8	0.9	0.5	0.4	1.1
SNR = 6	27.5	14.5	4.1	2.6	2.0	1.6	0.6	0.9	1.1
SNR = 4	25.6	13.8	5.1	3.2	2.5	1.7	0.8	1.1	1.1
SNR = 2	27.3	15.5	5.2	3.7	2.5	1.8	0.9	1.0	1.7
SNR = 1	32.2	16.1	5.8	4.3	3.2	1.9	1.5	1.1	2.2
SNR = 0.5	35.6	18.8	7.0	5.1	4.8	2.8	1.7	2.1	2.2

Figure 1. The average results of word recognition by a convolutional neural network under different noise levels when learning on the same speaker, the table shows the percentage of incorrect recognition.

One could see that it is enough to study on 7 samples for the error rate to fall below 2%, on 15 samples to falls below 1% and on 21 samples for 0.5% of errors. These results show what error values should be sought when recognizing samples of other speakers to exclude the effect of speaker dependence.

4.2 Other speakers recognition without noise

After that, recognition was carried out using samples of a single speaker. This means that the neural network was trained on the samples of one speaker and then the samples of all speakers were recognized on this neural network. When recognizing words from a training set, an average of 0.0% of errors is obtained, and for recognizing words from a test set, 5.9% of errors.

Further, recognition was carried out using samples of three speakers. When recognizing words from a training set, on average, 0.0% of errors are obtained, and for recognizing words from a test set, 1.7% of errors.

For recognition using a larger number of speakers, several training sets were constructed. Each set consists of samples of 7 different speakers. Further, on neural networks trained on compiled sets, a recognition was made on training and test samples. There is no errors when recognizing words from the training set, and on average, 0.6% of errors are obtained for recognizing words from the test set.

The Table 1 shows the total results of recognition of samples without noise when learning on a various number of speakers. From the results we can conclude that the percentage of errors steadily decreases with an increase in the number of speakers in the training set. A greater number of speakers allows you to get rid of speaker dependence, which positively affects the quality of recognition.

Table 1. Words recognition results on training sets containing a different number of speakers for the case of samples without noise.

Number of speakers	Percentage of errors
1 speaker	5.9%
2 speakers	2.4%
3 speakers	1.7%
4 speakers	1.2%
5 speakers	1.0%
6 speakers	0.8%
7 speakers	0.6%

4.3 Own speaker recognition with noise

This subsection presents the results of recognition of samples with noise. The noise was modeled as follows. For each sample, unique pieces of noise were selected from the noise record of the cockpit of a Boeing aircraft with an average volume of more than 80 dB. Also, the noise volume was normalized so that the signal-to-noise ratio was equal to some fixed value.

An experiment was conducted to recognize samples of that speaker, samples of which were used as a training set. As well as in the case of noise-free samples, samples of each speaker were divided into 2 parts: the convolutional neural network was trained on the first part, and the second part was used as a test set. The results of these recognitions are shown in the Figure 1. The columns show the number of entries for each word contained in the training set. The columns indicate the level of added noise. The results are averaged over all speakers.

It can be seen that the results monotonically deteriorate with increasing noise level, while the magnitude of the deterioration is negligible. For example, when using 15 samples in the training set, in the absence of noise, 0.9% of errors are obtained, and with a signal-to-noise ratio of 1, 1.9% of errors are obtained. The similar values for training on 9 samples are 1.8 and 3.2%, respectively. These results show what error values should be sought when recognizing samples with the noise of other speakers to eliminate the effect of speaker dependence.

4.4 Other speakers recognition with noise

After that, the usual recognition was carried out by one speaker with one additional variant of noise. Hereinafter, the noise volume was normalized so that the signal-to-noise ratio was 4. To increase the training set and to improve the learning process, various unique noise variants can be added to samples.

2 variants of the experiment were conducted. In the first variant, training process was conducted for the samples with noise, and samples without noise were recognized. The average error for the recognition of the speaker's samples on which training was conducted is 0.3%, and the average error for the samples of other speakers is 11.8%. In the second variant, both training and recognition were carried out on samples with noise. There were no errors on the samples on which the training was conducted, and the average error for the samples of other speakers was 9.7%. It can be seen from these results that for both recognition options, the error is higher than when using samples without noise. An experiment was also conducted in which training was conducted on samples without noise, and recognition on samples with noise. For this option, the number of errors was several times greater than for the previous ones, so this option will not be used in the future.

Then a similar experiment was carried out, but with the difference that for training one used samples with seven different added noise variants. For the first experiment, the average error in the recognition of the speaker's samples on which training was conducted is equal to 0.0%, and the average error for the samples of other speakers is 8.5%. For the second experiment, there were no errors on the samples on which the training was conducted, and the average error for the samples of other speakers was 8.2%. As can be seen from the results, the use of several noise variants significantly improves the quality of recognition.

After that, the neural network was trained on 3 speakers in noise conditions and subsequent recognition of samples. The first experiment involves the use of one variant of noise in training process, and the second experiment - 3 variants of noise. In both cases, the average error when recognizing the speaker's samples on which training was conducted is equal to 0.0%. For the case of samples recognition of other speakers, the error is 2.8% for the first experiment and 2.5% for the second.

From the results obtained in this and previous experiments, it can be concluded that the use of additional noise options added on the same source samples helps to improve the training process of the neural network and, as a result, the quality of recognition of speech commands. That is, practice shows that the more noise options are used, the better the results. The only limitation is the computing power of the computer used to train the neural network.

For the experiment on samples with added noise during training on 7 speakers, 3 variants of noise were used. When recognizing words from a training set, the average error is 0.0%, and for recognizing words from a test set, an average of 1.1% errors were obtained.

The Table 2 shows the total results of recognition of samples with noise when learning on a various number of speakers and a various amount of added noises.

Table 2. The total recognition results on the training set and on the test set when learning on a various number of speakers for cases of samples without noise and with noise, the table shows the percentage of incorrect recognition.

Number of speakers	Noise in training set	Number of noises	Percent of errors
1 speaker	without noise	1	11.8%
		7	8.5%
	with noise	1	9.7%
		7	8.2%
3 speakers	with noise	1	2.8%
		3	2.5%
7 speakers	with noise	1	1.2%
		3	1.1%

From the results we can conclude, similar to the conclusion in the previous subsection, that the percentage of errors steadily decreases for the same reasons with an increase in the number of speakers in the training base. It can also be concluded that an increase in the amount of added noises leads to a decrease in the number of errors, but this effect from an increase in the training set is less pronounced. A large number of noises in the training set reduces the effect of overfitting for a certain type of noise, which reduces the number of errors in the recognition of samples with other noise. In other words, "noise independence" is similar to speaker independence and is achieved by increasing the number of different noises in the training sample.

Both the results of experiments with noise and the results without noise turned out to be the best when using convolutional neural networks of deep learning. The disadvantages of this algorithm include the great computational complexity of network training. At the same time, recognition itself occurs almost instantly.

5. Conclusion

In this work, one proposed the convolutional neural network architecture, which showed good recognition results in solving several different cases with and without noise.

The first case was a recognition of samples of a speaker whose samples were also used in training process. The results showed that it is possible to achieve a recognition error of 2.0% by using 7 samples of each word in training set and a recognition error of 0.5% for 20 samples of each word.

The second case was to train on samples of one or more speakers and then to recognize sample of someone else's speaker. For this configuration, a share of errors is equal to 5.9% for a training sample of one speaker, 1.7% for a sample of 3 speakers and 0.6% for 7 speakers. The last result is very good, since the result of speaker-independent recognition comes close to the result of speaker-dependent recognition.

The third case was about testing the efficiency of convolutional neural networks during training and recognition of samples with noise. Here, unique parts of the cockpit noise were added to all samples. When recognizing samples of own speaker, the number of errors increased by about 2 times when using a signal-to-noise ratio of 1. A stable, but not very large increase in the number of errors with increasing noise level was also shown. When recognizing samples of other speaker, a share of errors was equal to 9.7% for learning from samples of one speaker with one added noise, and if 7 different noise samples were added to the training set, the error was reduced to 8.2%. For 3 speakers, the error rate was 2.8% for one noise sample and 2.5% for 3 noise samples. For 7 speakers these values were 1.2 and 1.1% respectively, which is an extremely low value for recognition under noise conditions.

Acknowledgements

This work was supported by Russian Foundation for Basic Research (RFBR), project 18-08-01142-a.

References

[1] Evdokimenkov V, Kim R, Krasil'shchikov M and Sebryakov G 2015 *Journal of Computer and Systems Sciences International* **4** 609–20
[2] Polyak B and Khlebnikov M 2017 *Automation and Remote Control* **3** 490–506
[3] Kolokolov A 2006 *Problemy upravleniya* **3** 13–8
[4] Rabiner L 1993 *Fundamentals of Speech Recognition*
[5] Benesty J, Sondhi M and Huang Y 2007 *Springer Handbook on Speech Processing and Speech Communication* (Heidelberg: Springer-Verlag Berlin)
[6] Schmidt-Nielsen A, Marsh E, Tardeli J, Gatewood P, Kreamer E, Tremain T, Cieri C and Wright J 2000 *Speech in Noisy Environments (SPINE) Evaluation Audio* (Philadelphia, PA: Linguistic Data Consortium)
[7] Kolokolov A and Lublinsky I 2015 *Automation and Remote Control* **10** 144–51
[8] Savchenko L 2014 *Informatsionno-upravlyayushchiye sistemy* **1** 23–31
[9] Korsun O and Poliev A 2016 *Journal of Computer and Systems Sciences International* **4** 609–18
[10] Oppenheim A, Buck J and Shafer R 2001 *Discrete-Time Signal Processing*
[11] TensorFlow: An open source machine learning framework for everyone 2018 https://www.tensorflow.org
[12] TFLearn: Deep learning library featuring a higher-level API for TensorFlow 2018 http://tflearn.org
[13] Dieleman S, van den Oord A, Korshunova I, Burms J, Degrave J, Pigou L and Buteneers P 2015 http://benanne.github.io/2015/03/17/plankton.html
[14] Truong T, Nguyen V and Tran M 2018 *ICPRAM* 675–82
[15] CIFAR-10 and CIFAR-100 datasets 2018 https://www.cs.toronto.edu/~kriz/cifar.html
[16] Andrew L 2017 https://beamandrew.github.io/deeplearning/2017/06/04/deep_learning_works.html