Slovenian command word speech recognition using transfer learning

Blaž Kovačič
blaz.kovacic@student.um.si
Faculty of Electrical
Engineering and Computer Science,
University of Maribor
Koroška cesta 46
SI-2000 Maribor, Slovenia

Borko Bošković
borko.boskovic@um.si
Faculty of Electrical
Engineering and Computer Science,
University of Maribor
Koroška cesta 46
SI-2000 Maribor, Slovenia

ABSTRACT

In this work, we present a speech recognition system using convolutional neural networks that are able to differentiate between four different Slovenian command words: naprej, nazaj, levo, desno. In neural network training, we intentionally limited ourselves to only 20 audio recordings per command word from a single speaker. We used transfer learning in conjunction with audio augmentation on a pre-trained model that we previously trained on 10 different English words with a total of about 25000 recordings from the Google Speech Commands Dataset v0.01. Using the transfer learning approach, we achieved highly accurate results on the test set with an accuracy of 0.988. Additionally, we demonstrate the soundness of our approach by comparing the results with those obtained when training an empty model from scratch, observing a substantial difference in results.

KEYWORDS

Speech recognition, transfer learning, human-computer interaction

1 INTRODUCTION

Speech recognition is an important interdisciplinary field that is present at every step of our everyday lives. It is used in home automatization with virtual assistants, in the transcription of medical documentation, in smart telephone answering machines, in the automatic generation of subtitles for deaf and hard of hearing, and in the speech controlled assistive technologies for disabled people.

Speech recognition technology goes back to the 50s of the previous century [5] when Bell Laboratories developed a system that was able to recognize spoken digits for a single speaker. We can generally differentiate between continuous speech recognition and isolated-word speech recognition, where the latter task is concerned with the recognition of a single word at a time. Today the technology in use for continuous speech recognition is mostly based on deep neural networks, convolutional neural networks (CNNs), and statistical models such as the Hidden Markov Model [1].

In our work, we focus on isolated-word speech recognition using CNNs, which means we recognize a single word in isolation. When building customized speech-controlled systems, such as an assistive technology solution for a specific person with special needs, we may be faced with limited availability of training data for our speech recognition system. This is especially true for the Slovenian language, where, to the best of our knowledge, doesn't exist a large database containing recordings of short command words. We address this issue in our work, by limiting ourselves to only 20 recordings per command word from a single speaker, and then use transfer learning and data augmentation techniques to develop a highly accurate speaker-dependent Slovenian command word speech recognition system.

The first step we take to develop such a system is to build a base model using large amounts of one-second-long recordings (about a total of 25000 recordings) using the Speech Commands Dataset v0.01 by Google [7]. We then use the technique of transfer learning on this model, by freezing all neural network layers except for the layers in last 5 blocks (corresponding to 14 out of a total of 27 layers) and then training those top layers using the limited data that we have. We demonstrate that using such an approach yields very favorable results and allows the model to generalize much better to the test data.

2 RELATED WORK

In the related work [6] on which we build upon and extend it using the idea of transfer learning, the authors used a deep learning approach for isolated-word speech recognition using CNNs. The authors recognized 10 different keywords such as "left", "right", "on", "off". They compared different approaches of neural network training: In the first approach they used raw audio data in conjunction with 1D CNN, whereas in the other two approaches they used Mel-spectrograms and Mel-frequency cepstral coefficients (MFCCs) as features that were input into a 2D CNN, where the latter approach yielded the best results (accuracy of 0.9619).

In [2], the authors used transfer learning for an automated speech recognition task, adapting a Wav2Letter CNN. They used an English base model to train a German model that outperformed the German baseline model trained from scratch. This can be especially helpful when limited amounts of training data and limited GPU memory are available; in addition, the training time is significantly reduced, and the final model accuracy is much higher.

In [3] the authors worked on a problem of keyword detection (similar to "Ok Google" detection in Google Assistant) where they compared three approaches: neural network with one hidden layer (a so-called "vanilla" neural network), a deep neural network with three hidden layers, and a CNN. The first approach is very fast, but such oversimplified network architecture did not yield optimal results; using the second approach the authors obtained an accuracy of 0.719, whereas using CNN yielded an accuracy of 0.945.

In [4] the authors worked on a task of large-vocabulary continuous speech recognition. Using raw audio signals, the authors trained a CNN to extract features (using convolutional filters) of basic word

DOI: https://doi.org/10.26493/scores23.04

units - phonemes. They relayed the output of the CNN to the next part of the neural network containing linear units that returned conditional probabilities of phonemes for each frame of speech recording. The authors then decoded this sequence of phonemes using a hidden Markov model into final words.

3 EXPERIMENTAL SETUP

3.1 Environment

We performed the experiment on Windows 11 using the GeForce RTX 3070 GPU with 8GB VRAM and central processing unit AMD Ryzen 7 5800H, 3.20GHz. We installed the TensorFlow 2.10 package for Python and executed code on WSL2 Ubuntu (Windows Subsystem for Linux). We used librosa library to obtain MFCC features, and audiomentations library for audio data augmentation.

3.2 Base model design

In the first part of the experiment, we build upon the work of Soliman, et. al. [6]. We use the Speech Commands Dataset v0.01 which contains one-second-long recordings of short English words such as "Happy" or "Marvin". For each word class there are a total of about 2500 recordings from various speakers. Just as authors in [6], we limit ourselves to the following 10 word classes: "yes" (2377), "no" (2375), "up" (2375), "down" (2359), "left" (2353), "right" (2367), "on" (2367), "off" (2357), "go" (2372) and "stop" (2380). In addition to these classes, we include the class for "Silence" ($6 \times 60s$ split into 360 one-second-long recordings) and "Other". The class "Silence" includes recordings of environmental noise, whereas the class "Other" contains 1000 one-second-long recordings of words from 10 classes other than those we trained it upon, such as "Happy" or "nine"; we used 100 recordings for each of 10 out-of-vocabulary classes. In the implementation, we use the 2D CNN architecture from [6] which is shown in Table 1. We sample the audio using a 16 kHz sampling frequency. We first put each audio recording through a pre-processing phase where we use a pre-emphasis high-pass filter to put more emphasis on higher frequencies of the audio, after which we z-normalize the audio by subtracting the mean and dividing each audio sample by the standard deviation. We then proceed with feature extraction where we use Mel-frequency cepstral coefficients (MFCCs) as features. For each time window, we extract 40 MFCC coefficients from the audio, using a window length of 25 ms and a hop length of 10 ms. The resulting CNN input shape is (40, 101, 1). We split data into training, validation and test sets using the ratios of 70:10:20. The neural network was trained in 200 epochs with a batch size of 64.

Block	Layer Type	Units	Kernel Size
1	Conv2D, BN, DO	32	(3, 3)
2	Conv2D, BN, MP, DO	32	(3, 3)
3	Conv2D, BN, DO	64	(3, 3)
4	Conv2D, BN, DO	64	(3, 3)
5	Conv2D, BN, MP, DO	64	(3, 3)
6	Conv2D, BN, DO	128	(3, 3)
7	Conv2D, BN, MP, DO	128	(3, 3)
8	Flatten	-	-
	Dense	1000	-
9	Dense	L2 (Softmax)	-

Table 1: Base model CNN architecture based on [6], where Conv2D corresponds to 2D convolutional layer, BN is Batch Normalization layer, MP is the Max Pooling 2D layer, and DO is 0.5 Dropout layer

3.3 Transfer learning approach

We recorded and classified one-second-long audio recordings into five classes that correspond to command words spoken in Slovenian language, namely: "naprej" (forward), "nazaj" (backward), "levo" (left), "desno" (right), and *Silence* class. The Silence class is composed of one-second-long recordings of environmental noise.

By design, we limited our training and validation data to 20 unique recordings per class, recorded with a smartphone for a single speaker. Additionally, we expanded the training and validation data (but not the test data) using audio augmentation technique, generating 100 additional recordings for each class. We used the audiomentations Python library to augment the recordings, varying the pitch by -2 to 2 semitones with a probability of 0.5, time stretching the audio by a rate between 0.8 and 1.25 with a probability of 0.5, and shifting the audio recordings by a fraction between -0.3 and 0.3 with a probability of 0.5. We split the resulting 120 recordings per class in ratio 85:15 among training and validation sets (giving us 102 samples/class in the training set, and 18 samples/class in the validation set).

The test data was recorded by the same speaker, but from a laptop microphone and at a different distance. In addition, the test data contains unique (not augmented) recordings, namely 100 unique recordings per class, each containing a range of natural variations in pitch and speed of pronunciation, which was done to obtain a sure estimate of model accuracy.

We then used the pre-trained base model and replaced the last softmax layer with a new one (for our 4+1 classes), and froze all layers in blocks 1-4 (see Table 1), training only layers from last 5 blocks (layers from *Conv2D 64* to *Dense 5 Softmax*). We decided upon training exactly 5 blocks empirically, as much fewer or much more than that had a negative affect on accuracy. In our experimental results, we demonstrate the significant difference that this approach makes in comparison to training all layers of a pre-trained model, as well as in comparison to model training from scratch.

For training, we used the Adam optimizer with a learning rate of 1e-3, batch size of 64, an early stopping callback with patience of 10, that monitors validation loss and prevents overfitting.

4 EXPERIMENTAL RESULTS

4.1 Base model

Training the base model persisted until the 105-th epoch, when it was interrupted by an early stopping mechanism, resulting in a training accuracy of 0.973, validation accuracy of 0.950, and a test accuracy of 0.954. The average class F1 measure on the test set was 0.956. Model accuracy with respect to the epoch number is given in Figure 1, and the confusion matrix is shown in Figure 2.

The obtained results are very much in line with those of the reference paper [6], and serve as a good basis on which to build upon using transfer learning.

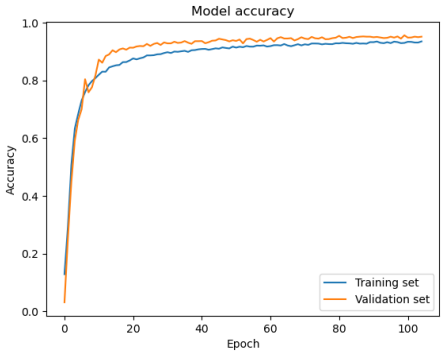

Figure 1: Base model train and validation accuracy with respect to the epoch number for 2D CNN using MFCC features.

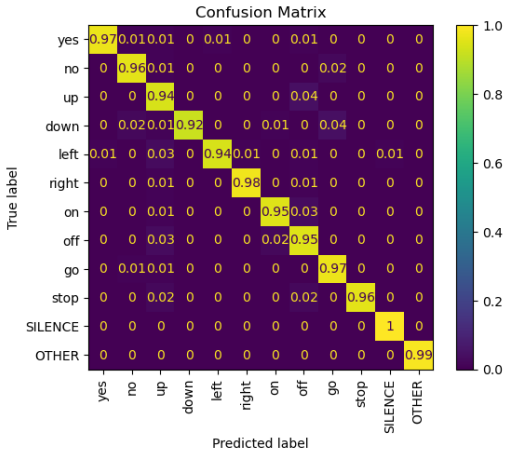

Figure 2: Base model confusion matrix for 2D CNN using MFCC features.

4.2 Transfer learning approach

We demonstrate the soundness of our approach by first displaying results of transfer learning when freezing all layers except for those in the last 5 blocks of the pre-trained base model, and then comparing that with the results obtained by training *all* layers of the pre-trained base model, which results in a worse model that exhibits lower accuracy on the test data. Finally, we demonstrate the unfeasibility of the approach where the model is trained without transfer learning using uninitialized weights, showing a substantial difference in accuracy.

4.2.1 Results when training last 5 blocks of the pre-trained model. The pre-trained base model was used and all layers except for those in last 5 blocks were frozen. The training was early-stopped after 99 epochs and lasted less than a minute. Training and validation accuracy of 1.00 were obtained, with a test accuracy of 0.988. The average class F1 score on the test set was 0.988. Model accuracy with respect to the epoch number is shown in Figure 3, and the confusion matrix is shown in Figure 4.

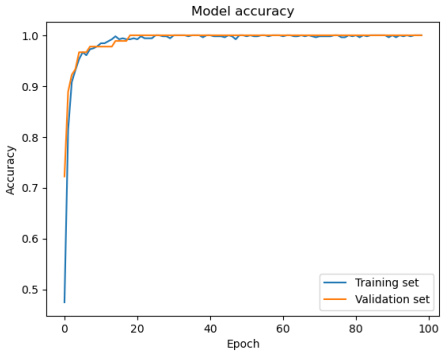

Figure 3: Transfer learning model (trained on layers from last 5 blocks) train and validation accuracy with respect to the epoch number for 2D CNN using MFCC features.

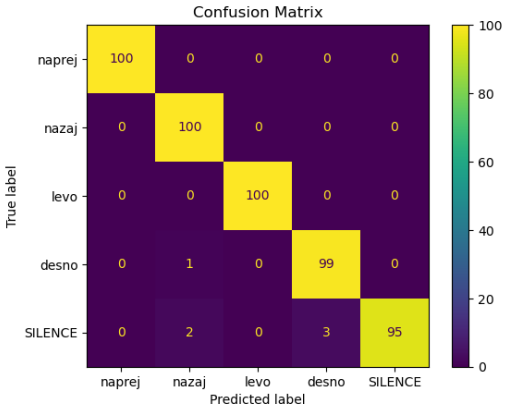

Figure 4: Transfer learning model (trained on layers from last 5 blocks) confusion matrix.

4.2.2 Results when training all layers of a pre-trained model. The pre-trained base model was used and all layers were trained. The training was early-stopped after 125 epochs. The obtained test accuracy of 0.930 was lower than when using the previous approach. The average class F1 score on the test set was 0.928. The confusion matrix for this setting is shown in Figure 5, demonstrating a somewhat degraded accuracy when compared to the model that was trained only on layers from the last 5 blocks.

4.2.3 Results without transfer learning. The original model architecture was used and all layers were trained from scratch. It was observed from high variations in validation accuracy with respect to the epoch number, that overfitting occurred, and that the model didn't generalize to the test data, which was observed from train accuracy being 0.9, validation accuracy of 0.92, and a test accuracy of 0.418, with an average F1 score on test set being 0.352. The large difference between the test and validation accuracy is most likely due to the fact that the test set was generated in a different way than the train and validation sets, and was thus more challenging.

To further test the feasibility of training from scratch with such a small dataset, we decided to implement a much smaller model with

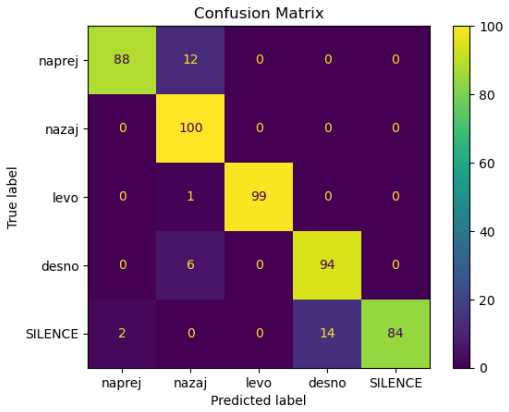

Figure 5: Transfer learning model (trained on all layers) confusion matrix.

the architecture consisting of Conv2D layer with 32 units, followed by MaxPolling2D layer and by Conv2D with 64 units, followed by MaxPooling2D layer, and then by Flattening and Dense 64 layer, and finally a Softmax layer with 5 units. The model was trained for 106 epochs; from the learning curve it was observed that model validation accuracy converged quickly and without large variations. Training accuracy and validation accuracy were both 1.0, whereas test accuracy was merely 0.695, and the average F1 score of 0.696 on the test set. Even though the validation accuracy was very high, the model wasn't able to predict the test data very well.

5 DISCUSSION

In summary, we can see that it is possible to make use of transfer learning in combination with audio data augmentation to train a convolutional neural network with very small amounts of training data, yielding a useful model that can be utilized in a command word speech recognition system with favorable accuracy.

We could see that by freezing an optimal amount of bottom CNN layers, which contain filters that detect less complex features than top layers, we can fine tune the base model and improve accuracy.

The approach, however, is not without limitations. One limitation that we observed was that the choice of command words that we decided to recognize could influence the system's ability to differentiate between them. Additionally, it was observed that the neural network seems to overgeneralize, assigning overly confident classification probability scores to erroneous classes when presented with "out-of-vocabulary" words, although this was observed to occur with the base model as well. This leaves room for further research into methods of out-of-vocabulary word detection when utilizing CNNs for spoken word command recognition.

6 CONCLUSION

We presented a speaker-dependent speech recognition system that is successfully able to differentiate between four different Slovenian command words (including a class for *silence*), trained on very limited amounts of training data, namely 20 recordings per word class. We achieved a high recognition accuracy of 0.988 by making use of the transfer learning approach, in combination with the audio data augmentation technique.

First we augmented the 20 recordings in each class by additional 100 recordings, by synthetically varying the pitch, time stretching and time shifting the audio recordings, giving a total of 120 recordings for each class in the training and validation sets. For training, we made use of a pre-trained convolutional neural network model which we previously trained on large amounts of recordings of 10 different word classes from the Google Speech Commands Dataset, using MFCCs as features. Obtained model accuracy was 0.954, which was very much in line with results of the reference paper [6]. We then used this model by training only the layers from its last 5 blocks (14 layers out of a total of 27 layers). To obtain a realistic classification accuracy score, we built a separate test set, using a different microphone and a different recording distance than in recordings utilized for the training and validation sets. This test set consisted of 100 unique recordings for each class, from a single speaker, and was not augmented.

We demonstrated that freezing the optimal amount of bottom layers yields better training results (accuracy of 0.988) than when training all layers of a pre-trained model (accuracy of 0.930). We have also shown that when using our small dataset, model training without transfer learning yielded much lower test accuracy scores, namely, 0.418 and 0.695 for the larger and smaller models, respectively.

We can conclude that by making use of the transfer learning in combination with data augmentation, we can compensate for small amounts of training data, allowing us to build a potentially useful system that can respond to the Slovenian spoken word commands from a single speaker. In future work we would like to research and compare methods that would allow for accurate detection of "out-of-vocabulary" words, such as the use of Mahalanobis distance or Bayesian neural networks. Additionally, we would like to research the model prediction accuracy and the optimal amount of recordings that would be required under the condition that same commands are spoken by different speakers.

REFERENCES

[1] Geoffrey Hinton, Li Deng, Dong Yu, George E. Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N. Sainath, and Brian Kingsbury. 2012. Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups. *IEEE Signal Processing Magazine* 29, 6 (2012), 82–97. https://doi.org/10.1109/MSP.2012.2205597

[2] Julius Kunze, Louis Kirsch, Ilia Kurenkov, Andreas Krug, Jens Johannsmeier, and Sebastian Stober. 2017. Transfer learning for speech recognition on a budget. *arXiv preprint arXiv:1706.00290* (2017).

[3] Xuejiao Li and Zixuan Zhou. 2017. Speech command recognition with convolutional neural network. *CS229 Stanford education* (2017), 31.

[4] Dimitri Palaz, Mathew Magimai.-Doss, and Ronan Collobert. 2015. Convolutional Neural Networks-based continuous speech recognition using raw speech signal. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 4295–4299. https://doi.org/10.1109/ICASSP.2015.7178781

[5] Melanie Pinola. 2011. *Speech Recognition Through the Decades: How We Ended Up With Siri.* https://www.pcworld.com/article/477914/speech_recognition_through_the_decades_how_we_ended_up_with_siri.html Accessed on 2023-03-19.

[6] Aljenan Soliman, Salah Mohamed, and Iman Abuelmaaly Abdelrahman. 2021. Isolated Word Speech Recognition Using Convolutional Neural Network. In *2020 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE)*. 1–6. https://doi.org/10.1109/ICCCEEE49695.2021.9429684

[7] Pete Warden. 2018. Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition. *CoRR* abs/1804.03209 (2018). arXiv:1804.03209 http://arxiv.org/abs/1804.03209