A neural attention model for speech command recognition

Douglas Coimbra de Andrade[a], Sabato Leo[b], Martin Loesener Da Silva Viana[c], Christoph Bernkopf[c]

[a]*Laboratory of Voice, Speech and Singing, Federal University of the State of Rio de Janeiro*
[b]*Adecco Italia S.P.A - GSK Vaccines Srl*
[c]*CERN*

Abstract

This paper introduces a convolutional recurrent network with attention for speech command recognition. Attention models are powerful tools to improve performance on natural language, image captioning and speech tasks. The proposed model establishes a new state-of-the-art accuracy of 94.1% on Google Speech Commands dataset V1 and 94.5% on V2 (for the 20-commands recognition task), while still keeping a small footprint of only 202K trainable parameters. Results are compared with previous convolutional implementations on 5 different tasks (20 commands recognition (V1 and V2), 12 commands recognition (V1), 35 word recognition (V1) and left-right (V1)). We show detailed performance results and demonstrate that the proposed attention mechanism not only improves performance but also allows inspecting what regions of the audio were taken into consideration by the network when outputting a given category.

Keywords: human voice, command recognition, attention mechanism, deep learning

1. Introduction

Currently, many human-computer interfaces (HCI) like Google Assistant, Microsoft Cortana, Amazon Alexa, Apple Siri and others rely on speech recognition. There is a significant amount of research in the field by all major companies, notably Google and Baidu (Amodei

Email addresses: douglas@cmsoft.com.br (Douglas Coimbra de Andrade), sabato.leo@gmail.com (Sabato Leo), martin.eduard.loesener@cern.ch (Martin Loesener Da Silva Viana), christoph.bernkopf@cern.ch (Christoph Bernkopf)

et al. (2015), Chiu et al. (2017)). However, these systems rely on powerful neural network models that usually run in the cloud due to the computation required to transform speech to text, perform natural language processing of user intent and react appropriately. Simple commands, such as "stop" and "go", that could be processed locally, go through the same processing stages. Therefore, industry applications which do not have the benefit of uninterrupted broadband internet connection cannot incorporate speech recognition in the same manner. The development of lightweight speech command models would enable the development of a multitude of novel engineering applications, such as voice-controlled robots for critical missions and assistive devices that can operate in areas without internet coverage. This aspect is particularly important when designing microcontrollers that support voice-driven commands (Zhang et al. (2017)).

This work introduces a novel attention-based recurrent network architecture designed to recognize simple speech commands, while still generating a lightweight model that can be loaded in mobile devices and run locally.

The main contributions of this work are:

1. Design of a novel recurrent architecture with attention that achieves state-of-the-art performance in command recognition and language identification from speech and is small enough to be run locally;
2. Visualization of attention weights and discussion about how attention improves accuracy and makes the speech recognition model explainable;
3. Source code (to be made available at https://github.com/.... after acceptance – blind review).

Results are presented using Google Speech Command datasets V1 and V2. For complete details about these datasets, refer to Warden (2018).

This paper is structured as follows: Section 1.1 discusses previous work on command recognition and attention models. Section 2 presents the proposed neural network architecture. Section 3 shows results obtained on various tasks related to Google Speech Command datasets V1 and V2, as well as attention and confusion matrix plots. Section 4 summarizes

this work and presents possible directions for future work.

1.1. Related Work

Identification of speech commands, also known as keyword spotting (KWS), is important from an engineering perspective for a wide range of applications, from indexing audio databases and indexing keywords (Tabibian et al. (2013), Sangeetha and Jothilakshmi (2014)) to running speech models locally in microcontrollers (Zhang et al. (2017)).

The development of neural attention models (Bahdanau et al. (2014), Vaswani et al. (2017)) increased performance on multiple tasks, especially those related to long sequence to sequence models. These models are extremely powerful ways to understand what parts of the input are being used by the neural network to predict outputs, as shown in the case of image captioning (Xu et al. (2015)). In the case of acoustic models, Connectionist Temporal Classification (CTC) loss shows good performance in English and Mandarin speech to text tasks (Amodei et al. (2015)). There is also work on sequence discriminative frameworks (Chen et al. (2018)). However, to the best of our knowledge, attention for single word recognition has not been investigated.

Command recognition using deep residual networks has been investigated in Tang and Lin (2017), Arik et al. (2017) and Sainath and Parada (2015). In particular, the problem of limited size architectures has been extensively explored in Zhang et al. (2017). The best results achieve over 95% accuracy in specific tasks using spectrogram methods. However, most models perform significantly worse than the proposed recurrent neural network (RNN) with attention even with a greater number of parameters. Moreover, the proposed attention mechanism makes results explainable and easy to interpret, which is fundamental for engineering applications and partly solves the problem of "black box in deep learning" (Lei et al. (2018)). Results using raw waveform without any Fourier analysis have also been investigated (Jansson (2018)).

Results are presented on accuracy of left vs right (i.e., identifying only the words "left", "right" or none of those two), 20 commands and 10 non-commands and all 35 words (McMahan and Rao (2017)). Note that while the first version of the dataset only has 30 words in

the test set, V2 has all 35 (denoted by "35-word task" in this paper).

2. Neural Network Implementation

The Keras interface (Chollet et al. (2015)) was used to implement all neural networks on top of Tensorflow backend (Abadi et al. (2015)). On top of that, Python library kapre (Choi et al. (2017)) is used to provide Keras layers for mel-scale spectrogram computation. The use of kapre library provides extreme versatility to change spectrogram and mel-frequency parameters without having to preprocess the audio in any way. As a result, the inputs to the models are raw WAV files converted to numpy arrays for efficient load with Keras generators.

2.1. Proposed Architecture

Since the audio files contain a single word command that can be anywhere in the $1s$ length of the WAV file, it is reasonable to assume that any model that is able to classify an audio should also be able to find what is the appropriate region of interest. Thus, the attention mechanism seems appropriate to this particular task.

The model starts by computing the mel-scale spectrogram of the audio using non-trainable layers implemented in the kapre library. The input to the model is the raw WAV data with original sampling rate of $\sim$16 kHz. Mel-scale spectrogram is computed using 80-band mel scale, 1024 discrete Fourier transform points and hop size of 128 points ($\sim$8 s). Similar parameters have been successfully used for audio synthesis (Wang et al. (2017)) and we noticed that they preserve the visual aspect of the voice formants in the spectrogram (which would allow a human specialist to evaluate the sound - Sundberg et al. (2013), Sundberg and Thaln (2015)).

After mel-scale spectrogram computation, a set of convolutions is applied to the mel-scale spectrogram (2D output) only in the time dimension to extract local relations in the audio file. A set of two bidirectional long short term memory (LSTM - Hochreiter and Schmidhuber (1997)) units is used to capture two-way (forward and backward) long term dependencies in the audio file.

At this point, one of the output vectors of the last LSTM layer is extracted, projected using a dense layer and used as query vector to identify what part of the audio is the most relevant. We choose to use the middle vector of the LSTM output since the voice command is expected to be centered in the audio files. This choice is arbitrary and any vector should work since the double stacked LSTM layers should be able to carry enough "memory".

Finally, the weighted average of the LSTM output is fed into 3 fully connected layers for classification. Figure 1 summarizes the architecture. For complete details about the implementation, please refer to the repository.

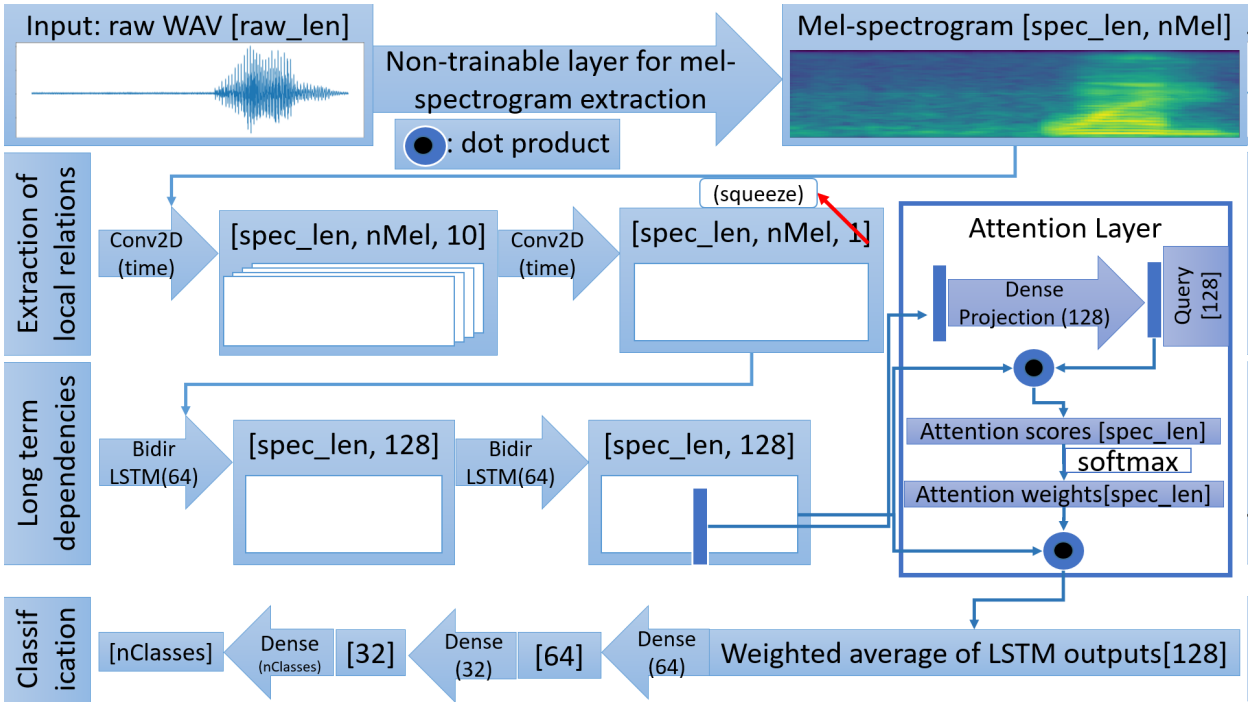

Figure 1: Proposed architecture: recurrent neural network with attention mechanism. Numbers between [brackets] are tensor dimensions. raw_len is WAV audio length (16000 in this case). spec_len is the sequence length of the generated mel-scale spectrogram. nMel is the number of mel bands. nClasses is the number of desired classes. The activation of the last Dense layer is softmax. The activation of the 64 and 32 dense classification layers is the rectified linear unit (relu).

3. Results

To facilitate comparison with previous results, 5 different tasks were considered:

- Recognition of 20 speech commands (+unknown) using Google Speech Dataset V2;
- Recognition of 20 speech commands (+unknown) using Google Speech Dataset V1;
- Recognition of 12 speech commands (+unknown) using Google Speech Dataset V1;
- Recognition of all 35 words (+unknown, only silence samples in this case) using Google Speech Dataset V1;
- Recognition of left-right words using Google Speech Dataset V1;

For each task, the proposed Attention RNN model was trained for a maximum of 40 epochs. The model with the best accuracy performance on the validation set was saved and training was stopped if no improvement was made in 10 consecutive epochs. Training was done using the "adam" algorithm (Kingma and Ba (2014)) with initial learning rate of 0.001 and decay of 0.4 every 10 epochs. The batch size used was 64. Tests in multiple training runs (3 to 5, depending on the task) show that the training procedure is consistent and standard deviation of accuracy results is 0.2% for all models. Each epoch takes approximately 180 s (V2) and 100 s (V1) to run in a Tesla K80 GPU.

Attention plots and confusion matrices are also presented to allow for better comparison of the results.

3.1. Speech Command Recognition Accuracy

The results obtained with the attention-RNN model are compared with left-right accuracy, 20 command accuracy and 35 word accuracy (McMahan and Rao (2017)), as shown in Table 1. Attention provides a substantial improvement on these tasks when compared to other models. On the V2 dataset, our results are 94.5% (20-cmd) and 93.9% (35-word) – significantly better than the 20-cmd baseline of 88.2% from Warden (2018).

Table 2 compares results of the attention RNN model when used to recognize only the 12 commands originally proposed in the Kaggle competition (Google Brain (2018)). Results are also presented for attention RNN trained and tested on V2 dataset.

Table 1: Accuracy results on the Google Speech Command Dataset V1. DenseNet-101 results from McMahan and Rao (2017). ConvNet results from Warden (2018). Our attention Model results on the Google Speech Command Dataset V2 are also reported in the last row.

	Accuracy (%)		
Model	20-cmd	35-word	left/right
DenseNet-121 No pretrain, no multiscale	81.32	80.13	89.19
DenseNet-121 Pretrained on UrbanSound8K, no multiscale	82.48	81.55	91.40
DenseNet-121 No pretrain, multiscale	82.22	82.11	88.54
DenseNet-121 Pretrained on UrbanSound8K, multiscale	85.52	84.35	95.32
ConvNet	85.4	N/A	N/A
Attention RNN (Ours)	**94.1**	**94.3**	**99.2**
Attention RNN (Ours, V2)	**94.5**	**93.9**	**99.4**

Table 2: Accuracy results on 12-commands from Google Speech Command Dataset V1. "res" model results from Warden (2018). ConvNet on raw WAV results from Jansson (2018). Depthwise Separable Convolutional Neural Network (DS-CNN) from Zhang et al. (2017).

Model	**Accuracy (%)**	**Trainable Parameters**
res15	95.8	238K
res26	95.2	438K
res8	94.1	110K
ConvNet on raw WAV	89.4	700K
DS-CNN	95.4	498K
Attention RNN (Ours)	**95.6**	202K
Attention RNN (Ours, V2)	**96.9**	202K

Furthermore, results obtained using the entire training dataset are compared with leaderboard results from Kaggle competition (Google Brain (2018)) using the held-out data released after the competition ended (Warden (2018)). Had this been an entry, the leaderboard score would have been 92.8%, which would rank #1 (assuming that the test set released is identical to the one used for scoring).

3.2. Attention Plots

One of the main advantages of deep learning models with attention is the possibility to explain the results and get intuition about what the model does. Like in the case of images (Xu et al. (2015)), plotting the attention weights allows visualization of what parts of the audio were most relevant for the classification.

Figures 2, 3 and 4 show attention weights along with waveform and mel-scale spectrogram. For better visualization, attention weights were plotted in a log-scale. Intuitively, one would expect that the network should put more emphasis on vowel transitions, since non-voiced regions (consonants) may be confused with background noise or even not present at all and regions where vowels are sustained do not carry extra information, i.e., a speaker might pronounce the /a/ sound in right for a long time before transitioning to the /i/ sound. Note that the model matches this intuition and attributes the highest weight to transitions.

3.3. Confusion Matrices

Confusion matrices are presented for the 20-cmd task and the 35-word recognition task on Google Speech Command dataset V1 and V2 (Figures 5, 6, 7 and 8 respectively). It is worth noting that words "three" and "tree" are often confused (which is expected given the similarity of the words), as well as "no" and "down". Proper identification of these words would require contextual information from other words in a sentence. This issue should not be relevant for an engineering application where the designers are allowed to pick possible commands: the choice of "three" and "tree" as possible commands would certainly be a poor choice due to how similar the words are (and also to the fact that non-native speakers sometimes are not even able to pronounce the /θ/ sound in **th**ree). The accuracy on those

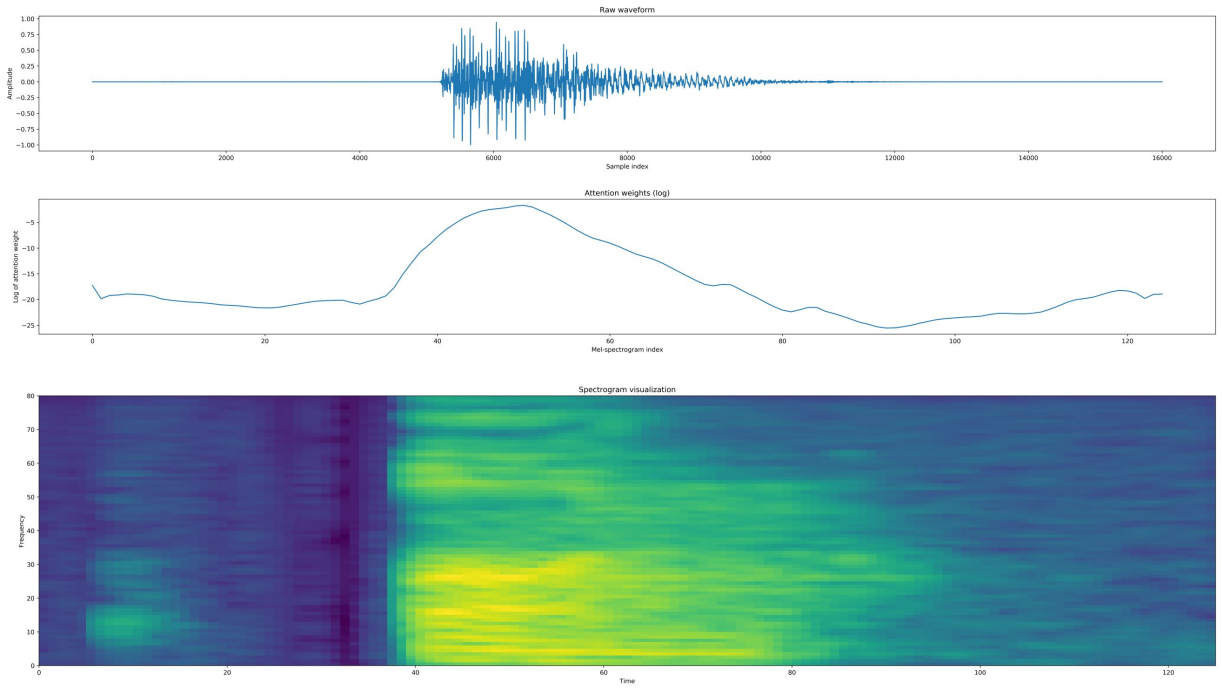

Figure 2: Waveform, mel-frequency spectrogram and attention weights for the word “on”

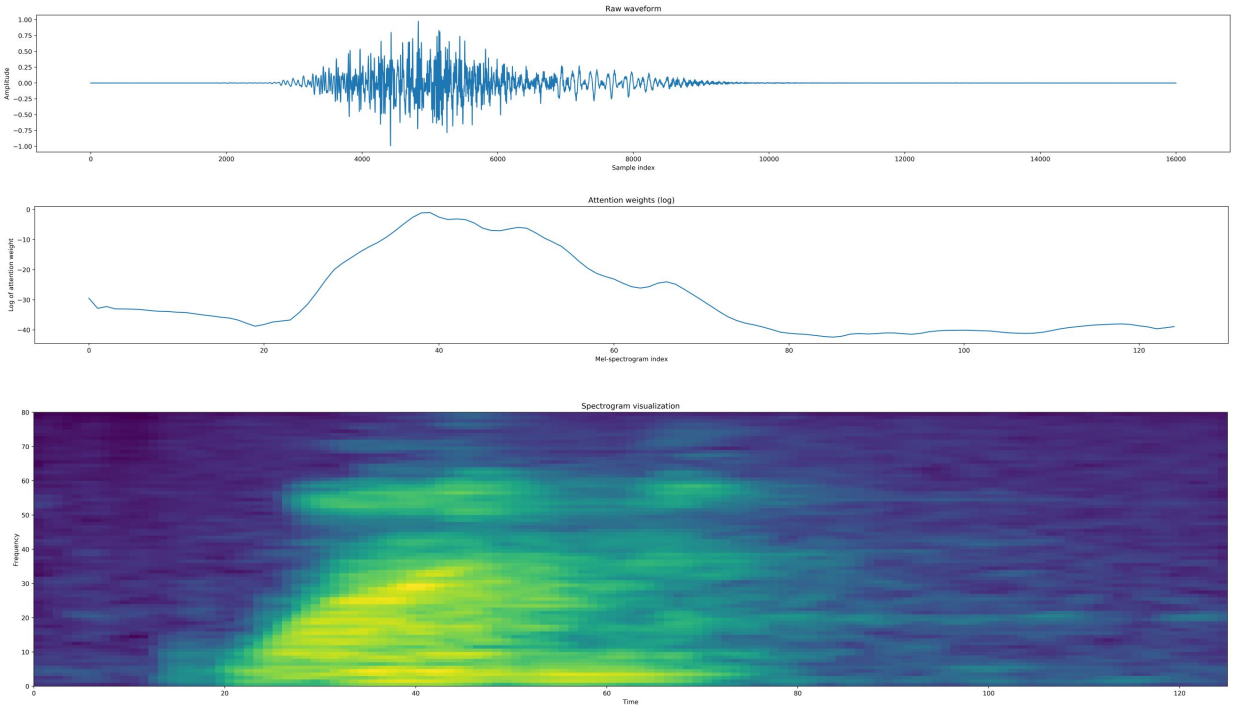

Figure 3: Waveform, mel-frequency spectrogram and attention weights for the word “one”

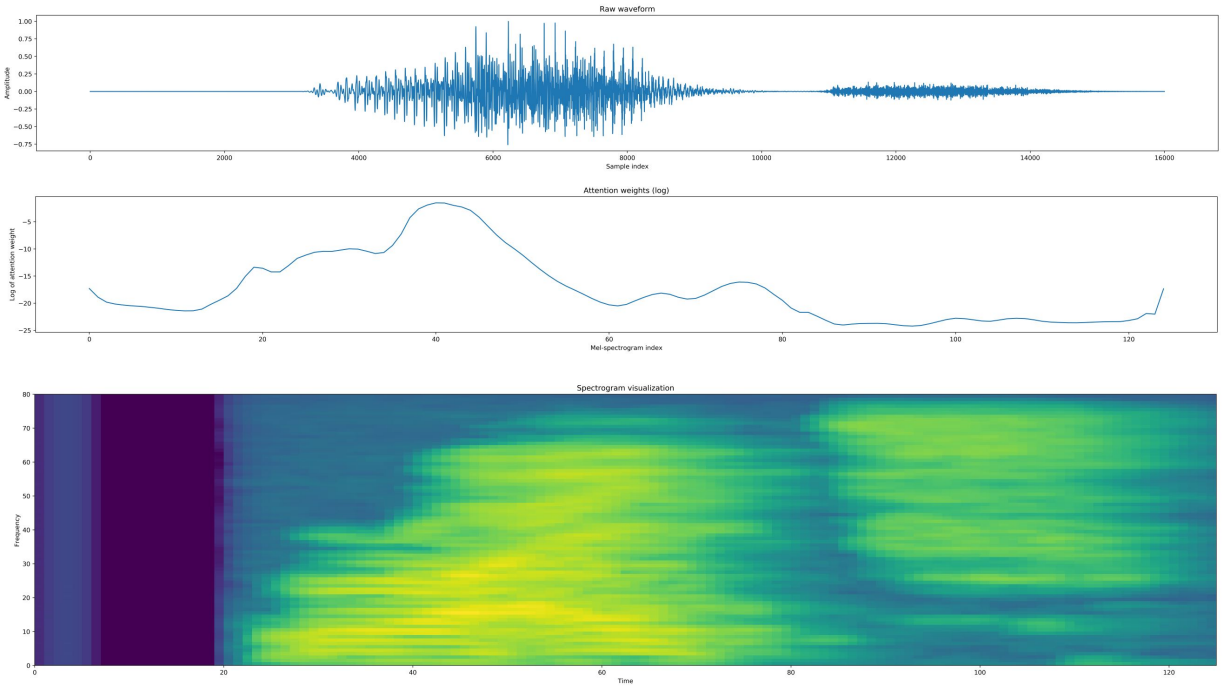

Figure 4: Waveform, mel-frequency spectrogram and attention weights for the word “right”

words is the lowest (approximately 90%). Note that in the 35-word task the accuracy on Speech Dataset V2 is $93.9 \pm 0.2\%$, slightly lower than V1 ($94.3 \pm 0.2\%$), because V2 has 35 words in the test set (as opposed to 30 in V1).

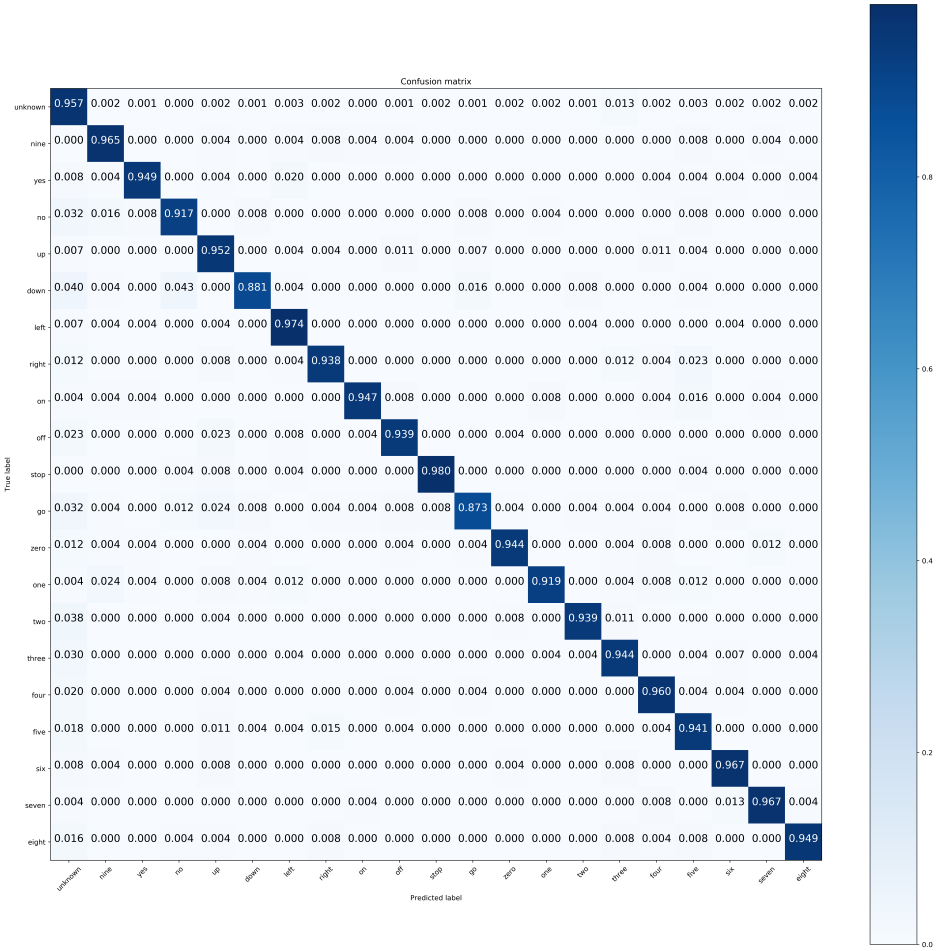

Figure 5: Confusion matrix for the 20 command task on Google Speech Dataset V1

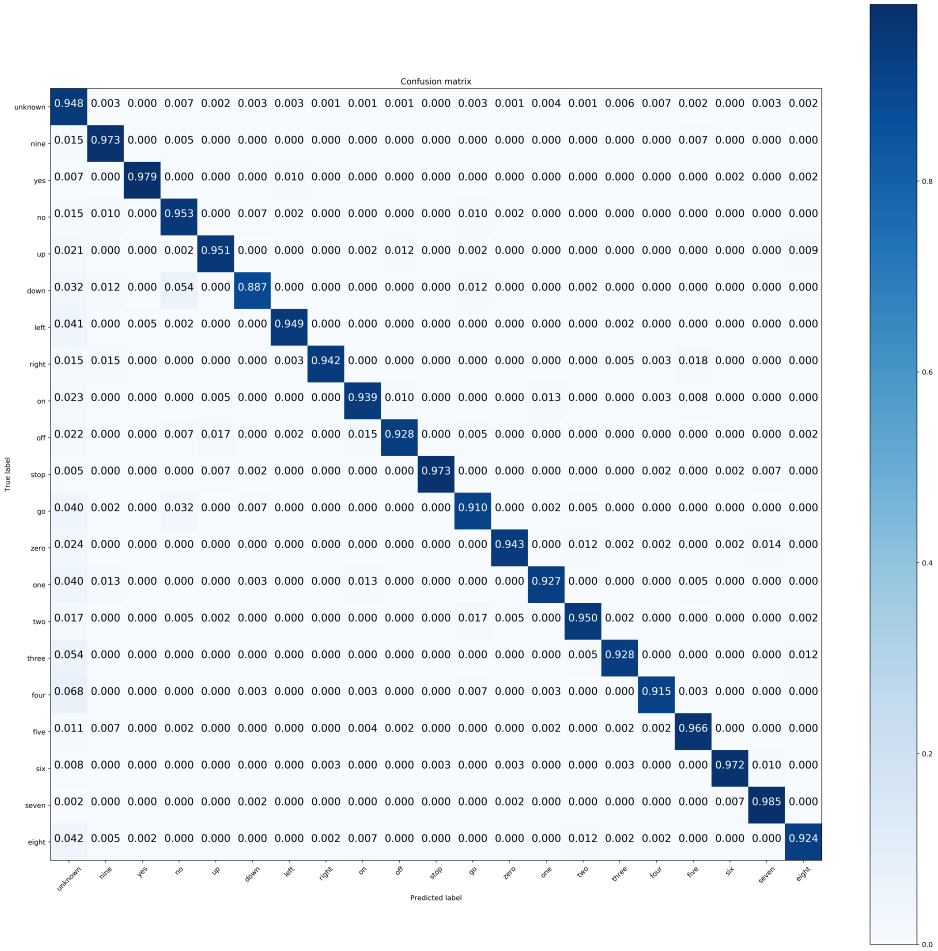

Figure 6: Confusion matrix for the 20 command task on Google Speech Dataset V2

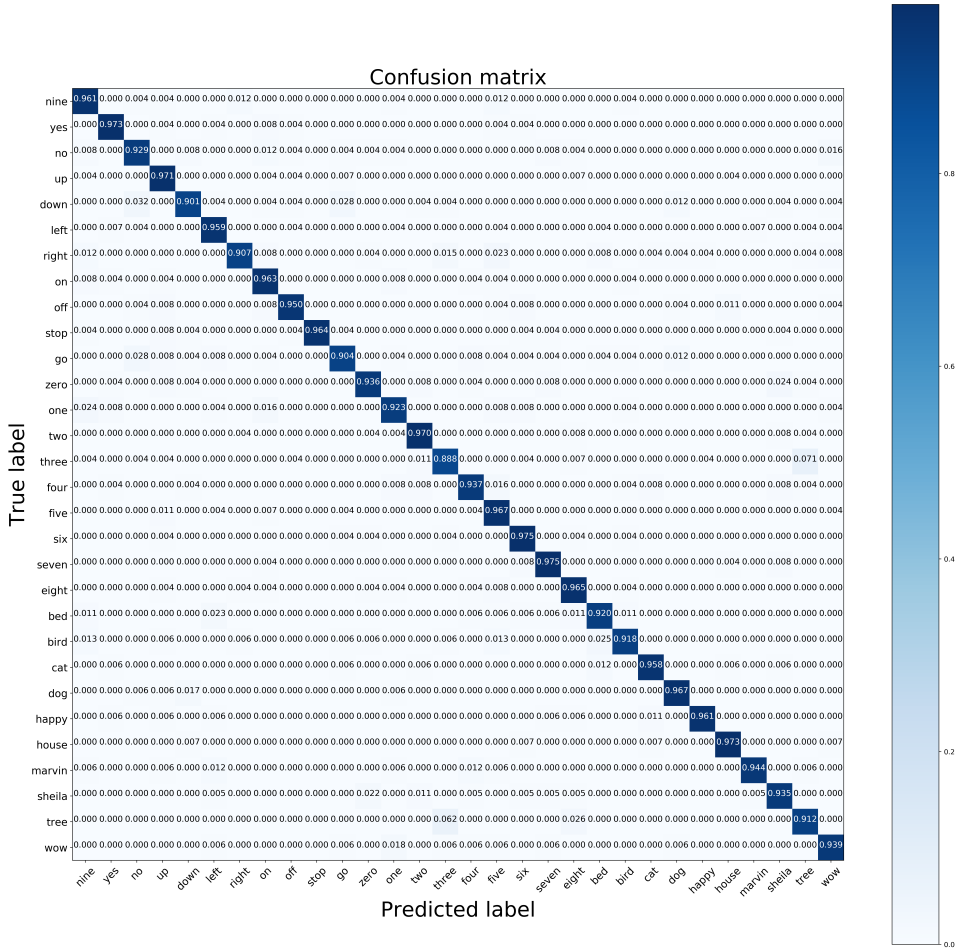

Figure 7: Confusion matrix for the 35 word task on Google Speech Dataset V1

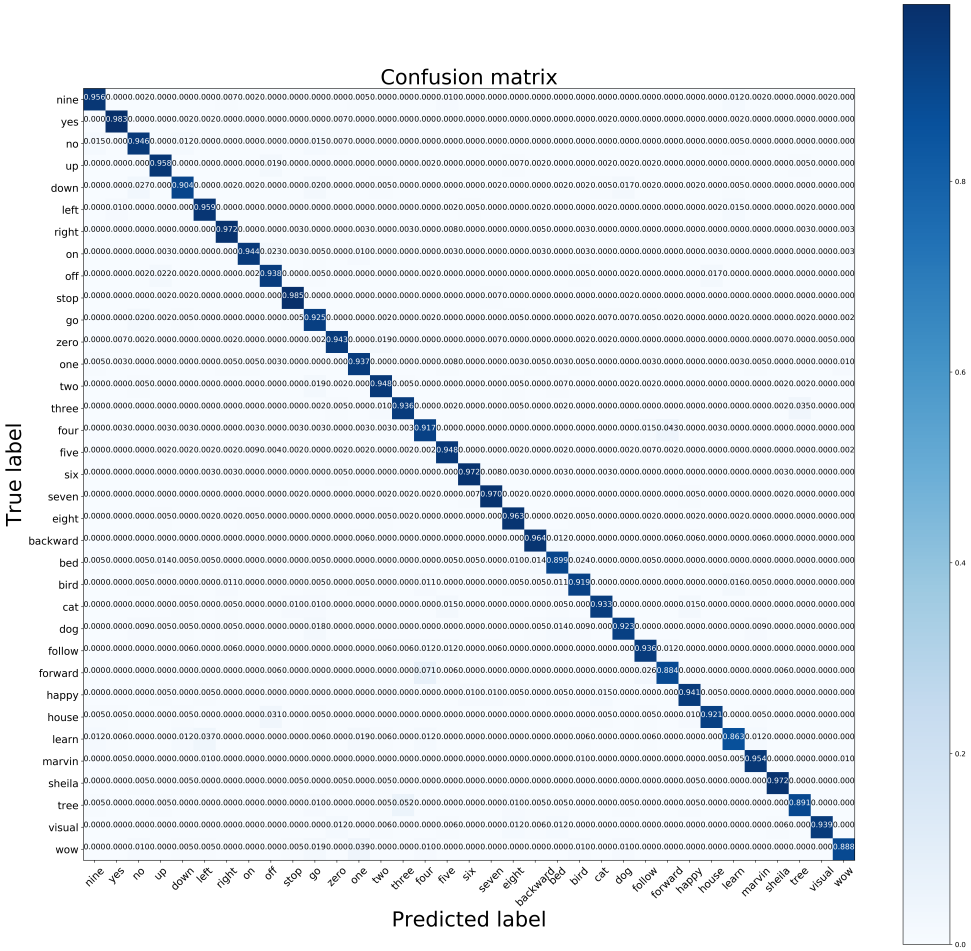

Figure 8: Confusion matrix for the 35 word task on Google Speech Dataset V2

4. Conclusion

Speech command recognition is present in a wide range of devices and utilized by many HCI interfaces. In many situations, it is desirable to obtain high accuracy, lightweight models that can run locally. In this work, we introduced a novel attention RNN architecture that achieves state-of-the-art performance on multiple KWS tasks while still keeping a small footprint in terms of trainable parameters. Source code is made available on github (to be posted) to enable further work.

The proposed architecture uses raw WAV files as inputs, computes mel-scale spectrogram using a non-trainable Keras layer, extracts short and long-term dependencies and uses an attention mechanism to pinpoint which region has the most useful information, that is then fed to a sequence of dense layers.

The Google Speech Commands datasets V1 and V2 Warden (2018) are used to demonstrate the effectiveness of the attention RNN approach. Attention RNN establishes a new state-of-the-art result on all tasks: 20-cmd, 12-cmd, 35-word and left-right. The accuracies are respectively 94.1%, 95.6%, 93.9% and 99.2% on the V1 dataset and 94.5%, 96.9%, 93.9% and 99.4% on the V2 dataset.

In engineering applications, being able to explain *what features* were used to select a particular category is a desirable element that is not available in previous neural network models. Our results demonstrate that the attention mechanism explains what parts of the audio are important for classification and also matches the intuition that regions of vowel transitions are relevant to recognize words. For completeness, confusion matrices are presented and show that the word pairs tree-three and no-down are difficult to identify without extra context.

Although data augmentation has been proven to be an important tool to increase model accuracy in visual tasks, the effectiveness of augmenting the audio samples with noise from other datasets was not explored. One possible direction of future work is to investigate the effect of incorporating multiple datasets and using pretrained models. It should also be possible to stack pairs of words for more complex commands and use a sequence-to-sequence

model or multiple attention layers. Further investigation will be conducted towards using the proposed attention RNN model for automatic language identification and detection of speech pathologies from audio.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X., 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.
URL `https://www.tensorflow.org/`

Amodei, D., Anubhai, R., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Chen, J., Chrzanowski, M., Coates, A., Diamos, G., Elsen, E., Engel, J., Fan, L., Fougner, C., Han, T., Hannun, A. Y., Jun, B., LeGresley, P., Lin, L., Narang, S., Ng, A. Y., Ozair, S., Prenger, R., Raiman, J., Satheesh, S., Seetapun, D., Sengupta, S., Wang, Y., Wang, Z., Wang, C., Xiao, B., Yogatama, D., Zhan, J., Zhu, Z., 2015. Deep speech 2: End-to-end speech recognition in english and mandarin. CoRR abs/1512.02595.
URL `http://arxiv.org/abs/1512.02595`

Arik, S. Ö., Kliegl, M., Child, R., Hestness, J., Gibiansky, A., Fougner, C., Prenger, R., Coates, A., 2017. Convolutional recurrent neural networks for small-footprint keyword spotting. CoRR abs/1703.05390.
URL `http://arxiv.org/abs/1703.05390`

Bahdanau, D., Cho, K., Bengio, Y., 2014. Neural machine translation by jointly learning to align and translate. CoRR abs/1409.0473.
URL `http://arxiv.org/abs/1409.0473`

Chen, Z., Qian, Y., Yu, K., 2018. Sequence discriminative training for deep learning based acoustic keyword spotting. Speech Communication 102, 100 – 111.
URL `http://www.sciencedirect.com/science/article/pii/S0167639317303631`

Chiu, C., Sainath, T. N., Wu, Y., Prabhavalkar, R., Nguyen, P., Chen, Z., Kannan, A., Weiss, R. J., Rao, K., Gonina, K., Jaitly, N., Li, B., Chorowski, J., Bacchiani, M., 2017. State-of-the-art speech recognition with sequence-to-sequence models. CoRR abs/1712.01769.
URL `http://arxiv.org/abs/1712.01769`

Choi, K., Joo, D., Kim, J., 2017. Kapre: On-gpu audio preprocessing layers for a quick implementation of deep neural network models with keras. In: Machine Learning for Music Discovery Workshop at 34th International Conference on Machine Learning. ICML.

Chollet, F., et al., 2015. Keras.
URL https://keras.io

Google Brain, 2018. Tensorflow speech recognition challenge.
URL https://www.kaggle.com/c/tensorflow-speech-recognition-challenge/

Hochreiter, S., Schmidhuber, J., Nov. 1997. Long short-term memory. Neural Comput. 9 (8), 1735–1780.
URL http://dx.doi.org/10.1162/neco.1997.9.8.1735

Jansson, P., 2018. Single-word speech recognition with convolutional neural networks on raw waveforms.
URL https://www.theseus.fi/bitstream/handle/10024/144982/Jansson_Patrick.pdf?sequence=1

Kingma, D. P., Ba, J., 2014. Adam: A method for stochastic optimization. CoRR abs/1412.6980.
URL http://arxiv.org/abs/1412.6980

Lei, D., Chen, X., Zhao, J., 2018. Opening the black box of deep learning. CoRR abs/1805.08355.
URL http://arxiv.org/abs/1805.08355

McMahan, B., Rao, D., 2017. Listening to the world improves speech command recognition. CoRR abs/1710.08377.
URL http://arxiv.org/abs/1710.08377

Sainath, T. N., Parada, C., 2015. Convolutional neural networks for small-footprint keyword spotting. In: INTERSPEECH.

Sangeetha, J., Jothilakshmi, S., 2014. A novel spoken keyword spotting system using support vector machine. Engineering Applications of Artificial Intelligence 36, 287 – 293.
URL http://www.sciencedirect.com/science/article/pii/S0952197614001821

Sundberg, J., L, F. M., Gill, B. P., 2013. Formant tuning strategies in professional male opera singers. Journal of Voice 27 (3), 278 – 288.
URL http://www.sciencedirect.com/science/article/pii/S0892199712002093

Sundberg, J., Thaln, M., 2015. Respiratory and acoustical differences between belt and neutral style of singing. Journal of Voice 29 (4), 418 – 425.
URL http://www.sciencedirect.com/science/article/pii/S0892199714002045

Tabibian, S., Akbari, A., Nasersharif, B., 2013. Keyword spotting using an evolutionary-based classifier and discriminative features. Engineering Applications of Artificial Intelligence 26 (7), 1660 – 1670.
URL http://www.sciencedirect.com/science/article/pii/S0952197613000511

Tang, R., Lin, J., 2017. Deep residual learning for small-footprint keyword spotting. CoRR abs/1710.10361.
URL http://arxiv.org/abs/1710.10361

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., Polosukhin, I., 2017. Attention is all you need. CoRR abs/1706.03762.

URL http://arxiv.org/abs/1706.03762

Wang, Y., Skerry-Ryan, R. J., Stanton, D., Wu, Y., Weiss, R. J., Jaitly, N., Yang, Z., Xiao, Y., Chen, Z., Bengio, S., Le, Q. V., Agiomyrgiannakis, Y., Clark, R., Saurous, R. A., 2017. Tacotron: A fully end-to-end text-to-speech synthesis model. CoRR abs/1703.10135.

URL http://arxiv.org/abs/1703.10135

Warden, P., 2018. Speech commands: A dataset for limited-vocabulary speech recognition. CoRR abs/1804.03209.

URL http://arxiv.org/abs/1804.03209

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A. C., Salakhutdinov, R., Zemel, R. S., Bengio, Y., 2015. Show, attend and tell: Neural image caption generation with visual attention. CoRR abs/1502.03044.

URL http://arxiv.org/abs/1502.03044

Zhang, Y., Suda, N., Lai, L., Chandra, V., 2017. Hello edge: Keyword spotting on microcontrollers. CoRR abs/1711.07128.

URL http://arxiv.org/abs/1711.07128