INTERSPEECH 2020
October 25–29, 2020, Shanghai, China

Squeeze for Sneeze: Compact Neural Networks for Cold and Flu Recognition

Merlin Albes[1], *Zhao Ren*[1], *Björn W. Schuller* [1,2], *Nicholas Cummins,*[1,3]

[1]Chair of Embedded Intelligence for Health Care & Wellbeing, University of Augsburg, Germany
[2]GLAM – Group on Language, Audio, & Music, Imperial College London, UK
[3] Department of Biostatistics and Health Informatics, IoPPN, King's College London, London, UK

nicholas.cummins@ieee.org

Abstract

In digital health applications, speech offers advantages over other physiological signals, in that it can be easily collected, transmitted, and stored using mobile and Internet of Things (IoT) technologies. However, to take full advantage of this positioning, speech-based machine learning models need to be deployed on devices that can have considerable memory and power constraints. These constraints are particularly apparent when attempting to deploy deep learning models, as they require substantial amounts of memory and data movement operations. Herein, we test the suitability of pruning and quantisation as two methods to compress the overall size of neural networks trained for a health-driven speech classification task. Key results presented on the Upper Respiratory Tract Infection Corpus indicate that pruning, then quantising a network can reduce the number of operational weights by almost 90 %. They also demonstrate the overall size of the network can be reduced by almost 95 %, as measured in MB, without affecting overall recognition performance.

Index Terms: Compact Neural Networks, Pruning, Quantisation, Computational Paralinguistics, Cold and Flu Recognition

1. Introduction

The recent coronavirus disease 2019 (COVID-19) pandemic has highlighted the need from remote digital health solutions [1]. One major challenge in creating such solutions, especially in digital health systems utilising machine learning, is the development of compact, resource-efficient inference models. This challenge is also of particular importance when considering the widespread use of Internet of Things (IoT) devices in digital health, as IoT devices are generally associated with low resource and low power environments [2]. Moreover, advances in low-complexity inference models that are deployable in IoT devices can help alleviate security and privacy concerns surrounding the use of these devices [3].

As in most areas of intelligent signal sensing, deep learning is emerging as the dominant modelling technique in digital health settings [4, 5]. Deep learning models have connection numbers measuring in the millions, require hundreds of megabytes of memory to store as well as generating substantial data movement operations to support their computation [6]. There is, therefore, a fundamental mismatch between the computational requirements needed to support deep learning and those available on IoT devices. Deep learning models, therefore, have to be run on servers and powerful workstations with the required resources. In turn, this demands the transmission of data from the collecting devices to the server, creating privacy concerns [3] and an over-reliance on network availability [7, 6, 8, 9].

A growing research direction is the development of approaches to optimise a large network until it is executable on a low resource device [7]. Many of these approaches focus on reducing the *memory footprint* (how much memory is required to store and run a network) and the *computational complexity* (the number of required calculations and their precision) of a network while preserving its accuracy. Such approaches can improve system performance while reducing the size of the model by up to 80 % [7]. To date, such techniques have shown promise across a range of different learning tasks, but their potential has not yet been realised for digital health applications.

The work presented in this paper focuses on creating low-resource neural networks using two well-established techniques: *pruning* and *quantisation*. Pruning is the process of removing unused connections and, eventually, neurons from a network. It has been widely used in a range of speech processing tasks, for example in speech recognition [10, 11, 12]; denoising, and enhancement [13, 14, 15]; and, emotion recognition [16]. The efficacy of quantisation, a lowering in the resolution of a network's weights and biases, has been established in similar applications [17, 18, 19]. Moreover, pruning and quantisation have often been used together [20, 13]. However, to the best of the authors' knowledge, these techniques have not been investigated in speech-health applications [4].

In this work, we test the capability of pruning and quantisation to reduce the model size whilst retaining the overall accuracy of a network trained for the task of cold and flu recognition from speech. For this task, we utilise the Upper Respiratory Tract Infection Corpus (URTIC) dataset as featured in the INTERSPEECH 2017 Computational Paralinguistics Challenge (ComParE) [21]. A range of different approaches have already been undertaken on this data, from conventional openSMILE [22] based systems [21, 23], to more contemporary deep learning systems [24]. As the aim of the presented work is to explore the advantages of network pruning and optimisation, we opted to train standard multi-layer dense neural networks, the suitability of which have been demonstrated for this data [25].

The rest of this paper is laid out as follows. The two compression methodologies are outlined in Section 2. Then, the key experimental settings are given in Section 3, and subsequent results and discussion given in Section 4. Finally, we conclude the paper and propose future work directions in Section 5.

2. Network Compression

Within digital health settings, low-resource models can provide fully-fledged low-complexity machine learning models capable of fast and accurate inferences on IoT devices. A multitude of methods exist for achieving this aim; this section introduces the two methods used in this work: *pruning* (Section 2.1) and *quantisation* (Section 2.2).

http://dx.doi.org/10.21437/Interspeech.2020-2531

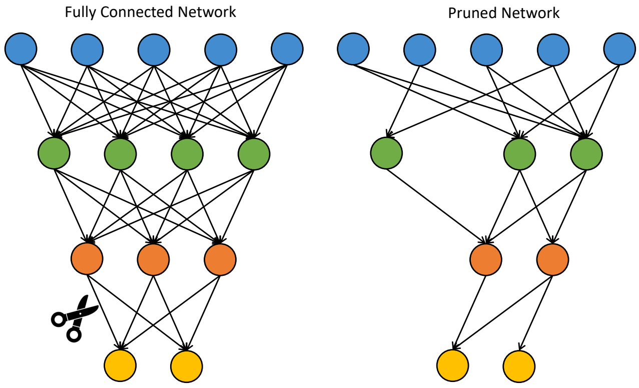


Figure 1: *Pruning is the process of creating a sparsely-connected model by identifying and removing redundant weights and neurons from a full model*

2.1. Network Pruning

Pruning removes unused connections and, eventually, neurons from a network (Figure 1). It involves omitting the weights that do not contribute to the actual output calculation and therefore, also have no contribution to the backpropagation. In principle, pruning does not affect the nodes needed to make correct predictions. Therefore, networks should maintain their accuracy as they are pruned. Moreover, analogous to dropout, the action of pruning often increases network generalisability [6, 8].

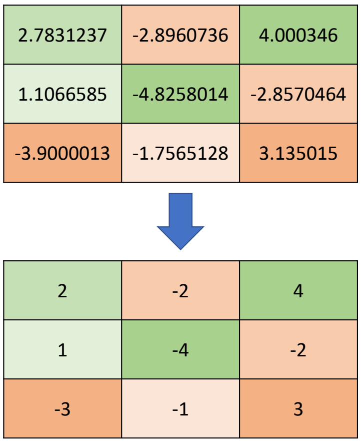
Pruning is achieved by first training a 'full' network to identify the importance of all connections. Then, connections falling below a certain, manually set, threshold are removed or fixed to be zero. Neurons that consequently have no incoming or outgoing connections can then be deleted. Using this process, Srinivas et al. were able to prune a LeNet-like architecture, trained for MNIST by up to 85 %, without degrading its performance [8]. They performed a similar experiment on AlexNet, demonstrating they could compress the size of the network by 35 % without affecting its performance [8].

To achieve stronger results, pruning can be run multiple times on a single network. Han et al. suggest a method for pruning redundant weights containing three steps [6]. First, a 'full' network is trained in order to identify important connections. Next, the network is pruned, and thirdly, the resulting sparse network is retrained to fine-tune the remaining weights. Results demonstrate that this process could reduce the parameter count of AlexNet by a factor of 9 and VGG-16 by 13 without any loss in accuracy [6].

2.2. Quantisation

Quantisation can be viewed as a lowering in resolution of the data type used to represent a network's weights and biases (Figure 2). Most contemporary deep learning approaches use float-64 or float-32 data types for storing their values, however, many studies have demonstrated that this level of precision is not necessary. In many cases, lowering the number of bits, for example by conversion to a float-16, or even int-8 representation, can yield performances matching the original network [9]. Some studies have even demonstrated the suitability of using single-bit representations [26].

Quantisation comes with other positive side effects. Firstly, it can considerably reduce the network's training time and overall inference delay, as for lower bit representation, the required mathematical operations are, in general, less time-consuming. Secondly, the lower resolution can help to improve generalisation by helping to prevent over-tuning.



2.7831237	-2.8960736	4.000346
1.1066585	-4.8258014	-2.8570464
-3.9000013	-1.7565128	3.135015

2	-2	4
1	-4	-2
-3	-1	3

Figure 2: *Example of the conversion of float-32 representations (top) into int-8 representations (bottom)*

Networks can be quantised by restricting all weight and bias values to low-bit-width integers during training and inference [9]. However, studies have suggested it can be more advantageous to first train the network at a higher resolution, like a float-32 data type, then quantise these values once it is ready for inference and deployed on the device [27]. We used this method in this work.

2.3. Concurrent Network Pruning and Quantisation

It is possible to use both compression methods together. This process is typically performed by first pruning, then quantising the remaining weights and biases [28]. Wu et al. used a combination of parameter pruning and a k-means based quantisation method to compact a speaker enhancement network to 10 % of its original size without overly affecting its performance [13]. Similarly, Shangguan et al. demonstrated that these techniques can be used to reduce the size of a recurrent neural network (RNN) – long-short term memory (LSTM) network trained for speech recognition to 57 % of its original size, also without overly affecting its performance [20].

3. Experimental Setting

To test the efficacy of network compression in a digital health scenario, we conduct our experiments on a corpus collected for assessing the affect of cold and flu on speech (Section 3.1). The model set-up (Section 3.2) and key setting regarding the compression techniques (Section 3.3), as well as the evaluation metrics (Section 3.4) are also given in this section.

3.1. Upper Respiratory Tract Infection Corpus

We used the *Upper Respiratory Tract Infection Corpus* (URTIC) in our experiments. The dataset was presented as part of the INTERSPEECH 2017 *Computational Paralinguistics Challenge* (ComParE) [21] and contains 28 652 audio recordings of 630 different subjects (382 male, 248 female). All recordings were made in quiet rooms with a microphone/headset/hardware setup. The mean age of the participants was 29.5 years, with a standard deviation of 12.1 years and a range of 12 to 84 years.

Participants undertook the German version of the *Wisconsin Upper Respiratory Symptom Survey* (WURSS-24) [29]. The

Table 1: *Partitioning of the Upper Respiratory Tract Infection Corpus as used for the Cold Sub-Challenge of the INTERSPEECH 2017 Computational Paralinguistics Challenge (top), and the rebalanced training and development partitions used in this work (below)*

#	Train	Devel	Test	$\sum$
Upper Respiratory Tract Infection Corpus (URTIC)				
C	970	1 011	895	2 876
NC	8 535	8 585	8 656	25 776
$\sum$	9 505	9 596	9 551	28 652
Balanced Train and Development Partitions				
C	1 869	432	–	2 301
NC	1 869	432	–	2 301
$\sum$	3 738	864	–	4 602

questionnaire is an evaluative illness-specific quality of life instrument that assesses the symptoms of the common cold. The audio recordings were split into two classes: chunks with a corresponding WURSS-24 equal to zero were assigned to *Non-Cold* (NC), while chunks with a corresponding WURSS-24 greater than zero were assigned to *Cold* (C).

We used the train/development/test partitioning of this dataset provided by the challenge organisers (Table 1). However, during initial tests, we were unable to find a suitable network architecture to achieve reasonable performance compared to the challenge baselines. Therefore, we redistributed the speech chunks in the training and development sets to balance the data distribution between the two classes (Table 1)[1]. To make our results comparable with those in the literature, we report our primary findings on the test-set only.

3.2. Classification Models

For the 2-class *Non-Cold* (NC) versus *Cold* (C) task, we created three different dense neural networks using Keras. All three models consist of a batch normalisation layer, multiple dense hidden layers with sigmoid activation and dropout, and a softmax layer to perform the classification. The three models enable the effectiveness of the compression techniques on different-sized networks to be observed. We ran experiments on a network with two hidden layers (Model A), five hidden layers (Model B), and 2×2 hidden layers (Model C) (Figure 3). These models have 39 128 332, 47 905 713, and 65 201 140 trainable parameters respectively.

3.3. Key Settings

We used Python 3.7.1 combined with TensorFlow 1.14.0 and the TensorFlow Model Optimisation Toolkit (version 0.1.3) on Windows 10 Pro. All experiments were performed on an AMD Ryzen 7 3700X (8C/16T) at stock speeds with 16 GB 3 200 MHz DDR4 RAM and a Samsung 960 EVO SSD.

We used the 6 373 dimensional ComParE feature set as supplied by the challenge organisers [21]. Each network was first trained normally for 40 epochs – in this case, one epoch means the network was exposed to all training examples once – with a batch size of 32 and an initial learning rate of 1^{-4}.

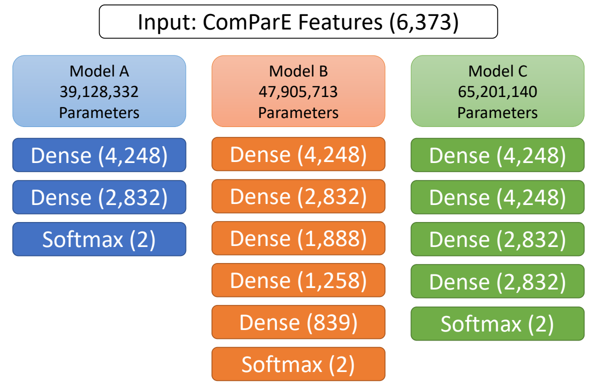


Figure 3: *An illustrative overview of the three different fully-connected architectures tested in our compression experiments. We tested networks with two hidden layers (Model A), five hidden layers (Model B), and 2×2 hidden layers (Model C)*

During training, the learning rate is updated with:

$$lr = \frac{lr_{initial}}{num\_epochs}. \qquad (1)$$

The model is compiled to run with the Adam optimiser and a binary cross entropy loss function.

In our first experiment, after initial training, the model was saved and then pruned for five epochs using the model optimisation tool provided by TensorFlow. Instead of deleting nodes, the toolkit fixes a percentage of all weights per layer to the value 0.0, depending on the current sparsity. This value is calculated using a polynomial decay function, meaning it initially grows fast, then, slowly plateaus to the target sparsity. We used the default TensorFlow settings, with the initial sparsity set to 0.5 and the target (final) sparsity set to 0.9 for all experiments. In a second experiment, the original network was quantised by converting it to a TensorFlow Lite FlatBuffer file (.flite), which sets the datatype of all of the model's weights to int-8. To further evaluate how well both approaches work together, a third experiment, which quantises the already pruned model, was also conducted.

3.4. Evaluation Metrics

As per the challenges [21], we report all classification results using the Unweighted Average Recall (UAR). We use McNemar's test [30] to check for significant changes in network performance due to our compression techniques. All models were compressed into .zip-files and the size in MB returned to give a quantitative metric of the effectiveness of our compression techniques. This measurement is needed, as the pruning technique used in our experiments (Section 3.3) does not delete weights, but instead fixes them to the value 0.0. We also report the number of weights set to zero in absolute values and as a percentage of the number of weights in the original networks. We also tested inference time, however, we observed very little change in this parameter, so do not report these results.

4. Results and Discussion

As mentioned in Section 3.1, we began initial experimentation using the original URTIC partitioning [21]. However, due to the large imbalance between the two classes, we were unable to identify suitable network architectures with these data splits. We, therefore, re-balanced the data and were able to identify

[1] balanced partitioning available on request

Table 2: *Comparison of UARs for uncompressed and compression networks on the rebalanced URTIC development set*

Model	*A*	*B*	*C*
Original	0.93	0.92	0.92
Pruned	0.93	0.91	0.89
Quantised	0.89	0.92	0.92
Quantised & Pruned	0.89	0.92	0.89

suitable models for compression (Section 3.2). The results of this work are given for completion (Table 2), but it should be noted they are incomparable with other works on this corpus. The key observation from this initial work is that compression has minimal effect on overall system performance.

To ensure comparability with other works on the URTIC corpus, a complete analysis of our results gained on the original test set is provided (Table 3). The first key observation is that all the compression results had minimal impact on system performance as measured by UAR; all systems scored in the range [0.65–0.68]. Indeed, the results of the McNemar's tests (not shown) revealed no significant differences between the results obtained on the original and compressed networks.

Pruning reduced the number of meaningful weights in the network by almost 90 %. Given the target sparsity was set to 0.9 (Section 3.3), the model optimisation worked as intended. The number of weights remaining for models A, B, and C was 3 978 664, 4 870 754, and 6 624 302 respectively. Pruning, therefore, results in a reduction in network size, as measured in MB (Section 3.4), by almost 80 %.

A similar effect was observed after quantising the networks, converting the weight and biases from float-32 into int-8 representations. Quantisation did not affect the total number of network parameters; these remained the same. It did, however, reduce the overall size of each network by approximately 75 %, without affecting accuracy. This size reduction is expected, considering that an int-8 representation only takes up 25 % of the space of a float-32 representation.

Finally, the combination of pruning, then quantising resulted in compression rates of approximately 95 %. Despite this considerable reduction in network size, the accuracy of the system is unaffected. The achieved compression rate can also be logically explained when observing that pruning initially results in a compression of approximately 79 %, which is then brought down to 95 % by the subsequent additional 75 % compression action of quantisation. The consistency in UAR values indicates that the two techniques do not interfere with each other, further justifying their combined use.

Our results match that in [28], who also observed the combination of pruning and quantisation as a highly efficient method to considerably reduce the overall size of a network without a loss of accuracy. A similar finding was observed in [19]. The authors binarised a convolutional recurrent neural network trained for speech-based emotion recognition. This action reduced the size of their network by 96 %, also without overly affecting its accuracy.

The UARs achieved by the compressed networks, despite being trained on a reduced amount of data, are not considerably reduced when compared to state-of-the-art systems in the literature. The official ComParE baseline was a UAR of 0.71 [21] while the winners of the challenge achieved a test set UAR of 0.72 [31]. It is worth noting that these two papers did not employ any compression techniques; the baseline is a fusion of three different systems, including an end-to-end network, while the winning entry utilised a three-layer feedforward network.

Table 3: *Comparison of UARs and other network compression metrics for uncompressed and compression networks on the URTIC test set*

Model	*A*	*B*	*C*
Hidden layers	*2*	*5*	*2×2*
Uncompressed Network			
UAR on test set	0.68	0.68	0.65
Trainable parameters	39×10^6	48×10^6	65×10^6
Size [in MB]	137.4	168.6	228.8
Pruned			
UAR on test set	0.67	0.67	0.66
Weights set to 0	35×10^6	43×10^6	59×10^6
Reduction in weights	89.8 %	89.8 %	89.8 %
Size [in MB]	28.8	35.1	47.5
Reduction in size	79.0 %	79.2 %	79.2 %
Quantised			
UAR on test set	0.68	0.68	0.65
Size [in MB]	32.1	40.0	55.1
Reduction in size	76.6 %	76.3 %	75.9 %
Pruned & quantised			
UAR on test set	0.67	0.67	0.66
Size [in MB]	7.0	8.8	11.8
Reduction in size	94.9 %	94.8 %	94.8 %

5. Conclusions

With an ever increasing need for remote digital health systems, the need for compact deep learning models capable of running remotely in embedded devices has also increased. The work presented in this paper demonstrated, on the Upper Respiratory Tract Infection Corpus, that the combination of pruning and quantisation can reduce network size by up to 95 % without overly affecting accuracy. We observed this effect on three differently sized networks, the largest of which, a 2 × 2 hidden layer dense network with 65 213 886 parameters. We were able to reduce this network from an initial size of 228.8 MB to only 11.8 MB with a final parameter count of 6 637 048. Such a reduction would make this network considerably more feasible for use in embedded devices. Future work will include the deployment of such models in embedded devices and will test how network compression affects other related performance metrics, such as execution time and energy consumption.

6. Acknowledgements

This work was supported by the Horizon H2020 Marie Skłodowska-Curie Actions Initial Training Network European Training Network (MSCA-ITN-ETN) project under grant agreement No. 766287 (TAPAS). This project also received funding from the Innovative Medicines Initiative 2 Joint Undertaking under grant agreement No. 115902, which receives support from the European Union's Horizon 2020 research and innovation program and EFPIA. The authors wish to thank Anton Winschel from Xitaso for his help and advice. Thanks to Dr Judith Dineley for copy-editing the text.

7. References

[1] P. Webster, "Virtual health care in the era of COVID-19," *The Lancet*, vol. 395, no. 10231, pp. 1180–1181, 2020.

[2] X. Xu, Y. Ding, S. X. Hu, M. Niemier, J. Cong, Y. Hu, and Y. Shi, "Scaling for edge inference of deep neural networks," *Nature Electronics*, vol. 1, no. 4, pp. 216–222, 2018.

[3] Y. Yang, L. Wu, G. Yin, L. Li, and H. Zhao, "A survey on security and privacy issues in internet-of-things," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1250–125, 2017.

[4] N. Cummins, A. Baird, and B. Schuller, "The increasing impact of deep learning on speech analysis for health: Challenges and Opportunities," *Methods*, vol. 151, pp. 41–54, 2018.

[5] A. Esteva, A. Robicquet, B. Ramsundar, V. Kuleshov, M. DePristo, K. Chou, C. Cui, G. Corrado, S. Thrun, and J. Dean, "A guide to deep learning in healthcare," *Nature Medicine*, vol. 25, no. 1, pp. 24–29, 2019.

[6] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 1135–1143.

[7] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "A Survey of Model Compression and Acceleration for Deep Neural Networks," arXiv, 2017. [Online]. Available: https://arxiv.org/abs/1710.09282

[8] S. Srinivas and R. V. Babu, "Data-free parameter pruning for Deep Neural Networks," arXiv, 2015. [Online]. Available: http://arxiv.org/abs/1507.06149

[9] S. Wu, G. Li, F. Chen, and L. Shi, "Training and Inference with Integers in Deep Neural Networks," arXiv, 2018. [Online]. Available: http://arxiv.org/abs/1802.04680

[10] P. Dong, S. Wang, W. Niu, C. Zhang, S. Lin, Z. Li, Y. Gong, B. Ren, X. Lin, Y. Wang *et al.*, "RTMobile: Beyond Real-Time Mobile Acceleration of RNNs for Speech Recognition," arXiv, 2020. [Online]. Available: https://arxiv.org/abs/2002.11474

[11] C. Liu, Z. Zhang, and D. Wang, "Pruning Deep Neural Networks by Optimal Brain Damage," in *Proceedings of INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association*. Singapore: ISCA, 2014, pp. 1092–1095.

[12] R. Takeda, K. Nakadai, and K. Komatani, "Node Pruning Based on Entropy of Weights and Node Activity for Small-Footprint Acoustic Model Based on Deep Neural Networks," in *Proceedings of INTERSPEECH 2017, 18th Annual Conference of the International Speech Communication Association*. Stockholm, Sweden: ISCA, 2017, pp. 1636–1640.

[13] J. Wu, C. Yu, S. Fu, C. Liu, S. Chien, and Y. Tsao, "Increasing Compactness of Deep Learning Based Speech Enhancement Models With Parameter Pruning and Quantization Techniques," *IEEE Signal Processing Letters*, vol. 26, no. 12, pp. 1887–1891, 2019.

[14] L. Xu, C. Choy, and Y. Li, "Deep sparse rectifier neural networks for speech denoising," in *2016 IEEE International Workshop on Acoustic Signal Enhancement (IWAENC)*. Xi'an, China: IEEE, 2016, 5 pages.

[15] F. Ye, Y. Tsao, and F. Chen, "Subjective Feedback-based Neural Network Pruning for Speech Enhancement," in *Proceedings of the 2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. Lanzhou, China: IEEE, 2019, pp. 673–677.

[16] M. E. Sánchez-Gutiérrez and P. P. González-Pérez, "Discriminative neural network pruning in a multiclass environment: A case study in spoken emotion recognition," *Speech Communication*, vol. 120, pp. 20–30, 2020.

[17] X. Chen, G. Liu, J. Shi, J. Xu, and B. Xu, "Distilled Binary Neural Network for Monaural Speech Separation," in *Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2018, 8 pages.

[18] Y. Hsu, Y. Lin, S. Fu, Y. Tsao, and T. Kuo, "A Study on Speech Enhancement Using Exponent-Only Floating Point Quantized Neural Network (EOFP-QNN)," in *Proceedings of the 2018 IEEE Spoken Language Technology Workshop (SLT)*. Athens, Greece: IEEE, 2018, pp. 566–573.

[19] H. Zhao, Y. Xiao, J. Han, and Z. Zhang, "Compact Convolutional Recurrent Neural Networks via Binarization for Speech Emotion Recognition," in *Proceedings of the 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Brighton, United Kingdom: IEEE, 2019, pp. 6690–6694.

[20] Y. Shangguan, J. Li, L. Qiao, R. Alvarez, and I. McGraw, "Optimizing Speech Recognition For The Edge," arXiv, 2019. [Online]. Available: https://arxiv.org/abs/1909.12408

[21] B. Schuller, S. Steidl, A. Batliner *et al.*, "The INTERSPEECH 2017 Computational Paralinguistics Challenge: Addressee, Cold & Snoring," in *Proceedings of INTERSPEECH 2017, 18th Annual Conference of the International Speech Communication Association*. Stockholm, Sweden: ISCA, 2017, pp. 3442–3446.

[22] F. Eyben, F. Weninger, F. Groß, and B. Schuller, "Recent Developments in openSMILE, the Munich Open-Source Multimedia Feature Extractor," in *Proceedings of the 21st ACM International Conference on Multimedia (MM)*. Barcelona, Spain: ACM, 2013, pp. 835–838.

[23] N. Cummins, M. Schmitt, S. Amiriparian, J. Krajewski, and B. Schuller, "You sound ill, take the day off: Classification of speech affected by upper respiratory tract infection," in *Proceedings of the 39th Annual International Conference of the IEEE Engineering in Medicine & Biology Society, EMBC 2017*. Jeju Island, South Korea: IEEE, July 2017, pp. 3806–809.

[24] D. Cai, Z. Ni, W. Liu, W. Cai, G. Li, and M. Li, "End-to-End Deep Learning Framework for Speech Paralinguistics Detection Based on Perception Aware Spectrum," in *Proceedings of INTERSPEECH 2017, 18th Annual Conference of the International Speech Communication Association*. Stockholm, Sweden: ISCA, Aug. 2017, pp. 3452–3456.

[25] M. Huckvale and A. Beke, "It sounds like you have a cold! Testing voice features for the Interspeech 2017 Computational Paralinguistics Cold Challenge," in *Proceedings of INTERSPEECH 2017, 18th Annual Conference of the International Speech Communication Association*. Stockholm, Sweden: ISCA, Aug. 2017, pp. 3442–3446.

[26] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized Neural Networks," in *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. Curran Associates, Inc., 2016, pp. 4107–4115.

[27] R. Krishnamoorthi, "Quantizing deep convolutional networks for efficient inference: A whitepaper," arXiv, 2018. [Online]. Available: https://arxiv.org/abs/1806.08342

[28] S. Han, H. Mao, and W. J. Dally, "Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding," arXiv, 2015. [Online]. Available: http://arxiv.org/abs/1510.00149

[29] B. Barrett, K. Locken, R. Maberry, J. Schwamman, R. Brown, J. Bobula, and E. A. Stauffacher, "The Wisconsin Upper Respiratory Symptom Survey (WURSS): a new research instrument for assessing the common cold," *The Journal of Family Practice*, vol. 51, no. 3, p. 265, March 2002.

[30] T. G. Dietterich, "Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms," *Neural Computation*, vol. 10, no. 7, pp. 1895–1923, 1998.

[31] G. Gosztolya, R. Busa-Fekete, T. Grósz, and L. Tóth, "DNN-Based Feature Extraction and Classifier Combination for Child-Directed Speech, Cold and Snoring Identification," in *Proceedings of INTERSPEECH 2017, 18th Annual Conference of the International Speech Communication Association*. Stockholm, Sweden: ISCA, 2017, pp. 3522–3526.