Private Emotion Recognition with Secure Multiparty Computation

Kyle Bittner,[1] **Martine De Cock,**[1,2] **Rafael Dowsley**[3]
[1]School of Engineering and Technology, University of Washington Tacoma
{kyleb29,mdecock}@uw.edu
[2]Dept. of Applied Mathematics, Computer Science and Statistics, Ghent University
martine.decock@ugent.be
[3] Faculty of Information Technology, Monash University
rafael.dowsley@monash.edu

Abstract

We present the first privacy-preserving solution for deep learning-based audio classification using Secure Multiparty Computation (MPC). Our approach allows to classify a speech signal of one party (Alice) with a deep neural network of another party (Bob) without Bob having access to Alice's speech signal in an unencrypted manner. As threat models, we consider both passive security and active security. We evaluate the efficiency-security-accuracy trade-off of the proposed solution in a use case for privacy-preserving emotion detection from speech with a convolutional neural network, answering a question that has remained open in the literature thus far, namely to what extent MPC-based protocols can enable provably secure and highly accurate real-time speech classification.

1 Introduction

Deep learning in audio signal processing, such as human voice audio signal classification, is a rich application area of machine learning (ML). Recordings of human speech, are automatically classified for various purposes, extending from user authentication, service and device control, surveillance, and marketing. The developing prevalence of speech audio processing technology stems from the ever-increasing demand of devices and programs that are "always-listening" – such as smartphones, televisions, and intelligent digital voice assistants – and the technological improvements in speech technology. While there are clear advantages to automated speech classification, application developers can gain knowledge beyond the professed scope from unprotected audio signal processing [35]. As stated in a recent survey paper by Nautsch et al., the continued success of speech technologies hinges upon the development of reliable and efficient privacy-preservation capabilities, specifically designed for the automatic processing of speech signals [26].

We propose the use of techniques from Secure Multiparty Computation (MPC) [9] to allow a user (Alice) to classify her speech signal with an ML model of a model owner (Bob), without Alice revealing her speech signal in plaintext, and without Bob disclosing his ML model in-the-clear, i.e. without encryption. MPC has already been used for privacy-preserving speaker and speech classification with hidden Markov models (HMMs) and Gaussian mixture models (GMMs) [27, 28, 29, 33]. While HMMs and GMMs were popular techniques for speech classification in the 1980s and 1990s, more recently deep learning has emerged as a state-of-the-art technique in this field. To the best of our knowledge, MPC-based secure classification of speech with deep neural networks has never been studied. It is this gap in the literature, which is also called out by Nautsch et al. [26], that we fill. Our results show that MPC based accurate real-time speech classification as would be needed for instance for digital voice assistants such as Apple's Siri, Amazon's Alexa, Google Home, and Microsoft's Cortana is within reach.

The strength of our solution derives both from the use of state-of-the-art MPC schemes [2, 8, 13, 17, 19] and the purposeful design of an MPC-friendly 1-dimensional convolutional neural network (CNN) model architecture that allows for efficient inference in a secure manner. While there is existing work on MPC-based inference with neural networks in the passive security setting [1, 15, 20, 23, 24, 30, 31, 32] and the active security setting [10, 18, 36], to the best of our knowledge, all existing work on MPC-based classification with CNNs is developed for and focused on 2-dimensional CNNs, which are commonly used for classification of images. While speech classification can be done with 2-dimensional inputs as well (spectrogram) and with neural network architectures other than CNNs, for speech classification we advocate the use of 1-dimensional CNNs for their smaller model size and efficiency during the secure inference process. We show, in a use case for human emotion detection from speech, that state-of-the-art accuracy can be obtained with a 1-dimensional CNN, and that this architecture lends itself well to efficient inference without leaking information about Alice's speech signal or Bob's model parameters. For the secure inference, we adapt the work that was done in SecureQ8 [10] for 2-dimensional CNNs (image classification) to 1-dimensional CNNs (speech classification).

Our approach is orders of magnitude faster than a previously proposed approach based on fully homomorphic encryption (FHE) with multilayer perceptrons [12, 34]. Prior to our work, MPC for speech recognition based on neural networks has been considered highly impractical due to an

assumed massive overhead in computation time and communication costs [4]. This has prompted research into the use of trusted execution environments [3, 4]. Contrary to this, the solution we present does not require special hardware, and is fast enough for use in real-time.

2 Methods

2.1 MPC-friendly ML Model Training Our assumption is that Bob has a set of audio files (speech signals) that are each annotated with a label, and he uses these to train an ML model that can assign a correct label to a previously unseen audio file (Alice's input). It is common in speech processing for classifiers to work on features extracted from the speech signal as opposed to work on the raw speech signal itself. These features and the software to extract them are widely known and publicly available. It is for example very common to convert a speech signal into a sequence of feature vectors of mel-frequency cepstral coefficients (MFCC) [11] that are extracted from sliding windows of consecutive speech. We demonstrate that we can train highly accurate ML models for speech classification based on these extracted feature vectors. That in itself is clear evidence that the feature vectors contain meaningful, private information that needs to be kept private during inference, as we do with MPC as described below.

We use the RAVDESS data set for emotion recognition from audio, using both song and speech versions. [21]. From this benchmark data set, we use 2,542 audio files with a length of $\sim$3.5 sec each. Each audio file is annotated with one of eight emotion labels: *neutral* (188), *calm* (376), *sad* (376), *happy* (376), *fearful* (376), *disgust* (192), *angry* (376), and *surprise* (192). We extract vectors of 40 MFCC features from each audio file with the librosa library [22], with default settings for all other parameters. We hold out 33% of the instances for testing, and train a CNN with two convolutional blocks on the remaining data. Both blocks have RELU activation, and the first one has an average pooling layer for downsampling. The convolutional blocks are followed by a dense layer with softmax activation. Model architecture details are provided in Fig. 1.

MPC based on secret sharing, as we use in this paper, performs secure computations on *integers* modulo q. The parameter values of a trained neural network are natively *real* numbers and need to be converted to integers. In deep learning, the conversion of floating-point data in the neural network to integers (quantization) is studied as an effective way to shrink the model size and to accelerate computation, e.g. on edge devices with limited memory and computational power (see e.g. [37]). The use of quantization is growing in popularity in research on privacy-preserving deep learning as well [1, 10, 30]. We use TensorFlow Lite's post-training integer quantization[1] to convert all CNN model parameters to 8-bit integers. The quantized trained model achieves 81.2% accuracy on the test data, which is state-of-the-art for speech emotion recognition. Issa et al. [14] for instance report 71.61% on the same RAVDESS dataset with a CNN approach based on MFCCs and Mel-scaled spectograms, while Nantasri et al. [25] report 82.3% for an evaluation on a simplified version of the RAVDESS dataset that includes seven emotion classes instead of the original eight.

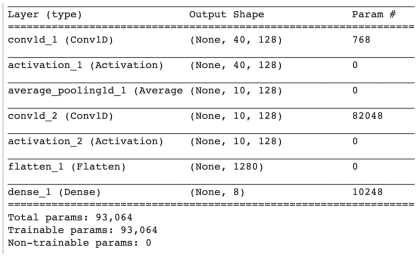
[1] https://www.tensorflow.org/lite/performance/post_training_integer_quant

2.2 Secure Inference with a 1-Dimensional CNN Protocols for MPC enable a set of parties to jointly compute the output of a function over each of the parties' private inputs, without requiring any of the parties to disclose their own inputs to anyone. It is natural to think of speech classification as a two-party computation (2PC) problem in which the parties are the user with the speech signal (Alice) and the model owner (Bob). MPC is concerned with the protocol execution coming under attack by an adversary which may corrupt one or more of the parties to learn private information or cause the result of the computation to be incorrect. MPC protocols are designed to prevent such attacks being successful, and can be mathematically proven to guarantee privacy and correctness. To this end, we follow the standard definition of the Universal Composability (UC) framework [5], in which the security of cryptographic protocols is analyzed by comparing a real world with an ideal world.

There exist a variety of MPC schemes, designed for different numbers of participants and offering various levels of security that correspond to different threat models. In a 2PC setting in which Alice and Bob execute a secure MPC protocol between themselves to perform the privacy-preserving speech classification, one corrupted party means that we are in the so-called scenario of *dishonest majority*. MPC protocols in the dishonest-majority setting are much more computationally expensive than protocols in an honest-majority setting, i.e. when more than half of the protocol participants are honest. Therefore, works on privacy-preserving inference have started considering the setting in which Alice and Bob outsource the secure computations to a set of 3 (3PC) or more servers, of which a majority is assumed to honest (e.g., [10, 18, 31, 36]).

Furthermore, a party can be corrupted in different ways. In the *passive security* setting (also known as *semi-honest* or *honest-but-curious adversaries*), the corrupted parties follow the specified protocol instructions, but they may try to learn additional information (i.e., information other than what can be inferred from their specified inputs and outputs) from the messages exchanged during the protocol execution. Secure MPC protocols prevent such information leakage. In the *active security* setting (also known as *malicious adversaries*), the parties may deviate from the protocol instructions in arbitrary ways, for instance by providing incorrect values on purpose. In this case, secure MPC protocols should prevent information leakage and detect devious behavior. Protection against such a stronger threat model comes at a higher computational cost.

We use the following MPC schemes (see also Table 1): SEMI [17], SEMI2K [8], MASCOT [17], SPDZ2K [8], ReplicatedPrime [2], Replicated2k [2], PsReplicatedPrime [19], PsReplicated2k [13]. In these schemes, all computations are done over integers in a field $\mathbb{Z}_p$ or a ring $\mathbb{Z}_{2^k}$. The choice of the MPC scheme and the underlying algebraic structure can have a substantial impact on the efficiency of the MPC



Layer (type)	Output Shape	Param #
conv1d_1 (Conv1D)	(None, 40, 128)	768
activation_1 (Activation)	(None, 40, 128)	0
average_pooling1d_1 (Average	(None, 10, 128)	0
conv1d_2 (Conv1D)	(None, 10, 128)	82048
activation_2 (Activation)	(None, 10, 128)	0
flatten_1 (Flatten)	(None, 1280)	0
dense_1 (Dense)	(None, 8)	10248

Total params: 93,064
Trainable params: 93,064
Non-trainable params: 0

Line (4), (11), and (12) in the code are only relevant for training, not for inference

```
(1)  model = Sequential()
(2)  model.add(Conv1D(128,5,padding='same',
     input_shape=(40,1)))
(3)  model.add(Activation('relu'))
(4)  model.add(Dropout(0.1))
(5)  model.add(AveragePooling1D(pool_size=(4)))
(6)  model.add(Conv1D(128, 5,padding='same',))
(7)  model.add(Activation('relu'))
(8)  model.add(Dropout(0.1))
(9)  model.add(Flatten())
(10) model.add(Dense(8))
(11) model.add(Activation('softmax'))
(12) opt = keras.optimizers.rmsprop(lr=0.00005,
     rho=0.9, epsilon=None, decay=0.0)
```

Figure 1: CNN architecture and Keras code snippet used to train the emotion recognition model on the RAVDESS data set

protocols (see Section 3). Operations done during the online phase of protocols in these MPC schemes rely on correlated randomness that is not dependent on specific inputs and can be generated during a so-called offline phase. The runtime results presented in Section 3 include both the offline and online phases; for deployment in a real-time audio classification system they can be improved by running the offline phase in advance.

Designing a secure solution based on MPC for inference comes down to representing the function that needs to be privately computed using the basic operations that are provided by the underlying MPC scheme (i.e., the addition and multiplication gates). Once this representation is found, the parties evaluate it gate by gate using existing procedures for private addition and private multiplication. The MPC protocol for secure inference with a 1-dimensional CNN consists of the following steps:

1. First, Alice secret shares her speech signal vector X; Bob secret shares his model parameters θ.
2. Next the parties compute a secret sharing $[\![Z]\!]$ of the output of the first convolutional layer, starting from the secret shared input $[\![X]\!]$ and the secret shared model parameters $[\![\theta]\!]$. To this end they need to perform Frobenius inner products and add bias terms. This boils down to performing multiplications and additions of values that are secret shared among the parties.
3. For secure evaluation of the RELU activation layer, the parties need to replace all negative values in $[\![Z]\!]$ by zeros. This is done on the quantized values using a secure comparison protocol derived from Catrina and De Hoogh [6], followed by a secure multiplication to either keep the original value or replace it by zero in an oblivious way.
4. Average pooling with a window size of P in a 1-dim CNN means that in every row in Z, each (non-overlapping) block of P adjacent elements is replaced by one cell, with the average value of the original block. To do average pooling, the parties first add the values in a block of Z by adding their own shares of these values. Next the parties need to divide the resulting sum $[\![s]\!]$ by P, to yield the average. The window size P is a hyperparameter of the model that is known by Bob. Bob secret shares the value of P, similarly to how he shares θ. For secure division of $[\![s]\!]$ by $[\![\mathrm{P}]\!]$, the parties use an iterative algorithm that is well known in the MPC literature [7].
5. The parties repeat steps 2-3-4 as needed, depending on the neural network architecture.
6. In a CNN, the output of the last convolutional block is flattened into a vector $\mathbf{x}$ of length d, and passed as input into a dense layer. The parties can each flatten their own shares of the values to construct $[\![\mathbf{x}]\!]$. Next, $[\![\mathbf{x}]\!]$ needs to be multiplied with a $d \times o$ matrix $[\![W]\!]$ that contains the weights of the dense layer, and a bias term $[\![b]\!]$ needs to be added. $[\![W]\!]$ and $[\![b]\!]$ have already been provided by Bob as inputs in Step 1. The output of the dense layer is a vector $\mathbf{y}$ of length o. The parties jointly compute $[\![\mathbf{y}]\!]$ by performing dot products and adding the bias term.
7. The class label inferred by the CNN is the index corresponding to the largest value in $\mathbf{y}$. In the final step, the parties obtain a secret sharing $[\![c]\!]$ of the class label by running a secure argmax protocol, which can be straightforwardly constructed using the above mentioned secure comparison protocol.

Throughout this process, all computations are done over encrypted data and model parameters, meaning that Alice does not learn anything about Bob's model weights or training examples, while Bob does not learn anything about Alice's speech signal.

When two fixed-point numbers are multiplied, the result has to be truncated to keep the correct fixed-point representation. In the case of prime fields this is done using either the deterministic truncation protocol of Catrina and De Hoogh [6] or the probabilistic truncation protocol of Catrina and Saxena [7]. In the probabilistic protocol, the probabilities that a number is rounded up or down are proportional to its distance to those bounds. The probabilistic truncation protocol eliminates a lot of invocations of the underlying secure comparison protocol, and therefore improves the efficiency. On the other hand, the probabilistic truncation negatively affects the accuracy of the secure classification as we will show in Section 3. In the case of binary fields, the truncation is done using the adaptations of the above deterministic and probabilistic truncation protocols that were introduced by Dalskov et al. [10]. In the procedures in which the amount of bits to be truncated needs to be kept secret, we use the proto-

Table 1: Accuracy and runtime results for privacy-preserving emotion detection. The accuracy results were obtained by holding 33% of the data out as test data. The classification runtimes are computed as an average over 10 inferences. Results are presented for F32s V2 and F72s V2 Azure instances.

		Truncation	**Accuracy**	**Active Security**		**Passive Security**	
				$\mathbb{Z}_{2^k}$	$\mathbb{Z}_p$	$\mathbb{Z}_{2^k}$	$\mathbb{Z}_p$
low-end VMs	honest majority, 3PC			PsReplicated2k	PsReplicatedPrime	Replicated2k	ReplicatedPrime
	with 3 F32s V2 instances	Probabilistic	73.8%	10.16 sec	9.97 sec	1.24 sec	4.18 sec
		Deterministic	81.2%	12.72 sec	12.44 sec	2.06 sec	4.86 sec
	dishonest majority, 2PC			SPDZ2K	MASCOT	SEMI2K	SEMI
	with 2 F32s V2 instances	Probabilistic	73.8%	250.9 sec	274.6 sec	27.6 sec	92.5 sec
		Deterministic	81.2%	370.0 sec	316.4 sec	40.5 sec	112.3 sec
high-end VMs	honest majority, 3PC			PsReplicated2k	PsReplicatedPrime	Replicated2k	ReplicatedPrime
	with 3 F72s V2 instances	Probabilistic	73.8%	1.35 sec	1.32 sec	0.15 sec	0.52 sec
		Deterministic	81.2%	1.61 sec	1.58 sec	0.26 sec	0.60 sec
	dishonest majority, 2PC			SPDZ2K	MASCOT	SEMI2K	SEMI
	with 2 F72s V2 instances	Probabilistic	73.8%	26.01 sec	28.36 sec	2.77 sec	9.56 sec
		Deterministic	81.2%	33.30 sec	32.28 sec	4.17 sec	11.55 sec

col of Dalskov et al. [10] to perform deterministic truncation by a secret value.

3 Results

We implemented our approach on top of the MP-SPDZ framework [16]. We report results for a variety of MPC schemes in Table 1. All benchmark and accuracy tests were completed on co-located F32s V2 and F72s V2 Azure virtual machines (VMs). We benchmarked our tests on two separate performance level machines to have a comparison of realistic runtimes today and into the future. A F32s V2 VM contains 32 cores, 64 GiB of memory, and up to a 14 Gbps network bandwidth between each VM. The F72s V2 VM represents computing power that could potentially be used more widespread in the future; it contains 96 cores, 144 GiB of memory, and a 30 Gbps connection speed between VM. Within each experiment, all computations were done on 2 or 3 VMs of the same kind. The first two rows with results in Table 1 for instance are for the scenario in which Alice and Bob outsource the computations to 3 low-end VMs (3PC), while the bottom two rows in Table 1 correspond to the scenario in which Alice and Bob execute the secure MPC protocol between themselves, each on a high-end VM (2PC).

The classification runtimes are computed as an average over 10 inferences, and they include the time needed for both the offline and the online phases. While privacy-preserving ML runtime results based on MPC are in the literature often only given in terms of runtime estimates, we stress that all runtime results presented in Table 1 are actual wall-clock time measurements for doing end-to-end secure classification.

In terms of **accuracy**, we observe that the accuracy results obtained with the deterministic truncation protocol are the same as the accuracy results in-the-clear (81.2%), while the probabilistic truncation protocol causes a significant drop in accuracy to 73.8%. These numbers are interesting by themselves: while Dalskov et al. [10] write that the use of a probabilistic truncation protocol may hurt classification accuracy, to the best of our knowledge, we are the first to evaluate and measure this drop in accuracy experimentally on a real-life data set.

In terms of **privacy**, MPC is very reliable: other than the result of the classification (which can be selectively revealed to the model owner, data owner, or a third party depending on the application), no information about the speech signal or the trained model parameters is leaked to any participant of the protocol. When performing oblivious speech classification, the price paid for keeping the data and the model private, is an increase in computational cost and runtime.

In terms of **runtime**, first we observe that the absolute runtimes that we obtain are an order of magnitude smaller (better) than the runtimes reported for image classification in [10]. This is because our overall neural network architecture is far more compact; the fact that we choose to use a 1-dimensional CNN instead of a 2-dimensional CNN contributes to this gain in speed. Secondly, and in line with common knowledge in the MPC literature, the fastest results are obtained when Alice and Bob outsource the computations to 3 semi-honest servers (3PC). As long as these servers do not collude with each other, they do not learn anything about Alice's speech signal or about Bob's trained model parameters. We have also included scenarios with stronger security assumptions in our study, namely, in increasing order of runtime: malicious adversaries with an honest majority (3PC), semi-honest adversaries with a dishonest majority (2PC), and malicious adversaries with a dishonest majority (2PC). Actively secure protocols remain secure even if one of the parties is a malicious adversary who deviates from the protocol specification. This makes these protocols most suitable for sensitive applications, even if they come at a notably higher computational cost. Ultimately the bottle-neck in execution time is network parameters between hosts, as network latency and packet loss has the greatest negative influence on performance. From our findings, we conclude that SMC is practically applicable with minor network limitations within intranet environments. hardware requirements to perform MPC on deep learning audio signal classifiers is within the ranges of today's hardware.

As performance of MPC protocols continue to improve, we expect increased feasibility of MPC in practical applications.

Acknowledgements

The authors would like to thank Marcel Keller for making the MP-SPDZ framework available, and for his assistance in the use of the framework. The authors would like to thank Microsoft for the generous donation of cloud computing credits through the UW Azure Cloud Computing Credits for Research program.

References

[1] Agrawal, N.; Shahin Shamsabadi, A.; Kusner, M. J.; and Gascón, A. 2019. QUOTIENT: two-party secure neural network training and prediction. In *ACM SIGSAC Conference on Computer and Communications Security*, 1231–1247.

[2] Araki, T.; Furukawa, J.; Lindell, Y.; Nof, A.; and Ohara, K. 2016. High-throughput semi-honest secure three-party computation with an honest majority. In *ACM SIGSAC Conference on Computer and Communications Security*, 805–817.

[3] Bayerl, S. P.; Frassetto, T.; Jauernig, P.; Riedhammer, K.; Sadeghi, A.-R.; Schneider, T.; Stapf, E.; and Weinert, C. 2020. Offline model guard: Secure and private ML on mobile devices. In *Design, Automation & Test in Europe Conference & Exhibition (DATE)*.

[4] Brasser, F.; Frassetto, T.; Riedhammer, K.; Sadeghi, A.-R.; Schneider, T.; and Weinert, C. 2018. VoiceGuard: Secure and Private Speech Processing. In *Interspeech*, 1303–1307.

[5] Canetti, R. 2000. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology* 13(1): 143–202.

[6] Catrina, O.; and De Hoogh, S. 2010. Improved primitives for secure multiparty integer computation. In *International Conference on Security and Cryptography for Networks*, 182–199.

[7] Catrina, O.; and Saxena, A. 2010. Secure computation with fixed-point numbers. In *International Conference on Financial Cryptography and Data Security*, 35–50.

[8] Cramer, R.; Damgård, I.; Escudero, D.; Scholl, P.; and Xing, C. 2018. SPD$\mathbb{Z}_{2^k}$: Efficient MPC mod 2^k for Dishonest Majority. In *Annual International Cryptology Conference*, 769–798.

[9] Cramer, R.; Damgard, I.; and Nielsen, J. 2015. *Secure Multiparty Computation and Secret Sharing*. New York: Cambridge University Press Print.

[10] Dalskov, A.; Escudero, D.; and Keller, M. 2020. Secure evaluation of quantized neural networks. *Proceedings on Privacy Enhancing Technologies* 2020(4): 355–375.

[11] Davis, S.; and Mermelstein, P. 1980. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal processing* 28(4): 357–366.

[12] Dias, M.; Abad, A.; and Trancoso, I. 2018. Exploring hashing and cryptonet based approaches for privacy-preserving speech emotion recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2057–2061.

[13] Eerikson, H.; Keller, M.; Orlandi, C.; Pullonen, P.; Puura, J.; and Simkin, M. 2020. Use your brain! Arithmetic 3PC for any modulus with active security. In *1st Conference on Information-Theoretic Cryptography (ITC 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

[14] Issa, D.; Demirci, M. F.; and Yazici, A. 2020. Speech emotion recognition with deep convolutional neural networks. *Biomedical Signal Processing and Control* 59: 101894.

[15] Juvekar, C.; Vaikuntanathan, V.; and Chandrakasan, A. 2018. GAZELLE: A low latency framework for secure neural network inference. In *27th USENIX Security Symposium*, 1651–1669.

[16] Keller, M. 2020. MP-SPDZ: A Versatile Framework for Multi-Party Computation. Cryptology ePrint Archive, Report 2020/521. https://eprint.iacr.org/2020/521.

[17] Keller, M.; Orsini, E.; and Scholl, P. 2016. MASCOT: faster malicious arithmetic secure computation with oblivious transfer. In *ACM SIGSAC Conference on Computer and Communications Security*, 830–842.

[18] Kumar, N.; Rathee, M.; Chandran, N.; Gupta, D.; Rastogi, A.; and Sharma, R. 2020. Cryptflow: Secure tensorflow inference. In *41st IEEE Symposium on Security and Privacy*, 336–353.

[19] Lindell, Y.; and Nof, A. 2017. A framework for constructing fast MPC over arithmetic circuits with malicious adversaries and an honest-majority. In *ACM SIGSAC Conference on Computer and Communications Security*, 259–276.

[20] Liu, J.; Juuti, M.; Lu, Y.; and Asokan, N. 2017. Oblivious neural network predictions via MiniONN transformations. In *ACM SIGSAC Conference on Computer and Communications Security*, 619–631.

[21] Livingstone, S. R.; and Russo, F. A. 2018. The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English. *PloS one* 13(5).

[22] McFee, B.; Raffel, C.; Liang, D.; Ellis, D. P.; McVicar, M.; Battenberg, E.; and Nieto, O. 2015. Librosa: Audio and music signal analysis in Python. In *Proceedings of the 14th Python in Science Conference*, volume 8, 18–25.

[23] Mishra, P.; Lehmkuhl, R.; Srinivasan, A.; Zheng, W.; and Popa, R. A. 2020. DELPHI: A Cryptographic Inference Service for Neural Networks. In *29th USENIX Security Symposium*.

[24] Mohassel, P.; and Zhang, Y. 2017. SecureML: A system for scalable privacy-preserving machine learning. In *IEEE Symposium on Security and Privacy (SP)*, 19–38.

[25] Nantasri, P.; Phaisangittisagul, E.; Karnjana, J.; Boonkla, S.; Keerativittayanun, S.; Rugchatjaroen, A.; Usanavasin, S.; and Shinozaki, T. 2020. A Light-Weight Artificial Neural Network for Speech Emotion Recognition using Average Values of MFCCs and Their Derivatives. In *17th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, 41–44. IEEE.

[26] Nautsch, A.; Jiménez, A.; Treiber, A.; Kolberg, J.; Jasserand, C.; Kindt, E.; Delgado, H.; Todisco, M.; Hmani, M. A.; Mtibaa, A.; et al. 2019. Preserving privacy in speaker and speech characterisation. *Computer Speech & Language* 58: 441–480.

[27] Pathak, M. A.; and Raj, B. 2013. Privacy-preserving speaker verification and identification using Gaussian mixture models. *IEEE Transactions on Audio, Speech, and Language Processing* 21(2): 397–406.

[28] Pathak, M. A.; Raj, B.; Rane, S. D.; and Smaragdis, P. 2013. Privacy-preserving speech processing: cryptographic and string-matching frameworks show promise. *IEEE Signal Processing Magazine* 30(2): 62–74.

[29] Portêlo, J.; Raj, B.; Abad, A.; and Trancoso, I. 2014. Privacy-preserving speaker verification using garbled GMMs. In *22nd European Signal Processing Conference (EUSIPCO)*, 2070–2074. IEEE.

[30] Riazi, M. S.; Samragh, M.; Chen, H.; Laine, K.; Lauter, K.; and Koushanfar, F. 2019. XONN: XNOR-based Oblivious Deep Neural Network Inference. In *28th USENIX Security Symposium*, 1501–1518.

[31] Riazi, M. S.; Weinert, C.; Tkachenko, O.; Songhori, E. M.; Schneider, T.; and Koushanfar, F. 2018. Chameleon: A hybrid secure computation framework for machine learning applications. In *Asia Conference on Computer and Communications Security*, 707–721. ACM.

[32] Rouhani, B. D.; Riazi, M. S.; and Koushanfar, F. 2018. DeepSecure: Scalable provably-secure deep learning. In *55th Annual Design Automation Conference (DAC)*.

[33] Smaragdis, P.; and Shashanka, M. 2007. A framework for secure speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing* 15(4): 1404–1413.

[34] Teixeira, F.; Abad, A.; and Trancoso, I. 2019. Privacy-preserving Paralinguistic Tasks. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 6575–6579.

[35] Tomashenko, N.; Srivastava, B. M. L.; Wang, X.; Vincent, E.; Nautsch, A.; Yamagishi, J.; Evans, N.; Bonastre, J.-F.; Noé, P.-G.; Todisco, M.; and Patino, J. 2020. The VoicePrivacy 2020 Challenge Evaluation Plan. https://www.voiceprivacychallenge.org/docs/VoicePrivacy_2020_Eval_Plan_v1_1.pdf.

[36] Wagh, S.; Gupta, D.; and Chandran, N. 2019. SecureNN: 3-party secure computation for neural network training. *Proceedings on Privacy Enhancing Technologies* 2019(3): 26–49.

[37] Yang, Y.; Deng, L.; Wu, S.; Yan, T.; Xie, Y.; and Li, G. 2020. Training high-performance and large-scale deep neural networks with full 8-bit integers. *Neural Networks* 125: 70–82.