EXPLORING HASHING AND CRYPTONET BASED APPROACHES FOR PRIVACY-PRESERVING SPEECH EMOTION RECOGNITION

Miguel Dias, Alberto Abad, Isabel Trancoso

Instituto Superior Técnico - Universidade de Lisboa
Instituto de Engenharia de Sistemas e Computadores - Investigação e Desenvolvimento (INESC-ID)

ABSTRACT

The outsourcing of machine learning classification and data mining tasks can be an effective solution for those parties that need machine learning services, but lack the appropriate resources, knowledge and/or tools to carry them out, in their own premises. This solution, however, raises major privacy concerns, in particular, when irrevocable biometric data such as speech is involved. In this work, we focus on the development of privacy-preserving schemes in a speech emotion recognition task, as a proof of concept that could be extended to other speech analytics tasks. Our aim is to prove that the implementation of privacy-preserving speech mining schemes in challenging tasks involving paralinguistic features is not only feasible, but also accurate. Using distance-preserving hashing techniques in a first approach, and homomorphic encryption in a second approach, we successfully protect sensitive data with little degradation costs regarding the accuracy of the predictive models.

***Index Terms*—** Cryptography, hashing, cryptonets, emotion recognition, paralinguistics

1. INTRODUCTION

In today's world, privacy is a very important matter. The increasing usage of biometrics for various machine learning tasks, and the fact that these are irrevocable and unique to each individual, results in major privacy issues. Thus, in order to keep up with the newest trends in biometric machine learning, namely *Machine Learning as a Service* (MLaaS) and other cloud-based computations, it is necessary to devise new and efficient ways to preserve the confidentiality of biometric data.

The voice of an individual conveys a great deal of information. In fact, information about the emotional and health status, age, gender and other types of both physical and psychological traits may be "mined" from the individual's voice. A party with malicious intent may resort to machine learning algorithms to extract this type of information and use it for their benefit.

In this paper, we provide two privacy-preserving emotion recognition frameworks that may be extended to other paralinguistic and speech analytics tasks [1, 2]. In a first approach, we adapt a Support Vector Machine (SVM) to work with hashes generated through Secure Modular Hashing (SMH) [3], while, in a second approach, we make use of the structure-preserving properties of a Somewhat Fully Homomorphic Encryption (SFHE) scheme to implement privacy in a neural network [4, 5]. By performing privacy-preserving emotion recognition on speech signals, we aim to effectively secure the biometric content of a recording while obtaining little to no degradation in classification scores. This broadens the spectrum in which privacy preserving schemes may be applied and promotes advancements in *Secure Machine Learning as a Service* (SMLaaS).

The remainder of this paper is organized as follows. Section 2 reports related studies on both emotion recognition and privacy-preserving schemes. Section 3 describes the dataset used in the experiments. Section 4 extends the SMH technique to an emotion recognition task. In Section 5, a neural network based on homomorphic encryption is implemented and tested. Finally, Section 6 shows the main conclusions and future work.

2. RELATED WORK

Privacy-preserving paralinguistic mining is still a relatively unexplored research topic. Nonetheless, several major works in related fields were crucial to the development of this work.

Secure binary embedding (SBE) [6] is a scheme based on nearest-neighbor search with secure randomized embeddings that uses quantized random projections. In short, it allows information about true vectors to be leaked if their corresponding SBE hashes are close enough to each other. Although this scheme provides information-theoretic security on its own, there is no guarantee that an attacker is not able to infer any information about the true vectors if he manages to get a hold of the secret keys, or if he has some prior knowledge about the true vectors. This scheme was later applied to some speech-related machine learning tasks, such as speaker verification [7] and query-by-example speech search [8] using i-vectors. Later, Abelino et. al. [3] provided a generalization of this method by replacing the standard quantization function of $Q(x) = x \mod 2$ with a more generalized one $Q(x) = x \mod k$, where k is chosen by the user. Note that, for these schemes, the number of input features needs to be the same across all samples and needs to be independent from the length of the recording.

Anyhow, the need to follow the neural network trend has interested several researchers to devise privacy-preserving schemes that would work on state-of-the-art Deep Neural Networks (DNN). Some types of DNN allow for features to be learned in the training phase by using deep convolutional networks [9, 10], thus diminishing the need for hand-engineered feature extraction. Moreover, these learned features may be better suited for the task at hand, since they are learned using a training set specific for that task. In fact, these deep convolutional networks have already been applied to a speech emotion recognition task [9], presenting state-of-the-art accuracy scores. These networks, however, require massive training datasets to be viable.

There are several techniques to preserve privacy in deep learning. These include the controlled leaking of information, such as the widely popular differential privacy [11], and computations over encrypted data [4, 5], using Homomorphic Encryption (HE) and other cryptographic primitives. Differential privacy was successfully applied to image recognition by Abadi et. al. [11], with insignificant degradation of accuracy. More recently, a new privacy-preserving

scheme using deep neural networks, namely cryptonets, was introduced by Dowlin et. al. [5]. These are deep neural networks that use an FHE scheme to make computations over encrypted data. The first somewhat efficient FHE scheme was proposed by Gentry [12]. Although they do not support encrypted learning, cryptonets mark an important milestone in SMLaaS, since a cloud containing an already trained model may effectively make predictions on client's encrypted data without inferring anything about the data itself. A light cryptonet (9 layers) was used on a hand-written digit database (MNIST [13]), achieving 99.00% accuracy [5] by using a square activation function (x^2) and replacing max-pooling layers with sum-pooling. Later, in order to reduce noise-related errors in deeper networks, Chabanne et. al. [4] proposed a solution based on the approximation of the ReLU activation function by polynomials, the use of batch normalization layers and the replacement of the max-pooling layer with an average-pooling layer. With this solution they obtained 99.30% for a deep Convolutional Neural Network (CNN) with 24 layers, although the state-of-the-at for the MNIST database exceeds 99.70%.

3. DATASET

The dataset considered for the experiments in this work is the latest version of the "Let's Go" dataset [14]. "Let's Go" [15] is a public dialog system that provides the bus schedule information for Pittsburgh's Port Authority buses during off-peak hours. The corpus was split into 3,005 recordings for training, 657 recordings for validation and 581 for testing. Each recording is labelled with an emotional status of either *neutral* or *angry*.

4. SECURE MODULAR HASHING

Secure Modular Hashing (SMH) [3] is a technique that hides and secures data by converting vectors into bit sequences using band-quantized random projections. SMH is based on the fact that if the Hamming distance between two hashed vectors is lower than a certain threshold, controlled mainly by a scaling factor Δ and modulus k, then, the hashes leak information about the true Euclidean distance between the feature vectors, otherwise no information is leaked. In order to generate the SMH hashes, one has to compute

$$h(\mathbf{x}) = Q(\lfloor \Delta^{-1}(\mathbf{A}\mathbf{x} + \mathbf{w}) \rfloor) = \\ = \lfloor \Delta^{-1}(\mathbf{A}\mathbf{x} + \mathbf{w}) \rfloor \mod k, \quad (1)$$

where $h(\mathbf{x})$ is the hashed product of the vector $\mathbf{x}$, $\mathbf{w} \sim \text{unif}(0, k)$, also known as dither, and $A \sim \mathcal{N}(0, 1)$ is an independent matrix. Δ is the scalar factor of the SMH that controls the variance of A. It is worth noticing that one can simply set the value of $k = 2$ to have a SBE scheme.

4.1. Baseline Using SVM

Due to its simplicity and the fact that, given a rather small amount of training data, it presents state-of-the-art results, we implement the baseline of the emotion recognition classifier using the SVM [16] algorithm with the Radial Basis Function (RBF) kernel. Moreover, a model that uses the Euclidean distance between features is required, since the SMH is based on the Euclidean distance being leaked through the hashes. Using previously scaled eGeMAPS [17] features along with the LIBSVM [18] software we obtain an accuracy score of 89.80% on the validation set and **82.79%** on the test set. eGeMAPS features were chosen for these experiments due to their good accuracy in paralinguistic tasks. In fact, the use of other feature sets, such as openSMILE [19], that contains about 6,000 features, would result in a poor generalization of the paralinguistic model, for such a limited training data set.

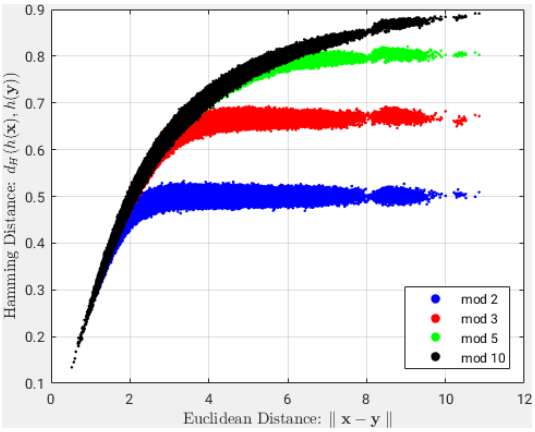

Fig. 1. Hamming distance as a function of the Euclidean distance for different values of k, $mpc = 64$ and $\Delta = 3$ for the database hashes.

4.2. Experiments Using SMH

4.2.1. RBF-Hamming Kernel

The SMH scheme is based on the Hamming distance between the hashes computed, therefore, it is necessary to adapt the SVM model to use Hamming distances instead of Euclidean distances. Thus, the RBF kernel needs to be modified to work with Hamming distances. Hence, the RBF-Hamming kernel would be then computed as

$$k(\mathbf{x}, \mathbf{y}) = e^{-\gamma . d_H^2(h(\mathbf{x}), h(\mathbf{y}))}, \quad (2)$$

where $d_H(h(\mathbf{x}), h(\mathbf{y}))$ corresponds to the normalized Hamming distance between the hashes $h(\mathbf{x})$ and $h(\mathbf{y})$, and γ corresponds to the scaling factor. Note that, for a given $\mathbf{A}$ and $\mathbf{x}$, the modified kernel closely approximates the conventional RBF for small $d(\mathbf{x}, \mathbf{y})$, but varies significantly when $d(\mathbf{x}, \mathbf{y})$ becomes larger.

4.2.2. Experimental Setup

It is known that the parameters Δ, M (number of samples) and k affect the behaviour of the SMH [3, 6], however, M by itself is not useful. An alternative, measurements per coefficient (mpc) or, in case $k = 2$, bits per coefficient (bpc) is used and computed as M/L, controlling the variance of the universal quantizer [20]. L refers to the dimensionality of the vector.

The SMH hashes were computed using MATLAB for different percentages of leakage, values of bpc/mpc and values of k. Afterwards, these hashes were used for training and classification on the modified SVM model. Notice that the leakage is defined as the total amount of hashes that leak information, i.e., that have their Hamming distance below the privacy threshold, which is empirically set by inspecting Fig. 1 and checking the position in which the Hamming distance stops being proportional to the Euclidean distance. For each $k = \{2, 3, 5, 10\}$ the privacy thresholds were set to 0.475, 0.65, 0.775 and 0.875, respectively. The leakage percentage for different sets of parameters k, Δ and $mpc = 32$ may be inspected in Table 1. From here it is possible to notice the influence k and Δ have on the leakage. The mpc may also influence the leakage, but on a smaller level [20].

Table 1. Leakage percentage of the test set for different parameters of Δ and k, with $mpc = 32$.

Leakage	$\Delta = 2$	$\Delta = 2.5$	$\Delta = 3$	$\Delta = 3.5$
$k = 2$	0.15 %	1.38 %	6.07 %	16.25 %
$k = 3$	27.69 %	56.58 %	77.02 %	88.96 %
$k = 5$	75.00 %	91.54 %	97.11 %	99.06 %
$k = 10$	98.78 %	99.86 %	99.98 %	99.99 %

Table 2. Accuracy results of the test set for different parameters of Δ and k, with $mpc = 32$.

Accuracy	$\Delta = 2$	$\Delta = 2.5$	$\Delta = 3$	$\Delta = 3.5$
$k = 2$	—	—	79.52 %	80.21 %
$k = 3$	**80.56 %**	81.58 %	81.75 %	82.80 %
$k = 5$	82.27 %	82.44 %	81.93 %	82.79 %
$k = 10$	82.61 %	82.61 %	82.61 %	82.79 %

4.2.3. Results and Discussion

The accuracy of a private SVM classifier based on SMH, is strongly related with the amount of information regarding the distance between feature vectors leaked by the hashes. Thus, it is fair to assume that, using the parameters from Table 1 the classification will be better as the leakage increases. Table 2 presents the classification results for the different parameters used in Table 1, with optimized C and γ for the training phase of the SVM classifier using the RBF-Hamming kernel.

As expected, for greater leakage values, the classification results tend to approximate the non-private results of 82.79% on the test set. Note that, for small leakages (less than 5%), the classifier is not able to effectively classify any of the utterances, thus always classifying them as *neutral*.

From both Tables 1 and 2, we note that a careful selection of the parameters mpc, k and Δ is necessary to achieve the best accuracy scores possible, while still providing strong security. The experiments performed led us to choose a compromise solution using $mpc = 32$, $\Delta = 3.5$ and $k = 2$ (highlighted in Table 2), since only 16.25% of the hashes leak information, which is sufficient for adequate security, and provides 80.21% accuracy, which presents approximately **2.5%** degradation from the non-private baseline. This is an acceptable degradation for the preservation of confidentiality. This scheme can, therefore, be used for encrypted testing and *Secure Joint Learning* (SJL) between several untrustworthy parties by sending hashes generated with the same keys to a trustworthy server that trains a joint model.

5. CRYPTOGRAPHIC NEURAL NETWORK

Cryptographic Neural Networks, or just cryptonets [4, 5], are simply ANNs that are able to perform operations over encrypted data. The main idea of using this privacy-preserving scheme is to unite the powerful deep learning models with a structure-preserving encryption scheme, such as FHE. However, FHE schemes only allow multiplications and additions to be performed on ciphertexts and, therefore, the ANN must be carefully analyzed and modified to only perform these FHE-friendly operations.

5.1. Baseline Using ANN

Artificial Neural Networks may range from simple and light Multilayer Perceptrons (MLP) to much more complex and deeper networks. For these experiments we considered a 2-layered MLP network. The inputs of these would be the previously scaled eGeMAPS features, for reasons previously mentioned, and the outputs would be either *neutral* or *angry*. The first Fully Connected (FC) layer has 100 hidden units, while the second has 50 hidden units. The activation function present at the end of each FC layer is the *Rectified Linear Unit* (ReLU), $f(x) = max(0, x)$. The learning algorithm is the gradient descent with a learning rate of 0.01 and weight decay of 0. The whole baseline system was implemented in C++.

By training and testing this model using the "Let's Go" emotional dataset, we obtained 88.58% accuracy on the validation set and **82.62%** on the test set. The accuracy results are similar to the results obtained with the SVM baseline model in Section 4.1, as expected, given the size of the training set.

5.2. Experiments Using Homomorphic Encryption

5.2.1. Polynomial Activation Function

In an inferring stage, the inputs will undergo a feedforward sweep of the network, passing through the activation function. This activation function (ReLU), however, is not compliant with FHE-friendly operations and, therefore, the activation function present in the inferring stage must be a polynomial approximation of the ReLU function. The accuracy scores will, then, be related to how good this approximation actually is, i.e. related to the degree of the polynomial approximation. Still, one must care for the encryption noise from homomorphic operations, that increases exponentially with the polynomial degree, and for the computational overhead that comes with a higher polynomial degree. A visual representation of the approximated ReLU polynomial functions for p equal to 2, 4 and 6 is illustrated in Fig. 2. These may be computed with the *polyfit* function of the *numpy* package, in Python.

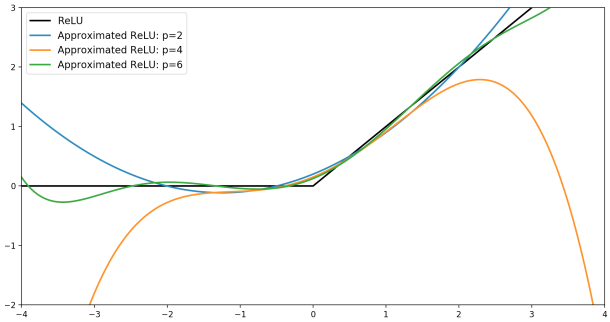

Fig. 2. Polynomial approximations of the ReLU (adapted [4]).

5.2.2. Pre-activation Normalization

Although the approximation of the activation function is crucial, it is not sufficient to guarantee good accuracy results and generalization. Notice that, by inspecting Fig. 2, the approximation is only viable within the interval $[-2, 2]$ (deviating exponentially outside of it) and, therefore, a normalization must be conducted to force the activation inputs to be within this viable interval. Common normalization techniques are incompatible with this approach, since one has to have access to the real unaltered feature vector and, in this case, all feature vectors are encrypted. To achieve a viable normalization, we use the batch normalization technique [21], which allows the network to learn the mean and variance of the training population, and the scale and shift parameters, that may later be used for the normalization in the inferring stage. Notice that, according to the law of large numbers, the mean and variance of the training population should give a rather accurate approximation of the mean and variance of the test feature vector, given a significant amount of training

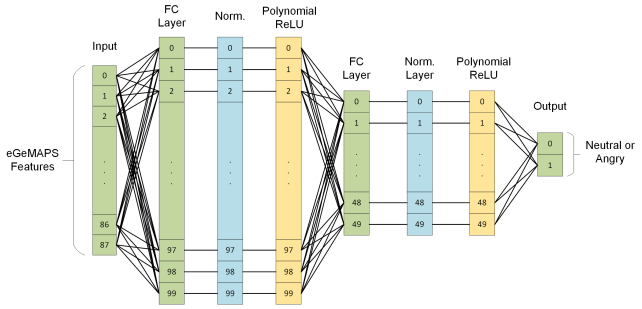

Fig. 3. Final network architecture.

samples. This normalization not only guarantees that all activation inputs fall within the viable interval, but also promotes a regularization of the model and faster training, due to the usage of higher learning rates.

5.2.3. Encryption Noise

Noise is a fundamental part of encryption. As soon as a plaintext is encrypted, it has a non-zero amount of noise. This noise will grow as homomorphic operations are performed. In fact, in the SEAL library, which is based on the Fan-Vercauteren homomorphic encryption scheme [22], each ciphertext is assigned a *noise budget*, which will decrease everytime an homomorphic operation is performed on it. While the noise budget is greater than zero, the ciphertext may be correctly decrypted, otherwise it becomes undecryptable. In higher degree polynomials and deeper networks with several layers, issues may arise due to the amount of homomorphic operations performed on a single ciphertext. In order to solve the noise problem, two things can be done. Either decrease the *plaintext modulus* parameter of the encryption process or increase both the *polynomial modulus* and *coefficient modulus* parameters. Increasing the *plaintext modulus* will significantly decrease the noise growth in homomorphic multiplication, but also increase the chance of the *plaintext polynomial* wrapping around the *plaintext modulus*. On the other hand, increasing both the *polynomial modulus* and *coefficient modulus* will effectively increase the amount of noise budget, while also increasing the security of the encryption. However, increasing these parameters will also increase the computational time of homomorphic encryptions.

5.2.4. Experimental Setup

With the modifications to our initial MLP, the network is ready for testing. For the encryption stage we used the SEAL [23] library. We used the library's fractional encoder with an *integer coefficient* of 64, *fraction coefficient* of 32 and *pool* of 3, leaving the rest as default. The encryption parameters were set to $x^{4096}+1$ for the *polynomial modulus*, 4096 for the *coefficient modulus* and 8 for the *plaintext modulus*, leaving the rest as default. We performed experiments for different polynomial degree approximations, and for a network with and without pre-activation normalization.

5.2.5. Results and Discussion

There are a lot of factors that may influence the accuracy of this cryptonet. These include, but are not limited to, the quality of the polynomial approximation, the loss of precision due to normalization and the encryption noise. Inspecting the accuracy results obtained, in Table 3, we notice a degradation around $\sim 2\% - 3\%$, comparing to the baseline of 82.62%. The introduction of the pre-activation normalization greatly increases the accuracy on the second degree polynomial, while slightly decreasing the accuracy for higher degree polynomials. Even though the eGeMAPS features were previously scaled to $[-1, 1]$, the entries of the polynomial ReLU might still fall outside the viable interval, significantly decreasing the accuracy of the classifier. As seen in Table 3 this effect is most dire in lower degree polynomials. Notice that, for other types of unscaled inputs, the accuracy for all polynomial degrees could be extremely low, since many inputs would fall outside the viable interval. Anyhow, the introduction of the normalization counters this effect by forcing the values to remain within the interval. In fact, an improvement is seen for the second degree polynomial of around 20%. Still, the loss of precision due to this normalization slightly decreases the accuracy for higher degree polynomials, which also tend to be slower computationally. These experiments led us to conclude that the pre-activation normalization plays a crucial role in the generalization of the networks to other inputs.

Table 3. Accuracy results of the tests performed on the encrypted test set for different values of p, with and without normalization.

Accuracy	$p=2$	$p=4$	$p=6$
Without Normalization	59.72 %	81.58 %	81.41 %
With Normalization	79.17 %	80.55 %	80.21 %

6. CONCLUSIONS AND FUTURE WORK

This work contributes with an extension of the privacy-preserving paralinguistic mining literature, advancements in the field of *Secure Machine Learning as a Service* (SMLaaS), development of the concept of *Secure Joint Learning* (SJL) with SMH, and, finally, encrypted testing in state-of-the-art classifiers such as SVMs and neural networks.

Both the SMH and cryptonet approaches effectively secure the client's speech biometric while performing emotion recognition onto it. The security of the SMH is based on the fact the randomly generated keys $\mathbf{A}$ and $\mathbf{w}$ are private, and on the assumption that $\mathbf{A}$ is non-invertible. In the case it is invertible, it would still be too computationally demanding to compute its inverse, even if the imposter had some prior knowledge about the true vector. Therefore, SMH provides a simple but strong information-theoretic security. On the other hand, HE offers security based on the hardness of Ring Learning with Errors (RLWE) problem [24]. An attacker would have to solve the *d-sample decision RLWE* problem, which is extremely hard. Thus, we prove that it is possible to devise privacy-preserving speech emotion recognition schemes, and expect that these results may be extended to other paralinguistic tasks, with little to no accuracy degradation. Moreover, these schemes may be implemented in cloud-based applications, promoting the development of SMLaaS. Future topics for research may be the implementation of end-to-end schemes for privacy-preservation in speech, encrypted learning in ANNs, deeper speech cryptonets, and finally, proving the non-invertibility of matrix $\mathbf{A}$ and understanding the role of k in Equation 1 for different scenarios. It might also be interesting to implement privacy-preserving schemes in state-of-the-art classifiers based on differential privacy.

7. ACKNOWLEDGEMENTS

The authors would like to acknowledge the very helpful suggestions of Bhiksha Raj, Abelino Jiménez and José Portêlo. This work was supported by national funds through Fundação para a Ciência e a Tecnologia (FCT) with reference UID/CEC/50021/2013, and by project H2020-EU.3.7 contract 653587 (LAW-TRAIN).

8. REFERENCES

[1] B. Schuller, S. Steidl, A. Batliner, F. Burkhardt, L. Devillers, C. MüLler, and S. Narayanan, "Paralinguistics in speech and language—state-of-the-art and the challenge," *Computer Speech & Language*, vol. 27, no. 1, pp. 4–39, 2013.

[2] B. Schuller, A. Batliner, S. Steidl, and D. Seppi, "Recognising realistic emotions and affect in speech: State of the art and lessons learnt from the first challenge," *Speech Communication*, vol. 53, no. 9, pp. 1062–1087, 2011.

[3] A. Jiménez, B. Raj, J. Portêlo, and I. Trancoso, "Secure modular hashing," in *Information Forensics and Security (WIFS), 2015 IEEE International Workshop on*. IEEE, 2015, pp. 1–6.

[4] H. Chabanne, A. de Wargny, J. Milgram, C. Morel, and E. Prouff, "Privacy-preserving classification on deep neural network." *IACR Cryptology ePrint Archive*, vol. 2017, p. 35, 2017.

[5] R. Gilad-Bachrach, N. Dowlin, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing, "Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy," in *International Conference on Machine Learning*, 2016, pp. 201–210.

[6] P. Boufounos and S. Rane, "Secure binary embeddings for privacy preserving nearest neighbors," in *Information Forensics and Security (WIFS), 2011 IEEE International Workshop on*. IEEE, 2011, pp. 1–6.

[7] J. Portêlo, A. Abad, B. Raj, and I. Trancoso, "Secure binary embeddings of front-end factor analysis for privacy preserving speaker verification." in *INTERSPEECH*, 2013, pp. 2494–2498.

[8] J. Portêlo, A. Abad, B. Raj, and I. Trancoso, "Privacy-preserving query-by-example speech search," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 1797–1801.

[9] G. Trigeorgis, F. Ringeval, R. Brueckner, E. Marchi, M. A. Nicolaou, B. Schuller, and S. Zafeiriou, "Adieu features? end-to-end speech emotion recognition using a deep convolutional recurrent network," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 5200–5204.

[10] B. Milde and C. Biemann, "Using representation learning and out-of-domain data for a paralinguistic speech task," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[11] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2016, pp. 308–318.

[12] C. Gentry *et al.*, "Fully homomorphic encryption using ideal lattices." in *STOC*, vol. 9, no. 2009, 2009, pp. 169–178.

[13] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[14] S. Ultes, M. J. P. Sánchez, A. Schmitt, and W. Minker, "Analysis of an extended interaction quality corpus," in *Natural Language Dialog Systems and Intelligent Assistants*. Springer, 2015, pp. 41–52.

[15] A. Raux, B. Langner, D. Bohus, A. W. Black, and M. Eskenazi, "Let's go public! taking a spoken dialog system to the real world," in *in Proc. of Interspeech 2005*. Citeseer, 2005.

[16] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[17] F. Eyben, K. R. Scherer, B. W. Schuller, J. Sundberg, E. André, C. Busso, L. Y. Devillers, J. Epps, P. Laukka, S. S. Narayanan *et al.*, "The geneva minimalistic acoustic parameter set (gemaps) for voice research and affective computing," *IEEE Transactions on Affective Computing*, vol. 7, no. 2, pp. 190–202, 2016.

[18] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011, software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[19] F. Eyben, M. Wöllmer, and B. Schuller, "Opensmile: the munich versatile and fast open-source audio feature extractor," in *Proceedings of the 18th ACM international conference on Multimedia*. ACM, 2010, pp. 1459–1462.

[20] J. M. L. Portêlo, "Privacy-preserving frameworks for speech mining," Ph.D. dissertation, Instituto Superior Técnico, 2015.

[21] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456.

[22] J. Fan and F. Vercauteren, "Somewhat practical fully homomorphic encryption." *IACR Cryptology ePrint Archive*, vol. 2012, p. 144, 2012.

[23] K. Laine, R. Player, and H. Chen, "Simple encrypted arithmetic library - seal v2.2," Technical report, September, Tech. Rep., 2017.

[24] V. Lyubashevsky, C. Peikert, and O. Regev, "On ideal lattices and learning with errors over rings," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2010, pp. 1–23.