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

SpecMark: A Spectral Watermarking Framework for IP Protection of Speech Recognition Systems

Huili Chen[1]*, Bita Darvish*[2]*, Farinaz Koushanfar*[1]

[1]UC San Diego, USA
[2]Microsoft Research, USA

huc044@ucsd.edu, bita.rouhani@microsoft.com, farinaz@ucsd.edu

Abstract

Automatic Speech Recognition (ASR) systems are widely deployed in various applications due to their superior performance. However, obtaining a highly accurate ASR model is non-trivial since it requires the availability of a massive amount of proprietary training data and enormous computational resources. As such, pre-trained ASR models shall be considered as the intellectual property (IP) of the model designer and protected against copyright infringement attacks. In this paper, we propose SpecMark, the first spectral watermarking framework that seamlessly embeds a *watermark* (WM) in the spectrum of the ASR model for *ownership proof*. SpecMark identifies the significant frequency components of the model parameters and encodes the owner's WM in the corresponding spectrum region before sharing the model with end-users. The model builder can later extract the spectral WM to verify his ownership of the marked ASR system. We evaluate SpecMark's performance using DeepSpeech model with three different speech datasets. Empirical results corroborate that SpecMark incurs negligible overhead and preserves the recognition accuracy of the original system. Furthermore, SpecMark sustains diverse model modifications, including parameter pruning and transfer learning.

Index Terms: speech recognition, intellectual property protection, spectral watermarking

1. Introduction

Automatic Speech Recognition (ASR) is a technology that allows humans to interact with machines using their voices. The emergence of Deep Learning (DL) techniques has revolutionized ASR systems and enabled their commercialization. Voice assistants including Google Home, Amazon Alexa, Microsoft Cortana, and Apple Siri are examples of ASR's wide deployment [1, 2, 3, 4]. The success of modern ASR systems relies on the superior performance of the underlying DL models [5, 6]. While current researches in this field mainly focus on increasing the accuracy of ASR models, we take an orthogonal perspective to ASR applications and investigate the *copyright concerns* of the pre-trained models. Training a highly accurate ASR model is expensive since this process requires: (i) Access to an enormous amount of proprietary training dataset; (ii) Allocating extensive computing resources and time [7, 8]. As such, the resulting ASR system shall be considered as the *Intellectual Property (IP)* of the model developer and protected to preserve the competitive advantage of the owner.

Regularization is a typical approach to increase the generalization capability of a DL model to unseen datasets [9, 10]. Prior works have explored regularization and adapted digital watermarking for *ownership proof* of Deep Neural Networks (DNNs). Existing DNN watermarking techniques can be categorized into two types based on the deployment scenario of the DL model. A line of works assumes the model internals are known in the watermark (WM) extraction stage (i.e., *'white-box'* setting) and insert the WM by training the DL model with additional *regularization* loss terms [11, 12, 13, 14]. In this case, the WM is typically a binary sequence. For instance, [12] modulates the distribution of static weights to encode the WM information while DeepSigns [11] manipulates the distribution of dynamic activations to insert the WM.

Another line of research targets at the *'black-box'* scenario where the DL model is employed as a remote oracle (i.e., only the input-output behavior is known) [15, 16, 17, 18, 19]. The model owner generates a secret WM key set (i.e., input-output pairs) and uses it to finetune the model. Here, the WM takes the form of *statistically biased responses* and is encoded in the decision boundary of the model. Note that all of the above mentioned watermarking methods require expensive model retraining and are shown to be vulnerable to careful model disturbance [20]. Such limitations motivate us to design a more efficient and resilient watermarking scheme.

Contributions. In this paper, we propose SpecMark, the first systematic *model-level spectral watermarking* framework that protects the IP of contemporary ASR systems. SpecMark encodes the ownership information in the *spectrum characteristics* of the ASR model while preserving the task accuracy. More specifically, we propose to *spread* the watermark over multiple random subsets of the significant spectra components of the model parameters to ensure that SpecMark is *robust* and *secure*. Furthermore, our framework is highly *lightweight* since it embeds the WM strategically in the *spread spectrum (SS)* of the ASR model without re-training it. We validate the feasibility and robustness of SpecMark using the DeepSpeech v2 [6] on AN4, Command Voice, and LibriSpeech datasets. Our spectral watermarking technique is compatible with existing DL-based ASR systems and paves the way for safe and reliable deployment of ASR systems.

2. Related Work

Previous works have identified IP concerns of DNNs and adapted digital watermarking techniques for ownership authentication in the DL domain. We categorize existing methods into two types based on the application scenarios of the DL model. We introduce each type in detail as follows.

White-box Watermarking. In the white-box setting,the pre-trained DL model for the intended task (computer vision, speech recognition, etc.) is shared with the end-users. This means that model internals including weight parameters and activation maps are publicly accessible. Such a deployment scenario is common with the increasing trend of knowledge exchange among the research community. [12] takes the first step of DNN watermarking and develops a customized regularization term to embed the watermark in the weight distribution

http://dx.doi.org/10.21437/Interspeech.2020-2787

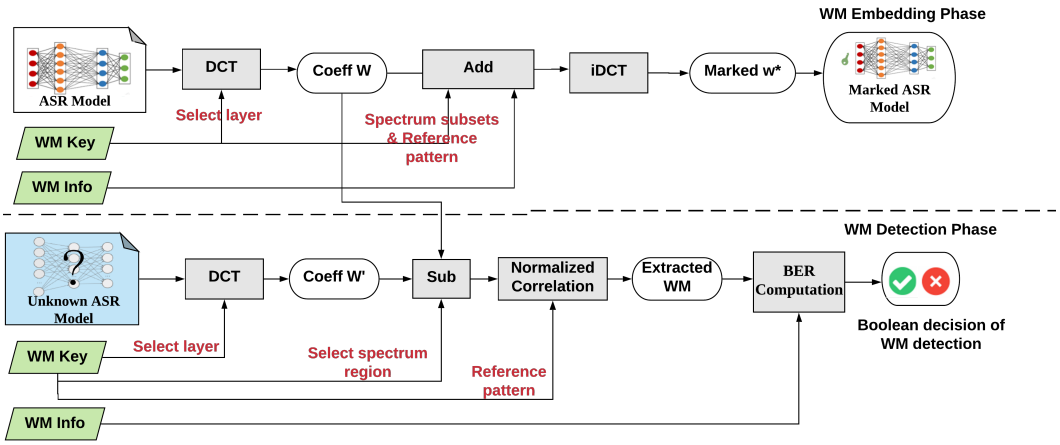

Figure 1: *Global workflow of SpecMark watermarking framework for ASR systems.*

of the selected DL layer. To improve the security and robustness, DeepSigns [11] proposes to insert the WM in the distribution of dynamic activations corresponding to the secret key input. DeepMarks [13] uses weight regularization and incorporates anti-collusion codes for WM design to enhance the watermark's resistance against averaging attacks.

Black-box Watermarking. In the black-box setting, the pre-trained DL model is employed as a remote service where the customer sends his data to the cloud server and receives the corresponding output. Since the DL model is only available as an oracle, prior works suggest to craft secret input-output pairs as the WM. To insert the WM in the model's decision boundary, the WM key set is used to finetune the model. As an example, [17] proposes to craft adversarial samples as the WM set, which results in high false alarm rates due to the transferability of adversarial examples. To resolve the issue, DeepSigns [11] generates random inputs and random labels as the WM key set.

Existing DL watermarking techniques have the following constraints: *(i) Application domain.* All of the above mentioned watermarking papers demonstrate their methods on image classification tasks. However, the intrinsic time-evolving nature and the representation form of speech signals distinguish ASR from image tasks. Such a discrepancy might render the watermarking techniques less effective or invalid for ASR systems; *(ii) High WM embedding overhead.* Present DL watermarking primitives embeds the WM via model re-training, which might be prohibitively costly; *(iii) Robustness.* Current watermarking schemes are susceptible to careful model disturbance such as transfer learning [20]. To address the above limitations, we propose SpecMark, the first practical and resilient model-level watermarking framework that is suitable for ASR systems.

3. Problem Statement

We define the problem of ASR model watermarking in this section. SpecMark assumes a *white-box* scenario where the model internals are known to the public. We formulate model-level watermarking as a *one-time, post-training* step where the objective is to embed a WM (a binary sequence in our work) in the parameter distribution of the ASR model. To be *practical* and effective in real-world ASR systems, the watermarking technique shall satisfy a set of criteria. We summarize these fundamental requirements in Table 1 and present a quantitative assessment of SpecMark's performance in Sec. 5.

Potential Attacks. The model owner inserts a secret watermark in his trained ASR model and shares the marked variant with the public. However, the marked model might undergo unintentional or deliberate model modifications in a practical deployment setting. The robustness criterion in Table 1 requires that the WM shall be resistant to potential disruptions and remains detectable. We consider three types of model disturbance attacks: *(i) Parameter pruning:* Parameters with small magnitudes can be zeroed out for computation savings without significant accuracy degradation [21, 22, 23]; *(ii) Model fine-tuning:* The converged model can be fine-tuned to find better local optima [24, 25, 26]; *(iii) Transfer learning:* A pre-trained model might be re-trained on a new dataset for the intended task [27, 28, 29]. We corroborate the robustness of SpecMark spread spectrum watermarking against these attacks in Sec. 5.

Table 1: *Requirements for an effective watermarking method of speech recognition systems.*

Requirement	Definition
Fidelity	Preserve the functionality of the original model.
Robustness	WM sustains possible model modifications.
Efficiency	Low overhead for WM embedding and detection.
Reliability	High detection rates of the embedded WM.
Integrity	Low false positive rates of WM detection.
Security	WM carrier is difficult to identify.

4. SpecMark Methodology

Figure 1 shows the global flow of SpecMark framework. From the high-level overview, SpecMark takes the pre-trained ASR model and a set of secret WM keys as the inputs. The marked variant of the ASR model is returned as the output. Our watermarking framework consists of two main phases: offline WM embedding and online WM detection. We detail the procedures of each step as follows.

4.1. Spectral WM Embedding

SpecMark *spreads* the WM information in the *significant spectrum components* of the target ASR model. Such an embedding mechanism features two advantages: *(i) Security*: Spreading the WM information over many frequency bins ensures that the energy change on a single bin is small and undetectable. The insertion location and content of the WM are only known to the owner, making it difficult to find out by the attacker using random guesses. *(ii) Robustness*: SpecMark's WM is encoded in the important frequency regions of the ASR model. Since feasible model modifications have to leave the significant spectra components intact to maintain high accuracy, the attacker can-

not remove our WM without performance degradation.

We define the WM as a binary bit sequence $\mathbf{b}$ of length T where $b_k = \{-1, +1\}, k = 1, ..., T$. To provide security guarantee, SpecMark's WM key has three components: (i) The layer position (denoted by l) whose parameters are selected to carry the WM information; (ii) The secret random seeds ($\mathbf{s}$) that are used to determine the frequency bins modulated by the WM; (iii) The secret reference pattern matrix $\mathbf{U}_{T\times M}$ where T is the length of the WM sequence and M is the number of frequency bins controlled by each WM bit. The k^{th} row of $\mathbf{U}$ is used as the reference vector $\mathbf{u_k}$ to carry the WM bit b_k. Note that elements in each row of $\mathbf{U}$ have equal probabilities of taking two values: $u_{i,j} = \{-\sigma_u, +\sigma_u\}$. We detail each step of SpecMark's WM embedding stage shown in Figure 1 below.

▪ **Identify Significant Spectra Components.** Given the layer position l in the WM key, SpecMark performs *DCT transformation* on the corresponding weight parameter w and obtains the frequency coefficients $\mathcal{W} = DCT(w)$. Since large values are less sensitive to additive alternations than small values, we select the top N largest elements of $\mathcal{W}$ as the tentative WM insertion locations and denote the resulting index set as I_N. Note that $M \ll N$ such that the spectra components controlled by each WM bit do not overlap with each other.

▪ **Encode WM in Random Spectra Subsets.** To enhance watermarking security, SpecMark embeds each WM bit in a random subset of spectra components with the highest values (found by I_N). The insertion location I_k (with size M) for the bit b_k is determined by Eq. (1) where s_k is the random seed from the WM key. To make the element-wise addition of frequency components feasible, we then use I_k^c to zero-pad the secret reference vector $\mathbf{u_k}$ as shown in Eq. (2). Here, I_k^c is the complement set of I_k where the whole set is the index range of $\mathcal{W}$. The padded variant $\tilde{\mathbf{u_k}}$ takes the corresponding value from $\mathbf{u_k}$ only when its current index exists in I_k. Finally, the entire WM sequence $\mathbf{b}$ is embedded in the significant part of the DCT coefficients $\mathcal{W}$ using Eq. (3).

$$I_k = RandomSelect(I_N, M, s_k), \tag{1}$$

$$\tilde{\mathbf{u_k}} = ZeroPad(\mathbf{u_k}, I_k^c), \tag{2}$$

$$\mathcal{W}^* = \mathcal{W} + \sum_{k=1}^{T} b_k \tilde{\mathbf{u_k}}, \tag{3}$$

▪ **Perform Inverse Frequency Transformation.** After embedding the WM in the selected bins of the important spectrum of the ASR model, we convert the resulting frequency map back to the spatial domain using inverse DCT: $w^* = iDCT(\mathcal{W}^*)$. The original weight parameter w of the secret layer l is replaced with w^* to obtain the marked ASR model.

4.2. Spectral WM Detection

In the online detection phase, the model owner queries the unknown ASR system and obtains its internal weights. Since the owner knows the WM insertion locations and content, he can concentrate the 'weak' WM signals spread over the particular frequency bins and extract the WM for authorship proof. We detail each step of WM detection shown in Figure 1 below.

▪ **Transform Queried Data to Frequency Domain.** Given the WM key, the model owner performs DCT on the weight parameter of the layer l of the queried model $\mathcal{W}^{'} = DCT(w^{'})$.

▪ **Compute Normalized Correlation.** As the developer of the original ASR system, the model owner has the DCT values $\mathcal{W}$ of the unmarked weights w. As such, he can compute the spectral difference $\Delta\mathcal{W}$ between the queried weight and the unmarked one in the DCT domain using Eq. (4). Then, the normalized correlation between $\Delta\mathcal{W}$ and each reference vector $\mathbf{u_k}$ is computed using Eq. (5). Note that $\|\mathbf{u_k}\| = \sigma_u^2$, since elements in $\mathbf{u_k}$ can only take the value of $-\sigma_u$ or $+\sigma_u$.

$$\Delta\mathcal{W} = \mathcal{W}^{'} - \mathcal{W}, \tag{4}$$

$$r_k^{'} = \frac{\Delta\mathcal{W} \cdot \tilde{\mathbf{u_k}}}{\|\tilde{\mathbf{u_k}}\|}. \tag{5}$$

▪ **Determine WM Existence.** After computing the normalized correlation r_k ($k = 1, .., T$) individually, the corresponding binary WM bit b_k is extracted by taking the sign of the correlation statistics as shown in Eq. (6). Finally, we compute the Bit Error Rate (BER) between the ground-truth WM sequence $\mathbf{b}$ and the extracted one $\mathbf{b}^{'}$. SpecMark's WM is successfully detected for ownership authentication only when $BER = 0$.

$$b_k^{'} = sign(r_k^{'}). \tag{6}$$

5. Evaluations

We present a comprehensive assessment of SpecMark's performance according to the watermarking requirements discussed in Table 1. The results are summarized in this section.

Experimental Setup. We demonstrate the effectiveness of SpecMark using the DeepSpeech v2 model [6] and three different speech datasets: AN4, Command Voice, as well as LibriSpeech [30]. To implement SpecMark's spread spectrum watermarking (detailed in Sec. 4.1), we use the following configuration: WM sequence length $T = 16$, candidate range of significant spectra components $N = 5000$, number of frequency bins controlled by each WM bit $M = 20$, and reference strength $\sigma_u = 0.5$. The hidden-hidden weights of the third LSTM layer of DeepSpeech is selected to carry the WM. Similar results are obtained when other layers are used for SpecMark's watermarking. We emphasize that *no model re-training is required* by SpecMark to embed the WM, making our framework lightweight. We use the same hyper-parameters (e.g., learning rate, batch size, and optimization level) as [30] for three WM removal attacks. Our evaluations are performed on Nvidia Titan Xp with 12 GiB memory. We repeat each set of experiments for 10 runs and report the average values in the following section.

5.1. Fidelity and Efficiency

Recall that fidelity requires the watermarking technique to preserve the accuracy of the pre-trained model. Table 2 summarizes the performance comparison results of the ASR system before and after SpecMark's WM embedding. The last two rows show the Frobenius norm of the weight perturbation introduced by WM insertion in the spatial and the DCT domain, respectively. One can see that SpecMark's spread spectrum watermarking primitive does **not impact the accuracy** of the original model, thus respects the fidelity criterion. This is due to the fact that our framework induces negligible disturbance on the weight parameters (small $\|\Delta\mathbf{w}\|$ in Table 2).

Table 2: *Fidelity evaluation of SpecMark. The WER and CER of the pre-trained baseline model and the watermarked variant are compared across different datasets.*

Datasets	**AN4**		**Command Voice**		**LibriSpeech**	
Models	**Baseline**	**Marked**	**Baseline**	**Marked**	**Baseline**	**Marked**
WER (%)	11.38	11.38	26.72	26.72	18.09	18.09
CER (%)	6.81	6.81	11.63	11.63	7.32	7.32
$\|\Delta\mathbf{w}\|$	0.20		0.20		0.16	
$\|\Delta\mathbf{W}\|$	9.11		9.11		7.07	

As for efficiency, we analyze the *runtime overhead* of SpecMark's WM embedding and detection procedure. According to the mechanism of SpecMark outlined in Sec. 4, we can see that

SpecMark has a *fixed computational overhead* for a specific watermarking configuration and a given target ASR system. Such *independence* of SpecMark's overhead with the dataset dimensionality suggests that our framework is **scalable** to large ASR tasks. In our experiments, the WM embedding and detection time is 97.67 and 10.98 millisecond for all three datasets, respectively. Compared with existing DL watermarking techniques [11, 12, 13], SpecMark features the highest **efficiency** since no model re-training is required.

5.2. Robustness

We discuss three possible attack scenarios in Sec. 3: parameter pruning, model fine-tuning, and transfer learning. In the following of this section, we validate SpecMark's robustness against these attacks with empirical results.

5.2.1. Robustness against Parameter Pruning

We perform standard parameter pruning (i.e., zero-out elements with the smallest magnitudes [21]) on all Convolutional and LSTM layers of the marked ASR model. Acceleration-oriented pruning pipeline conducts model re-training to compensate for accuracy loss induced by pruning. In our case, the attacker intends to use pruning for WM removal. It is very unlikely that the attacker has the original training data and the computing power to perform model re-training (otherwise he has less incentive to steal the ASR model.) As such, we measure the test accuracy and BER of WM detection of the pruned model without re-training. Figure 2 demonstrates SpecMark's robustness against parameter pruning on LibriSpeech dataset. One can observe that the BER of SpecMark's WM is *less sensitive* to parameter pruning compared to the accuracy metrics (i.e., WER and CER). As such, the adversary cannot remove the WM by excessive pruning while acquiring a functional ASR model. In our experiments, SpecMark spectral watermarking tolerates up to 99%, 90%, and 90% parameter pruning on AN4, Command Voice, and LibriSpeech datasets, respectively.

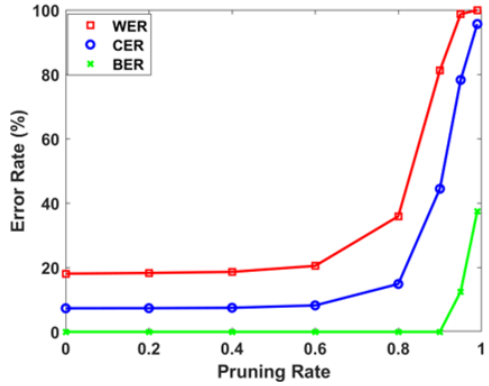

Figure 2: *SpecMark's robustness against parameter pruning.*

5.2.2. Robustness against Transfer Learning

Transfer learning is a popular practice that leverages the features extracted by a pre-trained DL model for a new task [27, 28]. More specifically, the user performs model re-training on his new dataset instead of training a model from scratch. In our robustness evaluation, the DeepSpeech model is first pre-trained on LibriSpeech dataset and marked by SpecMark. The transfer learning attack is then performed by re-training the marked model on AN4 dataset using the same configurations in [30]. Figure 3 shows the test accuracy of the marked DeepSpeech model on the new dataset (AN4) and the BER of WM detection during the transfer learning process. We can see that SpecMark's SS WM remains detectable (i.e., BER=0) even if the marked ASR model undergoes transfer learning. This **transferability** of SpecMark's WM makes it suitable for reliable technology exchange in the speech recognition domain.

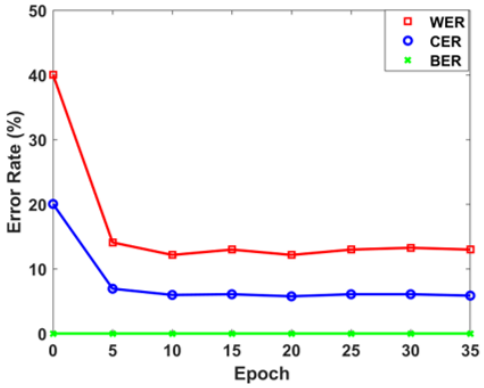

Figure 3: *SpecMark's robustness against transfer learning.*

5.2.3. Robustness against Model Fine-tuning

The nature of model fine-tuning determines that it introduces a *smaller* amount of perturbation to the marked weights compared to parameter pruning and transfer learning. Our evaluation results show that SpecMark still yields *zero BER* for the fine-tuned marked model across all three datasets, thus is **resilient** against model fine-tuning attacks. The detailed results are not shown here for simplicity.

5.3. Integrity

Recall that integrity requires the WM detection process to yield small false positive rates (see Table 1). This property is important since falsely claiming the ownership of an ASR model might lead to law disputes. To assess the integrity of SpecMark, we extract the watermark from unmarked ASR models following the procedures in Sec. 4.2. Table 3 shows the integrity evaluation results on LibriSpeech dataset while similar results are obtained on the other two datasets. 'Unmarked1' and 'Unmarked2' are models trained on the *same dataset* as the marked one (LibriSpeech in this case). 'Unmarked3' and 'Unmarked4' are models trained on different datasets (AN4 and Command Voice, respectively). We can see that SpecMark has **no false alarms** since the BER is non-zero for each unmarked model (regardless of the underlying training data). As such, our watermarking framework respects the integrity criterion.

Table 3: *Integrity evaluation of SpecMark on four different unmarked DeepSpeech models.*

Models	Marked	Unmarked1	Unmarked2	Unmarked3	Unmarked4
BER	0.	1.	0.5625	0.5	0.6875

6. Conclusion

In this paper, we propose SpecMark, the first spectral watermarking framework for speech recognition systems. SpecMark tackles an important and timely problem of Intellectual Property protection for ASR systems. For the first time, SpecMark demonstrates a lightweight, secure, and robust watermarking primitive that is suitable for ASR applications. Our proposed framework formulates model-level watermarking as a one-time, post-processing step and leverages spread spectrum watermarking to address the problem. SpecMark can be easily integrated within contemporary DL-based ASR systems without impacting their accuracy on the intended tasks. Experimental results on DeepSpeech model and various datasets corroborate that SpecMark respects the essential requirements for an effective watermarking approach.

7. References

[1] B. Li, T. N. Sainath, A. Narayanan, J. Caroselli, M. Bacchiani, A. Misra, I. Shafran, H. Sak, G. Pundak, K. K. Chin *et al.*, "Acoustic modeling for google home." in *Interspeech*, 2017, pp. 399–403.

[2] V. Kepuska and G. Bohouta, "Next-generation of virtual personal assistants (microsoft cortana, apple siri, amazon alexa and google home)," in *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*. IEEE, 2018, pp. 99–103.

[3] J. Hauswald, M. A. Laurenzano, Y. Zhang, C. Li, A. Rovinski, A. Khurana, R. G. Dreslinski, T. Mudge, V. Petrucci, L. Tang *et al.*, "Sirius: An open end-to-end voice and vision personal assistant and its implications for future warehouse scale computers," in *Proceedings of the Twentieth International Conference on Architectural Support for Programming Languages and Operating Systems*, 2015, pp. 223–238.

[4] M. B. Hoy, "Alexa, siri, cortana, and more: an introduction to voice assistants," *Medical reference services quarterly*, vol. 37, no. 1, pp. 81–88, 2018.

[5] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates *et al.*, "Deep speech: Scaling up end-to-end speech recognition," *arXiv preprint arXiv:1412.5567*, 2014.

[6] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," in *International conference on machine learning*, 2016, pp. 173–182.

[7] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," in *Advances in neural information processing systems*, 2016, pp. 901–909.

[8] A. Jain, A. Phanishayee, J. Mars, L. Tang, and G. Pekhimenko, "Gist: Efficient data encoding for deep neural network training," in *2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 2018, pp. 776–789.

[9] S. Gu, Y. Hou, L. Zhang, and Y. Zhang, "Regularizing deep neural networks with an ensemble-based decorrelation method." in *IJCAI*, 2018, pp. 2177–2183.

[10] D. Yu, K. Yao, H. Su, G. Li, and F. Seide, "Kl-divergence regularized deep neural network adaptation for improved large vocabulary speech recognition," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 7893–7897.

[11] B. Darvish Rouhani, H. Chen, and F. Koushanfar, "Deepsigns: An end-to-end watermarking framework for ownership protection of deep neural networks," in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, 2019, pp. 485–497.

[12] Y. Uchida, Y. Nagai, S. Sakazawa, and S. Satoh, "Embedding watermarks into deep neural networks," in *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval*, 2017, pp. 269–277.

[13] H. Chen, B. D. Rohani, and F. Koushanfar, "Deepmarks: A digital fingerprinting framework for deep neural networks," *arXiv preprint arXiv:1804.03648*, 2018.

[14] H. Chen, C. Fu, B. D. Rouhani, J. Zhao, and F. Koushanfar, "Deepattest: an end-to-end attestation framework for deep neural networks," in *Proceedings of the 46th International Symposium on Computer Architecture*, 2019, pp. 487–498.

[15] H. Chen, B. D. Rouhani, and F. Koushanfar, "Blackmarks: Blackbox multibit watermarking for deep neural networks," *arXiv preprint arXiv:1904.00344*, 2019.

[16] Y. Adi, C. Baum, M. Cisse, B. Pinkas, and J. Keshet, "Turning your weakness into a strength: Watermarking deep neural networks by backdooring," in *27th {USENIX} Security Symposium ({USENIX} Security 18)*, 2018, pp. 1615–1631.

[17] E. Le Merrer, P. Perez, and G. Trédan, "Adversarial frontier stitching for remote neural network watermarking," *Neural Computing and Applications*, pp. 1–12, 2019.

[18] J. Zhang, Z. Gu, J. Jang, H. Wu, M. P. Stoecklin, H. Huang, and I. Molloy, "Protecting intellectual property of deep neural networks with watermarking," in *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*, 2018, pp. 159–172.

[19] J. Guo and M. Potkonjak, "Evolutionary trigger set generation for dnn black-box watermarking," *arXiv preprint arXiv:1906.04411*, 2019.

[20] X. Chen, W. Wang, C. Bender, Y. Ding, R. Jia, B. Li, and D. Song, "Refit: a unified watermark removal framework for deep learning systems with limited data," *arXiv preprint arXiv:1911.07205*, 2019.

[21] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2015.

[22] S. Han, J. Kang, H. Mao, Y. Hu, X. Li, Y. Li, D. Xie, H. Luo, S. Yao, Y. Wang *et al.*, "Ese: Efficient speech recognition engine with sparse lstm on fpga," in *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, 2017, pp. 75–84.

[23] J.-H. Luo, J. Wu, and W. Lin, "Thinet: A filter level pruning method for deep neural network compression," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 5058–5066.

[24] N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway, and J. Liang, "Convolutional neural networks for medical image analysis: Full training or fine tuning?" *IEEE transactions on medical imaging*, vol. 35, no. 5, pp. 1299–1312, 2016.

[25] A. Nagabandi, G. Kahn, R. S. Fearing, and S. Levine, "Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 7559–7566.

[26] J. Chi, E. Walia, P. Babyn, J. Wang, G. Groot, and M. Eramian, "Thyroid nodule classification in ultrasound images by fine-tuning deep convolutional neural network," *Journal of digital imaging*, vol. 30, no. 4, pp. 477–486, 2017.

[27] D. C. Cireşan, U. Meier, and J. Schmidhuber, "Transfer learning for latin and chinese characters with deep neural networks," in *The 2012 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2012, pp. 1–6.

[28] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A survey on deep transfer learning," in *International conference on artificial neural networks*. Springer, 2018, pp. 270–279.

[29] H.-C. Shin, H. R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, and R. M. Summers, "Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning," *IEEE transactions on medical imaging*, vol. 35, no. 5, pp. 1285–1298, 2016.

[30] S. Naren, "Speech recognition using deepspeech2." https://github.com/SeanNaren/deepspeech.pytorch, 2020.