Lightweight Protection for Privacy in Offloaded Speech Understanding

Dongqi Cai
Beiyou Shenzhen Institude

Shangguang Wang
Beiyou Shenzhen Institude

Zeling Zhang
Beiyou Shenzhen Institude

Felix Xiaozhu Lin
University of Virginia

Mengwei Xu
Beiyou Shenzhen Institude

ABSTRACT

Speech is a common input method for mobile embedded devices, but cloud-based speech recognition systems pose privacy risks. Disentanglement-based encoders, designed to safeguard user privacy by filtering sensitive information from speech signals, unfortunately require substantial memory and computational resources, which limits their use in less powerful devices. To overcome this, we introduce a novel system, XXX, optimized for such devices. XXX is built on the insight that speech understanding primarily relies on understanding the entire utterance's long-term dependencies, while privacy concerns are often linked to short-term details. Therefore, XXX focuses on selectively masking these short-term elements, preserving the quality of long-term speech understanding. The core of XXX is an innovative differential mask generator, grounded in interpretable learning, which fine-tunes the masking process. We tested XXX on the STM32H7 microcontroller, assessing its performance in various potential attack scenarios. The results show that XXX maintains speech understanding accuracy and privacy at levels comparable to existing encoders, but with a significant improvement in efficiency, achieving up to 53.3× faster processing and a 134.1× smaller memory footprint.

1 INTRODUCTION

Privacy concern for cloud speech service The volume of speech data uploaded to the cloud for spoken language understanding (SLU) is steadily increasing [1, 2, 12], particularly in ubiquitous wimpy devices where textual input is inconvenient [3, 17, 41], e.g., home automation devices [32], smartwatches [37], telehealth sensors [22] and smart factory sensors [29] . However, exposing raw speech signal to the cloud raises privacy concerns [42]. It was revealed that contractors regularly listened to confidential details in Siri recordings to improve its accuracy [4]. This included private discussions, medical information, and even intimate moments.

There are many aspects of potential privacy leakage in cloud-based SLU. Among them: biometric or contextual privacy leakage have been well studied and somewhat solved by removing information relevant to such tasks without compromising the SLU accuracy [18, 35]; transcript protection (especially sensitive entities) is more challenging since it is deeply entangled with the SLU task itself. As shown in Figure 1, this paper focus on ensuring that cloud-based systems could efficiently classify the intent of SLU task (e.g., scheduling appointments or controlling home devices) while refraining from identifying the concrete entities (e.g., unintended names or passwords) in the spoken utterance, i.e., high word error rate (WER) of Automatic Speech Recognition (ASR) task. This is also a setting commonly used in speech privacy protection [9, 15, 16, 42, 44].

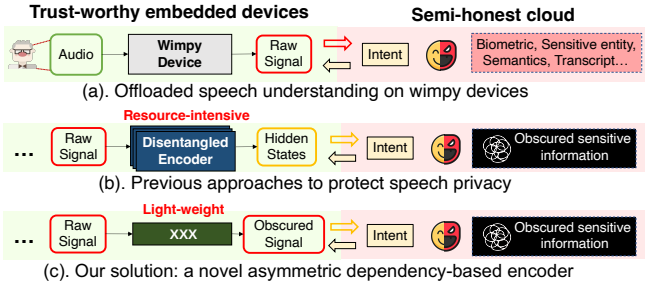


Figure 1: Illustration of offloaded speech understanding on wimpy devices and its privacy protection.

Prior approaches A prevalent method for private speech processing is employing *encoders*[1] based on disentanglement representation learning [9, 28, 34, 44], as illustrated in Figure 1(b). Those encoders extract the speech representations using pre-trained acoustic models, e.g., wav2vec [9, 40], conformer [26, 34] and Preformer [20, 44]. Furthermore, they promote representation disentanglement through adversarial training [25]. For example, PPSLU [44] uses a 12-layer transformer-based Preformer as its encoder.

As a result, disentanglement-based encoders still demand considerable computational resources, often exceeding tens of GFLOPs, to achieve effective disentanglement [11]. They are also memory-intensive, often comprising tens of millions of parameters. Consequently, they are unsuitable for

[1]Note that these encoders are not specifically transformer encoders; rather, they can be implemented using any NNs to encode speech signals.

embedded devices with limited memory and are prone to be the victim of mobile OSes' low memory killer mechanisms [10]. Moreover, it takes time-consuming adversarial training to disentangle the encoded representation for each specific SLU task. This aspect limits the flexibility and scalability for emerging SLU tasks. More motivating details will be presented in §2.2.

In this paper, we aim to achieve real-time, privacy-preserving speech understanding task offloading on wimpy devices like STM32H7 microcontroller [5] with only 1MB RAM. This goal necessitates a novel encoder design that must be both lightweight and effective in filtering out sensitive information on such devices, as illustrated in Figure 1(c).

Our solution: XXX We therefore present XXX, a **SI**mp**L**e **ENC**od**E**r designed for efficient privacy-preserving SLU offloading, as depicted in Figure 3. It is based on the *asymmetric dependency* observation: SLU intent extraction (e.g., scenario identification) typically requires only long-term dependency knowledge across the entire utterance, while ASR task (e.g., recognizing individual words or phrases) needs short-term dependency, as confirmed by our experiments in §3.1. Based on it, XXX strategically partitions the utterance into several segments, selectively masking out the majority to enhance privacy by obscuring short-term details, without significantly damaging the long-term dependencies. The processed audio waveform is then transmitted to the cloud for SLU intent analysis. Additionally, we integrate a differential mask generator, inspired by interpretable learning methods [19], to optimize performance by automatically identifying how many and which segments to mask.

Results We deploy XXX on the STM32H7 microcontroller [5] and assess its performance using the SLURP dataset [13] in both black-box and white-box attack environments. XXX achieves 81.2% intent classification accuracy on SLURP, surpassing previous privacy-preserving SLU systems by up to 8.3%. Regarding privacy protection, XXX offers comparable security to earlier systems, with a word error rate of up to 81.6% and an entity error rate of 90.7% under malicious ASR attacks. Even against white-box attacks, where attackers is strongly assumed to have the same encoder structure and weights as XXX, plus partial malicious client data, XXX maintains 67.3% word error rate and 64.3% entity error rate. Additionally, XXX proves to be resource-efficient and feasible for wimpy devices, using only 394.9KB of memory and taking just 912.0ms to encode a 4-second speech signal. Integrated with RPI-4B for a fair comparison, XXX uses up to 134.1× less memory and operates up to 53.3× faster than prior systems. The accuracy of XXX is only 7% lower than unprotected SLU systems.

Contribution We have made the following contributions.

- Based on the observation of asymmetric dependency between SLU and ASR tasks, we propose XXX, a simple yet effective encoder system for conducting privacy-preserving SLU.
- We are the first to retrofit interpretable learning methods to automatically configure the masking process for a better balance between privacy and utility in speech understanding tasks.
- We evaluate XXX on a wimpy MCU and demonstrate its effectiveness under various attack scenarios.

2 RELATED WORK AND BACKGROUND

2.1 Privacy-preserving SLU

Spoken Language Understanding (SLU) is a critical component of modern voice-activated systems, responsible for interpreting human speech and translating it into structured, actionable commands. For instance, when a user says, "Set a meeting for tomorrow at 10 AM," the SLU system might map this to a structured intent such as `{scenario: Calendar, action: Create_entry}`.

Evolution of SLU Systems The evolution of SLU systems has seen a shift from traditional two-component systems, comprising ASR and Natural Language Understanding (NLU), to modern end-to-end neural networks [27, 39]. These advanced systems bypass the intermediate textual representation and directly map speech signals to their semantic meaning, enhancing efficiency and reducing error propagation. A typical end-to-end SLU model features an encoder, often with convolution and attention-based elements, and a decoder, including a transformer decoder and a connectionist temporal classification decoder. Many SLU systems incorporate encoders from pre-trained ASR models like HuBERT [45], replacing the original ASR decoder with one tailored for SLU tasks.

Threat Model Our threat model aligns with prior work [9, 44] where users (the victims) actively offloads their audio data to the cloud server (the adversary) for intended SLU tasks. Upon receiving the data, the adversary may employ automatic speech recognition to transcribe the audio and identify private entities [15, 16, 42]. Note that the transcriptions are often exceedingly detailed, containing much more information than the users intend to disclose. The goal of this paper is to ensure that the victims can reliably obtain the predefined SLU intent from the adversary, while preserving the adversary from discerning sensitive details or private entities in the transcript.

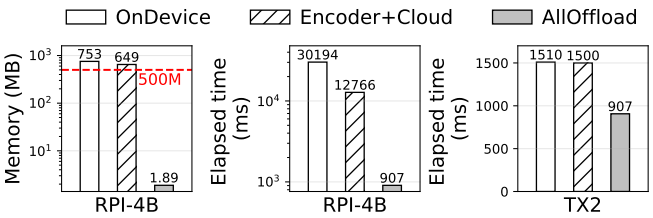


Figure 2: Cost of disentangling-based encoders [44] for a 4-second audio inference.

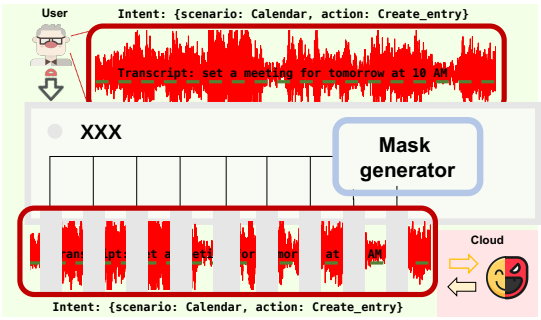


Figure 3: XXX overview. Red hard line represents the long-term dependency, while the green dotted line represents the short-term dependency.

2.2 Inefficiency of Existing Approaches

Privacy-preserving methods Crypto-based approaches, such as HE [48] and MPC [24], have been proposed to provide encrypted computation. Unfortunately, they are technically slow and thus impractical for deployment on wimpy audio devices due to the significant increase in computation and communication complexity. For example, MPC-based PUMA [21] takes 5 minutes to complete one token inference, which is far too slow for real-time. Voice conversion is another method to protect speech content. Pr$\varepsilon\varepsilon$ch [8] integrates voice conversion with GPT-based generated noise protect privacy, but it is far from feasible for deployment on wimpy devices. Traditional peripheral devices, such as ultrasonic microphone jammers (UMJ), are designed to obscure raw speech by inserting non-linearity noise, thereby preventing illegal eavesdropping[15, 23]; however, they also corrupt speech semantics as well. A emerging and prevailing strategy is disentangling-based encoders [9, 28, 44]; they aim to create a disentangled and hierarchical representation of the speech signal devoid of sensitive data. But we reveal their performance issue next.

We conduct preliminary experiments to measure the resource consumption of the disentangling-based encoder of a pre-trained SLU model on a Raspberry Pi 4B (RPI-4B) [6] and Jetson TX2 (TX2) [7]. Our key observation is that disentangling-based privacy-preserving SLU system is too resource-intensive for practical deployment. As illustrated in Figure 2, a disentanglement encoder consumes 648.7MB memory and 12.8s for complete one inference on RPI-4B. Even in the strong TX2 with GPU, the encoder still takes 593.0ms to complete one inference. Considering the network latency, the end-to-end latency of the disentangling-based SLU offloading system only saves 0.7% wall-clock time compared to the no offloading OnDevice inference, with a similar memory footprint over 500M.

Implications Disentangling-based encoders is slow and memory-intensive due to the complex encoder structure designed to separate sensitive information from the speech signal. Given the limited resource of wimpy devices, it is not practical for common privacy-preserving SLU scenarios. To enable practical privacy-preserving SLU, the encoder structure and the inference process need to be simplified.

3 XXX DESIGN

3.1 System Design and Rationales

We introduce XXX to efficiently scrub raw audio for privacy-preserving SLU, as depicted in Figure 3. The key idea of XXX is simple and novel: it masks out a portion of audio segments before sending them to the cloud for SLU tasks. This design is based on an unique observation shown in Figure 4(c): when a portion of audio segments is masked out, the ASR model becomes incapable to recognize the phonemes in the masked frames, while the SLU model can still recognize the intent.

The rationales behind XXX Why is XXX able to protect the sensitive entity privacy while maintaining SLU accuracy? This capability is rooted in the asymmetrical dependency between the ASR and SLU task.

Speech is composed of many meta phonemes, and the generation of a single meta phoneme depends on its adjacent frame [42]. *Dependency* is defined as the length of frame that a model's output depends on. Figure 4 shows each phoneme is mainly dependent on a few frames, indicating short-term dependency. This phenomenon is referred to as "peaky behavior" in the ASR literature [47]. In contrast, an SLU model utilizes an attention-based decoder [45] to capture the relationship between the entire utterance and the intent, implying that the intent is long-term dependent on the whole utterance.

Formally, XXX is a simple encoder based on asymmetrical dependency-based masking. This simple masking encoder is defined as: $\hat{x} = x \odot \mathbb{Z}$, where x is the input audio signal, $\odot$ represents the element-wise multiplication, $\hat{x}$ is the masked audio signal and $\mathbb{Z}$ is the binary masking vector with the same dimension as x. $\mathbb{Z}$ consists of k uniform portion, with all 0s or 1s in one portion to mask-out or preserve the complete

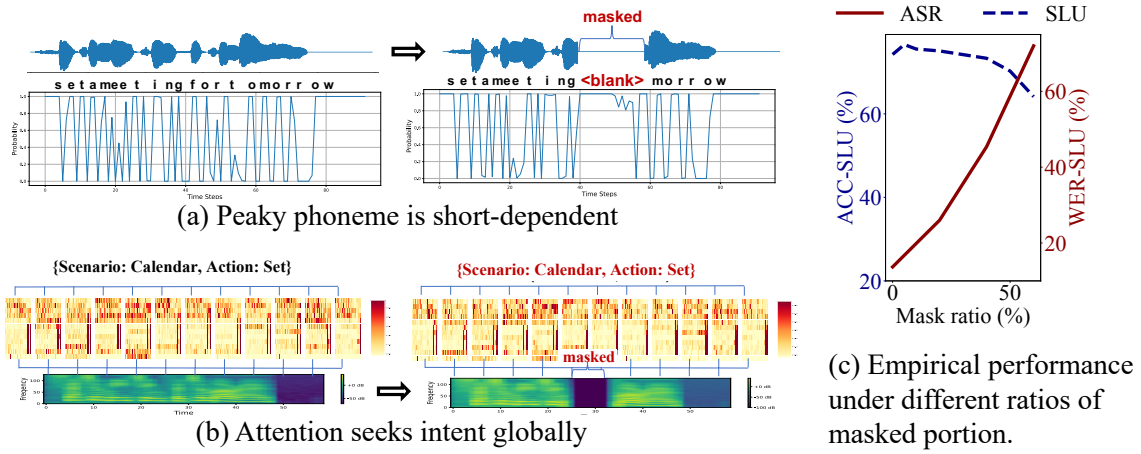


Figure 4: Foundation of XXX: asymmetrical dependency. (a). ASR task is short-term dependent on the peaky phoneme probability. (b). SLU task is long-term dependent on knowledge from the whole utterance. (c). Empirical results.

adjacent frames, respectively. This simple encoder forms the basis of XXX's efficiency and privacy-preservation capacity, enabling secure offloading of speech understanding tasks on wimpy devices.

The configuration challenges: Figure 4(c) demonstrates that the ratio of masked portion plays a crucial role in balancing the privacy (ASR-WER) and utility (SLU-ACC). Currently, XXX employs a trivial masking mechanism, necessitating clients to undertake a time-intensive hyper-parameter adjustment about the extent and location of masking. Incorrect masking configurations can result in significant loss of global long-term dependency, negatively affecting SLU accuracy, or insufficient masking of sensitive information, thus compromising privacy. Therefore, we face critical questions: how many and which portions should be masked?

3.2 Online Configurator for XXX

To address these challenges, we derive a differential mask generator from the interpretable learning [19] as a online configurator for XXX. This automatically generate the masking vector $\mathbb{Z}$. The mask generator is trained to identify how many and which portions to mask, optimizing the privacy-utility balance.

Differentiable mask generator The configurator model aims to minimize the discrepancy between masked and original output by generating a mask $\mathbb{Z}$ Formally, we define the number of unmasked portions as $\mathcal{L}_0$ loss:

$$\mathcal{L}_0(\phi, x) = \sum_{i=1}^{n} \mathbf{1}_{[\mathbb{R}_{\neq 0}]}(\mathbb{Z}_i) \tag{1}$$

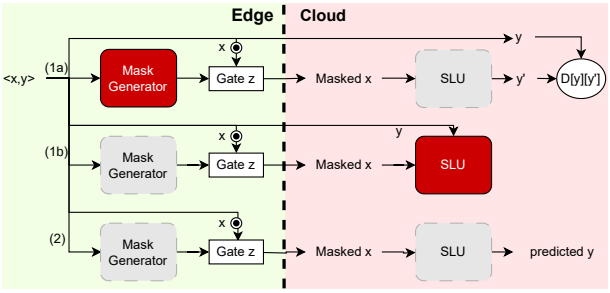


Figure 5: XXX workflow. (1) *Offline phase*: (1a) Training mask generator and (1b) adapting cloud SLU model to it; (2) *Online phase*: Conducting could inference with the masked x. Only masked input audio x and insensitive intent label y are exposed to the cloud.

where ϕ is the mask generator, $\mathbf{1}(\cdot)$ is the indicator function. We minimize $\mathcal{L}_0$ for dataset $\mathcal{D}$, ensuring that predictions from masked inputs resemble those from the origin model:

$$\min_{\phi} \sum_{x \in \mathcal{D}} \mathcal{L}_0(\phi, x) \tag{2}$$

$$\text{s.t. } \mathrm{D}_{\star}[y \| \hat{y}] \leq \gamma \quad \forall x \in \mathcal{D} \tag{3}$$

where $\hat{y} = f(\hat{x}), y$ is the tokenized label, $\mathrm{D}_{\star}[y\|\hat{y}]$ is the KL divergence and the margin $\gamma \in \mathbb{R}_{>0}$ is a hyperparameter.

Given that $\mathcal{L}_0$ is discontinuous and has zero derivative almost everywhere, and the mask generator ϕ requires a discontinuous output activation (like a step function) for binary masks, we utilize a sparse relaxation to binary variables [14, 30] instead of the binary mask during training.

Holistic workflow As shown in Figure 5, XXX encompasses two phases:

(1) *Offline phase*: (**1a**) First, XXX trains a differentiable mask generator. The client selects a mask generator model, potentially a submodule of a pre-trained ASR model, such as HuBERT's CNN feature extractor. A small gate model is then integrated with this submodule. The combined model processes the input audio and generates a mask. This mask selectively conceals parts of the input, ensuring retention of only vital SLU information while hiding sensitive data. The masked input is then forwarded to either a trusted cloud service or a local SLU model for obtaining masked output. The mask generator is fine-tuned to minimize the discrepancy between the masked output logits and the original intent, as defined in Equation (1-3).

(**1b**) Second, XXX adapts the cloud model . Here, the client forwards the masked input and a specific SLU intent (e.g., "set alarm") to the cloud-based SLU model. The model undergoes fine-tuning to adapt to the masked inputs. This process includes adjusting the model parameters for accurate recognition and response to SLU commands based on the masked input.

(2) *Online phase*: In online speech understanding, the client sends the masked input to the cloud SLU model. Using the adapted model, the cloud-based SLU accurately identifies and executes the intended SLU action or response.

Configurator cost analysis Training the differentiable mask generator is affordable for the client. Our experiments indicate that convergence is achieved with approximately 200 audio samples, equivalent to 800 seconds of audio. This process takes up to 40 seconds on an A40 GPU. Adapting the SLU model to each mask generator is a one-pass effort. This adaptation is relatively trivial, especially when starting from a fine-tuned SLU model rather than building from scratch. This aspect of the process incurs minimal cost compared to the training of the cloud SLU model. Moreover, these costs can be amortized over a large number of edge users in the long run, making it an economically viable solution.

Remark Note that the mask generator is not developed for tagging sequences at a semantic level. Rather, its design focuses on identifying segments that are more relevant to the SLU task. This task is essentially a relatively straightforward binary classification problem, which is proven to be effective in prior interpretable learning literature [14, 19] and lightweight enough for real-time inference.

4 IMPLEMENTATION AND METHODOLOGY

We have fully implemented the XXX prototype atop SpeechBrain [38], a PyTorch-based and unified speech toolkit. As prior work [45], we use SpeechBrain to train the differential mask generator and simulate the cloud training process. After that, we deploy the trained mask generator into the embedded devices and evaluate the end-to-end performance.

Hardware and environment Offline training is simulated on a server with 8 NVIDIA A40 GPUs. The trained mask generator is deployed into the STM32H7 [5] or Raspberry PI 4 (RPI-4B) [6]. STM32H7 is a wimpy microcontroller with 1MB RAM. RPI-4B is a popular development board with 4GB RAM. We embed the approaches not feasible to fit in the STM32H7 into the Raspberry PI 4.

Models We design four types of mask generator structures: (1) Random: a random binary vector generator with 50% portion masked; (2) XXX-S: a learnable mask generator with only one MLP gate; (3) XXX-M: a learnable mask generator with one HuBERT encoder layer and one MLP gate; (4) XXX-L: a learnable mask generator with three HuBERT encoder layer and one MLP gate. As for the cloud SLU model, we simulate it using the SoTA end-to-end SLU model [45]. It replaces the ASR decoder of pre-trained HuBERT with SLU attentional decoder.

Dataset and Metrics We run our experiments on SLURP [13] with 102 hours of speech. SLURP's utterances are complex and closer to daily human speech. We select scenario classification accuracy to measure the SLU understanding performance (ACC-SLU). We choose large-scale English reading corpus LibriSpeech [33] for the multi-task protection scenario, following prior work [44]. In the multi-task protection scenario, not only the SLU command utterance (SLURP) but also the background or the subsequent utterance (LibriSpeech) are uploaded to the cloud. WER is used to measure the attack performance. More specifically, we utilize WER-SLU to measure the attacker's capacity to recognize the word information in the uploaded SLU audio itself, and WER-ASR as the WER of recognized accompanying audio, i.e., LibriSpeech dataset. We also report the private entity recognition error rate (EER) to ensure that the cloud model is not able to recognize the private information in the speech signal.

Baselines We compare XXX to the following alternatives: (1) OnDevice means the cloud SLU model is downloaded and run locally on the client device. (2) AllOffload means the raw audio is uploaded to the cloud for SLU inference. (3) VAE [9] is the vanilla variational auto-encoder method that uses adversarial training to disentangle the private information from speech signal. (4) PPSLU [44] is the state-of-the-art disentangling-based SLU privacy-preserving system, which uses 12 transformer layers to separate the SLU information into a part of the hidden layer and only sends those hidden layers to the cloud for SLU inference.

Attack scenarios. We use three attacks encompassing both black-box and white-box attacks: (1) Azure represents a black-box attacker scenario, in which the masked audio is

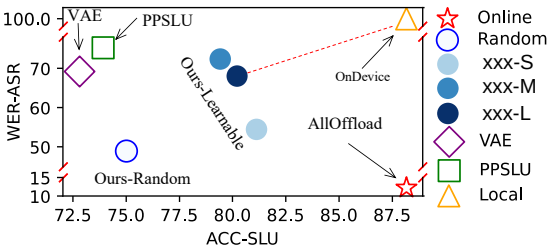


Figure 6: Performance of different privacy-preserving SLU approaches. `OnDevice` offloads no signals to the cloud and thus has the best privacy protection (WER=100).

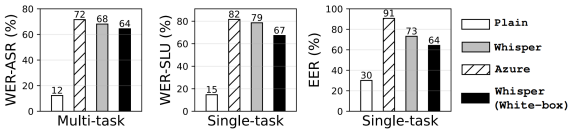


Figure 7: XXX privacy-preserving capacity under different attack models.

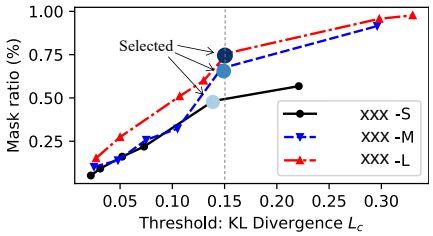


Figure 8: Effect of threshold with different mask generators.

transmitted to Microsoft's ASR model, Azure, for automatic speech recognition [31]. We propose three attack scenarios encompassing both black-box and white-box attacks: (2) `Whisper` simulates a cloud-based ASR model. This black-box attacker uses the pre-trained *Whisper.medium.en* model [36], directly downloaded from HuggingFace [46]. (3) `Whisper(White-box)` constitutes a white-box attack. Here, we hypothesize that certain users are malicious and disclose the mask generator's structure and weights, along with their own audio data, to the `Whisper` attack model. `Whisper(White-box)` then utilizes this collected data from malicious users to adapt the pre-trained *Whisper.medium.en* model to the specific masking pattern.

Hyper-parameters During the offline phase in Figure 5, we use the Adam optimizer with a learning rate of 1e-5 and a batch size of 4. For the inference step, we use the batch size of 1 to simulate the real streaming audio input scenario. The end-to-end cloud SLU latency is measured by invoking Azure APIs following previous work [43]. KL threshold λ is set as 0.15 for all mask generators. Attack model is set as `Whisper` without special declaration.

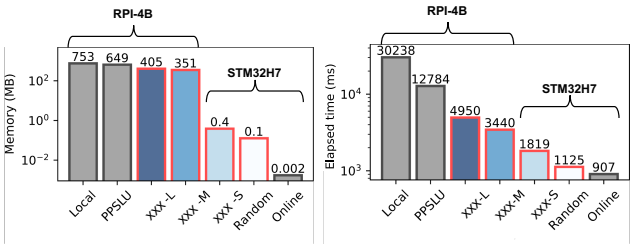
5 EVALUATION

5.1 End-to-end performance

`XXX` achieves comparable accuracy performance and privacy protection capacity to previous encoders. As shown in Figure 6, we compare the accuracy of `XXX` with all baselines. Is observed that `XXX` could achieve up to 81.1% accuracy, with less than 7% accuracy loss compared to unprotected `AllOffload` and local `OnDevice` SLU model. Its rationale is that we mainly mask the short-dependent frames that does not significantly affect the SLU performance. We also compare the performance of `XXX` with the state-of-the-art privacy-preserving SLU system, i.e., `PPSLU` [44]. `XXX` achieves 7.2% higher accuracy than `PPSLU` which tries to apply complex non-linear transformation to the hidden layer to prevent malicious re-construction, but this might also damage part of the SLU information. In terms of privacy preservation, our learnable mask generator achieves up to 78.6% WER using `XXX-L`, indicating a privacy-preserving capacity on par with `PPSLU`. Furthermore, we complete the inference with much lower delays and memory footprint as will be shown in Figure 9.

`XXX` is resistant to different attack models. As illustrated in Figure 7, `XXX` increases the SLU-WER from 14.7% to 78.6% under the attack model `Whisper`. As for the online attack model `Azure`, `XXX` increases the SLU-WER from 14.7% to 81.6%. According to our returned service details, we find that over 50% of the sent audios are tagged as "*ResultReason.NoMatch*", which means audios are recognized as null utterances by the Azure ASR model. `Whisper(White-box)` is a white-box attack model, which means the attacker has the same mask generator structure and weights as the `XXX`. We still achieve more than 50% SLU-WER under this attack model. This is because even `Whisper(White-box)` is fine-tuned to fill some of the missing frames, it still could not recover the private missing frames because ,masking the short-dependent frames have fundamentally destroy the raw audio signal. It is not possible to re-construct the phoneme without knowing any speech information. In the last row, we show the high entity error rate to demonstrate that the private entity is not leaked.

`XXX` scales to better privacy-accuracy trade-off with a larger mask generator. We explore the impact of the threshold γ of `XXX` under different mask generator structures. As shown in Figure 8, the threshold γ controls the trade-off between the privacy and utility. When γ is small, the mask generator is more conservative, leading to higher the utility a lower the masking portion. As we have discussed in Section 3, a lower rate of masking portions leads to higher possibility of privacy entity leakage. When γ is large, the mask generator is more aggressive, enhancing privacy.



(a) Memory footprint **(b) End-to-end latency**

Figure 9: Comparison of resource cost in different SLU approaches. Ours are highlighted in red.

Another way to achieve more practical privacy-utility balance is using a more complex mask generator structure, e.g., XXX-L. It achieves higher utility with the same privacy level compared to XXX-S, albeit with less efficiency, as shown in § 5.2.

5.2 System cost

XXX protects the private entities efficiently. Different from prior encoders using complex disentanglement model, XXX only requires a light-weight mask generator to scrub the private information. The size of this generator varies according to different mask generator structures. For the smallest mask generator, XXX-S, it only requires a 394.9KB memory footprint, and could successfully embed into the wimpy STM32H7 with 2MB RAM. XXX is efficient not only in terms of memory footprint but also in latency. XXX-S completes the local encoding with only 912.2ms on the wimpy STM32H7. For a fair comparison, we embed XXX-S into RPI-4B and find that it is 18.1× faster and 134.1× less memory footprint than PPSLU. Even with the strong mask generator XXX-L, XXX achieves up to 7.5× lower encoding latency and consumes 1.9× less memory compared to OnDevice.

6 CONCLUSIONS

XXX is an efficient and privacy-preserving end-to-end SLU system based on the asymmetrical dependency between ASR and SLU. XXX selectively mask the short-dependent sensitive words while retaining the long-dependent SLU intents. Together with the differentiable mask generator, XXX shows superior end-to-end inference speedup and privacy protection under different attack scenarios.

REFERENCES

[1] https://openai.com/blog/chatgpt-can-now-see-hear-and-speak.
[2] https://huggingface.co/models?sort=downloads.
[3] https://safeatlast.co/blog/siri-statistics/.
[4] https://www.cnbc.com/2019/08/28/apple-apologizes-for-listening-to-siri-conversations.html.
[5] https://www.st.com/en/microcontrollers-microprocessors/stm32h7-series.html.
[6] https://www.raspberrypi.com/products/raspberry-pi-4-model-b/.
[7] https://developer.nvidia.com/embedded/jetson-tx2.
[8] Shimaa Ahmed, Amrita Roy Chowdhury, Kassem Fawaz, and Parmesh Ramanathan. Preech: A system for privacy-preserving speech transcription. arXiv preprint arXiv:1909.04198 v2, 2019.
[9] Ranya Aloufi, Hamed Haddadi, and David Boyle. Privacy-preserving voice analysis via disentangled representations. In Proceedings of the 2020 ACM SIGSAC Conference on Cloud Computing Security Workshop, pages 1–14, 2020.
[10] Android. Android: Low memory killer daemon. https://source.android.com/docs/core/perf/lmkd, 2022.
[11] Siddhant Arora, Siddharth Dalmia, Xuankai Chang, Brian Yan, Alan Black, and Shinji Watanabe. Two-pass low latency end-to-end spoken language understanding. arXiv preprint arXiv:2207.06670, 2022.
[12] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. Advances in neural information processing systems, 33:12449–12460, 2020.
[13] Emanuele Bastianelli, Andrea Vanzo, Pawel Swietojanski, and Verena Rieser. Slurp: A spoken language understanding resource package. arXiv preprint arXiv:2011.13205, 2020.
[14] Jasmijn Bastings, Wilker Aziz, and Ivan Titov. Interpretable neural predictions with differentiable binary variables. arXiv preprint arXiv:1905.08160, 2019.
[15] Yike Chen, Ming Gao, Yimin Li, Lingfeng Zhang, Li Lu, Feng Lin, Jinsong Han, and Kui Ren. Big brother is listening: An evaluation framework on ultrasonic microphone jammers. In IEEE INFOCOM 2022-IEEE Conference on Computer Communications, pages 1119–1128. IEEE, 2022.
[16] Peng Cheng and Utz Roedig. Personal voice assistant security and privacy—a survey. Proceedings of the IEEE, 110(4):476–507, 2022.
[17] Leigh Clark, Philip Doyle, Diego Garaialde, Emer Gilmartin, Stephan Schlögl, Jens Edlund, Matthew Aylett, João Cabral, Cosmin Munteanu, Justin Edwards, et al. The state of speech in hci: Trends, themes and challenges. Interacting with computers, 31(4):349–371, 2019.
[18] Trung Dang, Om Thakkar, Swaroop Ramaswamy, Rajiv Mathews, Peter Chin, and Françoise Beaufays. A method to reveal speaker identity in distributed asr training, and how to counter it. In ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 4338–4342. IEEE, 2022.
[19] Nicola De Cao, Michael Schlichtkrull, Wilker Aziz, and Ivan Titov. How do decisions emerge across layers in neural models? interpretation with differentiable masking. arXiv preprint arXiv:2004.14992, 2020.
[20] Keqi Deng, Songjun Cao, Yike Zhang, and Long Ma. Improving hybrid ctc/attention end-to-end speech recognition with pretrained acoustic and language models. In 2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), pages 76–82. IEEE, 2021.
[21] Ye Dong, Wen-jie Lu, Yancheng Zheng, Haoqi Wu, Derun Zhao, Jin Tan, Zhicong Huang, Cheng Hong, Tao Wei, and Wenguang Cheng. Puma: Secure inference of llama-7b in five minutes. arXiv preprint arXiv:2307.12533, 2023.
[22] Lloyd E Emokpae, Roland N Emokpae, Wassila Lalouani, and Mohamed Younis. Smart multimodal telehealth-iot system for covid-19 patients. IEEE Pervasive Computing, 20(2):73–80, 2021.
[23] Ming Gao, Yike Chen, Yajie Liu, Jie Xiong, Jinsong Han, and Kui Ren. Cancelling Speech Signals for Speech Privacy Protection against Microphone Eavesdropping. Association for Computing Machinery, New York, NY, USA, 2023.
[24] Oded Goldreich. Secure multi-party computation. Manuscript. Preliminary version, 78(110):1–108, 1998.
[25] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. Advances in neural information processing

systems, 27, 2014.

[26] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, et al. Conformer: Convolution-augmented transformer for speech recognition. arXiv preprint arXiv:2005.08100, 2020.

[27] Parisa Haghani, Arun Narayanan, Michiel Bacchiani, Galen Chuang, Neeraj Gaur, Pedro Moreno, Rohit Prabhavalkar, Zhongdi Qu, and Austin Waters. From audio to semantics: Approaches to end-to-end spoken language understanding. In 2018 IEEE Spoken Language Technology Workshop (SLT), pages 720–726. IEEE, 2018.

[28] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114, 2013.

[29] Naveen Kumar and Seul Chan Lee. Human-machine interface in smart factory: A systematic literature review. Technological Forecasting and Social Change, 174:121284, 2022.

[30] Christos Louizos, Max Welling, and Diederik P Kingma. Learning sparse neural networks through l_0 regularization. arXiv preprint arXiv:1712.01312, 2017.

[31] Microsoft. Azure asr. https://azure.microsoft.com/en-us/products/ai-services/speech-to-text/.

[32] Nombulelo CC Noruwana, Pius Adewale Owolawi, and Temitope Mapayi. Interactive iot-based speech-controlled home automation system. In 2020 2nd International Multidisciplinary Information Technology and Engineering Conference (IMITEC), pages 1–8. IEEE, 2020.

[33] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an asr corpus based on public domain audio books. In 2015 IEEE international conference on acoustics, speech and signal processing (ICASSP), pages 5206–5210. IEEE, 2015.

[34] Cal Peyser, Ronny Huang Andrew Rosenberg Tara N Sainath, Michael Picheny, and Kyunghyun Cho. Towards disentangled speech representations. arXiv preprint arXiv:2208.13191, 2022.

[35] Jianwei Qian, Haohua Du, Jiahui Hou, Linlin Chen, Taeho Jung, and Xiang-Yang Li. Hidebehind: Enjoy voice input with voiceprint unclonability and anonymity. In Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems, pages 82–94, 2018.

[36] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In International Conference on Machine Learning, pages 28492–28518. PMLR, 2023.

[37] Joel M Raja, Carol Elsakr, Sherif Roman, Brandon Cave, Issa Pour-Ghaz, Amit Nanda, Miguel Maturana, and Rami N Khouzam. Apple watch, wearables, and heart rhythm: where do we stand? Annals of translational medicine, 7(17), 2019.

[38] Mirco Ravanelli, Titouan Parcollet, Peter Plantinga, Aku Rouhe, Samuele Cornell, Loren Lugosch, Cem Subakan, Nauman Dawalatabad, Abdelwahab Heba, Jianyuan Zhong, et al. Speechbrain: A general-purpose speech toolkit. arXiv preprint arXiv:2106.04624, 2021.

[39] Subendhu Rongali, Beiye Liu, Liwei Cai, Konstantine Arkoudas, Chengwei Su, and Wael Hamza. Exploring transfer learning for end-to-end spoken language understanding. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 35, pages 13754–13761, 2021.

[40] Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. wav2vec: Unsupervised pre-training for speech recognition. arXiv preprint arXiv:1904.05862, 2019.

[41] Suranga Seneviratne, Yining Hu, Tham Nguyen, Guohao Lan, Sara Khalifa, Kanchana Thilakarathna, Mahbub Hassan, and Aruna Seneviratne. A survey of wearable devices and challenges. IEEE Communications Surveys & Tutorials, 19(4):2573–2620, 2017.

[42] Ke Sun, Chen Chen, and Xinyu Zhang. " alexa, stop spying on me!" speech privacy protection against voice assistants. In Proceedings of the 18th conference on embedded networked sensor systems, pages 298–311, 2020.

[43] Rongxiang Wang and Felix Lin. Efficient deep speech understanding at the edge. arXiv preprint arXiv:2311.17065, 2023.

[44] Yinggui Wang, Wei Huang, and Le Yang. Privacy-preserving end-to-end spoken language understanding. In Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, pages 5224–5232, 2023.

[45] Yingzhi Wang, Abdelmoumene Boumadane, and Abdelwahab Heba. A fine-tuned wav2vec 2.0/hubert benchmark for speech emotion recognition, speaker verification and spoken language understanding. arXiv preprint arXiv:2111.02735, 2021.

[46] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. arXiv preprint arXiv:1910.03771, 2019.

[47] Albert Zeyer, Ralf Schlüter, and Hermann Ney. Why does ctc result in peaky behavior? arXiv preprint arXiv:2105.14849, 2021.

[48] Chengliang Zhang, Suyi Li, Junzhe Xia, Wei Wang, Feng Yan, and Yang Liu. Batchcrypt: Efficient homomorphic encryption for cross-silo federated learning. In 2020 USENIX annual technical conference (USENIX ATC 20), pages 493–506, 2020.