# SGSM: A Foundation-model-like Semi-generalist Sensing Model

Tianjian Yang[1,2,3], Hao Zhou[1,2,3], Shuo Liu[1], Kaiwen Guo[1,2,3], Yiwen Hou[1], Haohua Du[4], Zhi Liu[5], and Xiang-Yang Li[1,2,3]

[1]School of Computer Science and Technology, University of Science and Technology of China
[2]CAS Key Laboratory of Wireless-Optical Communications, University of Science and Technology of China
[3]Deqing Alpha Innovation Institute, Huzhou, Zhejiang, China
[4]School of Cyber Science and Technology, Beihang University
[5]Department of Computer and Network Engineering, The University of Electro-Communications, Tokyo 182-8585, Japan

## ABSTRACT

The significance of intelligent sensing systems is growing in the realm of smart services. These systems extract relevant signal features and generate informative representations for particular tasks. However, building the feature extraction component for such systems requires extensive domain-specific expertise or data. The exceptionally rapid development of foundation models is likely to usher in newfound abilities in such intelligent sensing. We propose a new scheme for sensing model, which we refer to as semi-generalist sensing model (SGSM). SGSM is able to semiautomatically solve various tasks using relatively less task-specific labeled data compared to traditional systems. Built through the analysis of the common theoretical model, SGSM can depict different modalities, such as the acoustic and Wi-Fi signal. Experimental results on such two heterogeneous sensors illustrate that SGSM functions across a wide range of scenarios, thereby establishing its broad applicability. In some cases, SGSM even achieves better performance than sensor-specific specialized solutions. Wi-Fi evaluations indicate a 20% accuracy improvement when applying SGSM to an existing sensing model.

***Keywords*** Mobile computing · Machine learning

## 1 Introduction

Intelligent sensing systems have shown remarkable performance on many environmental perception (e.g., liquid recognition [1], soil moisture estimation [2], temperature monitoring [3]) and human activity (e.g., fall detection [4], vital sign estimation [5], location tracking [6]) tasks, becoming the core component of smart physical-related services, such as smart city and smart manufacturing. However, the current cost of designing intelligent sensing systems is relatively high since the models were designed to solve specific tasks with expensive expert knowledge [7] or a substantial amount of domain-specific data [8], one at a time.

Foundation models [9] – the latest generation of artificial intelligence (AI) models – are intuitively used to generalize the model for numerous downstream tasks, which are trained on large multimodal datasets. They can solve entirely new tasks which the models are never explicitly trained for. Although the foundation models paradigm perform well in computer vision or natural language processing area, applying them in the intelligent sensing area is still challenging for two reasons.

First, **it is difficult to generate or access massive and diverse sensing datasets**. Massive high-quality data is crucial for foundation model applications, such as computer vision [10] and natural language processing [9]. However, this requirement is often unmet in the sensing field. To overcome this challenge, signal processing methods based on spectrum analysis are traditionally used, such as Discrete Fourier Transform (DFT), Discrete Wavelet Transform

(DWT), and Hilbert-Huang Transform (HHT). Unfortunately, these methods focus more on general features and require considerable expertise to apply them to specific tasks. This lack of standardized guidelines makes it difficult to efficiently utilize these handcrafted methods and the datasets transformed by them in AI models.

Secondly, **the sensors work on complex principles and the sensing tasks are ever-changing.** The task-specific machine learning models are largely developed for handling this, but most of the existing solutions lose the generality and have to re-train for different scenarios. For example, a wearable exercise monitoring model may be trained on an IMU dataset where samples are annotated as human activity. This model can only predict the exercise situation, but the same dataset with different labels could carry out other diagnostic model for some common health problem, like pneumonia.

Fundamentally, the challenges come from the phenomenon of *feature drifts*, wherein the extracted features associated with the data analysis method are not well-suited for subsequent tasks. Therefore, to alleviate such a phenomenon, the intuitive idea is to offer a scheme that guides the design of specific sensing systems by *using the advantages of both handcrafted and machine learning approaches* – utilizing the data handling knowledge from ready-made signal processing methods and the task depicting ability from machine learning models.

Nevertheless, the realization of such a scheme is not obvious. Utilizing existing human knowledge to enhance foundation model performance remains an ongoing and unresolved subject. Furthermore, the intrinsic complexity associated with human's and models' contexts have not been thoroughly investigated [11]. There is currently a lack of standardized guidelines for extracting data handling knowledge from signal processing methods, and experience in generalizing the ability of task-specific models in the field of signal processing. The scheme must not only effectively incorporate both components, but also facilitate interactive and iterative adjustments between them that are both uncertain and subject to change.

In this paper, we propose SGSM, a novel intelligent sensing scheme to realize versatile models with limited labelled data and little signal processing knowledge. It works in a foundation-model-style, i.e., a *generalized* model to support numerous *specialized* downstream tasks. To achieve this target, SGSM utilizes a two-phase autoencoder (AE) structure to combine the knowledge and annotated data. In the first phase, SGSM provides *generalized* representations by extracting the knowledge of existing signal processing methods with a series of AEs denoted as COMPRESSORs. They unify the outputs of diverse handcrafted methods into a shared feature space and learn to generate task-irrelative representations with unlabelled data. In the second phase, SGSM supports *specialized* tasks by guiding the selection of methods with a denoising AE denoted as the MIXER which can automatically evaluate the performance of various method combinations regarding to a specific downstream task, using the representations of the annotated data from the COMPRESSORs.

The main contributions of the paper are:

- To the best of our knowledge, SGSM is the first foundation-like model to leverage the handcrafted knowledge and machine learning approaches in the intelligent sensing area, requiring only restricted annotated data: in this way we can achieve the same accuracy as existing works with zero expert knowledge and less amount of annotated training data.
- We propose a semi-automated intelligent sensing paradigm which can adaptively construct the sensing model, whatever the sensor is, whatever the sensing objectives are. SGSM does not base on particular characteristics derived from signal types, which supports the robustness of the scheme.
- We build prototypes of SGSM and test them with real world datasets. Experimental results on two heterogeneous sensors (acoustic and Wi-Fi) show that our scheme functions across a wide range of scenarios, thereby establishing its broad applicability. In some cases, SGSM even achieves better performance than sensor-specific specialized solutions. Specifically, Wi-Fi evaluations indicate a 20% accuracy improvement when applying SGSM to an existing sensing model.

## 2 Methodology

In this section, we will first give the problem formulation. Then, we will illustrate the basic idea of how SGSM works and provide illustrative case studies as support.

### 2.1 Problem Formulation

In the scenario of *semi-generalist intelligent sensing*, a generic task consists of a sensing objective $o$, a sensor and its corresponding signal dataset $S$, and a related labeled dataset $D_S^o$. It is noteworthy that $D_S^o$ has the same data type with $S$ and labeled for $o$.

Given an instance $(o, S, D_S^o)$ and the handcrafted signal processing methods $F = \{f_1, f_2, \cdots, f_n\}$ as input, our goal is to identify which subset $F' \subseteq F$ has the optimal performance with deep learning classification methods. To do so, we need to solve two co-related sub-problems. The first one is to find the optimal subset $F'$ on the dataset $D_S^o$,

$$\phi(F', D_S^o) \geq \max_{\hat{F} \in \mathcal{P}(F) - F'} \left\{ \phi\left(\hat{F}, D_S^o\right) \right\} + \varepsilon, \tag{1}$$

where $\phi$ itself can be interpreted as proper metrics such as accuracy and F1-score, and $\varepsilon$ is an error coefficient.

The second is to guarantee the performance on the real task dataset $S$ being at least same with on $D_S^o$ for chosen $F'$,

$$|\phi(F', S) - \phi(F', D_S^o)| \leq \varsigma, \tag{2}$$

where $\varsigma$ is an error coefficient.

## 2.2 Solution Sketch

The key to solve problems (1) and (2) is to provide a proper way to evaluate the performance of $\hat{F}$ on $D_D^o$, which leads to two requirements. First, SGSM should be capable of normalizing the output of various signal processing methods so that it can deal with heterogeneous sequences which vary in scale and granularity. Second, SGSM should provide evaluation results of all method selections conveniently and cost-efficiently.



Figure 1: Two-phase structure of SGSM.

The proposed system satisfies these requirements by applying a two-phase structure. The first phase, COMPRESSORs, extracts compressed representations from transformed sequences generated by processing methods. COMPRESSORs learn to standardize and refine given information, and are trained with unlabeled data to enhance robustness. The second phase, the MIXER, combines representations from all method channels and integrates them into embeddings $SGSM(\hat{F}, D_S^o)$, meanwhile providing a masking interface for subsequent evaluation.

## 2.3 Phase One - COMPRESSORs

The first phase, COMPRESSORs, focuses on dealing with task-irrelevant knowledge provided by existing signal processing methods. COMPRESSORs extract information and normalize them for later use. COMPRESSORs are a series of autoencoders, which have been part of the historical landscape of neural networks for decades and were traditionally used for dimensionality reduction or feature learning [12]. In SGSM, signal processing methods are applied to unlabeled data $U$ for transformed results $f_i(U)$. COMPRESSORs receive the results and generate the latent codes COMPRESSOR$(f_i(U))$. By learning from the transformed results of unlabeled data, COMPRESSORs manage to refine the core knowledge of corresponding methods. Moreover, by processing heterogeneous information into standardized code vectors, COMPRESSORs provide normalized preparation for later processing.

## 2.4 Phase Two - MIXER

The second phase, the MIXER, is responsible for using the refined codes provided by the COMPRESSORs to generate final embeddings according to method selections. The embeddings of *task-relevant* datasets can be used for subsequent evaluations. To fulfill the MIXER, we combine an autoencoder with masking policies, i.e., a masked autoencoder. Masking policies come from the need to adapt various method configurations $\hat{F}$. We regard researchers selecting methods as applying "noise" to corresponding channels of COMPRESSORs. In this way, not using methods is equivalent to masking the related codes. With masking policies, the only challenge left is to combine information from masked

codes. We apply masked autoencoders, denoising autoencoders (DAE) which receive corrupted data as input and predict the original, uncorrupted data as output. A DAE is intended not only to denoise but to learn a good internal representation as a side effect [12]. If we regard the method configurations as noise, training a DAE with masking policies enables the DAE to deal with the "noise", i.e., varying method combinations, in advance. Thus, the system needs no extra training to generate embeddings for particular downstream data. In conclusion, by applying masked autoencoders, the system can combine information from different COMPRESSOR channels and generate embeddings with various method selections.

## 3 System Design

In this section, we will first describe design details of COMPRESSORs and the MIXER. After that, instructions on how to apply SGSM in specific tasks will be given. The overall workflow for embedding generation and extra workflow for pre-training are depicted in Fig. 2.

### 3.1 COMPRESSOR

An instance of SGSM involves several COMPRESSORs, each of which corresponds to a signal processing method. Without loss of generality, we assume that a set of methods $\{f_i | 1 \leq i \leq n\}$ is involved, where each method is a mapping $f_i : \mathbf{R}^L \to \mathbf{R}^{L'_i}$. $f_i(\boldsymbol{x})$ denotes the transformed output of a signal sequence $\boldsymbol{x}$ by $f_i$. The autoencoder of $f_i$ is composed of encoder $Enc^i$ and decoder $Dec^i$. For $f_i(\boldsymbol{x})$, $Enc^i$ generates a code of a fixed length:

$$\boldsymbol{v}_{\boldsymbol{x}}^i = Enc^i \left( f_i(\boldsymbol{x}) \right), \tag{3}$$

where $\boldsymbol{v}_{\boldsymbol{x}}^i \in \mathbf{R}^d$. The length is called fixed because all encoders share the same length of the codes. By this constraint, COMPRESSORs transform sequences of all signal transformation results into a unified data space. Sequentially, the decoder $Dec^i$ accepts $\boldsymbol{v}_{\boldsymbol{x}}^i$ as input and outputs a vector of length $L'_i$:

$$\boldsymbol{y}_{\boldsymbol{x}}^i = Dec^i \left( \boldsymbol{v}_{\boldsymbol{x}}^i \right). \tag{4}$$

The loss function of this autoencoder is given by:

$$\mathcal{L}^i(\boldsymbol{x}) = \frac{1}{L'_i} \left( \boldsymbol{y}_{\boldsymbol{x}}^i - f_i(\boldsymbol{x}) \right)^2 + 1 - \cos \left( \boldsymbol{y}_{\boldsymbol{x}}^i, f_i(\boldsymbol{x}) \right). \tag{5}$$

The loss function can be considered as a combination of the mean squared error and the cosine embedding loss between the input and the reconstruction outcome.



Figure 2: SGSM's workflow for embedding generation and extra workflow for pre-training in SGSM.

Fig. 3 depicts the structure of the COMPRESSOR. Multiple layers are employed, including convolutional layers, ReLU layers, linear layers, etc. The encoder of COMPRESSOR consists of two parts. The first part aims to extract and refine information by increasing the number of channels. The second part aims to map the output of the first part to codes

Figure 3: Network structure of COMPRESSOR.



Figure 4: Network structure of MIXER.

of a uniform length $d$. By assuring $d < L'_i$, the encoder must learn a compressed representation of input, which also explains why they are called COMPRESSORs.

The COMPRESSOR learns to compress and reconstruct transformed signal sequences based on the aforementioned loss function. The limitation of $d < L'_i$ requires the undercomplete AE to extract valuable information instead of plunging into the trivial solution of identity mapping.

## 3.2 MIXER

There is one MIXER in every instance of SGSM. Assume that a series of COMPRESSORs are trained. For each raw signal sequence $\boldsymbol{x}$, COMPRESSORs provide $n$ codes $\{\boldsymbol{v}_{\boldsymbol{x}}^i | 1 \leq i \leq n\}$. Concatenating these codes in a certain order results in a vector $\boldsymbol{V}$ of length $nd$. Let $\boldsymbol{V} = \boldsymbol{v}_{\boldsymbol{x}}^1 \oplus \boldsymbol{v}_{\boldsymbol{x}}^2 \oplus \cdots \oplus \boldsymbol{v}_{\boldsymbol{x}}^n$, where $\oplus$ indicates the concatenating operation. Applying one of the two masking policies produces a masked vector $\boldsymbol{V}'$.

- Global-mask policy: $\boldsymbol{V}'$ is the outcome of setting a certain percentage of the bits of $\boldsymbol{V}$ to zero. In our experiments, this percentage is set to 10%.

- Channel-mask policy: $\boldsymbol{V}'$ is the outcome of setting bits of randomly selected channels of $\boldsymbol{V}$ to zero. Each channel has a 50% possibility to be masked, except that masking all channels is meaningless and thus forbidden. For example, if the third channel is selected when masking $\boldsymbol{V} = \boldsymbol{v}_{\boldsymbol{x}}^1 \oplus \boldsymbol{v}_{\boldsymbol{x}}^2 \oplus \boldsymbol{v}_{\boldsymbol{x}}^3$, then $\boldsymbol{V}' = \boldsymbol{v}_{\boldsymbol{x}}^1 \oplus \boldsymbol{v}_{\boldsymbol{x}}^2 \oplus \boldsymbol{0}$.

The possibility of the global-mask policy being applied is 80% while that of the channel-mask policy is 20%. While training the MIXER, each vector $\boldsymbol{V}$ is masked independently every time it is fed into the MIXER. These masked vectors $\boldsymbol{V}'$ are input of the MIXER. The loss function of the MIXER is:

$$\mathcal{L}(\boldsymbol{V}) = \frac{1}{L} \left(\boldsymbol{y}_{\boldsymbol{V}'} - \boldsymbol{V}\right)^2 + 1 - \cos\left(\boldsymbol{y}_{\boldsymbol{V}'}, \boldsymbol{V}\right), \tag{6}$$

where $\boldsymbol{y}_{\boldsymbol{V}'}$ is the reconstruction output of the MIXER and $L$ is the length of it. Output of MX's encoder is the final embeddings $\boldsymbol{E}$. It is noteworthy that only channel-mask policy is applied when generating embeddings for subsequent tasks. The embeddings have the same length as input, i.e. $nd = \dim \boldsymbol{V} = \dim \boldsymbol{E}$.

Fig. 4 depicts the structure of the MIXER. The encoder and the decoder are identical. They first expand the dimension to $4nd$, and then reduce it back to $nd$. Unlike the COMPRESSORS, the MIXER is not an undercomplete autoencoder. However, the noising process make sure that the inputs and the reconstruction targets of the MIXER are different. In other words, identity mapping is naturally not an expected result of training the MIXER. Therefore, the MIXER does not fall into the trivial solution and will always produce meaningful outcomes if converges.

### 3.3 Applying SGSM in particular tasks

Pre-training a SGSM involves no particular labelled data. To implement SGSM, the *pre-training* procedures are as follows:

1. Select the processing method set $F = \{f_i | i \in [1, n]\}$.
2. Collect an unlabeled dataset $U$ containing signal samples from the same type of sensors as the particular tasks. Apply the methods in $F$ for transformed datasets $\{f_i(U)\}$.
3. Train $n$ COMPRESSORS with transformed datasets separately. Apply COMPRESSORS to transformed datasets for coded datasets $\{\boldsymbol{v}_U^i\}$. Concatenate the coded datasets to gather the concatenation dataset $\boldsymbol{V}_U$.
4. Train a MIXER with the concatenation dataset.

To apply SGSM in a particular task $S$ about an annotated dataset $D_S^o$, researchers only have to:

5. Apply the MIXER to the annotated dataset $D_S^o$ for embeddings $SGSM(\hat{F}, D_S^o)$ with user-specific channel masks. The embeddings are the final outcomes of SGSM. Gather evaluation performance for all $\hat{F} \in \mathcal{P}(F)$ to find the best-performing method combination $F'$.

For users, SGSM doesn't participate in additional training because there is no need for fine-tuning. Instead, a pre-trained SGSM simply offers embeddings of annotated data with user-specific channel masks. If researchers want to evaluate various configurations, the only work is to generate embeddings with corresponding masks. During this process, SGSM is fixed without the need for adjustment. Moreover, a trained SGSM can be utilized repeatedly since only step (5) involves particular datasets. Researchers can use it to generate wanted embeddings across tasks without any extra pre-training. In contrast, knowledge-based traditional solutions require experts to manually design and test countless statistical features provided by processing methods, while data-driven non-reusable deep learning based models need repetitious training to adapt to different transformed datasets.

## 4 Evaluation

We evaluate SGSM with acoustic and Wi-Fi signals. It is noteworthy that we don't necessarily compare with SOTA approaches in all experiments. The reason is that the experiments are designed to demonstrate SGSM's general capability under different situations and perspectives, other than solely for acheiveing the best performance. To be more specific, Acoustic experiments show SGSM can achieve comparable performance as traditional approaches, while Wi-Fi experiments are to show SGSM's potential to work with other complicated models and their embeddings to achieve outstanding performance.

### 4.1 Experiments on Acoustic Sensing

#### 4.1.1 Datasets

We use **ESC-US** and **ESC-50** [13] for the evaluation. ESC-50 dataset comprises of 2,000 short environmental recordings split equally among 50 classes. ESC-US is an additional dataset including 250,000 recordings without labels. We use ESC-US as the unlabeled dataset for pre-training and ESC-50 for the downstream classification task.

#### 4.1.2 Evaluation Metrics and Baselines

[14] evaluates the test accuracy of applying MFCCs with a convolutional network on the ESC-50 dataset so we use it as the metric. Although more recent works on this task have reported much better performance, they generally apply heavy neural networks and other information modalities, severly deviating from our evaluation aim, so we will not use their results.

### 4.1.3 Implementation Details

SGSM is implemented using Python and PyTorch [15]. It is trained in a PC with 1 NVIDIA GeForce RTX 3060 GPU, 32 GB memory, and an Intel(R) Core(TM) i5-11500 2.70GHz CPU. For each clip, we generate the log Mel-spectrogram which has two dimensions: frequency and time. We split the result along the frequency axis into four parts, each of which occupies a quarter of the whole band. We then reduce each part with addition, resulting in four sequences. We use the unlabeled dataset to train a prototype. The length of the codes is set to 128. Each COMPRESSOR is trained for 50 epochs and the MIXER is trained for 100 epochs. The learning rates are 0.001 for both COMPRESSORs and MIXER, while the batch sizes are 64 and 128 respectively. We use the same strategy as the baseline to evaluate the performance. We use SGSM-XXXX to denote the usage of four channels. The channel order is from the lowest frequency sequence to the highest. For example, SGSM-TFFF indicates that only the channel of the lowest frequency band is used.

### 4.1.4 Performance



Figure 5: Ratio of SGSM's accuracy to baseline's. The red line represents the baseline (100%).

Ratio of SGSM's accuracy to baseline's is depicted in Fig. 5, where the red line represents the baseline (100%). This evaluation is aimed to show that SGSM can achieve a comparable result with a commonly acknowledged method. It is depicted in Fig. 5 that the optimal performance among all mask configurations reaches similar classification accuracy with the baseline. Besides, we also find that the fourth channel, i.e. the highest frequency sequence, reports a relatively poor performance. Sound clips in ESC-50 include many human and animal sounds, where the sound frequencies are lower and their features are clearer. The environmental noises that have a higher frequency, however, are generally more chaotic. This might explain the performance difference between the low and high frequency channels.



Figure 6: Performance of different method combinations. Combining simple methods and AutoFi results in better performance.

## 4.2 Experiments on Wi-Fi Sensing

### 4.2.1 Datasets

**NTU-Fi** [16, 17] is a dataset that includes both human activity recognition (HAR) and human identification (Human ID) tasks. The data has a high resolution of subcarriers (114 per pair of antennas). Both datasets are separated into training and testing datasets. In this evaluation, we use the whole NTU-Fi HAR as the unlabeled dataset, the NTU-Fi Human-ID training dataset as the subsequent task training dataset, and the NTU-Fi Human-ID testing dataset as the subsequent task testing dataset.

#### 4.2.2 Evaluation Metrics and Baselines

**AutoFi** [18] is a Wi-Fi sensing model based on a self-supervised learning algorithm. It can generate an embedding for each Wi-Fi CSI sample. We train an instance of Auto-Fi with MLP backbone on NTU-Fi HAR without complicated parameter adjustments and fine-tuning. Although the model doesn't achieve the performance reported in [18], it is enough for us to conduct our evaluation. Since the evaluation metric is the testing accuracy in [18], we decide to use it as well.

#### 4.2.3 Implementation Details

In this evaluation, we test SGSM's ability to combine simple processing methods and complex embeddings. We build two prototypes of SGSM. The hardware setup is the same as the one in acoustic signal experiments. The first prototype, denoted as SGSM-A, has five channels whose signal processing methods are: DFT, DWT, Raw, HHT, and Periodogram. We treat each CSI sample as a combination of 342 signal sequences, corresponding to the 342 channels. We let each COMPRESSOR generates a code of length 128, resulting in the final embeddings being in the shape of $(342, 128 \times 5)$. The second prototype, denoted as SGSM-B, has six channels, where the extra channel comes from the embeddings of AutoFi. We train an extra COMPRESSOR with AutoFi embeddings and a new MIXER to combine all six channels. Thus, SGSM-B generates embeddings in the shape of $(342, 128 \times 6)$.

For evaluation, we first test AutoFi on NTU-Fi Human-ID. Since the embeddings are 1-dimensional, we use an MLP network including three fully connected layers as the classifier. Then, SGSM-A and SGSM-B use a convolutional network as the classifier because of the extra dimension. The classifiers are adjusted to having comparable sizes. All classifiers are trained on the NTU-Fi Human-ID training dataset for 150 epochs, and the accuracy are tested on the testing dataset. To denote results with different masks, we use SGSM-A-XXXXX and SGSM-B-XXXXXX. The highest five bits of both correspond to DFT, DWT, Raw, HHT, and Periodogram, respectively. The lowest bit for SGSM-B corresponds to the AutoFi channel.

Table 1: Accuracy of AutoFi, SGSM-A, and SGSM-B

| Method | Accuracy |
|---|---|
| AutoFi | 74.33 |
| SGSM-B-FFFFFT | 92.44 |
| SGSM-A-TTTTT | 67.00 |
| SGSM-B-TTTTTT | 83.71 |
| SGSM-A-FFFTF | 88.48 |
| SGSM-B-FFFTFT | 94.37 |

#### 4.2.4 Performance

Classification performance is depicted in Fig. 6 and Tab. 1. In Fig. 6, we group the accuracy results by the mask configurations of the first five channels. For example, group TTFTT includes accuracy of SGSM-A-TTFTT, SGSM-B-TTFTTF, and SGSM-B-TTFTTT. In group FFFFF, only performance of SGSM-B-FFFFFT is provided, because masking all channels is meaningless. In theory, combining information from five simple methods with the other AutoFi embeddings should result in a similar, if not better, performance compared to using them separately. In Tab. 1, we can see that SGSM-B-TTTTTT, i.e., combining codes from all channels, outperforms AutoFi and SGSM-A-TTTTT. This indicates that SGSM-B successfully incorporates information from codes of simple methods and codes of complicated embeddings.

It is noteworthy that most masking combinations in SGSM-A report a better performance in SGSM-B with the sixth channel of AutoFi, as depicted in Fig. 6. This indicates that AutoFi do capture features of the signals that the five simple methods don't. Besides, we notice that the masking combination with the best performance is not opening all channels, but the one opening HHT and AutoFi. It is possible that this combination extracts the most useful features of NTU-Fi Human-ID. Finally, it is worth mentioning that SGSM-B-FFFFFT performs significantly better than AutoFi alone. This implies that the MIXER manages to discover latent information of AutoFi embeddings with the aid of other method channels and recover them for subsequent classification. Such results further demonstrate MIXER's effectiveness.

## 5 Conclusion

This paper introduces SGSM, a new paradigm for sensing model. When compared to typical systems, SGSM can tackle a variety of tasks semi-automatically using less task-specific labelled data. Experiments conducted on two heterogeneous sensors show that SGSM works in various conditions, proving its broad applicability. Acoustic experiments show SGSM can achieve comparable performance as traditional approaches. Wi-Fi evaluations demonstrate that applying SGSM to an existing sensing model improves accuracy by 20%.

## References

[1] Fei Shang, Panlong Yang, Yubo Yan, and Xiang-Yang Li. Liqray: non-invasive and fine-grained liquid recognition system. In *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking*, pages 296–309, 2022.

[2] Usman Mahmood Khan and Muhammad Shahzad. Estimating soil moisture using rf signals. In *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking*, pages 242–254, 2022.

[3] Baicheng Chen, Huining Li, Zhengxiong Li, Xingyu Chen, Chenhan Xu, and Wenyao Xu. Thermowave: a new paradigm of wireless passive temperature monitoring via mmwave sensing. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, pages 1–14, 2020.

[4] Jay Chen, Karric Kwong, Dennis Chang, Jerry Luk, and Ruzena Bajcsy. Wearable sensors for reliable fall detection. In *2005 IEEE Engineering in Medicine and Biology 27th Annual Conference*, pages 3551–3554. IEEE, 2006.

[5] Zhe Chen, Tianyue Zheng, Chao Cai, and Jun Luo. Movi-fi: motion-robust vital signs waveform recovery via deep interpreted rf sensing. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*, pages 392–405, 2021.

[6] Xiang Li, Daqing Zhang, Qin Lv, Jie Xiong, Shengjie Li, Yue Zhang, and Hong Mei. Indotrack: Device-free indoor human tracking with commodity wi-fi. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 1(3):1–22, 2017.

[7] Jiao Liu, Guanlong Teng, and Feng Hong. Human activity sensing with wireless signals: A survey. *Sensors*, 20(4), 2020.

[8] Huatao Xu, Pengfei Zhou, Rui Tan, Mo Li, and Guobin Shen. Limu-bert: Unleashing the potential of unlabeled data for imu sensing applications. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*, pages 220–233, 2021.

[9] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[10] James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. Improving image generation with better captions, 2023.

[11] Andrea Tocchetti and Marco Brambilla. The role of human knowledge in explainable ai. *Data*, 7(7):93, 2022.

[12] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[13] Karol J Piczak. Esc: Dataset for environmental sound classification. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 1015–1018, 2015.

[14] Yusuf Aytar, Carl Vondrick, and Antonio Torralba. Soundnet: Learning sound representations from unlabeled video. *Advances in neural information processing systems*, 29, 2016.

[15] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[16] Jianfei Yang, Xinyan Chen, Han Zou, Dazhuo Wang, Qianwen Xu, and Lihua Xie. Efficientfi: Towards large-scale lightweight wifi sensing via csi compression. *IEEE Internet of Things Journal*, 2022.

[17] Dazhuo Wang, Jianfei Yang, Wei Cui, Lihua Xie, and Sumei Sun. Caution: A robust wifi-based human authentication system via few-shot open-set gait recognition. *IEEE Internet of Things Journal*, 2022.

[18] Jianfei Yang, Xinyan Chen, Han Zou, Dazhuo Wang, and Lihua Xie. Autofi: Towards automatic wifi human sensing via geometric self-supervised learning. *IEEE Internet of Things Journal*, 2022.