AI-based Acoustic Defect Detection for Speaker Manufacturing

Yitong Lu
Stanford University
yitonglu@stanford.edu

Abstract

In this work, we develop a robust AI-based system to detect speaker defects in an acoustic device manufacturing pipeline. Our solution replaces manual auditory inspection with a spectrogram-based classification model, trained and deployed in real factory settings. The project progresses from a simple CNN baseline trained on 300 manually labeled samples, to a self-supervised learning pipeline utilizing over 8000 unlabeled samples. Through rigorous data collection, selective audio augmentation, and fine-tuning with conservative relabeling standards, we improved classification accuracy from 94.27% to 96.43%, and eliminated false positives. Our model integrates seamlessly into SoundCheck, enabling microphone-based real-time classification on a laptop with latency under 1 second. This work demonstrates the feasibility of deep learning–driven acoustic quality control, offering significant time savings and improved reliability in factory workflows.

1. Introduction

Speaker defect detection is a critical task in acoustic device manufacturing. This project aims to develop a robust AI-based system to replace manual inspections, reduce human fatigue, and improve consistency. The system processes raw audio recordings and classifies them into normal or abnormal categories using spectrogram-based features.

This project is a real-world application in collaboration with an OEM manufacturing company. The AI system will be integrated into the production line of a mid-range speaker model. We estimate that it will reduce annual inspection costs by approximately \$3.5k per production line and decrease the inspection time per unit from 5 seconds to just 1 second. In addition, it reduces training time, minimizes inconsistency among inspectors, and mitigates errors caused by fatigue and inexperience. These human-induced errors—especially those stemming from fatigue or insufficient expertise—are significant, as we observed substantial discrepancies between judgments made by factory workers and those made by professional audio engineers.

2. Related Work

After careful investigation, we found that there is currently no open-source model specifically designed for speaker defect detection. This may be due to a prevailing belief that traditional signal processing techniques are more reliable than AI-based methods in this domain. In conversations with professionals at companies such as AIZIP and NexaAI—both of which specialize in audio AI—they confirmed that, as of now, no robust solutions exist for this task. For large tech companies, this problem likely falls outside their core priorities within the supply chain. Meanwhile, startup companies often lack access to sufficient real-world factory testing data to develop effective models. Given this context, I believe my project represents a valuable and meaningful exploration into an underdeveloped yet impactful area.
Currently, our speaker testing process relies on a **sweeping frequency** method, where an input signal is played from low to high frequency, and the resulting sound is analyzed. We have two primary approaches for evaluating speaker quality:

- **Manual auditory inspection:** Evaluation is performed by factory workers. The reliability of this method depends heavily on the workers' expertise and factory management, as discussed earlier.
- **Automated quality control systems:** Many factories utilize commercial QC systems from companies such as Klippel and SoundCheck to detect speaker defects. For example, the diagram on the right illustrates SoundCheck's detection algorithm. It assesses perceptual distortion by analyzing the harmonic structure of the output signal and applying PEAQ (Perceptual Evaluation of Audio Quality) to produce a perceptual distortion curve (RB/PRB), which reflects human auditory perception of anomalies.

  However, these systems also face several limitations:

  – Thresholds must be manually configured by engineers; inaccurate settings can result in high false positive rates.

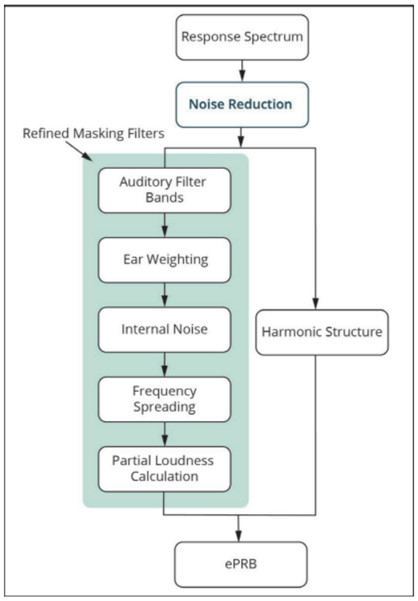

Figure 1. SoundCheck's detection algorithm

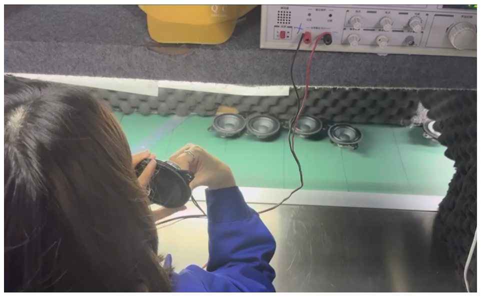

Figure 2. worker check the speaker quality by manual inspections

- The system is sensitive to background noise and may fail to detect subtle anomalies or distortions.
- The use of sweep-tone signals can be time-consuming, impacting production efficiency.

For our project, although the main model was trained independently, we utilized pretrained EfficientNetV2 and ResNet-18 models for feature extraction during our experiments.

3. Data

All of our data were collected directly from a real factory production line. Each sample is a WAV audio file approximately one second in length. No data were sourced online.
The labeling follows a standard binary classification scheme: **OK** (quality passed) and **NG** (not good). We also have a third label, **Not sure**, which indicates disagreement among engineers regarding the quality of a sample.
Labeling proved to be the most difficult and time-consuming part of the project. The process is inherently

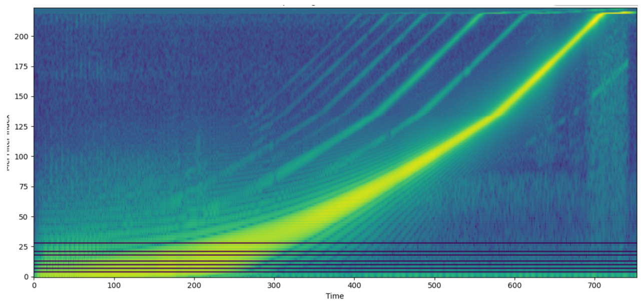

Figure 3. spectrogram with 5 bands merges together

subjective, and audio quality is a continuous property — there is no clear-cut boundary between good and bad. Another major challenge was dataset imbalance, as the factory naturally produces far more good samples than defective ones. To address this, I invested significant effort in curating a dataset of 1000 samples with an approximately balanced 50-50 distribution between OK and NG labels, resulting in a relatively reliable and fair training set.
We also collected a set of 8,000 unlabeled samples, most of them are suppose to belong to "OK", which proved valuable for self-supervised learning. Some of these samples originate from different types of speakers, providing additional diversity to the dataset.

3.1. Data Preprocessing

One of the most important components of our pipeline is the transformation of raw WAV audio files into spectrograms. This conversion allows us to leverage a wide range of deep learning techniques from the computer vision domain to train our binary classification model effectively.
To convert raw audio into spectrograms, we apply a Mel-scale transformation that closely aligns with human auditory perception. Specifically, we divide the frequency range into four bands with varying Mel filter bank resolutions:

- **0–250 Hz:** 4 filter banks
- **250–6000 Hz:** 130 Mel filter banks
- **6000–16000 Hz:** 85 Mel filter banks
- **16000–48000 Hz:** 5 filter banks

This filter configuration places higher resolution in perceptually important frequency ranges, allowing us to preserve more detailed information where human hearing is most sensitive.

4. Methods

4.1. Supervised Learning

The first version of the model was trained on a dataset consisting of 200 **OK** samples and 100 **NG** samples. To address the class imbalance, the **NG** samples

Figure 4. the speakers we using to generate training data

were oversampled to create a balanced training set. The model architecture consisted of two convolutional layers, each followed by max pooling, and a series of fully connected layers, culminating in a softmax output over two classes. This baseline model achieved an initial accuracy of 94.29%.

4.2. Data Augmentation

To further improve performance, I expanded the dataset by collecting and carefully labeling additional samples to maintain class balance.
I also implemented three types of audio data augmentation:

- **Volume scaling:** Adjusting amplitude using 4 filter banks.
- **Background noise addition:** Using 7 real background noise recordings collected from the factory production line.
- **Random trimming:** This is particularly tricky, as the original audio clips are only one second long. I allowed a maximum trim fraction of 0.05 at the beginning and end of each clip—longer trimming risks removing essential information.

Other commonly used audio augmentation techniques, such as time stretching, frequency shifting, or spectral distortion, were deliberately avoided. Since the goal of this model is to analyze the healthiness of the frequency distribution, such augmentations could fundamentally alter the spectral content and render the augmented samples unreliable.

4.3. Self-supervised Learning

Next, I experimented with open-source pretrained models such as **EfficientNetV2** and **ResNet18**. I fine-tuned

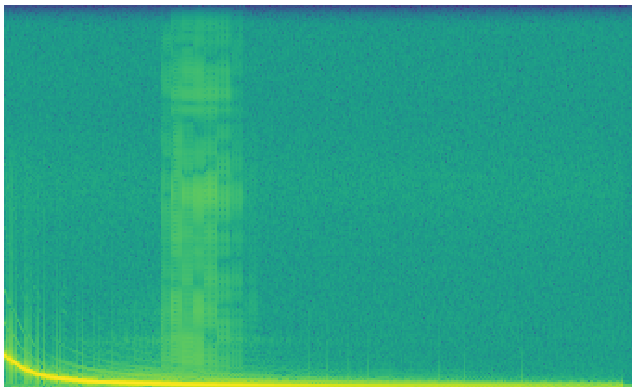

Figure 5. spectrogram from OK sample

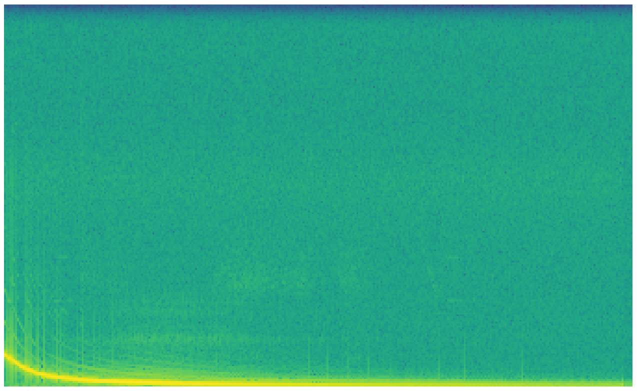

Figure 6. Spectrogram from NG sample

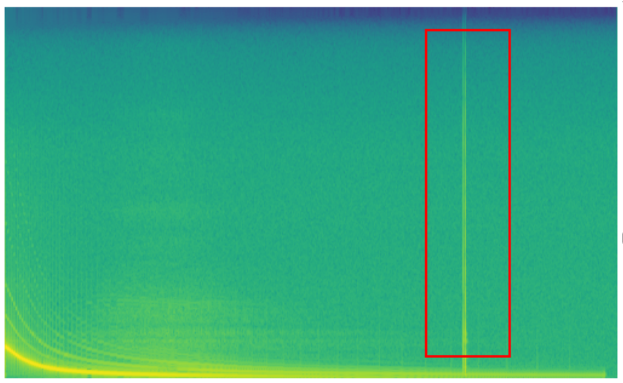

Figure 7. Spectrogram from OK sample with background noise

these models using both the original and augmented versions of my dataset. However, the classification accuracy remained around **94%**, and in some cases, particularly with data augmentation, the accuracy dropped to approximately **80%**.

As a last resort, I collected a large amount of **unlabeled data** to expand the dataset as much as possible. This included samples from different types of speakers to increase diversity. Rather than relying on general-purpose pretrained models, I trained a custom upstream model using **self-supervised learning**.

In this approach, I divided each spectrogram image into a 10×5 grid. Then I randomly masked 10%, 20%, or 30% of the grid cells and trained the model to **reconstruct** the original spectrogram. The loss function was defined as the **pixel-wise difference** between the original and reconstructed spectrogram images.

$$\mathcal{L}_{\text{L1}} = \frac{1}{3HW} \sum_{c=1}^{3} \sum_{i=1}^{H} \sum_{j=1}^{W} \left| \hat{S}_{c,i,j} - S_{c,i,j} \right| \tag{1}$$

L1 loss are used because spectrograms often have smooth regions with sparse high-energy components (like curves+). L1 loss preserves sharp edges and fine structures better than MSE. MSE would tend to blur those clear lines, which I don't want.

This self-supervised pretraining strategy encouraged the model to learn **spectrogram-specific patterns** relevant to our task, resulting in a more specialized and effective feature extractor.

5. Experiments

The development of a robust model for speaker defect detection was a gradual, iterative process. I began by training a simple convolutional neural network (CNN) on a small dataset of 200 OK and 100 NG samples. To address class imbalance, I oversampled the NG class. This baseline model, composed of two convolutional layers followed by max pooling and fully connected layers with a softmax output, achieved a promising accuracy of 94.27%.

Motivated by this result, I fine-tuned a pretrained ResNet18 on a larger, balanced dataset of 500 OK and 500 NG samples. However, the accuracy slightly decreased to 93.07%, indicating that off-the-shelf pretrained features may not be well-aligned with the domain-specific patterns in spectrograms.

I then experimented with EfficientNetV2, fine-tuning it on a heavily augmented dataset comprising 2500 augmented samples per class. Unfortunately, the accuracy dropped sharply to 80.5%, suggesting that aggressive augmentation distorted the frequency structure critical for classification.

5.1. Lower False Positive

Realizing that further training epochs led to overfitting and that the accuracy plateaued around 94%, I suspected that the bottleneck was no longer architectural, but rather inherent to the quality of the training data. I began to believe that my dataset contained latent inconsistencies and ambiguous labeling that limited model performance.

To address this, I thoroughly revisited the dataset. I reclassified any samples in the OK group that exhibited even subtle signs of defects—such as short, faint glitches that are easy for humans to overlook—into the NG group. Additionally, I re-labeled all samples previously marked as "not sure" into the NG category. This effectively imposed a much stricter definition of quality and introduced a more conservative, clearly defined boundary between OK and NG classes. As a result, the labeling standard became both visually sharper and more consistent across the dataset.

Using this refined labeling strategy, I manually constructed a new dataset consisting of 450 OK and 750 NG samples. Training a CNN on this improved dataset led to a notable boost in performance, achieving an accuracy of 95.23%. Remarkably, this stricter definition of NG also completely eliminated false positives in the test set.

To further improve performance, I designed a self-supervised learning strategy. I collected a large corpus of unlabeled data, including recordings from various speaker types. Each spectrogram was divided into a 10×5 grid, and 10%, 20%, or 30% of the grid patches were randomly masked. A model was then trained to reconstruct the missing regions using a pixel-wise L1 loss. This self-supervised model was subsequently fine-tuned on the labeled data, yielding the highest accuracy of 96.67%. This step successfully that self-supervised learning outperform generalized pretrained model.

This step-by-step progression demonstrated that model success was not solely determined by architecture complexity, but by data quality, domain alignment, and the ability to learn meaningful spectrogram-specific representations.

5.2. Failure Case Analysis

During model evaluation, we identified a class of failure cases in which the model consistently misclassified borderline samples—such as the spectrogram shown in Figure8 as OK, despite subtle but real signs of acoustic defects. These spectrograms typically exhibit a near-normal harmonic structure but contain faint distortions, short-lived dropout bands, or slightly irregular harmonics that are easily overlooked by human annotators.

Initially, these samples were labeled as OK due to their overall visual similarity to clean signals. However, upon closer inspection and repeated listening, we recognized that such signals are indicative of mild speaker anomalies, such as intermittent leakage or high-frequency instability. To address this ambiguity, we reclassified these edge cases from OK to NG and also reassigned all "not sure" samples to the NG category, thereby enforcing a stricter and more conservative labeling standard.

In the specific case shown in Figure8, after converting the waveform into a spectrogram, we observed a pronounced spread of magnitude in the first few milliseconds. This spread, concentrated in the low-frequency region, is difficult to detect by ear due to its subtlety and short duration. However, after consulting with an experienced audio

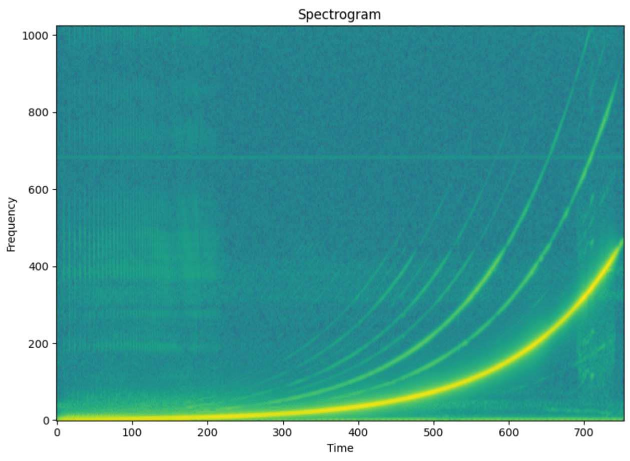

Figure 8. An sample that used to have OK label and then changed to NG

engineer, we determined that this behavior is indeed symptomatic of a defect and should be labeled as NG under our revised criteria.

This stricter labeling strategy led to more consistent training signals and improved the model's overall accuracy and robustness. In particular, it eliminated false positives entirely in the final CNN model trained on the relabeled dataset.

5.3. Generalization and Limitations

To evaluate the generalizability of our model, we conducted tests on audio data from two additional speaker types not present in the final supervised training set, one tiny size speaker and another type of mid-speaker. The model maintained high accuracy on both 87%, demonstrating a potential good transferability across speaker hardware variants. This suggests that the spectrogram-based features learned by the network might be robust to changes in speaker response characteristics. In the future, a one-to-all model to test different kinds of speaker might be possible.

However, the model was found to be somewhat sensitive to background noise conditions. In particular, factory recordings with excessive ambient machinery noise or overlapping human voices occasionally caused classification instability. This indicates a limitation in the diversity of environmental conditions represented in the training data. Although data augmentation with backgournd noise is tried, its shows more harmfulness than usefulness in our cases.

5.4. Result Table

- Model 1
  - Model: CNN
  - Training set: 200 OK + 100 NG
  - Accuracy: 94.27%
  - False Positive: $6.67\%(1/15)$
- Model 2
  - Model: ResNet18 + warmup + fine-tune
  - Training set: 500 OK + 500NG
  - Accuracy: 93.07%
  - False Positive: $10.67\%(8/75)$
- Model 3
  - Model: EfficientNetV2 + warmup + fine-tune
  - Training set: 500 OK+ 2500 OK(Augmented) + 500NG + 2500 NG(Augmented)
  - Accuracy: 80.5%
  - False Positive: $22.00\%(99/450)$
- Model 4
  - Model: EfficientNetV2 + warmup + fine-tune
  - Training set: 450 OK+ 750NG
  - Accuracy: 95.23%
  - False Positive: $0\%(0/112)$
- Model 5
  - Model: Self-supervised Vision Pretraining with Local Masked Reconstruction + CNN
  - Training set: 1. self-supervised phase: 5000 unlabeled data; 2.Supervised phase: 450 OK+ 750NG
  - Accuracy: 96.43%
  - False Positive: $0\%(0/112)$

6. Implementation

I have successfully integrated my model into **Soundcheck**, which supports real-time audio input directly from a microphone. The complete pipeline is as follows:

Speaker arrives $\rightarrow$ Microphone captures audio $\rightarrow$ Convert to spectrogram $\rightarrow$ Binary classification (with minimized false positives) $\rightarrow$
• If classified as **OK**: proceed to the next production step
• If classified as **NG**: the sample is flagged for human double-check

Since the model is designed to minimize false positives, it tends to be conservative and may occasionally overkill—classifying borderline OK samples as NG. Therefore, a manual review is still required for NG outputs. However, given that NG cases are naturally rare in the production line, this approach still significantly reduces inspection time.

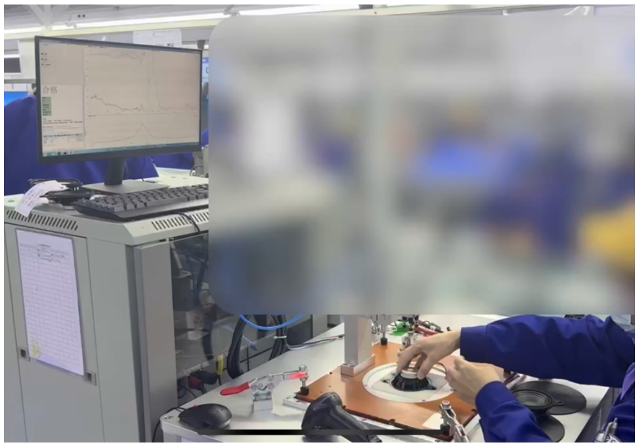
Figure 9. Worker using computer checking quality of speaker

Unlike many embedded machine learning deployments, we did not face major constraints regarding quantization, latency, or model size. The factory does not require us to deploy the model on low-power devices such as Raspberry Pi. Instead, our "embedded" environment is a laptop, which is compact enough for mobile production lines and powerful enough for real-time processing. On a local laptop, the model runs consistently and completes inference in under 1 second. This setup demonstrates that our goals of time-saving and efficiency improvement are practically achievable.

I will demonstrate the entire pipeline—from sound input to real-time decision output—during my poster session.

Looking ahead, I plan to collaborate further with the factory this summer to explore the possibility of real-time calibration in the actual production environment and evaluate whether our target efficiency gains can be fully realized.

Conclusion

In this project, I developed and deployed a machine learning system for detecting speaker defects through spectrogram analysis. Starting with a simple CNN model and a small, imbalanced dataset, I gradually improved performance through systematic iterations in data collection, labeling quality, model architecture, and training methodology. Along the way, I identified the limitations of generic pretrained models and data augmentation techniques, and ultimately adopted a self-supervised learning approach that enabled the model to learn spectrogram-specific features more effectively.

Careful relabeling and stricter quality standards further enhanced model consistency, leading to an accuracy of 95.23% with zero false positives in the final CNN model. The entire system was successfully integrated into a real-time production pipeline using Soundcheck, enabling fast, reliable classification with inference time under one second on a local laptop.

This pipeline demonstrates the feasibility of applying deep learning in industrial audio quality inspection. Looking ahead, I plan to continue collaborating with the factory to validate this system in actual production settings and explore real-time calibration strategies to achieve measurable cost and time savings.

References

- Defard, Thomas et al. "PaDiM: A Patch Distribution Modeling Framework for Anomaly Detection and Localization." *arXiv preprint arXiv:2011.08785*, 2021.
- Tan, Mingxing and Le, Quoc V. "EfficientNetV2: Smaller Models and Faster Training." *arXiv preprint arXiv:2104.00298*, 2021.
- Chen, Jun et al. "Efficient Self-supervised Vision Pretraining with Local Masked Reconstruction." *arXiv preprint arXiv:2206.00790*, 2025.