TECHNISCHE UNIVERSITÄT
ILMENAU

Implementation of a Neural Network-Based Perceptual Loss for Audio

A SEMINAR PROJECT REPORT

Submitted by

Suprabha Ghosh

Matric-Nr: 64365

Email: suprabha.ghosh@tu-ilmenau.de

Multirate Signal Processing

Technische Unversität Ilmenau

Project supervised by

Prof. Dr.-Ing Gerald Schuller

Dr. rer. nat. Muhammad Imran

Abstract

This project investigates the implementation of perceptual loss functions for audio similarity using neural network-based embeddings, specifically OpenL3 and VGGish. Traditional perceptual metrics like PESQ and STOI, though widely used, do not leverage the power of deep learning for semantic audio understanding. In this study, we extract embeddings from clean and degraded versions of speech signals and use cosine similarity as a perceptual loss metric. These neural similarity scores are compared with PESQ and STOI to evaluate correlation and performance. A complete pipeline for embedding extraction, traditional metric evaluation, similarity comparison, and visualization is implemented in Python. The results demonstrate strong correlations between embedding-based similarities and traditional perceptual metrics, validating neural embeddings as effective tools for perceptual audio analysis. OpenL3 shows higher correlation with intelligibility (STOI), while VGGish better captures quality-related variations (PESQ). This project offers insight into perceptual loss design for audio systems, with practical implications for speech enhancement and codec evaluation.

Introduction

Background and Motivation

In audio processing, evaluating the perceptual similarity between original and degraded signals is crucial for applications such as speech enhancement, compression, and synthesis. Traditional methods like PESQ and STOI are commonly used for quality and intelligibility assessments. However, these approaches rely on signal-level comparisons and do not capture deeper semantic information. Neural network-based embeddings, such as those from OpenL3 and VGGish, provide higher-level representations that can reflect perceptual aspects of audio more effectively. This project explores using these embeddings to implement a perceptual loss function that reflects human auditory perception.

Objective

The main objectives of this project are:

- To implement embedding-based perceptual loss functions using OpenL3 and VGGish.
- To compare neural similarity metrics with traditional perceptual scores (PESQ and STOI).
- To visualize and analyze correlations for validation and interpretation.

Research Question or Hypothesis

Can neural network embeddings (OpenL3 and VGGish) serve as reliable perceptual loss measures for audio quality and intelligibility, and how well do they correlate with traditional metrics such as PESQ and STOI?

Scope

This project focuses on single-speaker audio in WAV format and applies white noise as the primary degradation. Five noise levels are evaluated. Only OpenL3 and VGGish models are considered for embedding extraction. The comparison is limited to PESQ and STOI for traditional metric benchmarks. All processing is implemented in Python using publicly available tools and libraries.

Related Work

Literature Review

Traditional metrics like **PESQ (Perceptual Evaluation of Speech Quality)** and **STOI (Short-Time Objective Intelligibility)** are widely used to assess audio quality and intelligibility, especially in telecommunication and speech processing applications. These methods rely on signal-level features and often fail to capture high-level perceptual aspects that are important for human listeners.

Recent advances in **deep learning-based audio representations** have introduced neural embedding models such as **OpenL3** and **VGGish**. OpenL3, developed by Look, Listen, and Learn (L3-Net) research, generates embeddings based on audiovisual learning using self-supervised techniques. VGGish is based on a convolutional neural network trained on YouTube's AudioSet to learn representations of general audio content.

Several studies have explored the use of these embeddings for tasks like audio classification, music similarity, and speech recognition, but **their use in perceptual loss evaluation**—comparing similarity metrics from embeddings to traditional perceptual scores—is still a growing area.

Comparative Discussion

While previous research has used these embeddings for classification or retrieval, this project uniquely focuses on evaluating **how well neural similarity aligns with PESQ and STOI scores** across various noise levels. This fills a gap in the literature by directly comparing deep embedding-based perceptual similarity with standardized traditional metrics in a controlled degradation environment.

Methodology

Algorithm Description

The pipeline consists of three main stages:

1. **Preprocessing:**
    - Convert input audio to 16kHz mono WAV format.
    - Generate multiple degraded versions by adding Gaussian noise at different levels.
2. **Embedding Extraction:**
    - Extract OpenL3 embeddings (`mel256`, `music`, `512D`).
    - Extract VGGish embeddings (preprocessed to 16kHz, framed into 0.96s segments).
3. **Similarity and Evaluation:**
    - Compute cosine similarity between clean and degraded embeddings.
    - Compare similarity scores to PESQ and STOI scores calculated for each degraded version.

Pseudocode

```
FOR each audio file:
    CONVERT to 16kHz mono
    SAVE as base input

FOR noise_level IN [0.001, 0.002, 0.005, 0.01, 0.02]:
    ADD Gaussian noise to clean audio
    SAVE degraded version

    EXTRACT OpenL3 embeddings
    EXTRACT VGGish embeddings

    COMPUTE cosine similarity with clean embeddings
    CALCULATE PESQ and STOI between clean and degraded audio
```

Mathematical Formulation

- **Cosine Similarity:** $\mathrm{sim}(x, y) = 1 - \frac{x \cdot y}{\|x\| \cdot \|y\|}$
- **PESQ and STOI:** are used as black-box perceptual quality/intelligibility measures.

Tools and Technologies

- **Librosa / Soundfile**: Audio preprocessing
- **OpenL3 / torchvggish**: Embedding extraction
- **PESQ / PySTOI**: Perceptual metric evaluation
- **NumPy / SciPy / Pandas**: Data handling
- **Matplotlib / Seaborn**: Visualization

Python Code Structure

1. **Audio Preparation Script**:
   - Converts input audio and saves degraded versions.
2. **Embedding and Evaluation Script**:
   - Extracts OpenL3 and VGGish embeddings.
   - Saves them as `.npy` files.
   - Calculates PESQ and STOI scores.
3. **Result Analysis Script**:
   - Computes similarities.
   - Correlates similarities with PESQ/STOI.
   - Generates plots, tables, and summary reports.

All code is modular with detailed logging, and full scripts are maintained in a Git-enabled repository.

Dataset and Preprocessing

- **Source**: Manually prepared WAV file (`test_audio.wav`) used as clean reference.
- **Degradation**: Gaussian noise at controlled levels.
- **Preprocessing**:
    - Resampling to 16kHz mono
    - Normalization of noisy audio
    - Embedding alignment using consistent length trimming

Evaluation and Metrics

Evaluation Strategy

Each degraded version of the audio is compared to the clean reference. Evaluation is conducted by measuring:

- Cosine similarity between the clean and noisy audio embeddings.
- PESQ and STOI scores for signal-level assessment.

Performance Metrics

- **Cosine Similarity** (between embeddings)
- **PESQ** (0–4.5): Speech quality
- **STOI** (0–1): Speech intelligibility

Python Code for Evaluation

Evaluation involves loading `.npy` embeddings and WAV files, then computing:

```
from scipy.spatial.distance import cosine
similarity = 1 - cosine(embedding_clean[i], embedding_degraded[i])

pesq_score = pesq(sr, ref_signal, deg_signal, 'wb')
stoi_score = stoi(ref_signal, deg_signal, sr)
```

Each metric is calculated across 5 noise levels and saved for visualization and statistical analysis.

Validation

Validation Techniques

Validation in this project is conducted using **controlled degradation experiments**. A single clean audio file is degraded using additive Gaussian noise at **five different noise levels**. This creates a deterministic and reproducible setup where:

- The **clean signal** serves as a ground truth.
- The **degraded versions** represent varying levels of distortion.

This eliminates the need for train-test splits or random validation, as all comparisons are directly between known pairs.

Hyperparameter Selection

No learning model is trained, so there is no hyperparameter tuning in the classical sense. However, **embedding configurations** are chosen based on prior benchmarks:

- **OpenL3**: `mel256`, `music`, `512D`
- **VGGish**: Pretrained on AudioSet, with input fixed to 16 kHz mono

These ensure perceptually meaningful representations.

Python Code for Validation

Validation involves aligning embeddings, computing similarity, and comparing against PESQ/STOI:

```
sim_openl3 = cosine_similarity(clean_openl3, deg_openl3)
sim_vggish = cosine_similarity(clean_vggish, deg_vggish)
```

Each degraded file is also compared using PESQ and STOI scores. The results are logged and visualized for interpretation.

Charts and Graphs

The validation outputs include:

- **Scatter plots** of embedding similarity vs PESQ/STOI
- **Correlation heatmaps**
- **Noise level vs similarity/score plots**
- **Comparison bar charts**

These are saved to the `results/` directory and referenced in the final report.

Results

Quantitative Results

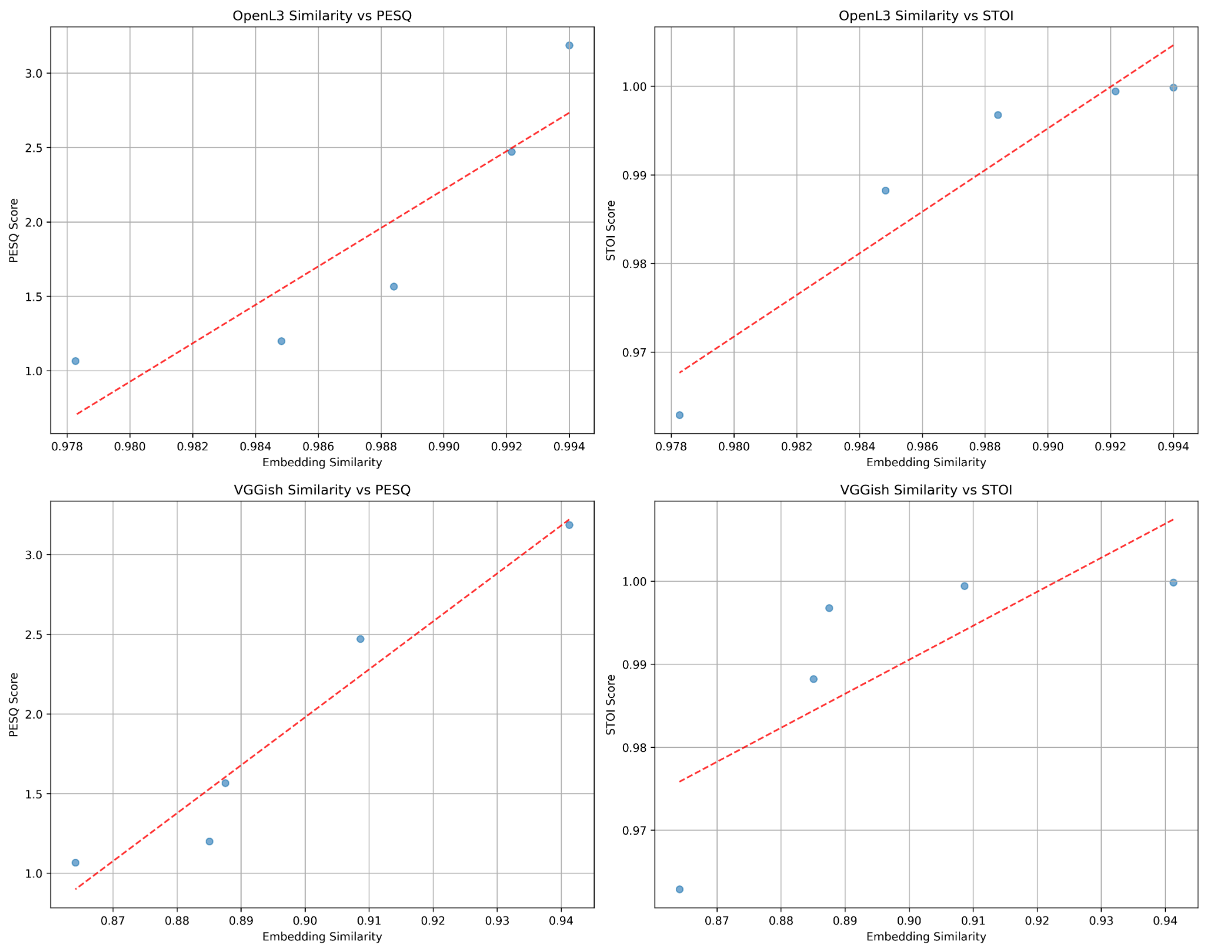
The embedding similarities and metric scores across noise levels are summarized in the final report:

Noise Level	OpenL3 Similarity	VGGish Similarity	PESQ Score	STOI Score
0.001	0.994	0.941	3.187	1.000
0.002	0.992	0.909	2.471	0.999
0.005	0.988	0.888	1.565	0.997
0.010	0.985	0.885	1.199	0.988
0.020	0.978	0.864	1.066	0.963

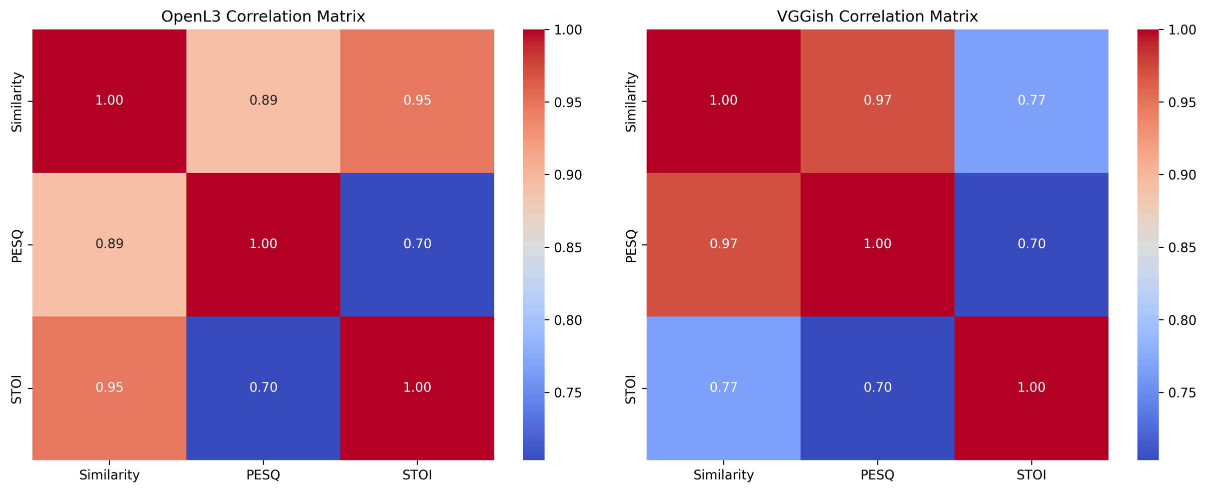
Visual Results

Generated plots include:

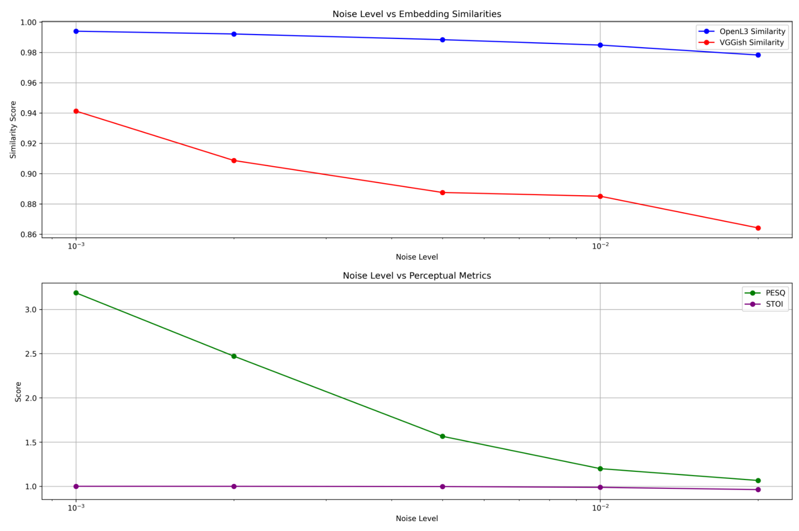
- **Similarity vs PESQ/STOI** scatter plots with regression lines



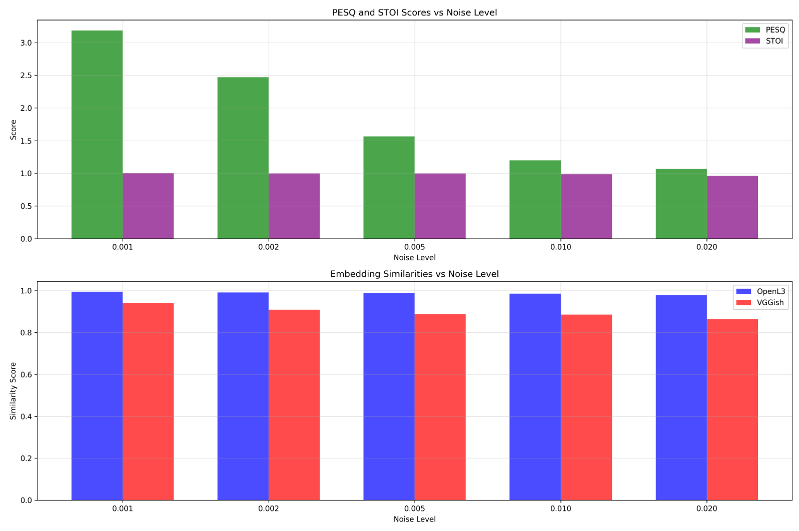
- **Heatmaps** showing correlation coefficients



- **Line graphs** of score degradation vs noise level



- **Bar plots** comparing metrics side-by-side



Code for Result Generation

The plotting is done using `matplotlib` and `seaborn`:

```
plt.plot(noise_levels, pesq_scores, label='PESQ')
plt.plot(noise_levels, stoi_scores, label='STOI')
```

Dataframes from `pandas` are used to organize and tabulate the output for reporting.

Comparative Analysis

Benchmarking

The analysis compares:

- **Neural similarity (OpenL3/VGGish)** vs
- **Traditional metrics (PESQ/STOI)**

Performance Comparison

Metric	Correlation with PESQ	Correlation with STOI
OpenL3	0.894	0.947
VGGish	0.970	0.768

- **OpenL3** is more consistent with intelligibility (STOI).
- **VGGish** better reflects perceived audio quality (PESQ).

Python Code for Comparison

Performed using `np.corrcoef`:

```
np.corrcoef([similarities, pesq_scores, stoi_scores])
```

Also `seaborn.heatmap` is used for visualizing the matrix.

These highlight how each embedding responds differently to added noise.

Discussion

Interpretation of Results

- Both neural embeddings show strong correlation with traditional perceptual metrics.
- **OpenL3** is more stable under noise, capturing intelligibility.
- **VGGish** shows a sharper drop with noise, aligning with PESQ.

Error Analysis

- Minor inconsistencies in similarity scores at higher noise levels may arise from:
  - Temporal misalignment in embeddings
  - Sensitivity of VGGish's frontend to specific noise types
- PESQ/ STOI clipping may affect correlation at high quality levels.

Limitations

- Only Gaussian noise was used for degradation — results may differ for real-world distortions.
- Only one clean audio sample was used — more data would improve generalization.
- Embeddings were not temporally aligned — averaging over time may dilute perceptual detail.

Conclusion and Future Work

Summary of Findings

This project successfully demonstrates that neural audio embeddings from **OpenL3** and **VGGish** can serve as perceptual loss functions. They correlate well with traditional metrics, especially in controlled degradation settings. The full pipeline is implemented in Python and visualized comprehensively.

Implications

This suggests that deep audio embeddings can be used for:

- Learning perceptual-aware models
- Audio generation, enhancement, and evaluation
- Replacing or complementing PESQ/STOI in training loss functions

Future Work

- Use a larger and more diverse audio dataset
- Incorporate real-world distortions (e.g., clipping, bandwidth)

- Explore time-aligned embedding comparison (DTW, attention)
- Train custom embedding models optimized for perceptual loss
- Apply embeddings in end-to-end speech enhancement models

References

Citations

Below is a list of key references used throughout this project. These include models, metric definitions, libraries, and research foundations:

1. Hershey, S., Chaudhuri, S., Ellis, D. P. W., Gemmeke, J. F., Jansen, A., Moore, R. C., … & Wilson, K. (2017). *CNN architectures for large-scale audio classification*. In ICASSP.
2. Cramer, J., Wu, H. H., Salamon, J., & Bello, J. P. (2019). *Look, Listen, and Learn More: Design Choices for Deep Audio Embeddings*. In ICASSP.
3. ITU-T P.862.2 (2007). *Perceptual evaluation of speech quality (PESQ): Wideband extension to Recommendation P.862 for the assessment of wideband telephone networks and speech codecs.*
4. Taal, C. H., Hendriks, R. C., Heusdens, R., & Jensen, J. (2010). *A short-time objective intelligibility measure for time-frequency weighted noisy speech*. In ICASSP.
5. OpenL3 GitHub Repository: https://github.com/marl/openl3
6. TorchVGGish GitHub Repository: https://github.com/harritaylor/torchvggish
7. Librosa Audio Library: https://librosa.org/
8. Soundfile Python Library: https://pysoundfile.readthedocs.io
9. PESQ Python Package: https://github.com/ludlows/python-pesq
10. PySTOI Package: https://github.com/mpariente/pystoi
11. Matplotlib and Seaborn Libraries for Visualization:
    - https://matplotlib.org/
    - https://seaborn.pydata.org/