EE 413: Applied Digital Signal Processing

Project: Audio Signal Classification

Abdullah F. Al-Battal, PhD

Department of Electrical Engineering
King Fahd University of Petroleum and Minerals

Spring, 2025 (242)

Project Overview

- **Focus:** Audio signal classification using DSP techniques
- **Team structure:** Groups of 4 students
- **Duration:** 2-3 weeks
- **Dataset:** Free-Spoken Digit Dataset (FSDD)
- **Application of course topics:**
  - Topic 1: Signals and System Review
  - Topic 2: Discrete Fourier Transform and Applications
  - Topic 3: Wavelet Transform
- **Key deliverables:**
  - GitHub code repository
  - Team presentation
  - Teammates evaluation
  - Peer evaluation

Learning Objectives

By completing this project, you will:

- Apply theoretical DSP concepts to real-world audio signals
- Implement and compare multiple signal processing techniques:
    - Time-domain analysis
    - Frequency-domain analysis (DFT/STFT)
    - Time-frequency analysis (Wavelet Transform)
- Design and evaluate feature extraction methods for classification
- Gain hands-on experience with Python for audio signal processing
- Develop technical presentation and collaborative coding skills
- Learn professional GitHub usage for code sharing and documentation

Dataset: Free-Spoken Digit Dataset (FSDD)

Dataset characteristics:

- "MNIST of audio"
- Spoken digits (0-9)
- Multiple speakers
- Multiple recordings per digit
- WAV files (8kHz mono)
- Short recordings (approximately 1 second each)
- Clean audio without background noise

Download: `https://github.com/Jakobovski/free-spoken-digit-dataset`

Advantages:

- Well-structured for classification tasks
- Allows focus on DSP technique comparison
- Manageable dataset size
- Clear ground truth labels
- Speaker variability provides interesting challenge
- Widely used benchmark in audio ML

Project Requirements: Signal Analysis

Implement at least three feature extraction approaches:

1. **Time-Domain Analysis**
   - Extract statistical features (zero-crossing rate, energy, etc.)
   - Implement basic filtering techniques
   - Apply signal envelope extraction
   - Analyze temporal patterns in speech signals
2. **Frequency-Domain Analysis**
   - Implement Short-Time Fourier Transform (STFT)
   - Extract spectral features (centroid, bandwidth, etc.)
   - Address practical DFT issues (windowing, zero-padding)
   - Consider formant analysis for digit recognition
3. **Wavelet Transform Analysis**
   - Implement discrete wavelet transform
   - Extract wavelet-based features
   - Compare with STFT results
   - Analyze time-frequency resolution benefits

Project Requirements: Classification

1. **Classification System**
   - Develop a classification system using extracted features
   - Implement a simple machine learning classifier
   - Use appropriate evaluation metrics
2. **Analysis and Comparison**
   - Compare effectiveness of different feature extraction methods
   - Identify which digits are easier/harder to classify
   - Analyze how DSP techniques affect classification performance
   - Discuss trade-offs between methods

Timeline and Deliverables

Week 1:

- Dataset download and exploration
- Time-domain analysis implementation
- Initial feature extraction
- Set up GitHub repository

Week 2:

- Frequency-domain analysis
- Wavelet transform implementation
- Refine feature extraction
- Develop classification system

Week 3:

- Complete performance evaluation
- Comparative analysis
- Finalize GitHub repository
- Prepare presentation

Final Deliverables:

- GitHub code repository
- 10-minute presentation + 5 min Q&A
- Individual Contribution Statement and Teammate Evaluation
- Peer Evaluation Forms

GitHub Repository Requirements

- **Repository Structure:**
  - Well-organized directory structure
  - Clear README.md with project description and setup instructions
  - requirements.txt with all dependencies
- **Code Quality:**
  - Well-commented Python code
  - Modular design with clear function/class responsibilities
  - Error handling and edge cases considered
- **Documentation:**
  - Function docstrings explaining purpose, parameters, and returns
  - Markdown files explaining methodology
  - Sample results and visualizations
  - Usage examples

Presentation Guidelines

10-minute presentation + 5 minutes Q&A

- All team members must participate equally
- Required content:
  - Introduction (1-2 minutes)
  - Time-domain analysis approach (1-3 minutes)
  - Frequency-domain analysis approach (1-3 minutes)
  - Wavelet transform approach (1-3 minutes)
  - Classification results and comparison (1-3 minutes)
  - Conclusions and future work (1 minute)
- Include relevant visualizations
- Demonstrate code functionality with examples
- Be prepared to explain implementation details
- Focus on DSP concepts and their application

Evaluation Criteria

Presentation (70%):

- Technical content and depth (20%)
- Implementation details (15%)
- Results analysis (15%)
- Visualizations and demos (10%)
- Delivery and organization (10%)

Code Quality (30%):

- Correctness and completeness (10%)
- Code organization (5%)
- Documentation (5%)
- Reproducibility (5%)
- GitHub organization (5%)

Note: Individual grades will be adjusted based on peer evaluations, teammate evaluations and presence during other teams presentations (submitting peer evaluations).

Teammate Evaluation Process

- Each team member will complete a confidential teammate evaluation form
- Evaluations will assess:
  - Technical contribution
  - Reliability and meeting attendance
  - Teamwork and communication
  - Initiative and leadership
  - Problem-solving abilities
- Team members will indicate contribution for each member
- Individual grades may be adjusted up or down based on evaluations
- **Maximum adjustment factor:** 0.7 to 1.1 of team grade capped at the project full mark score [Minimum factor can can be further lowered below 0.7 if temamates report lack of participation of a specific teammate that is validated/proven]
- Self-evaluation will be compared with peer assessments

Peer Evaluation Process

- Each student will complete a project evaluation form that is completed and submitted during presentations.
- The project evaluation form will follow the presentation evaluation criteria.
- Each student will give a score ranging from 1 (lowest) to 5 (highest) depending on the quality of the project being evaluated.
- Maximum adjustment factor: 0.9 to 1.1 of team grade capped at the project full mark score.

Individual Contribution Statement

- Each team must submit one document detailing:
  - Breakdown of tasks and responsibilities
  - Specific contributions of each team member
  - Percentage allocation of work
  - Description of team dynamics and challenges
- The statement should be agreed upon by all team members
- This statement will be compared with individual peer evaluations
- Significant discrepancies may result in follow-up discussions
- Be honest and fair in your assessments

Implementation Examples

Time-Domain Features:

- Zero-crossing rate
- Root mean square (RMS) energy
- Temporal envelope
- Autocorrelation coefficients

Frequency-Domain Features:

- Mel-frequency cepstral coefficients (MFCCs)
- Spectral centroid and bandwidth
- Formant frequencies
- Harmonic-to-noise ratio

Wavelet Features:

- Wavelet energy by decomposition level
- Wavelet entropy
- Statistical measures of wavelet coefficients
- Relative wavelet energy

Visualization Examples:

- Waveform plots
- Spectrograms
- Scalograms (wavelet visualizations)
- Feature distribution plots
- Confusion matrices

Potential Challenges

Common Challenges:

- Speaker variability in pronunciation
- Selecting optimal window sizes for STFT
- Choosing appropriate wavelet basis functions
- Feature selection and dimensionality
- Finding discriminative features for similar-sounding digits
- Balancing computation efficiency with feature quality

Tips for Success:

- Start with exploratory data analysis to understand the dataset
- Implement simpler techniques first before advancing to complex ones
- Create visualization tools early to help with debugging
- Test on small subsets before scaling to the full dataset
- Document your work continuously
- Maintain regular team communication

Technical Guidance

- **Python Libraries:**
  - `librosa`: Audio processing and feature extraction
  - `numpy` and `scipy`: Numerical computations
  - `pywavelets`: Wavelet transform implementation
  - `matplotlib`: Visualization
  - `scikit-learn`: Machine learning algorithms
- **DSP Concepts to Focus On:**
  - Sampling and quantization effects
  - Windowing functions and their impact
  - Spectral leakage and zero-padding
  - Time-frequency resolution trade-offs
  - Multi-resolution analysis with wavelets
- **Remember:** Connect your implementation to course concepts

Important Reminders

- **GitHub Repository:**
  - Create repository early
  - Use meaningful commit messages
  - All team members should contribute
  - Submit final repository link via Blackboard
- **Team Collaboration:**
  - Schedule regular meetings
  - Distribute tasks evenly
  - Document decisions and progress
- **Deadlines:**
  - Presentations: TBD via Blackboard
- **Generative AI Policy:** Check the following slides

Generative AI Tools Policy

Generative AI Tools Policy: Overview

- This project recognizes the growing role of AI tools in our daily lives.
- Rather than prohibiting these tools, we aim to:
  - Provide guidance on appropriate usage
  - Ensure you develop core DSP competencies
  - Balance innovation with academic integrity
  - Prepare you for real-world engineering environments
- **AI tools should enhance your learning, not replace it !**
- The policy establishes clear boundaries for permitted and restricted uses

Permitted Uses of AI Tools

AI tools may be used for:

- Debugging assistance for non-core functionality
  - Example: Finding syntax errors in visualization code
- Help with general programming syntax and best practices
  - Example: Asking about Python file handling or NumPy functions
- Documentation improvement suggestions
  - Example: Enhancing README clarity or function docstrings
- Generating basic visualization code templates
  - Example: Creating matplotlib or seaborn plotting templates
- Explaining concepts to enhance your understanding [**Note:** Be careful of Hallucinations and use reference-based interactions only.]

Restricted Uses of AI Tools

AI tools should NOT be used for:

- Core DSP algorithm implementation
  - You must write your own STFT, wavelet transform implementations
- Feature extraction methods
  - The feature selection and extraction logic must be your own
- Classification logic
  - The machine learning model setup and evaluation must be your work
- Key analytical interpretations
  - Analysis of results and conclusions must reflect your team's thinking

Documentation Requirements

All AI assistance must be transparently documented:

- **In-code documentation:**
  - Add comments before AI-assisted sections:
  - Example: # The following visualization code was created with assistance from ChatGPT
- **GitHub README section:**
  - Create a dedicated "AI Assistance" section that details:
    - Which specific parts received AI assistance
    - What tools were used (e.g., ChatGPT, GitHub Copilot)
    - How the team built upon or modified the AI suggestions
    - How you verified and understood the assisted code
- Honesty about AI usage will not reduce your grade if within permitted scope

Learning Objectives and AI Usage

Remember that the primary goal is your learning:

- **Understanding is paramount**
  - Be prepared to explain every line of code in your project
  - If you can't explain it, you shouldn't be using it
- **AI as a learning tool**
  - Use AI to deepen your understanding, not substitute for it
  - Ask AI to explain concepts, then implement in your own way [**Note:** Be careful of Hallucinations and use reference-based interactions only.]
- **Skill development**
  - The course aims to develop your DSP implementation skills
  - These skills are valuable and can't be outsourced to AI
- **When in doubt**
  - Consult with the instructor about appropriate AI usage
  - Better to ask than risk academic integrity issues

Evaluating AI-Assisted Work

How AI usage will factor into your evaluation:

- **Transparency is valued**
  - Properly documented AI usage within permitted scope won't affect grading
  - Undisclosed AI usage may be considered academic dishonesty
- **During (& After) presentations**
  - You'll be asked questions about implementation details
- **Code review**
  - AI-assisted sections will be reviewed for compliance with policy
  - Focus will be on how you built upon AI suggestions
- **Learning outcomes**
  - Final evaluation focuses on your mastery of DSP concepts
  - Appropriate AI use can and should enhance, not diminish, this mastery

Examples: Appropriate vs. Inappropriate AI Usage

Appropriate:

- Asking AI to explain different window functions for STFT, then implementing yourself
- Using AI to help debug visualization code that isn't displaying correctly
- Getting suggestions for better code organization or documentation
- Asking AI to help format your README more clearly

Inappropriate:

- Asking AI to implement your wavelet transform feature extraction
- Having AI write your classification algorithm
- Using AI to interpret your results and draw conclusions
- Requesting complete code solutions for core project requirements

Important Note

The more central the component is to demonstrating DSP concepts, the less AI assistance should be involved.