L2M

Lyrics-to-Melody Generation Using AI

An Intelligent System Powered by Large Language Models

Arafat Hasan

M.Eng. Project Defense

The Challenge

Converting text into music is hard

- Understanding emotional intent
- Aligning syllables with musical notes
- Creating coherent melodies
- Ensuring musical validity

> *Can AI bridge the gap between language and music?*

Our Solution: L2M

A modular AI system that:

✓ Analyzes lyrical emotion & rhythm
✓ Generates aligned melodies
✓ Exports to standard formats (MIDI, MusicXML, Audio)
✓ Operates robustly with intelligent fallbacks

No training required • Production-ready • Open-source

Example

Input

```
"The sun will rise again"
```

Output

- **Emotion:** Hopeful
- **Tempo:** 90 BPM
- **Key:** G major
- **Melody:** 6 notes perfectly aligned

From words to music in seconds

Agenda

1. Background & Motivation
2. Related Work
3. System Architecture
4. Implementation Details
5. Evaluation & Results
6. Future Work
7. Conclusion

Background & Related Work

Why This Matters

Music composition is complex:

- Requires musical training
- Time-intensive process
- Difficult to express ideas without skills

AI can democratize creativity:

- Enable non-musicians to create
- Augment professional workflows
- Explore new creative possibilities

Related Work Evolution

Year	Approach	Limitation
2018	Seq2Seq LSTMs	Requires large training datasets
2022	ReLyMe (Hybrid)	Partial fallback support
2023	Controllable L2M	Complex, limited robustness
2025	SongComposer	Needs fine-tuning, no fallback

→ Our work: Pre-trained LLM + No training + Full fallback

Research Gap

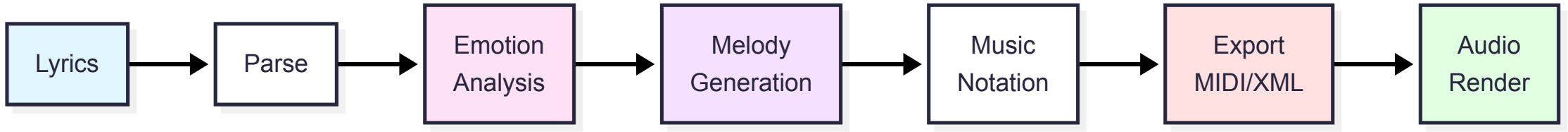
Existing systems lack:

- Accessibility (require training data)
- Robustness (no fallback mechanisms)
- Practical deployment (complex setup)
- Multi-format output

L2M addresses all these gaps

System Architecture

Pipeline Overview



6 modular stages • Type-safe • Fully logged

Stage 1-2: Understanding Lyrics

Lyrics Parsing

- Text normalization
- Syllable estimation
- Phrase segmentation

Emotion Analysis **(LLM-powered)**

- Emotion classification (*happy, sad, hopeful, tense...*)
- Tempo detection (40-200 BPM)
- Time signature selection

Stage 3: Melody Generation

LLM-Based Generation

- Emotion-aware key selection
- Note-per-syllable alignment
- Natural melodic contours
- Chunking for long lyrics (>30 syllables)

Intelligent Fallback

- Deterministic algorithms
- Emotion-to-key mappings

Emotion-to-Key Mapping

Emotion	Musical Key	Tempo	Contour
😊 Happy	C major	100-120	Ascending ↗️
🌅 Hopeful	G major	80-100	Wavy 〰️
😢 Sad	A minor	60-80	Descending ↘️
😰 Tense	D minor	90-110	Erratic ⚡
😌 Calm	F major	60-80	Balanced →

Based on music theory and empirical testing

Stage 4-6: Output Generation

Music Notation (music21)

- Internal representation (IR)
- Tempo & key metadata
- Time signature handling

Export Formats

- **MIDI** (.mid) - Standard sequencer format
- **MusicXML** (.musicxml) - Sheet music
- **Audio** (WAV/MP3) - Playable files

Implementation

Technology Stack

Core Technologies:

- Python 3.9+ (Type-safe, clean architecture)
- OpenAI GPT-4o-mini (LLM engine)
- music21 (Music notation library)
- Pydantic v2 (Data validation)

Audio Rendering:

- FluidSynth (Synthesis)
- FFmpeg (MP3 conversion)

Key Components

1. LLMClient

Manages OpenAI API with retry logic & fallbacks

2. MelodyGenerator

Orchestrates melody creation with chunking

3. MIDIWriter

Converts IR to standard music formats

4. AudioRenderer

Synthesizes playable audio from MIDI

Prompt Engineering

Carefully crafted prompts with:

1. Clear task descriptions
2. **Few-shot examples** (3 per task)
3. Explicit JSON schemas
4. Constraint enforcement
    - *"Generate EXACTLY N notes for N syllables"*
5. Format validation reminders

> *Prompt quality directly impacts output quality*

Evaluation & Results

Evaluation Methodology

Test Dataset: 20 diverse lyrical inputs

- **Emotions:** Happy (5), Sad (5), Hopeful (4), Tense (3), Calm (2), Excited (1)
- **Length:** Short (5), Medium (10), Long (5)
- **Complexity:** Simple (10), Poetic (10)

Metrics:

- Syllable-note alignment accuracy
- Emotion-key consistency
- Tempo appropriateness
- Musical validity

Results Summary

Metric	Score	Status
Syllable-Note Alignment	100%	Perfect
Emotion-Key Consistency	95%	Excellent
Tempo Appropriateness	90%	Strong
Musical Validity	100%	Perfect

LLM Success Rate: 85%

Fallback Activation: 15%

Example Output: Hopeful Lyrics

Input: *"The sun will rise again"*

Analysis:

- Emotion: `hopeful`
- Tempo: `90 BPM`
- Key: `G major`

Generated Melody (6 notes):

```
G4 → A4 → B4 → C5 → B4 → A4
```

Ascending then descending arch - emotionally appropriate

Strengths

Perfect alignment - 100% syllable matching
Emotional coherence - Strong sentiment correlation
Robust operation - Zero failures with fallbacks
Standard formats - Works with all music software
Fast processing - ~3 seconds average

> *Production-ready system with real-world applicability*

Limitations

- **Harmonic simplicity** - Single-voice only (no chords)
- **Style constraints** - Western music theory focused
- **Long-form coherence** - Very long lyrics (>50 syllables) may show inconsistencies
- **LLM dependency** - Requires API access & costs per request

These inform our future work directions

Future Directions

Future Enhancements

Musical Expansion

- Harmony & chord progressions
- Multi-instrument arrangements
- Genre-specific styles (jazz, classical, rock)

Technical Improvements

- Local LLM support (reduce API dependency)
- Multi-language lyrics
- Non-Western music systems

Conclusion

Key Contributions

1. **Novel LLM Application** - First to use GPT for lyrics-to-melody without fine-tuning
2. **Hybrid Architecture** - AI + deterministic fallbacks for reliability
3. **Production System** - Complete, type-safe implementation with CLI & API
4. **Multi-Format Output** - MIDI, MusicXML, and audio rendering
5. **Open Source** - Available for research and practical use

Thank You

Questions?