mllm-shap: A Shapley Value Explainability Platform for Text-Audio Multimodal Large Language Models

Jakub Muszyński[1]* **Paweł Pozorski**[1]* **Maria Ganzha**[1]
[1]Warsaw University of Technology, Warsaw, Poland
{jakub.muszynski2, pawel.pozorski}.stud@pw.edu.pl
maria.ganzha@pw.edu.pl
0009-0000-2797-6044 (JM) 0009-0007-0728-1043 (PP) 0000-0001-7714-4844 (MG)

Abstract

We present mllm-shap, an open-source Python platform for researchers and ML practitioners that extends Shapley value (SV) explainability from text-only large language models to multimodal LLMs (MLLMs) that jointly process text and audio. Building on the token-level SV framework introduced by TokenSHAP, mllm-shap addresses three challenges absent in the text-only setting: (1) modality-aware coalition masking that handles the coexistence of text tokens and dense audio encoder frames within a single input, (2) multi-turn conversation tracking with per-token role and modality metadata, and (3) audio token grouping via phonetic alignment that reduces the coalition space by 10–50×. The platform ships as a pip-installable package[1] implementing five SV estimation strategies – including a Complementary Contributions estimator with Neyman-optimal allocation that outperforms Monte Carlo baselines – together with an interactive web GUI for real-time attribution visualization. To our knowledge, mllm-shap is the first publicly available framework for complete, reproducible SV-based explainability of text-audio MLLMs. The package is MIT-licensed with full source code on GitHub[2,3] and a demonstration video included as supplementary material.

1 Introduction

Shapley values (SV; Shapley, 1951) have become a standard tool for model-agnostic input attribution in machine learning, popularized by SHAP (Lundberg and Lee, 2017) for tabular and image models and extended to text-only LLMs by TokenSHAP (Goldshmidt and Horovicz, 2024). Unlike perturbation-based alternatives such as LIME (Ribeiro et al., 2016) or gradient-based methods like Integrated Gradients (Sundararajan et al., 2017), SVs uniquely satisfy efficiency, symmetry, and additivity axioms while requiring only black-box access. Yet as production systems increasingly deploy multimodal LLMs (MLLMs) that process both text and speech end-to-end – in customer service, healthcare, and accessibility – no existing tool supports SV-based explainability for these models (Zhao et al., 2024).

Applying TokenSHAP's approach directly to text-audio MLLMs fails for three reasons. First, **granularity mismatch**: a short utterance produces hundreds of dense audio encoder frames alongside a handful of text tokens, making the coalition space intractable (2^{150+} vs. 2^{10}). Second, **modality-aware masking**: removing a text token means dropping it from the token sequence, but "removing" an audio segment requires zeroing encoder frames at specific time boundaries – these are fundamentally different operations that must be handled within a single coalition. Third, **conversational structure**: real MLLM interactions involve multi-turn dialogues with system prompts, user messages, and assistant responses across both modalities; attribution must track which turn and modality each token belongs to.

mllm-shap addresses all three gaps as an integrated platform with two main components:

1. A **model-agnostic Python package** that extends SV computation to multimodal, multi-turn conversational inputs, with modality-aware masking, feature-unit metadata tracking, and five SV estimation strategies including a Neyman-optimal allocator (§2).
2. An **interactive web GUI** with cost estimation, session persistence, and real-time attribution visualization for both text and audio (§4).

* Equal contribution.
[1]https://pypi.org/project/mllm-shap/
[2]https://github.com/Pawlo77/MLLM-Shap
[3]https://github.com/mvishiu11/shap-mllm-explainer

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations), pages 387–396

We demonstrate the platform on LFM2-Audio-1.5B (Liquid AI, 2025), showing concrete attribution outputs for text and multi-turn inputs (§3), and report evaluation results across 593 SV analyses on a single consumer GPU (§5).

2 System Architecture

`mllm-shap` follows a modular, pipeline-oriented design (Figure 1) that separates four concerns: (i) model interaction via *connectors*, (ii) modality-aware coalition mask generation, (iii) utility function evaluation, and (iv) SV estimation. This separation ensures that estimator comparisons differ only in the approximation method while all other components remain fixed.

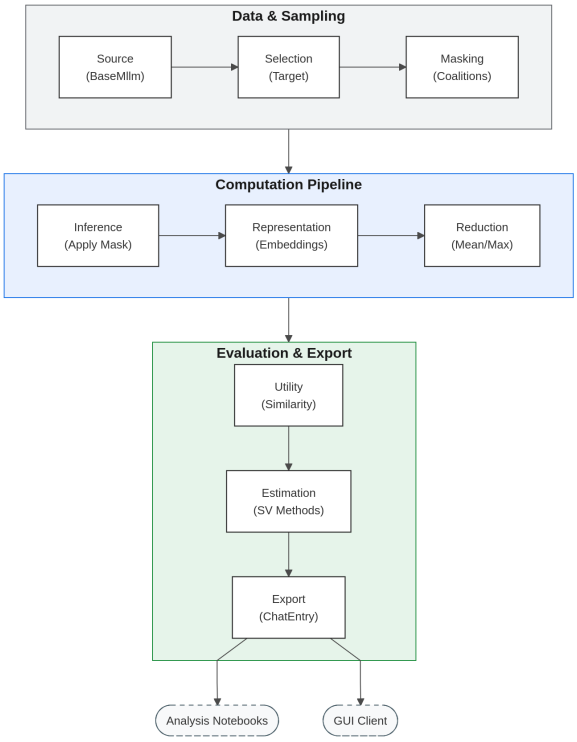

Figure 1: Package pipeline: chat construction with per-token metadata, modality-aware mask generation, connector-based inference, utility evaluation, and SV estimation. Results export to analysis notebooks or the GUI client.

Connectors and the Black-Box Interface. Model integration is encapsulated behind a `BaseMllmModel` interface that translates between package-level chat objects and model-specific I/O formats. Only input–output access is required – no gradients or attention maps – making the system compatible with any MLLM exposing a generate API. A reference connector for HuggingFace `transformers` is provided, with LFM2-Audio-1.5B as the default model.

Feature-Unit Tracking. A core design contribution is the *feature unit*: each token in the pipeline carries structured metadata – modality (TEXT or AUDIO), role (USER, SYSTEM, or ASSISTANT), turn index, and positional information. This metadata flows through the entire pipeline, enabling modality-aware masking, role-based filtering (system tokens are excluded from attribution by default), and per-turn aggregation for multi-turn conversations. Table 2 illustrates this: only user-content tokens (Role 0) receive SV values, while system/special tokens (Role 2) are correctly excluded, and turn indices enable separate analysis of each conversational exchange. Excluding tokens from coalitions within the scope of this framework means they are always fed to the model. That is, final input prompts consist of explainable tokens present in the given coalition (those with a non-NaN final SV) and all non-explainable tokens in their original order.

Modality-Aware Masking. A critical challenge in multimodal SV estimation is removing features in a way that remains in-distribution for the MLLM. The platform implements two distinct masking strategies based on feature type. For *text*, explainable tokens are physically removed from the input string; however, non-explainable role-markers (e.g., `<|im_start|>`) and turn-delimiters are preserved in every coalition to maintain the structural integrity required by chat-tuned MLLMs, ensuring the model always perceives a valid conversational state. For *audio*, physical removal of segments would create temporal discontinuities and acoustic "clicks" that confuse the model's encoder; instead, `mllm-shap` masks by zeroing the amplitude of the raw waveform or latent frames within the time boundaries of the selected segment (Covert et al., 2021). This dual-strategy design ensures that both modalities can participate in the same coalition game while each receives masking appropriate to its signal structure.

Audio Token Grouping via SGPA. To resolve the granularity mismatch, the platform integrates Spectrogram-Guided Phonetic Alignment (Pozorski et al., 2026), a four-stage pipeline that maps dense audio encoder frames to word-aligned

segments. SGPA combines CTC-based forced alignment (Graves et al., 2006) via Wav2Vec2-XLSR-53 (Baevski et al., 2020) with spectrogram-guided boundary refinement using local energy and spectral flux cues, then aggregates character-level boundaries into word-level segments. This reduces the effective coalition space by 10–50×: in controlled experiments, SGPA compressed the player set from ∼50 native audio tokens to ∼7 word-aligned segments, yielding a 43× reduction in model evaluations and bringing per-sample wall-clock time from ∼30 minutes to roughly one minute on a consumer GPU. Crucially, SGPA provides a model-agnostic explanation unit: by grounding attributions in phonetic segments rather than model-specific latent frames, `mllm-shap` enables direct, black-box explainability comparisons between models with different audio architectures. We note that SGPA necessarily changes the cooperative game being solved – attributions are defined with respect to externally aligned segments rather than model-native audio units – and refer the reader to Pozorski et al. (2026) for a full diagnostic characterization of this trade-off.

SV Estimators. The platform provides five estimation strategies sharing a common callable interface for seamless benchmarking: **Exact** (2^n enumeration for tractable inputs), **Monte Carlo** with optional first-order omission constraints (Goldshmidt and Horovicz, 2024), **Complementary Contributions** (CC) using stratified sampling (Mitchell et al., 2022), **Neyman-CC** extending CC with Neyman-optimal variance-adaptive allocation (Neyman, 1934), and **Hierarchical** recursive decomposition for long sequences. Full algorithmic descriptions of each estimator are provided in Appendix A. Benchmarking on 27 inputs where exact SV is tractable (9–10 tokens) confirms that Neyman-CC dominates, achieving $> 90\%$ accuracy at sampling fractions as low as 0.15 (Figure 2).

Utility Functions. The platform supports three utility functions: Hidden State Similarity for white-box debugging, Cosine Similarity for semantic-level evaluation (Reimers and Gurevych, 2019), and TF-IDF Weighted Cosine Similarity (the default) for cross-architecture benchmarking in a reduced feature space. Detailed descriptions are in Appendix A.

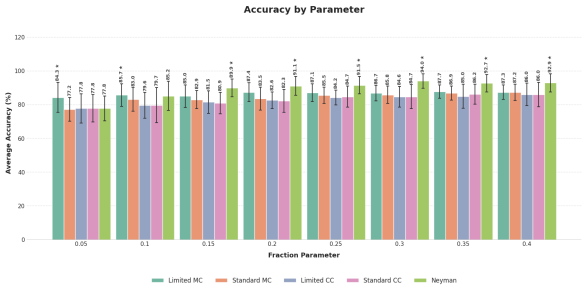

Figure 2: Approximation accuracy vs. exact SV (27 samples, 9–10 tokens). The Neyman-based estimator dominates at all sampling fractions tested.

3 Example Walkthrough

We illustrate the platform with two concrete examples on LFM2-Audio-1.5B, highlighting capabilities absent in existing SV tools.

Single-Turn Attribution. Listing 1 shows a minimal API workflow. For the text prompt *"Who developed reCAPTCHA?"*, Table 1 shows the resulting token-level SVs. Note that the tokenizer splits "reCAPTCHA" into subword units ("Rec", "apt", "cha"); the platform handles this transparently, and the SVs reveal that "developed" (ϕ=0.110) and the first subword of the target entity ("Rec", ϕ=0.357) carry high attribution – consistent with the model attending to the verb–object pair that determines the answer.

Table 1: Token-level SV for *"Who developed reCAPTCHA?"* (text-to-text, LFM2-Audio-1.5B, Neyman estimator). System and punctuation tokens are automatically filtered.

ID	Token	SV (ϕ)
1	Who	0.080
2	developed	0.110
3	Rec	0.357
4	apt	0.273
5	cha	0.174

Cross-Modal Attribution. To demonstrate multimodal reasoning, we analyze a transcription task where the input consists of a 3-second English audio recording and the text instruction: *"Transcribe the audio."*. As shown in Figure 3, the AUDIO modality, represented by segments aligned via SGPA, carries the primary attribution (ϕ=0.81). In contrast, the text tokens for "transcribe" carry secondary attribution (ϕ=0.10). The framework successfully decouples the source of information (the audio signal) from

Listing 1: Minimal SV attribution with `mllm-shap`.

```python
from mllm_shap.connectors.liquid import (
    LiquidAudio
)

from mllm_shap.shap.monte_carlo import
     LimitedMcShapExplainer
from mllm_shap.shap.compact import Explainer
from mllm_shap.connectors.enums import Role
import torch
from pprint import pprint

# Load model and create chat
model = LiquidAudio(device=torch.device("cuda"))
chat = model.get_new_chat()
chat.new_turn(Role.USER)
chat.add_text("Who developed Recaptcha?")
chat.end_turn()

# Run SV estimation
explainer = Explainer(
    model=model,
    shap_explainer=LimitedMcShapExplainer()
)
result = explainer(chat=chat)

pprint(result.full_chat.get_conversation())
```

the instructional context (the text prompt).

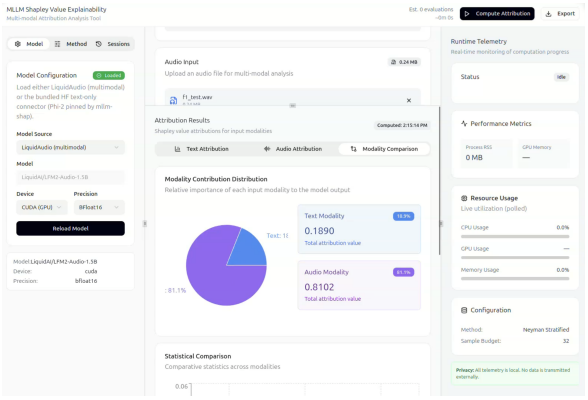

Figure 3: Web interface: chart of total SV attribution by modality based on simple text input of *"Transcribe the audio."* and 3-second English audio recording.

Multi-Turn Conversation Attribution. Table 2 shows output from a two-turn dialogue: *"Who are you?"* followed by *"Can you repeat?"*. This demonstrates three capabilities unique to `mllm-shap`: (a) **Turn tracking**: each token is tagged with its turn index, enabling per-turn and cross-turn attribution analysis. (b) **Role-based filtering**: special tokens (`<|im_start|>`, `<|im_end|>`) and role markers are assigned Role 2 (system) and automatically excluded from SV computation, while user-content tokens (Role 0) receive attributions. (c) **Structured output**: the full token–SV–role–turn table is exportable as a DataFrame for downstream statistical analysis.

In this example, "Who" (ϕ=0.231) and "you" (ϕ=0.230) dominate Turn 1, while "are" receives zero attribution – the model's response to an identity question is driven by the interrogative and the pronoun, not the copula. In Turn 3, "repeat" (ϕ=0.217) carries the highest SV, consistent with it being the only novel semantic content (the model must recognize a repetition request).

Table 2: Multi-turn attribution for *"Who are you? | Can you repeat?"*. Role 0 = user content (attributed), Role 2 = system/special (excluded) – denotes tokens excluded from SV computation by design.

ID	Token	SV (ϕ)	Role	Turn
0	`<|im_start|>`	—	2	1
1	user	—	2	1
3	Who	0.2313	0	1
4	are	0.0000	0	1
5	you	0.2300	0	1
7	`<|im_end|>`	—	2	1
9	`<|im_start|>`	—	2	3
10	user	—	2	3
12	Can	0.1546	0	3
13	you	0.1676	0	3
14	repeat	0.2166	0	3
16	`<|im_end|>`	—	2	3

4 Interactive Web Interface

The GUI (Figure 4) provides a complete explainability workflow without writing code. It follows a client–server architecture: a React (Meta Platforms, 2013) frontend communicates with a FastAPI[4] backend via REST API, with PostgreSQL for session persistence and Nginx for routing.

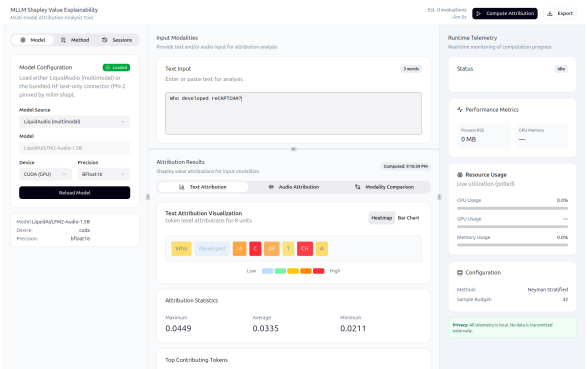

Figure 4: Web interface: model and method configuration (left), multimodal chat with text and audio input (center), attribution heatmaps and time-aligned audio plots (right).

[4] https://fastapi.tiangolo.com/

Asynchronous Execution. Since SV estimation requires hundreds of model forward passes, the backend implements an asynchronous task queue using FastAPI's background workers. When a user requests an explanation, the request is immediately acknowledged with a unique task ID and the estimation process is dispatched to a GPU-bound worker, preventing HTTP timeouts and allowing the frontend to poll for real-time progress updates visualized via a progress bar.

Session Persistence. The PostgreSQL database stores not only chat history but also the FeatureUnit metadata for every token and audio segment. This enables *session resumption* – a researcher can close the browser during a long-running Neyman-CC estimation and return later to find results fully populated – and *post-hoc analysis*, where computed attributions can be re-visualized using different GUI modes (e.g., switching from a modality-level pie chart to a token-level heatmap) without re-running the computation.

Workflow and Visualization. The interface guides users through the following steps: configure model and SV method → interact via text or audio → receive an upfront cost estimate (number of model calls and estimated runtime) → compute attributions → visualize results as token heatmaps or time-aligned audio plots → export for analysis. For audio inputs, the GUI provides a dedicated *Audio Attribution Timeline* (Figure 5) that displays time-aligned attribution intensity overlaid on the waveform, allowing researchers to identify which temporal segments of the audio signal most influenced the model's output. The right panel provides real-time telemetry (CPU/GPU/memory utilization), the active SV configuration, and a privacy notice confirming that all computation remains local. The GUI is containerized via Docker Compose (Appendix B) for reproducible single-command deployment.

5 Evaluation

We evaluate `mllm-shap` along three axes: estimator accuracy, platform-scale feasibility, and coverage of system capabilities.

Estimator Accuracy. On 27 samples where exact SV is tractable (9–10 tokens), the Neyman estimator with first-order omission achieves >90% mean accuracy at 30% sampling fraction (∼150 coalitions), outperforming standard MC and CC

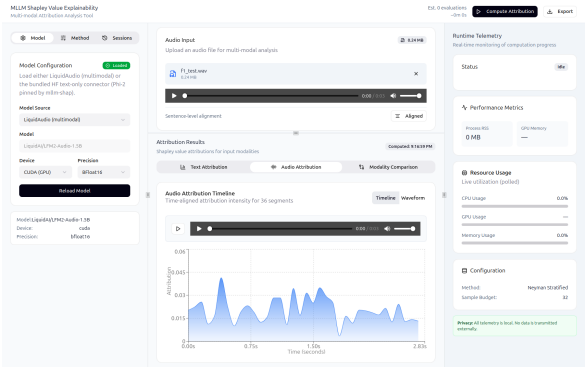

Figure 5: Audio attribution view: time-aligned SV intensity plot for a 3-second audio input, with model configuration (left), attribution results with audio/text/-modality comparison tabs (center), and runtime telemetry with method configuration (right).

at all tested budgets (Figure 2). Across all estimators, the modular pipeline ensures that comparisons isolate the approximation method: connector behavior, masking policies, and utility evaluation remain fixed, so accuracy differences reflect only the estimation strategy.

System Feasibility. We validated `mllm-shap` by performing 593 SV analyses on a single NVIDIA RTX 4080 GPU (16 GB VRAM), covering 100 VoiceBench (Chen et al., 2024) samples across five modality configurations (text-to-text, speech-to-text with male/female voices, and speech-to-speech variants) and 93 multilingual samples from Infinity-Instruct (Li et al., 2025) in three languages (English, Spanish, French). Table 3 summarizes key runtime statistics. Single-sentence audio-to-text tasks complete in ∼30 s, while multi-sentence interleaved prompts, which expand the coalition space by 10–20×, require ∼400 s. The Neyman allocator consistently reached its variance-adaptive second phase in the majority of runs, confirming that even the default budget allocation ($3n^2$ model evaluations) provides sufficient initialization for stable estimates. Speech-to-speech modes incur ∼2× the runtime of speech-to-text due to audio decoding overhead, establishing a practical ceiling for interactive use.

Capability Coverage and Reproducibility. Across the 593 analyses, all core platform capabilities were exercised at scale: cross-modal attribution (300 analyses), SGPA audio segmentation (200), multi-turn tracking (100), and multilingual support across English, Spanish, and French (93) with no language-specific pipeline

Table 3: System feasibility summary across 593 SV analyses on a single consumer GPU. Runtime is mean wall-clock time per sample using the Neyman estimator.

Configuration	Mean calls	Mean time (s)
Single-sentence S2T	60	30
Single-sentence S2S	59	67
Single-sentence T2T	155	91
Multi-sentence (interleaved)	1,005–1,174	399–451
Multilingual S2T	235	105
Multilingual T2T	606	205

Listing 2: Adding a custom SV estimator. Three abstract methods must be implemented.

```
from mllm_shap.shap.base import BaseShapExplainer

class MyEstimator(BaseShapExplainer):
    def _get_next_split(self, n, device,
                        generated_masks_num,
                        existing_masks=None):
        # return Tensor[1, n] for next coalition
        ...
    def _get_num_splits(self, n):
        # return total number of masks to sample
        ...
    def _calculate_shap_values(self, masks,
                               similarities,
                               device):
        # return Tensor[n] of attributions
        ...
```

modifications required (Appendix D). The full results are released as a **multimodal attribution dataset** on HuggingFace[5], providing a reproducible baseline for future multimodal XAI research. All experiments were executed through a reproducible pipeline with YAML-based configuration management, per-sample checkpointing with resume support, and deterministic inference defaults (temperature = 0, top-k = 1, see Appendix C).

6 Discussion

We address several practical concerns relevant to practitioners adopting the platform.

Extending the Framework. The modular pipeline (Figure 1) is designed so that each component is independently extensible. A custom SV estimator subclasses `BaseShapExplainer` and implements three abstract methods: `_get_next_split` (coalition sampling logic), `_get_num_splits` (budget calculation), and `_calculate_shap_values` (attribution from masks and similarities), as shown in Listing 2. A custom similarity function extends `BaseEmbeddingSimilarity` with a single `__call__` method that compares a base embedding against masked-output embeddings, allowing researchers to substitute domain-specific notions of output similarity (e.g., factuality scores, task-specific metrics) without touching the rest of the pipeline. New modalities or model families enter the system through the `BaseMllmModel` connector interface: only input–output access is required.

Connectors for Discrete-Codec MLLMs. The reference connector targets continuous audio frontends (LFM2-Audio-1.5B), but the architecture cleanly accommodates models built on discrete audio codecs (e.g., EnCodec, Mimi, or semantic-token variants). Two changes are required, both local to the connector. First, the audio-masking primitive shifts from amplitude-zeroing of the waveform to token replacement in the codec sequence: a coalition member is "removed" by substituting its codec tokens with a designated mask token, the codebook's silence centroid, or repeated padding tokens, depending on what the upstream model treats as in-distribution. Second, the feature-unit granularity may map directly onto codec tokens when their rate is already comparable to phonetic segments, in which case SGPA can be bypassed; otherwise SGPA still applies as a pre-tokenization segmentation layer because it operates on the raw waveform. The feature-unit metadata abstraction, the estimator interfaces, the utility functions, and the GUI are unchanged.

Out-of-Distribution Behavior under Amplitude Masking. Zeroing the raw waveform within SGPA-derived boundaries introduces signal discontinuities that could in principle push the audio encoder out of its training distribution. We mitigate this implicitly through the SGPA segmentation procedure itself: boundary refinement uses local energy minima and spectral-flux cues (§2), which favor naturally low-energy regions of the signal, reducing click artifacts at mask edges. Across the 593-analysis evaluation suite we did not observe pathological model outputs (e.g., empty generations, repetition loops, or attribution distributions inconsistent with held-out exact SVs on the 27-sample tractable subset). We emphasize this is an empirical observation on LFM2-Audio's continuous frontend; codec-based mod-

[5] https://huggingface.co/datasets/Pawlo77/mllm-shap

els – where the encoder is by construction more sensitive to any out-of-vocabulary token – are expected to require the token-replacement strategy described above, and we leave systematic characterization to future work.

Deployment Without a Local GPU. The reference deployment co-locates inference and the explainability backend on a single workstation with a CUDA device. Two alternatives exist within the connector abstraction: the connector accepts an arbitrary `torch.device`, so CPU-only execution is functional for small text-only inputs (albeit at substantially higher runtime); more practically, a custom connector can dispatch generate calls to a remote inference endpoint (e.g., a self-hosted vLLM or TGI server, or a HuggingFace Inference Endpoint), in which case the explainability backend itself runs on commodity hardware and only orchestrates coalitions and aggregates results.

Relation to Other Explainability Methods. `mllm-shap` does not subsume gradient methods such as Integrated Gradients (Sundararajan et al., 2017), which require white-box access; perturbation surrogates such as LIME (Ribeiro et al., 2016), which lack SV axiomatic guarantees; or attention-based interpretability, whose faithfulness remains contested (Jain and Wallace, 2019). The platform's contribution is that SVs are the only currently available explainability primitive satisfying the efficiency, symmetry, and additivity axioms while remaining well-defined over interleaved text-audio multi-turn dialogue.

7 Related Systems

Table 4 compares `mllm-shap` with existing SV-based tools. While the original SHAP (Lundberg and Lee, 2017) is the industry standard, it lacks native support for the variable-length, autoregressive nature of LLMs. TokenSHAP (Goldshmidt and Horovicz, 2024) addressed text-only LLMs but cannot handle the "interleaved" nature of multimodal conversations or the high-dimensionality of raw audio encoder frames. Prior reviews of SHAP-based methods in NLP (Mosca et al., 2022) have similarly focused exclusively on text modalities, while attention-based interpretability approaches (Jain and Wallace, 2019) remain debated in their faithfulness and do not naturally extend to cross-modal settings where text and audio operate over different representational spaces.

Table 4: Comparison with existing SV-based explainability tools.

Feature	SHAP	TokenSHAP	Ours
Text LLMs	✓	✓	✓
Audio modality	×	×	✓
Variable-length seq.	×	✓	✓
Multi-turn chat	×	×	✓
Audio segmentation	×	×	✓
Neyman allocation	×	×	✓
Interactive GUI	×	×	✓
Reproducibility infra	×	×	✓
Open-source	✓	✓	✓

8 Conclusion and Availability

We presented `mllm-shap`, the first open-source platform for Shapley value-based explainability of text-audio multimodal language models. By extending the token-level SV framework of TokenSHAP with modality-aware masking, multi-turn conversation tracking, efficient audio token grouping via SGPA, and an interactive GUI, the system makes previously intractable multimodal attribution analysis accessible on consumer hardware. Future work includes extending connector support to additional MLLM architectures (e.g., vision-language models) and conducting large-scale human evaluation studies.

The platform is distributed under the MIT license. The `pip`-installable package,[6] source repositories,[2,3] API documentation,[7] and the multimodal attribution dataset[4] are all publicly available.

Ethical Considerations

The system is designed to improve transparency and accountability of AI systems. No personal or sensitive data is collected by the tool; all experiments used publicly available datasets and open-source models. We note that SV-based attributions, while principled, should not be treated as ground-truth causal explanations of model behavior – they measure marginal contribution under a specific utility function and coalition structure. Additionally, SGPA segmentation introduces an external player partition that does not necessarily align with model-internal representations.

[6] **PyPI:** https://pypi.org/project/mllm-shap/

[7] https://pawlo77.github.io/MLLM-Shap/

Limitations

All experiments use a single MLLM (LFM2-Audio-1.5B); although the connector interface is model-agnostic, generalization to other architectures (e.g., Whisper-based or codec-based speech models) remains unvalidated. Despite SGPA's $43\times$ coalition-space reduction, complex multi-sentence multimodal inputs still require $\sim$400 s per sample on consumer hardware, limiting real-time interactivity for long inputs. Only text and audio modalities are currently supported; vision or video would require new masking and alignment strategies. Our evaluation isolates estimator accuracy against exact SVs but does not include head-to-head comparison with LIME, Integrated Gradients, or attention-based interpretability on a common faithfulness benchmark (e.g., ERASER); such a comparison is necessary before strong claims can be made about the practical utility of the produced explanations relative to alternatives, and is a direct target for future work.

Acknowledgments

We gratefully acknowledge Polish high-performance computing infrastructure PLGrid (HPC Centers: ACK Cyfronet AGH, WCSS) for providing computer facilities and support within computational grant no. PLG/2026/019124.

We thank the anonymous reviewers for their constructive feedback, which substantively improved the discussion of extensibility, out-of-distribution behavior, and deployment options. We also thank the LFM2-Audio team at Liquid AI for releasing model weights under a permissive license, and the maintainers of the VoiceBench and Infinity-Instruct datasets used in our evaluation.

References

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Advances in Neural Information Processing Systems*, volume 33, pages 12449–12460.

Yiming Chen, Xianghu Yue, Chen Zhang, Xiaoxue Gao, Robby T. Tan, and Haizhou Li. 2024. Voicebench: Benchmarking LLM-based voice assistants. *arXiv preprint arXiv:2410.17196*.

Ian Covert, Scott M Lundberg, and Su-In Lee. 2021. Explaining by removing: A unified framework for model explanation. *Journal of Machine Learning Research*, 22(209):1–90.

Roni Goldshmidt and Miriam Horovicz. 2024. Tokenshap: Interpreting large language models with monte carlo shapley value estimation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 369–376.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3543–3556.

Jijie Li, Li Du, Hanyu Zhao, Bo-wen Zhang, Liangdong Wang, Boyan Gao, Guang Liu, and Yonghua Lin. 2025. Infinity instruct: Scaling instruction selection and synthesis to enhance language models. *arXiv preprint arXiv:2506.11116*.

Liquid AI. 2025. Lfm2-audio: Liquid foundation models for audio. https://www.liquid.ai/liquid-foundation-models/lfm-2-audio.

Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, volume 30.

Meta Platforms. 2013. React: A JavaScript library for building user interfaces. https://github.com/facebook/react.

Rory Mitchell, Joshua Cooper, Eibe Frank, and Geoffrey Holmes. 2022. Approximating the shapley value without marginal contributions. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(7):7773–7780.

Edoardo Mosca, Ferenc Szigeti, Stella Tragianni, Daniel Gallagher, and Georg Groh. 2022. SHAP-based explanation methods: A review for NLP interpretability. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4593–4603.

Jerzy Neyman. 1934. On the two different aspects of the representative method: The method of stratified sampling and the method of purposive selection. *Journal of the Royal Statistical Society*, 97(4):558–625.

Paweł Pozorski, Jakub Muszyński, and Maria Ganzha. 2026. Sgpa: Spectrogram-guided phonetic alignment for feasible shapley value explanations in multimodal large language models. *Preprint*, arXiv:2603.02250. Submitted to Interspeech 2026.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pages 3973–3983.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144.

Lloyd S. Shapley. 1951. *Notes on the N-Person Game; II: The Value of an N-Person Game*. RAND Corporation, Santa Monica, CA.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 3319–3328.

Haiyan Zhao, Hanjie Chen, Fan Yang, Ninghao Liu, Huiqi Deng, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, and Mengnan Du. 2024. Explainability for large language models: A survey. *ACM Transactions on Intelligent Systems and Technology*, 15(2).

A SV Estimator and Utility Function Details

Estimator Descriptions. The five SV estimation strategies provided by `mllm-shap` are:

- **Exact:** Full 2^n enumeration of all coalitions, restricted to tractable inputs (typically $n \leq 15$). Serves as the ground-truth reference for estimator benchmarking.
- **Monte Carlo (MC):** Random coalition sampling with optional first-order omission constraints, following Goldshmidt and Horovicz (2024). Omission constraints ensure that each player is excluded from at least one sampled coalition, reducing the variance of individual SV estimates.
- **Complementary Contributions (CC):** A stratified sampling approach where each coalition evaluation contributes to the marginal utility of all included players and their complements simultaneously (Mitchell et al., 2022), approximately doubling the information extracted per model call.
- **Neyman-CC:** An extension of CC that utilizes Neyman-optimal allocation (Neyman, 1934) to distribute the sampling budget across strata according to their variance, significantly reducing estimation error for a fixed budget. The estimator operates in two phases: an initial uniform exploration phase ($3n^2$ evaluations by default) followed by a variance-adaptive allocation phase.
- **Hierarchical:** A recursive decomposition strategy designed for long sequences that exceed the computational limits of flat estimators. Tokens are grouped into higher-level features, SVs are computed at the group level, and then recursively refined within groups of interest.

Utility Functions. The platform supports three utility functions tailored to different research goals: **(U1) Hidden State Similarity** directly compares model-internal activations between full and masked inputs, enabling white-box debugging of how coalitions affect latent representations. **(U2) Cosine Similarity** computes sentence-level semantic similarity using external embeddings (Reimers and Gurevych, 2019), providing a model-external measure of output equivalence. **(U3) TF-IDF Weighted Cosine Similarity** operates over decoded text, weighting terms by their TF-IDF scores so that attribution focuses on the information-dense tokens that define the model's factual output.

B GUI Deployment Architecture

The system architecture (Figure 6) is orchestrated via Docker Compose to manage the lifecycle of three distinct services: an Nginx reverse proxy for routing, a FastAPI backend that loads models from HuggingFace Hub and performs GPU-bound SV computation, and a PostgreSQL database for session CRUD operations. The GUI is containerized for reproducible single-command deployment on any system with a compatible NVIDIA Container Toolkit installation.

C Experimental Pipeline and Reproducibility

To support the 593 analyses reported in Section 5, we developed a robust experimental orchestrator (Figure 7) organized around four key components: **Configuration Management** uses YAML-based specifications to pin model versions, estimator hyperparameters, and dataset revisions. The **Deterministic Compute Engine** enforces deterministic inference settings (temperature = 0, top-k = 1). A **Checkpointing Layer** persists results

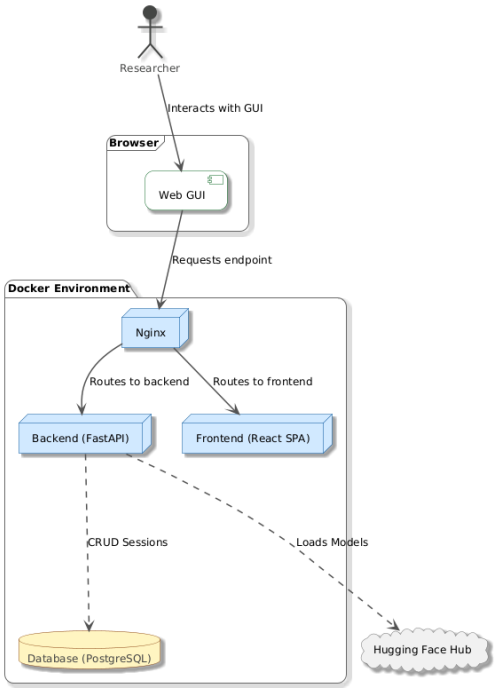

Figure 6: GUI deployment architecture: three-container Docker Compose setup with Nginx routing, FastAPI backend loading models from HuggingFace Hub, and PostgreSQL for session CRUD.

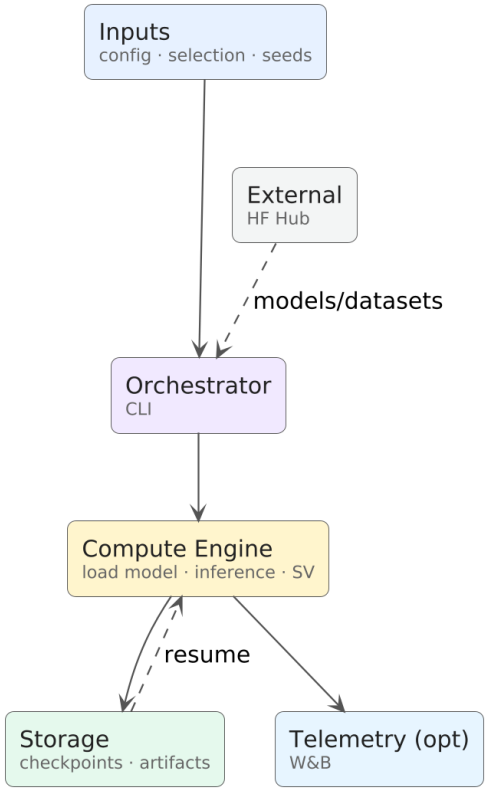

Figure 7: Experimental infrastructure: YAML-configured orchestrator dispatches jobs to the GPU compute engine with per-sample checkpointing.

incrementally, enabling resume from the last completed sample if a run is interrupted. The **Artifact Hierarchy** produces a structured output directory per run containing a `spec.json`, checkpoint states, and individual JSON results for every sample with full metadata and attribution vectors. All evaluation experiments were conducted on a single consumer-grade workstation equipped with an NVIDIA RTX 4080 (16 GB VRAM) and an Intel Core i9-13900K CPU.

D Capability Coverage

Table 5 provides a detailed breakdown of the platform capabilities exercised across the 593-analysis evaluation suite. Each capability corresponds to a distinct engineering challenge solved by the platform: multi-turn tracking was validated on the full VoiceBench multi-sentence set (4 modes $\times$ interleaved prompts); cross-modal attribution was exercised across all speech configurations.

E Attribution Data Schema

The platform exports results in a structured JSON format (Table 6) that encapsulates the complete state of the Shapley game.

Table 5: Platform capabilities exercised across the evaluation suite. Each row corresponds to a system feature validated at scale.

Capability	**Analyses**	**Source**
Text-only attribution	200	VB + II
Cross-modal (text+audio)	300	VoiceBench
SGPA audio segmentation	200	VoiceBench
Multi-turn tracking	100	VB multi-sent.
Multilingual support	93	Infinity-Inst.

Table 6: Structure of the exported attribution JSON.

Field	**Type**	**Description**
`session_id`	UUID	Unique identifier for the analysis session.
`metadata`	Dict	Model name, estimator type, and budget.
`features`	List	Array of Feature-Units (ID, modality, role).
`values`	List	Raw ϕ values corresponding to each unit ID.
`turn_map`	Dict	Mapping of turn indices to feature IDs.
`utility`	Float	Final utility score of the grand coalition.
`telemetry`	Dict	GPU memory usage and wall-clock time.