Leveraging Federated Machine Learning to Detect Unlicensed Training

Kay Yan

23dw32@queensu.ca
Queen's University
Canada

March 8, 2025

1 Introduction

Split Learning (Vertical FL)

We selected the split learning paradigm because it best aligns with the requirements of this compliance check. The neural network architecture divides across different parties when using split (or vertical) federated learning. The client stores the initial model components and input data while the server holds the rest of the model layers and produces the outputs. The cut layer in split learning exchanges intermediate activations ("smashed data") between parties while raw inputs and complete model parameters remain undisclosed. This setup matches our scenario: The record label (client) feeds its music data into the initial layers of a compliance check model while the AI company (server) processes this data using its own model weights to complete the forward pass. Sensitive information remains confidential because the record label cannot access the company's model details while the company cannot access the label's raw audio.

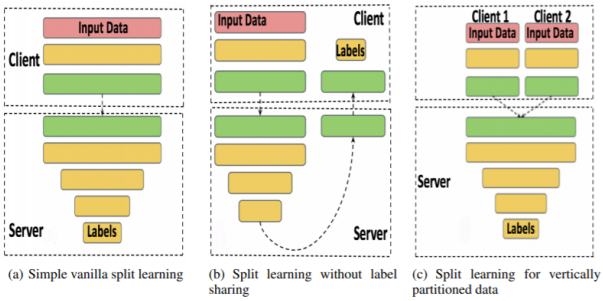

Figure 1: Illustration of the split learning configuration [1].

Figure 1: Example split learning configurations. (a). For instance, in this setting, each client trains a partial model up to a specific layer called the 'cut layer'. Only the intermediate features at the cut layer (boundary between yellow and green) are sent to the server, and gradients at that cut layer are returned to the client for training. This allows joint model processing without sharing raw data.

Structure: For compliance checking, we configure a two-part neural network: The record label runs a feature extractor on its audio data locally, up to a cut layer. The extractor could be a lightweight CNN or audio encoder that transforms the song into an intermediate feature representation. The AI company attaches a corresponding detector head (the remaining layers) that takes the intermediate feature and produces a compliance result (for example, a likelihood score or an "infringement" prediction). During an infringement check, the process works as follows:

- The record label's system takes one of its songs (or a unique fingerprint derived from it) as input and forward propagates through the local cut-layer of the network. This yields an encoded representation of the song. Crucially, this encoding is abstract — it does not reveal the raw audio, but it captures patterns the later layers can analyze [1].

- The "smashed" features at the cut layer are securely transmitted to the AI company's server. The AI company then forward-propagates that activation through the rest of the model (or a special compliance check sub-network) using its private model parameters. For example, the AI company's model (or detector head) might compute a similarity or likelihood that the input audio was part of its training data.
- The final output (which might be an encrypted or blinded result – see privacy measures below) is sent back to the record label. This output could be, for instance, a probability or an error metric indicating how closely the AI model's knowledge matches the input song. If the value is above a certain threshold, it suggests the model was likely trained on that song (possible copyright infringement), if not, it suggests no memorization of that content.

The split learning method enables every participant to access only the information they need. The record label reveals only derived features from its music files while the AI company keeps its model weights private and does not get the raw input. Split learning offers enhanced model privacy protection compared to horizontal FL because each party has access to only parts of the model which prevents full visibility of the entire model to any single entity. [3]. It is ideal when parties hold different modalities of data (here, one has a model or model updates, the other has data to test) – effectively a form of federated inference on combined inputs. Through split federated learning record labels and AI companies engage in a compliance check workflow that mirrors model training/inference activities while preventing data pooling. Through a shared split model the organizations process data collaboratively to identify copyright overlaps while maintaining local storage of proprietary information.

2 Privacy-Preserving Techniques for Secure Compliance

Ensuring privacy is paramount: The music owned by record labels and the parameters of the AI company's model must remain hidden throughout the federated compliance check. Our system employs multiple cryptographic and privacy methods to protect the federated process:

Secure Multi-Party Computation (SMPC): SMPC enables multiple participants to perform joint computations over their input data while keeping those inputs hidden from each other [4]. Our design uses SMPC protocols to enable collaborative analysis between AI model and record label's data. For example, the record label and AI company can employ an SMPC framework (like Facebook's CrypTen or Microsoft SEAL) to evaluate the AI's model on the label's song in a secret-shared manner. The model's computations (matrix multiplications, etc.) are performed on encrypted or secret-shared values, so the AI company never sees the actual audio features and the record label never sees the raw model outputs. This could be implemented with an additive secret sharing scheme: the record label secret-shares the intermediate features with the AI company (or a neutral server), and the AI company secret-shares its model parameters. They then perform the forward-pass computation by exchanging masked values. At the end of the computation, only the final result (e.g., a risk score) is revealed (and only to the party authorized to see it). SMPC essentially functions like a virtual trusted calculator: for instance, it can let the AI company privately evaluate its model on the record label's data [4], or allow multiple labels to perform a joint aggregate audit without sharing individual data. The process ensures that proprietary data and models stay secure and undisclosed throughout the compliance check.

Homomorphic Encryption (HE): Homomorphic encryption [5] enables users can perform calculations on encrypted data to receive encrypted results that decryption is possible only via owner of the secret key. HE can be used in selected areas of the compliance pipeline to improve data protection. For example, a record label can encrypt an audio feature vector with their public key and send it to the AI company. The AI company then runs its model on the encrypted data without ever decrypting it, using partially homomorphic operations (addition, multiplication on ciphertexts). The outcome is an encrypted infringement indicator, which only the record label can decrypt to see the result. Throughout this process, the AI company learns nothing about the input or the output, since all intermediate data remains encrypted. Fully Homomorphic Encryption (FHE) schemes (which allow arbitrary computations on ciphertexts) can be heavy for deep learning, hence we use optimizations like leveled HE or Partially Homomorphic Encryption (PHE) for specific operations to keep overhead reasonable [6]. For instance, computing a simple dot-product similarity or a reconstruction error between the model's output and a target song can be done under HE if we linearize the operation. Non-linear operations (like activations) can be handled either by the split learning approach (so that they occur on the AI's side in plaintext on already encrypted inputs) or by efficient garbled circuits if needed. By carefully choosing which parts of the computation to encrypt, we ensure a balance between privacy and performance. The cryptographic strength of HE means even if communications are intercepted, the content (songs or model responses) remains unintelligible without the decryption key.

Differential Privacy (DP): We incorporate differential privacy to protect against information leakage in any shared outputs or updates. DP works by adding carefully calibrated random noise to results so that the presence or absence of any single data record is indistinguishable. In our context, the AI music company could

train its model with DP-SGD (differentially private stochastic gradient descent), which would ensure the model does not memorize specific training examples (like a particular song) too exactly. This pre-emptively protects against infringement, because a DP-trained model is unlikely to regurgitate any one song verbatim. Even during the federated compliance check, DP can be applied: for example, if multiple record labels participate in a joint audit, the aggregated compliance metrics can have noise added before being revealed, so that no single label learns specifics about another label's queries. The record label's query results themselves could be noised if we only need a yes/no answer with high confidence. Importantly, the noise levels are set such that they do not obscure true infringements but hide minute details of the model's behavior. DP ensures that any one song (even if it were in training) has a limited influence on the output, preventing the exposure of exact memorized content. This technique is computationally cheap (just noise addition) and scales well, complementing heavier cryptographic methods by reducing how much sensitive information even exists in the computed results.

Zero-Knowledge Proofs (ZKP): Zero-knowledge proofs allow a party to prove a statement about data or computations without revealing the data itself [7]. We utilize ZKPs to make the compliance process verifiable and legally defensible. For instance, after training, the AI company can generate a zero-knowledge proof of training that attests "This model was trained only on licensed data and did not include Record Label X's songs" without revealing anything about the training data or model parameters. Recent advances in ZKPs for ML (zkML) enable proving properties of models, such as training steps or dataset membership, in a computationally feasible way [8][15]. Concretely, the AI company could commit to the dataset it used (e.g., via a cryptographic hash or Merkle root of all training data) and then provide a ZK-SNARK proof that none of the record label's song hashes are in the committed dataset. This is akin to a zero-knowledge set membership test – proving a set intersection is empty without revealing the actual sets. Alternatively, the AI company can prove that it followed a prescribed training procedure (for example, a training run with differential privacy enabled, or only using a specific approved dataset) [8]. The record label (or a regulator) can verify this proof and be mathematically assured of compliance. Another use of ZKP in our system is for the infringement test itself: if the record label gets a negative result (no infringement detected), the AI company could output a ZKP that the test was carried out honestly on the model in question (preventing an AI company from swapping in a different "clean" model just for the test). Although generating ZK proofs for deep learning computations can be resource-intensive, we confine their use to periodic checks or final audits to keep it tractable. The outcome is that compliance checks are trustless – the record label doesn't have to trust the AI company's word, they have cryptographic proof of either compliance or violation, which is crucial for legal defensibility.

By combining these measures – SMPC/secure computation for processing data, HE for data encryption in transit and compute, DP for output privacy, and ZKP for process verification – we create a robust privacy-preserving compliance system. Each technique is chosen to minimize performance hits: for example, we use partial HE and secret-sharing (which are faster than full FHE), we add only small noise for DP, and we generate ZK proofs for high-level properties rather than every single operation. The overall design ensures that at no point is sensitive information exposed in plain form, yet all parties can collaboratively achieve the goal of detecting unlicensed training.

When the model training is complete, the AI company can package a compliance report: this might include the ZK proofs of training, differential privacy parameters used, and summary of any internal compliance tests. The record labels, through the federated system, get the ability to verify this report and test the model themselves, yielding high assurance that if the model passes, it truly did not use unlicensed music.

To avoid high financial cost for any single entity, the federated compliance system can be managed by a neutral third-party service or consortium of stakeholders. This service can maintain the secure aggregation server and coordinate cryptographic key management. Using cloud computing with hardware acceleration (like FPGAs for HE, or SGX secure enclaves as a backup option) can speed up cryptographic operations, reducing runtime and therefore cost. Also, many cryptographic libraries are open-source and optimized, meaning the main expense is computing time. With model compression and batching, we ensure that even large models can be handled with commodity hardware given some time (hours, not days, for a thorough audit of a big model against millions of song samples, for instance). The design favors one-time heavy computations (like proof generation or full-catalog scan) only when absolutely necessary (e.g., a legal dispute), whereas routine compliance checks can be much lighter (sampling a subset of songs, using partial evaluations, etc., to get a quick assurance).

3 Workflow

Finally, we outline the end-to-end execution of the federated learning compliance system, from setup to verification:

Step 0: Initial Setup and Key Exchange – All participating entities (the AI company and one or more record labels, or an auditor) set up the cryptographic environment. This involves generating encryption keys (public/private key pairs for HE for each label, key shares for SMPC, etc.) and exchanging any public parameters. They also agree on the model split architecture and the protocol (which cut layer, what format features will be,

what threshold constitutes a violation, etc.). For example, the AI company publishes the architecture of the compliance model or the fingerprinting method it will use. A central coordination server (could be run by a neutral party or consortium) may exist to facilitate scheduling and key management, but it will not see any raw data or models.

Step 1: Registration of Data Commitments – The AI company commits to its training dataset and model. It computes a commitment hash (or Merkle root) of all training data it used. This is submitted to a smart contract or to the record labels in a ledger so that it's fixed (the company can't later change it). Likewise, each record label prepares a fingerprint database of their copyrighted songs – e.g., a set of audio hashes or embeddings – and commits to those (so that they can't maliciously add more songs later just to trap the AI company). These commitments will be used in ZK proofs later. This stage ensures both sides "lock in" the items of interest without revealing them.

Step 2: Local Model Training (AI Company) – The AI company trains its AI music model on its own data (e.g., publicly licensed music, user-generated music, etc.). This is done using its standard pipeline, possibly with differential privacy and logging as described. No external interaction is needed during core training, so no overhead is incurred here aside from any self-chosen privacy technique. Once the model is trained (or at certain checkpoints), it is saved for audit. Let's assume the model is now ready to be checked for compliance.

Step 3: The coordinator server notifies the AI company and relevant record label(s) that a check will happen. They establish a secure session. The AI company provides the server-side model for the split learning inference – typically, this means loading the second part of the model on a secure computation server. If using SMPC, the AI company secret-shares or encrypts its model weights with the computation service (or among multiple servers). If using a TEE (trusted execution environment) as an aid, the model could be loaded into an enclave. In any case, the AI company does not give the model in plaintext to the label, it only makes it available in the secure protocol. The record label in turn prepares its input data for the check. For instance, it selects a batch of 100 songs (or segments) that it strongly cares about. The label either keeps these on its local machine (for split learning) or encrypts them with homomorphic encryption (if the model will process them directly in encrypted form). All parties confirm readiness.

Step 4: Federated Inference/Processing – The record label's client-side application now goes through the selected songs one by one (or in batches). For each song, it does the following:

- Compute the feature representation (e.g., passes it through the local cut-layer of the model or simply prepares the raw input if using HE directly).
- Send the intermediate activation to the AI company's model server over an encrypted channel (TLS + the values might already be secret shares or encrypted numbers). If using pure HE, send the encrypted audio/features to the server.
- The AI company's server (or the joint MPC nodes) then perform the forward pass on the encrypted/secret shared data through the remaining network layers. For example, it computes the output logits or reconstruction of the input. Since the model is large, this computation is optimized as discussed (maybe using GPU, etc.).
- The server returns the encrypted result of the inference back to the record label. This might be the log-likelihood of the sequence, a set of output audio tokens, or a high-level "yes/no" flag in secret-shared form.

Step 5: Compliance Metric Computation – The record label now decrypts or reconstructs the results from Step 4. If the result was an encrypted likelihood score, the label decrypts it with its HE secret key. If it was done via MPC, the label combines its share of the result with the shares from the server to obtain the final number. Now the label has, for each song tested, a metric indicating how strongly the model reacted. The label compares these metrics to the expected range for non-members. For instance, if a certain song has a model likelihood far above a threshold (meaning the model highly likely has seen it [9], the system flags this song as a potential infringement. In practice, the label might set a threshold based on a statistical confidence (e.g., "if probability ¿ 0.9 that this was in training, flag it"). They could also use an internal classifier on the outputs – for example, if the output was the model trying to continue the song, the label can measure similarity between the continuation and the original. Some systems might automate this: e.g., compute a cosine similarity between audio embeddings of the original and the generated continuation. High similarity would yield a flag. These calculations are done on the label's side, so no privacy issue arises. The outcome of this step is a compliance report: perhaps a table of songs vs. scores, highlighting any that exceed the infringement threshold.

Step 6: Result Sharing and Proof Generation – Now the record label has preliminary results. If all songs are in the clear (no suspicious scores), the AI model likely did not use any of the label's data. The record label can then cryptographically sign an attestation that "We, Label X, have tested Model Y on [date] and found no evidence of training on Label X's catalog." This attestation can be shared with the AI company as part of a compliance certificate. On the other hand, if any song was flagged, the system can escalate. The record

label can notify the AI company (most likely through the protocol, without revealing which song in plaintext, at least initially). They might say "Song ID #5 from our hashed list appears to have been used in training. We request an explanation or remediation." At this stage, the AI company has the option to contest or accept. If contesting, this is where zero-knowledge proofs or additional verification come in. The AI company might invoke the previously computed commitment of its training set and perform a private set intersection (PSI) with the label's song in question. PSI can definitively show if that song (or its fingerprint) was in the training set, without the AI co learning which song it is (if done properly). If PSI comes out positive, it's proof of infringement. If PSI is negative but the model's behavior was still highly suspect, it could indicate the model learned something very close to the song (e.g., an overfitted surrogate). In either case, the parties now have cryptographic evidence. Optionally, they can involve a neutral auditor who reviews the evidence (the auditor could be given access to the song under NDA and maybe run a targeted test themselves for confirmation).

- The AI company can produce a ZK-proof that the model tested was indeed the one corresponding to the committed training hash. This prevents a scenario where the company trained a second "decoy" model without the label's songs just to pass the test. The proof would show that the weights of the deployed model are a result of training on the committed dataset (or at least that they match a certain hash that was committed). Such a proof might use zk-SNARKs as described in the proof-of-training concept [8].
- If the result is clean, the AI company might also produce a ZK-proof that none of the label's songs (from a committed list) appear in its training set. This could use a zk-proof of set disjointness, which might be heavy, but perhaps they only do it for a small set of top songs.

Step 7: Compliance Outcome – After analysis, one of two outcomes occurs. No Infringement Detected: All checks pass. Or, one or more labels detected their content in the model. In this case, the system can automatically provide evidence to the AI company and a regulator.

Step 8: Ongoing Monitoring – The federated system remains available for future checks.

Throughout this process, all actions (from key exchange to final verification) are designed to be auditable and repeatable. Each cryptographic message or proof can be logged (in encrypted form) to provide a trace in case of disputes. The combination of federated learning structure and advanced privacy techniques ensures that compliance verification is done scientifically and rigorously, minimizing trust and subjectivity. The result is a feasible, efficient, and privacy-preserving federated system that upholds copyright law without stifling the development of AI models. By balancing the load between parties and using cutting-edge cryptography, the solution scales to real-world industry usage – enabling record labels to defend their intellectual property and AI companies to innovate with accountability.

References

[1] MIT Media Lab, "Split Learning: Distributed and collaborative learning," [Online]. Available: `https://www.media.mit.edu/projects/distributed-learning-and-collaborative-learning-1/overview/`.

[2] Restack.io, "Federated learning vs split learning," [Online]. Available: `https://www.restack.io/p/federated-learning-answer-vs-split-learning-cat-ai`.

[3] C. Thapa, M. A. P. Chamikara, S. Camtepe, and L. Sun, "SplitFed: When federated learning meets split learning," *arXiv preprint arXiv:2004.12088*, Apr. 2020. doi: `https://doi.org/10.48550/arxiv.2004.12088`

[4] B. Knott, S. Venkataraman, Awni Hannun, S. Sengupta, M. Ibrahim, and van, "CrypTen: Secure Multi-Party Computation Meets Machine Learning," *arXiv preprint arXiv:2109.00984*, Sep. 2021, doi: `https://doi.org/10.48550/arxiv.2109.00984.`

[5] J. Ma, S. Naas, S. Sigg, and X. Lyu, "Privacy-preserving federated learning based on multi-key homomorphic encryption," *Int. J. Intell. Syst.*, vol. 37, no. 9, pp. 5880–5901, Apr. 2021, doi: `https://doi.org/10.1002/int.22818`.

[6] [6]S. A. Rieyan et al., "An advanced data fabric architecture leveraging homomorphic encryption and federated learning," *Information Fusion*, vol. 102, Feb. 2024, doi: `https://doi.org/10.1016/j.inffus.2023.102004`.

[7] R. Nguyen, "Zero Knowledge Proofs in Machine Learning: A Comprehensive Guide," SotaZK, Oct. 22, 2024. [Online]. Available: `https://sotazk.org/insights/zero-knowledge-proofs-in-machine-learning-a-comprehensive-guide/`.

[8] S. Garg et al., "Experimenting with Zero-Knowledge Proofs of Training," *CCS: Computer and Communications Security*, pp. 1880–1894, Nov. 2023, doi: https://doi.org/10.1145/3576915.3623202.

[9] S. Antebi, E. Habler, A. Shabtai, and Y. Elovici, "Tag&Tab: Pretraining Data Detection in Large Language Models Using Keyword-Based Membership Inference Attack," *arxiv.2501.08454*, Jan. 2025, doi: https://doi.org/10.48550/arxiv.2501.08454.

[10] R. S. Roman, P. Fernandez, A. Deleforge, Y. Adi, and Romain Serizel, "Latent Watermarking of Audio Generative Models," *arXiv:2409.02915*, Jan. 2024, doi: https://doi.org/10.48550/arxiv.2409.02915.

[11] N. Baracaldo and H. Shaul, "Federated Learning Meets Homomorphic Encryption," *IBM Research Blog*, Feb. 09, 2021. [Online]. Available: https://research.ibm.com/blog/federated-learning-homomorphic-encryption.

[12] "Audio-Visual Fingerprinting - ExCITe Center," *ExCITe Center*, 2025. [Online]. Available: https://drexel.edu/excite/innovation/met-lab/Expressive%20Robotics/Audio-Visual%20Fingerprinting/.

[13] J. George and Ashok Jhunjhunwala, "Scalable and robust audio fingerprinting method tolerable to time-stretching," *International Conference on Digital Signal Processing (DSP)*, pp. 436–440, Jul. 2015, doi: https://doi.org/10.1109/icdsp.2015.7251909.

[14] W. Yang et al., "FedZKP: Federated Model Ownership Verification with Zero-knowledge Proof," *arXiv:2305.04507*, Jan. 2023, doi: https://doi.org/10.48550/arxiv.2305.04507.

[15] B.-J. Chen, Suppakit Waiwitlikhit, I. Stoica, and D. Kang, "ZKML: An Optimizing System for ML Inference in Zero-Knowledge Proofs," *Association for Computing Machinery*, pp. 560–574, Apr. 2024, doi: https://doi.org/10.1145/3627703.3650088.