[Early Bird Access] TechJam 2025 Tracks & Problem Statements

631 654

📌 Congratulations on gaining exclusive Early Bird access to our Problem Statements ahead of the public release! *Not sure how to or where to start? Want to learn more from our Problem Statement setting teams?*

Join our **tech workshop webinars** to learn more about the Tracks that interest you! Look forward to key takeaways about the problems, our technologies, and the impact your projects can bring. You are highly encouraged to attend all the workshops that align with your interests, so as to make a more informed decision in selecting the track that works best for you.

- **Date**: 27 August 2025, Wednesday (GMT+8)
- **Time**: See below, under each Track
- **Webinar Link**: vc-sg.larkoffice.com/j/295831165

Got a question you hope our speakers will address during their webinars?

Submit your questions here: Workshops Q&A Form

1. Filtering the Noise: ML for Trustworthy Location Reviews

✨ **Workshop Webinar with Q&A** will be held on **27 Aug, 1-1.30pm**.

Introduction

Online reviews play a crucial role in shaping public perception of local businesses and locations. However, the presence of irrelevant, misleading, or low-quality reviews can distort the true reputation of a place. With the proliferation of user-generated content, ensuring the quality and relevancy of reviews is more important than ever.

This hackathon challenges students to leverage Machine Learning (ML) and Natural Language Processing (NLP) to automatically assess the quality and relevancy of Google location reviews, aligning them with a set of well-defined policies. The ultimate goal is to improve the reliability of review platforms and enhance user experience.

Problem Statement

Design and implement an ML-based system to evaluate the quality and relevancy of Google location reviews. The system should:

- Gauge review quality: Detect spam, advertisements, irrelevant content, and rants from users who have likely never visited the location.
- Assess relevancy: Determine whether the content of a review is genuinely related to the location being reviewed.
- Enforce policies: Automatically flag or filter out reviews that violate the following example policies:
  - No advertisements or promotional content.
  - No irrelevant content (e.g., reviews about unrelated topics).
  - No rants or complaints from users who have not visited the place (can be inferred from content, metadata, or other signals).

Motivation & Impact

- For Users: Increases trust in location-based reviews, leading to better decision-making.
- For Businesses: Ensures fair representation and reduces the impact of malicious or irrelevant reviews.
- For Platforms: Automates moderation, reduces manual workload, and enhances platform credibility.

Data Sources

Data Sources	Details

Public Datasets	• Google Review Data: Open datasets containing Google location reviews (e.g., Google Local Reviews on Kaggle: https://www.kaggle.com/datasets/denizbilginn/google-maps-restaurant-reviews) • Google Local review data: https://mcauleylab.ucsd.edu/public_datasets/gdrive/googlelocal/ • Alternative Sources: Yelp, TripAdvisor, or other open review datasets for supplementary training.
Student-Crawled Data	• Students are encouraged to crawl additional reviews from Google Maps (in compliance with Google's terms of service). • Example of scraping Google Reviews: https://www.youtube.com/watch?v=LYMdZ7W9bWQ
Additional Signals	• GPS Location: Metadata about the reviewed place. • User Metadata: (If available) Such as review history, timestamps, etc.
Internal Evaluation Data	• The developed algorithms will be evaluated using TikTok Local Service's internal review dataset to assess real-world efficacy.

Task Requirements

Participants must:

1. Preprocess and Clean Data: Remove noise, handle missing values, and standardize formats.
2. Feature Engineering:
   - Textual features (NLP): sentiment, topic modeling, keyword extraction, etc.
   - Metadata features: review length, posting time, user history, GPS proximity, etc.
3. Policy Enforcement Module: Implement logic to detect policy violations.
4. Model Development:
   - We do not restrict the input structure of the model, but please ensure that all input details are clearly documented.
   - Train / Prompt Engineering ML/NLP models (e.g., transformers, LSTM, classical ML) for classification and relevancy scoring.
   - Optionally, build an ensemble or multi-task learning approach.
5. Evaluation and Reporting:
   - Evaluate on provided and internal datasets.
   - Report precision, recall, F1-score, and other relevant metrics.
   - Provide a summary of findings and recommendations.

Example Policies

Policy Type	Description	Example Violation
No Advertisement	Reviews should not contain promotional content or links.	"Best pizza! Visit www.pizzapromo.com for discounts!"
No Irrelevant Content	Reviews must be about the location, not unrelated topics.	"I love my new phone, but this place is too noisy."
No Rant Without Visit	Rants/complaints must come from actual visitors (inferred via content/metadata).	"Never been here, but I heard it's terrible."

Suggested Workflow

This serves as a guide to teams taking up this problem statement

a. Data Collection: Obtain and preprocess Google location reviews.

b. Exploratory Data Analysis: Understand data distribution, common violations, etc.

c. Feature Engineering: Extract both textual and non-textual features.

d. Modeling: Build and train ML/NLP models.

e. Policy Module: Implement rule-based or ML-based policy detectors.

f. Evaluation: Test on both public and TikTok internal datasets.

g. Reporting: Summarize findings, discuss limitations, and suggest improvements.

Possible Detailed Workflow for the 72-Hour Challenge

Day	Details
1	**Data Pipeline Setup & Initial Modeling** • **Data Collection:** Utilize the provided Google Review dataset as the primary data source. • **Prompt Engineering for LLMs:** Design prompts to classify each review according to the predefined policy categories (e.g., advertisement, irrelevant content, rants without visit). • **Prototype Pipeline with Hugging Face:** Build a basic review classification pipeline using Hugging Face models (e.g., Gemini, Qwen). Run inference on the collected reviews using either: ◦ Fine-tuned classification models ◦ Few-shot LLMs with prompts
2	**Evaluation & Label Refinement** • **Result Analysis:** Compare model predictions against ground-truth labels (if available). Evaluate precision, recall, and F1-score for each violation type. • **Handling Missing Labels:** If no ground truth labels are provided: Use a more advanced LLM (e.g., GPT-4o) to generate pseudo-labels. Alternatively, conduct manual annotation for a small subset to serve as a validation set. • **Iterate and Refine:** Use evaluation results to refine prompts, adjust model thresholds, or improve labeling strategies.
3	**Prepare all deliverables for submission** • Complete all deliverables for submission and presentation • Address any outstanding tasks from Day 1 and Day 2

Tools & Technologies

- Programming Languages: Python (preferred), R, or others.
- NLP Libraries: HuggingFace Transformers, spaCy, NLTK, Gensim.
- ML Frameworks: scikit-learn, TensorFlow, PyTorch.
- Data Processing: pandas, NumPy.
- Visualization: matplotlib, seaborn, Plotly.
- Crawling (if needed): Scrapy, BeautifulSoup (respecting TOS).

Deliverables

1. **Text Description (Devpost Submission)**

Provide a written description that clearly explains the features, functionality, and relevance of your project. This description must include:

- A clear explanation of how your solution addresses the problem of assessing the quality and relevancy of location-based reviews
- The specific problem statement tackled from the challenge prompt
- Development tools used (e.g., VSCode, Colab, Jupyter, etc.)
- APIs used (e.g., OpenAI GPT-4o, Google Maps API, etc.)
- Libraries and frameworks used (e.g., Hugging Face Transformers, PyTorch, scikit-learn, pandas, etc.)
- Assets and datasets used (e.g., Google Local Reviews dataset, manually labeled data)

2. **Public GitHub Repository**

Include a link to your public GitHub repository, which must contain:

- Well-structured, commented code covering all components: data collection, preprocessing, modeling, policy enforcement, and evaluation
- A README file that includes:
  - Project overview
  - Setup instructions
  - How to reproduce results
  - Team member contributions (if applicable)

3. **Demonstration Video**

Teams are expected to submit a short demo video that:

- Clearly shows the system functioning on the device or environment it was built for (e.g., inference results, dashboard demo, model predictions)
- Is uploaded to YouTube and set to public visibility
- Is linked in the Devpost text description
- Does not contain third-party trademarks or copyrighted content without permission

Track Flexibility Note: If this track's focus (e.g., backend modeling or NLP pipelines) makes it impractical to demonstrate a front-end interface, we will accept a walkthrough video showing API usage, inference examples, or result analysis instead.

4. **Interactive Demo (Optional for Bonus Points 🌟)**

You may also include an interactive component (e.g., web app, dashboard, or notebook demo) showcasing:

- Review classification output
- Policy violation detection interface
- Real-time prediction examples

Resources

You may use the following:

Resources	Details
Frames	• https://huggingface.co/docs/huggingface_hub/en/package_reference/inference_client
Models	• Gemma 3 12b ◦ https://huggingface.co/google/gemma-3-12b-it • Qwen3 8b ◦ https://huggingface.co/Qwen/Qwen3-8B

2. Harnessing MLLM for Next-Generation UI Automation Testing

✨ **Workshop Webinar with Q&A** will be held on **27 Aug, 1.45-2.15pm**.

Problem Statement