Enunciado de la tercera práctica

Fecha entrega: Domingo 29 de diciembre, 23:55 a través de AG

Tarea: Asistente inteligente para el transporte público

En esta práctica, desarrollarás un prototipo de asistente inteligente para el transporte público utilizando detección de eventos basada en audio para mejorar el entorno de los pasajeros, atendiendo las necesidades de los usuarios y mejorando la accesibilidad.

Implementarás el reconocimiento de tres sonidos utilizando aprendizaje por transferencia con YamNet, construyendo tu propio dataset para entrenar un clasificador personalizado. Utilizarás también LLMs para generar mensajes personalizados para los usuarios de la red de Cercanías de Madrid.

Este ejemplo demuestra cómo el aprendizaje por transferencia basado en audio y los LLM pueden utilizarse para desarrollar sistemas que satisfagan objetivos fundamentales de entornos de inteligencia ambiental:

- Adaptarse a las necesidades específicas del transporte público, ofreciendo soluciones contextuales y dinámicas.
- Promover un entorno más respetuoso y accesible, mejorando la convivencia entre los pasajeros.
- Incorporar tecnologías avanzadas en la vida cotidiana, mostrando el potencial de la inteligencia ambiental para resolver problemas reales.

Sonidos y acciones

- Sonido 1: Golpes de Asiento al Levantarse

  Detectar golpes o movimientos bruscos en los asientos al levantarse para recordar a los pasajeros que tengan cuidado con los asientos plegables o automáticos, ayudando a prevenir accidentes o daños. Generar mensajes que se muestren en pantallas digitales del vagón. Generar mensajes adicionales en inglés, francés, etc.

- Sonido 2: Sonidos de alarma

  Detectar alarmas de emergencia o sonidos inusuales y proporcionar notificaciones inclusivas, considerando a personas con problemas auditivos.

- Sonido 3: Equipaje rodante en zonas no permitidas

  Detectar sonidos asociados al movimiento de maletas con ruedas o carritos en áreas donde está prohibido (como zonas de descanso o escaleras). Esto ayuda a mejorar la seguridad y a mantener la comodidad de otros pasajeros. Mostrar mensajes en pantallas junto a las áreas afectadas.

Preparación del conjunto de datos

Dado que estos sonidos no forman parte de conjuntos de datos comunes como ESC-50, será necesario crearlos o adquirirlos, grabando muestras específicas de los siguientes eventos:

- Golpes de asiento al levantarse: Graba sonidos producidos por el movimiento brusco de asientos plegables o automáticos en diferentes escenarios. Incluye variaciones en intensidad y velocidad del golpe.
- Sonidos de alarma: Captura ejemplos de alarmas comunes en contextos de transporte público, como alarmas de emergencia, alertas de puertas o sonidos inusuales, asegurándote de incluir variaciones de tono y frecuencia.
- Equipaje rodante en zonas no permitidas: Graba sonidos de ruedas de maletas o carritos moviéndose sobre diferentes superficies (madera, baldosas, alfombras) en espacios similares a los de estaciones o vagones.

También puedes complementar las grabaciones con clips de audio recuperados de fuentes online para aumentar la diversidad de las muestras.

Tendrás que recopilar 20 muestras de audio de 5 segundos para cada sonido (golpes de asiento, alarmas, equipaje rodante) y etiquetarlas adecuadamente. Incluye algunas muestras de ruido de fondo, como conversaciones bajas, ruido ambiental del tren o sonidos mecánicos, para mejorar la robustez del sistema. Esto permitirá que el clasificador sea más confiable en escenarios reales.

Utiliza YAMNet para extraer embeddings para cada muestra de sonido. Entrena un clasificador simple utilizando estos embeddings para categorizar los tres tipos de sonido seleccionados. Finalmente, implementa un sistema que escuche audio en tiempo real y clasifique los sonidos, activando los mensajes correspondientes según el evento detectado.

Integración de modelos de lenguaje

Utiliza transformers.js directamente en el navegador para interactuar con modelos generativos de lenguaje y generar sugerencias y respuestas contextualmente relevantes. Proporciona al modelo información contextual, como el tipo de sonido detectado, la ubicación o el nivel de urgencia, como parte del prompt de entrada para mejorar la personalización de las respuestas. Se tendrán que generar prompts efectivos para interactuar con el modelo generativo de lenguaje en transformers.js, adaptándose al contexto detectado. Esto significa que cada prompt debe incluir información relevante sobre el evento ocurrido para obtener respuestas claras, útiles y adecuadas al entorno.

Ejemplo:

```
Tipo de sonido: Golpes de asiento
Ubicación: Vagón 4
Urgencia: Moderada

Un pasajero ha golpeado el asiento al levantarse en el vagón 4. Genere un mensaje educado para mostrar en pantallas que recuerde a los pasajeros manejar los asientos con cuidado.
```

Respuesta:
“Por favor, asegúrese de manejar los asientos con cuidado al levantarse para evitar golpes o accidentes. Gracias por su colaboración.”

Además, implementa un sistema de traducción automática para mostrar mensajes en múltiples idiomas, garantizando accesibilidad a una audiencia diversa. Complementa esta funcionalidad con un servicio opcional de text-to-speech para aquellos casos en los que las notificaciones auditivas sean adecuadas y necesarias.

Normas de entrega:

Se entregará en AG un fichero comprimido (.zip) con:

- Código fuente: Incluye todos los archivos necesarios para ejecutar la aplicación (HTML, CSS, y JavaScript).
- Memoria (.pdf): El documento debe contener:
    - Descripción clara del flujo de trabajo, desafíos enfrentados y soluciones implementadas.
    - Evaluación personal sobre lo aprendido en la práctica.
    - Conexión entre el contenido de la práctica y su aplicabilidad en la vida real o en futuros proyectos.
- Breve video enseñando la interacción con el sistema

Puntuación

- Detección de sonidos y clasificación: 1.2 punto
    - Generación del dataset, configuración y uso de yamnet
- Integración de modelos generativos: 1 punto
    - Elección de los modelos, Implementación con transformers.js, Generación de prompts relevantes
- Experiencia de usuario: 0.3
    - Reconocimiento de sonidos, traducciones en al menos dos idiomas, consideraciones de accesibilidad
- Memoria: 0.5