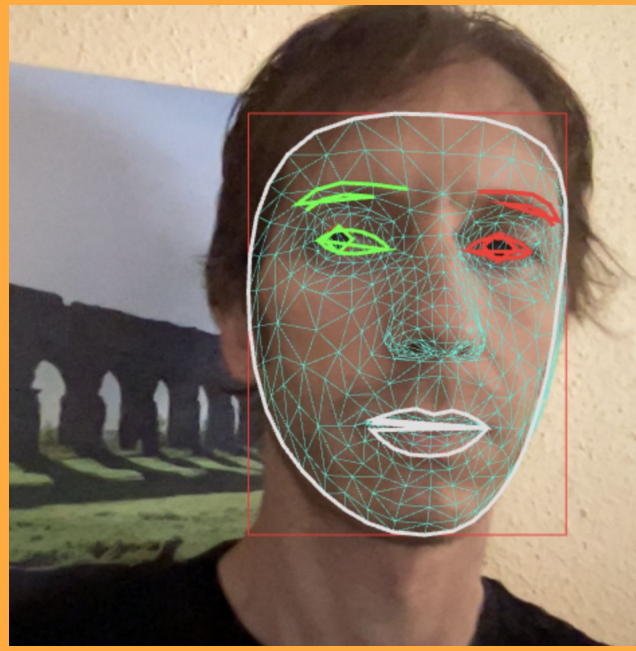
Inteligencia Ambiental

Máster Universitario en Inteligencia Artificial Aplicada



Aprendizaje por transferencia en TFJS

uc3m | Universidad Carlos III de Madrid

Contenido

1. YAMNet
2. Dataset de eventos sonoros ESC50
3. Aprendizaje por transferencia con YaMNet

1. YAMNet

YAMNet: Yet Another Mobile Network

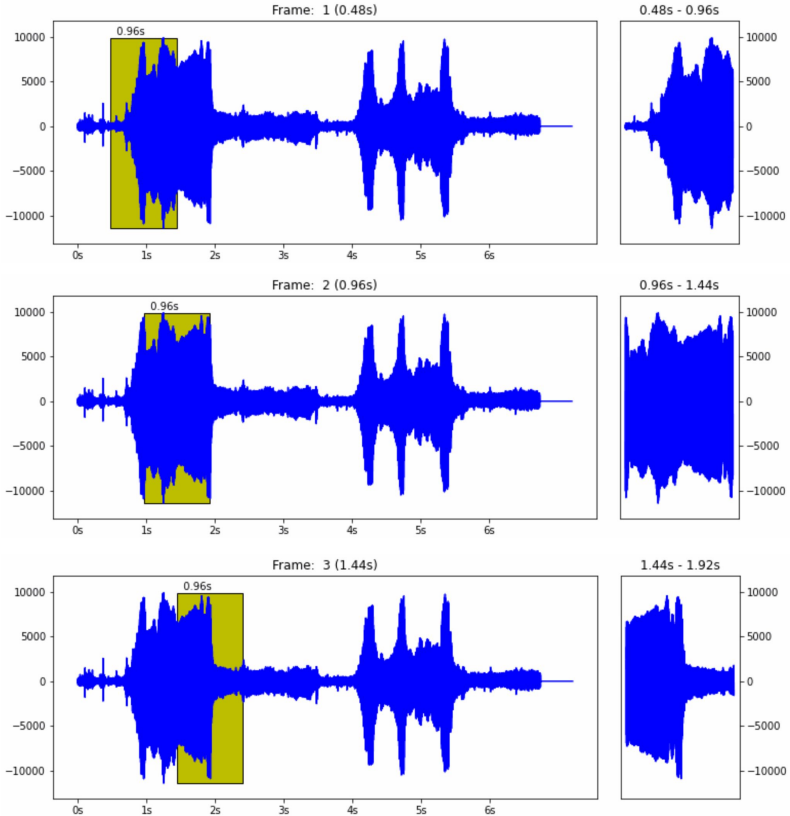
https://www.kaggle.com/models/google/yamnet

- Modelo de aprendizaje automático para **clasificar** eventos de audio realizando predicciones independientes para cada uno de los 521 eventos de audio de la ontología AudioSet
- Usa la arquitectura MobileNet v1 y fue entrenado usando el corpus AudioSet
- **Funciona con clips de audio de 16Khz**
- Se puede utilizar como extractor de características para crear rápidamente clasificadores de audio especializados sin requerir una gran cantidad de datos etiquetados y sin tener que entrenar un modelo grande de principio a fin
    - La salida de representaciones del audio (**embeddings**) 1024D de YAMNet se puede utilizar como las características de entrada de otro modelo que luego se puede entrenar con una pequeña cantidad de datos para una tarea específica

```
const model = await tf.loadGraphModel(modelUrl, { fromTFHub: true });
const waveform = tf.zeros([16000 * 3]);
const [scores, embeddings, spectrogram] = model.predict(waveform);
scores.print(verbose=true);  // shape [N, 521]
embeddings.print(verbose=true);  // shape [N, 1024]
spectrogram.print(verbose=true);  // shape [M, 64]
// Find class with the top score when mean-aggregated across frames.
scores.mean(axis=0).argMax().print(verbose=true);
// Should print 494 corresponding to 'Silence' in YAMNet Class Map.
```

YAMNet entrada

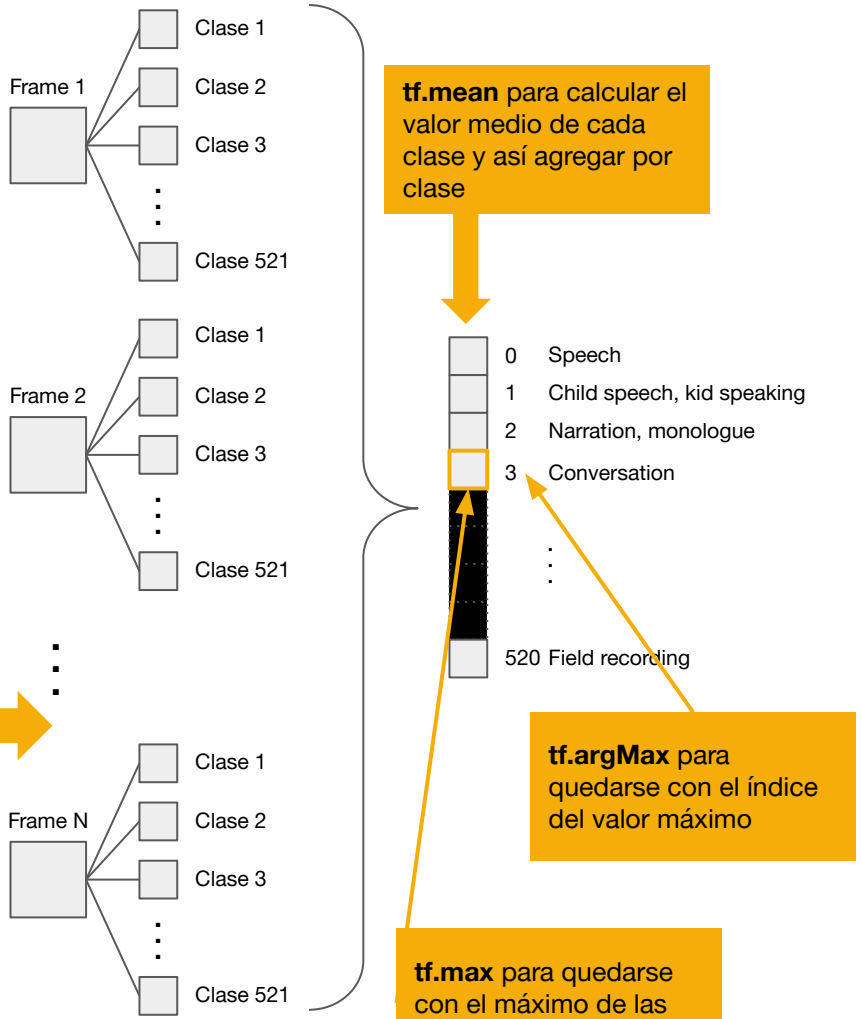
- El modelo acepta un **Tensor 1D** de tipo **'float32'** que contiene un **forma de onda** de un audio de **longitud arbitraria**, representada como muestras **mono** de **16 kHz** en el rango **[-1.0, +1.0]**
- Internamente, la forma de onda se recorta en fotogramas (frame) con ventanas deslizantes de 0,96 segundos de duración y saltos de 0,48 segundos, y luego se ejecuta el modelo en un lote de estos fotogramas
- Para determinar la clasificación del clip de audio, las puntuaciones se pueden agregar por clase a través de los fotogramas (por ejemplo, usando la agregación media). Finalmente, para encontrar la clase con la puntuación más alta, se toma el máximo de las 521 puntuaciones agregadas



https://blog.tensorflow.org/2021/03/transfer-learning-for-audio-data-with-yamnet.html

YAMNet entrada

- El modelo acepta un **Tensor 1D** de tipo **'float32'** que contiene un **forma de onda** de un audio de **longitud arbitraria**, representada como muestras **mono** de **16 kHz** en el rango **[-1.0, +1.0]**
- Internamente, la forma de onda se recorta en fotogramas (frame) con ventanas deslizantes de 0,96 segundos de duración y saltos de 0,48 segundos, y luego se ejecuta el modelo en un lote de estos fotogramas
- Para determinar la clasificación del clip de audio, las puntuaciones se pueden agregar por clase a través **de los fotogramas** (por ejemplo, usando la agregación media). Finalmente, para encontrar la clase con la puntuación más alta, se toma el **máximo** de las 521 puntuaciones agregadas



Ejemplo: utilizar YAMNet para clasificación de audio desde microfono

Código en AG: https://aulaglobal.uc3m.es/mod/resource/view.php?id=5273082

- Capturar flujo de audio desde micrófono
- Activar la clasificación de audio con YAMNet cada 3 segundos

```
recorder.port.onmessage =  async(e) => {
    const inputBuffer = Array.from(e.data);

    if (inputBuffer[0] === 0) return;

    timeDataQueue.push(...inputBuffer);

    const num_samples = timeDataQueue.length;
    if (num_samples >= SAMPLE_RATE * NUM_SECONDS) {
        const audioData = new Float32Array(timeDataQueue.splice(0, SAMPLE_RATE * NUM_SECONDS));
        console.log("Start classification");
        const audioTensor = tf.tensor(audioData);
        const [scores, embeddings, spectrogram] = model.predict(audioTensor);
        scores.mean(axis=0).argMax().print(verbose=true);
    }
}
```

YAMNet salida

- El modelo devuelve un array que contiene tres Tensores de tipo **'float32'**:
    - **scores**: tensor de forma **(N, 521)** que contiene las puntuaciones por fotograma para cada una de las 521 clases de AudioSet compatibles con YAMNet
    - **spectrogram**: tensor de forma **(NUM_SPECTROGRAM_FRAMES, 64)** que representa el espectrograma mel de la forma de onda completa. Estas son las características de la pista de audio pasadas al modelo. NUM_SPECTROGRAM_FRAMES es el número de fotogramas producidos a partir de la forma de onda al deslizar una ventana de análisis de espectrograma de 0,025 segundos de duración con un salto de 0,01 segundos, y 64 representa el número de **bandas Mel**
    - **embeddings**: tensor de forma **(N, 1024)** que contiene embeddings por fotograma. El array de embeddings, tensor 1D **(1024)**, es la salida agrupada promedio que alimenta la capa clasificadora final
- El índice de columna (0-520) del tensor de puntuaciones se asigna al nombre de clase de AudioSet correspondiente utilizando el mapa de clase de YAMNet, que está disponible como un archivo CSV en GitHub
- El array de embeddings son las **características que se utilizan para aprendizaje por transferencia**

2. Dataset de eventos sonoros ESC50

Dataset ESC50

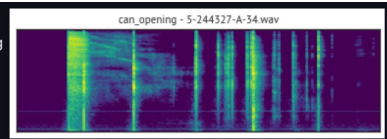
https://github.com/karolpiczak/ESC-50

https://www.karolpiczak.com/papers/Piczak2015-ESC-Dataset.pdf

- Conjunto de datos para la clasificación de sonido ambiental
- 2000 grabaciones de audio ambientales **(44.1 KHz, mono**)
- **50 clases** (**40 clip de audio por clase**) de 5 grandes categorías (10 clases por categoría): Sonidos de animales, Paisajes sonoros naturales y sonidos del agua, Sonidos humanos (no habla), Sonidos interiores/domésticos, Ruidos exteriores/urbanos
- Cada grabación tiene una duración de **5 segundos** y proviene originalmente del proyecto Freesound
- El conjunto de datos se ha organizado previamente en 5 particiones para la validación cruzada comparable, asegurándose de que los fragmentos de 5 segundos de un mismo archivo de origen estén contenidos en una sola partición

The **ESC-50 dataset** is a labeled collection of 2000 environmental audio recordings suitable for benchmarking methods of environmental sound classification.

The dataset consists of 5-second-long recordings organized into 50 semantical classes (with 40 examples per class) loosely arranged into 5 major categories:



Animals	Natural soundscapes & water sounds	Human, non-speech sounds	Interior/domestic sounds	Exterior/urban noises
Dog	Rain	Crying baby	Door knock	Helicopter
Rooster	Sea waves	Sneezing	Mouse click	Chainsaw
Pig	Crackling fire	Clapping	Keyboard typing	Siren
Cow	Crickets	Breathing	Door, wood creaks	Car horn
Frog	Chirping birds	Coughing	Can opening	Engine
Cat	Water drops	Footsteps	Washing machine	Train
Hen	Wind	Laughing	Vacuum cleaner	Church bells
Insects (flying)	Pouring water	Brushing teeth	Clock alarm	Airplane
Sheep	Toilet flush	Snoring	Clock tick	Fireworks
Crow	Thunderstorm	Drinking, sipping	Glass breaking	Hand saw

3. Aprendizaje por transferencia con YaMNet

Tutorial 5: clasificación de sonidos personalizados

https://docs.google.com/document/d/1Y4Ixuck2iq9y7WCKv_oYV7JDr7YhIhGueCaBeMrp_OA

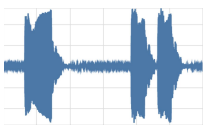
Aprender a:

- Utilizar las Web Audio API para
  - Cargar ficheros de audio y obtener su representación como forma de onda
  - Utilizar el micrófono como fuente de audio
- Obtener las características de un fichero de audio del módelo YAMNet para su clasificación
- Realizar aprendizaje por transferencia utilizando las representaciones (embeddings) generadas por YAMNet

Realizar clasificaciones

Utilizamos YAMNet como extractor de características para crear rápidamente clasificadores de audio especializados sin requerir una gran cantidad de datos etiquetados

forma de onda



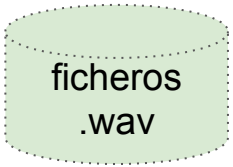
embeddings
[N_FRAMES,1024]

Predicción de categoría para cada fotograma
[N_FRAMES, N_CLASSES]

Fuente de audio (ficheros o micrófono) → **YAMNet** → **Clasificador personalizado** →

Preparar datos de entrenamiento del modelo

```
const trainDataArray = await (await fetch("data/trainData.json")).json();
```



Leer cada fichero WAV

```
for (let i = 0; i < trainDataArray.length; i++) {
```

```
const audioData = await
getTimeDomainDataFromFile(`data/audio/
${trainDataArray[i].fileName}`,
context);
```

Extraer
- Forma de onda
- Clase (ID)

Extraer las características del audio con YAMNet

```
const embeddings = await
getEmbeddingsFromTimeDomainData(yamnet,
audioData);
```

Añadir ejemplo a
- array de entrada (características)
- array de salida (clase)

```
for (let j = 0; j < embeddingsArray.length; j++)
{
  INPUT_DATA.push(embeddingsArray[j]);

OUTPUT_DATA.push(trainDataArray[i].classNumber);
}
```

Entrenar el modelo con las características y la clasificación esperada

```
createAndTrainCustomAudioClassificationModel (yamnet,
trainDataArray)
```

Resumen

- Yamnet es un modelo de aprendizaje automático para clasificar eventos de audio realizando predicciones independientes para cada uno de los 521 eventos de audio de la ontología AudioSet
- Yamnet se puede utilizar como extractor de características para crear rápidamente clasificadores de audio especializados sin requerir una gran cantidad de datos etiquetados