Procesamiento de Señales de Audio

Muestreo · Nyquist · FFT · Espectrogramas · Filtros Mel · MFCCs
Preprocesamiento · Dataset ESD · Balanceo de Clases

Tesis CIMAT — Fernández Morales, I. (2024)
Dir: Dr. Hugo Arnoldo Mitre Hernández

Pipeline Completo: Del Sonido a los MFCCs

El procesamiento de audio para clasificación emocional transforma una señal temporal cruda en una representación compacta (MFCCs) que captura las propiedades perceptuales del habla humana, reduciendo la dimensionalidad 25× sin pérdida de información relevante.

Dimensiones de la señal en cada etapa

Entrada: señal cruda de 2 segundos a 16 kHz = 32,000 muestras (float32 → 128 KB)

Salida: matriz MFCC de 20 × 63 = 1,260 coeficientes (float32 → 4.9 KB) — Compresión ≈ 25×

Muestreo Digital y Teorema de Nyquist

Definición: Muestrear es tomar valores discretos de una señal continua a intervalos regulares T = 1/fs.

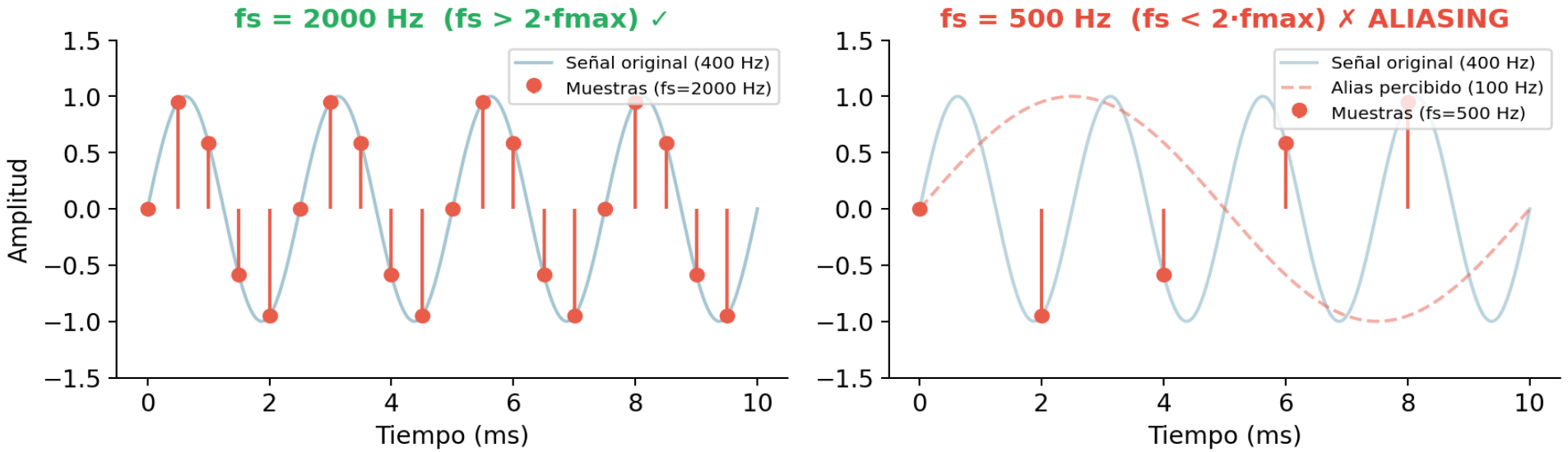
Teorema de Nyquist-Shannon (1949)

$$f_s \geq 2 \cdot f_{max}$$

Una señal de banda limitada con frecuencia máxima fmax puede ser reconstruida perfectamente a partir de sus muestras si y solo si la frecuencia de muestreo fs es al menos el doble de fmax. La frecuencia fn = fs/2 se denomina

Si fs < 2·fmax → ALIASING

Las frecuencias superiores a fs/2 se "pliegan" (fold) sobre las inferiores, generando componentes fantasma que no pueden distinguirse de frecuencias reales. Este artefacto es irreversible.



Aplicación: Muestreo para Voz Humana

Cálculo paso a paso

1. Rango de la voz humana:
   - Frecuencia fundamental (F0): 85–255 Hz
   - Formantes y armónicos: hasta ~8,000 Hz
   - Rango total audible: 20–20,000 Hz

2. Aplicar Nyquist:
   fmax(voz) ≈ 8,000 Hz
   fs ≥ 2 × 8,000 = 16,000 Hz

3. Elección: fs = 16,000 Hz (16 kHz)
   — Estándar para procesamiento de voz
   — Igual al ESD (Emotional Speech Database)

4. Duración: 2 segundos por ventana
   N = fs × duración = 16,000 × 2 = 32,000 muestras

5. Memoria requerida (float32, 4 bytes/muestra):
   32,000 × 4 = 128,000 bytes ≈ 125 KB
   ✓ Dentro del límite de 160 KB del ESP32

Frecuencias de muestreo estándar

8,000 Hz	Telefonía	fmax = 4 kHz
16,000 Hz	Voz / nuestro proyecto	fmax = 8 kHz
22,050 Hz	Radio AM	fmax = 11 kHz
44,100 Hz	CD (Red Book)	fmax = 22 kHz
48,000 Hz	Video/cine profesional	fmax = 24 kHz
96,000 Hz	Estudio de grabación	fmax = 48 kHz

Nota: tesis original (ESP32-WROVER-E)

Por limitaciones de SRAM (160 KB/variable), se usó fs = 4,096 Hz con duración de 2 s:

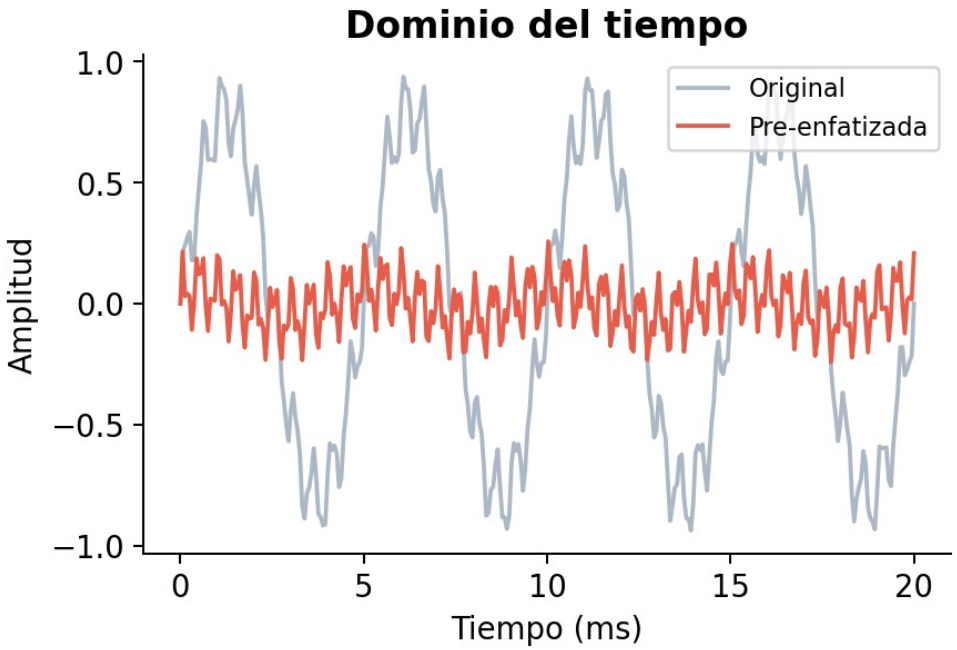
N = 4,096 × 2 = 8,192 muestras
Memoria = 8,192 × 4 = 32,768 bytes ≈ 32 KB

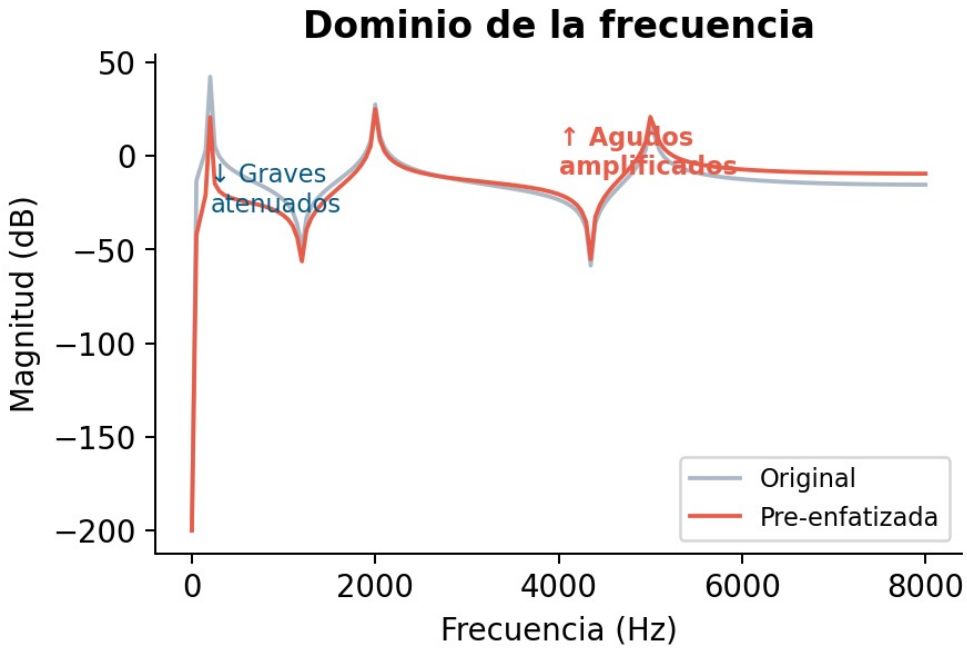
Se eligió potencia de 2 para optimizar FFT.

Paso 1: Pre-énfasis — Filtro Paso Alto

$$y[n] = x[n] - \alpha \cdot x[n-1] \qquad \text{donde } \alpha = 0.97 \text{ (coeficiente de pre-énfasis)}$$

Propósito: compensar el sesgo espectral glótico. Las cuerdas vocales producen un espectro con caída de ~6 dB/octava — las frecuencias graves dominan por naturaleza. El pre-énfasis ecualiza la energía espectral amplificando los agudos (consonantes fricativas: s, f, t, k) que son clave para la discriminación emocional.





Pre-énfasis: Detalle Matemático

Análisis en Z del filtro FIR

Función de transferencia:

$$H(z) = 1 - \alpha \cdot z^{-1}$$

Respuesta en frecuencia:

$$H(e^{j\omega}) = 1 - \alpha \cdot e^{-j\omega}$$

$$|H(e^{j\omega})| = \sqrt{(1 - 2\alpha \cdot \cos(\omega) + \alpha^2)}$$

Para $\omega = 0$ (DC): $|H| = 1 - 0.97 = 0.03$ (atenúa 30 dB)
Para $\omega = \pi$ (Nyquist): $|H| = 1 + 0.97 = 1.97$ (amplifica 6 dB)

Frecuencia de corte (-3 dB):

$fc = fs/(2\pi) \cdot \arccos((1+\alpha^2)/(2\alpha))$

$fc \approx 0.015 \cdot fs \approx 240$ Hz (para fs=16 kHz)

El filtro es un paso alto de primer orden
con ganancia unitaria en frecuencias altas.

Ejemplo numérico completo

```
Entrada x = [100, 102, 98, 105, 103, 97]

y[0] = x[0] = 100                (primer
elemento)
y[1] = 102 - 0.97 × 100 = 5.00
y[2] = 98  - 0.97 × 102 = -0.94
y[3] = 105 - 0.97 × 98  = 9.94
y[4] = 103 - 0.97 × 105 = 1.15
y[5] = 97  - 0.97 × 103 = -2.91

Salida y = [100, 5.00, -0.94, 9.94, 1.15,
-2.91]
```

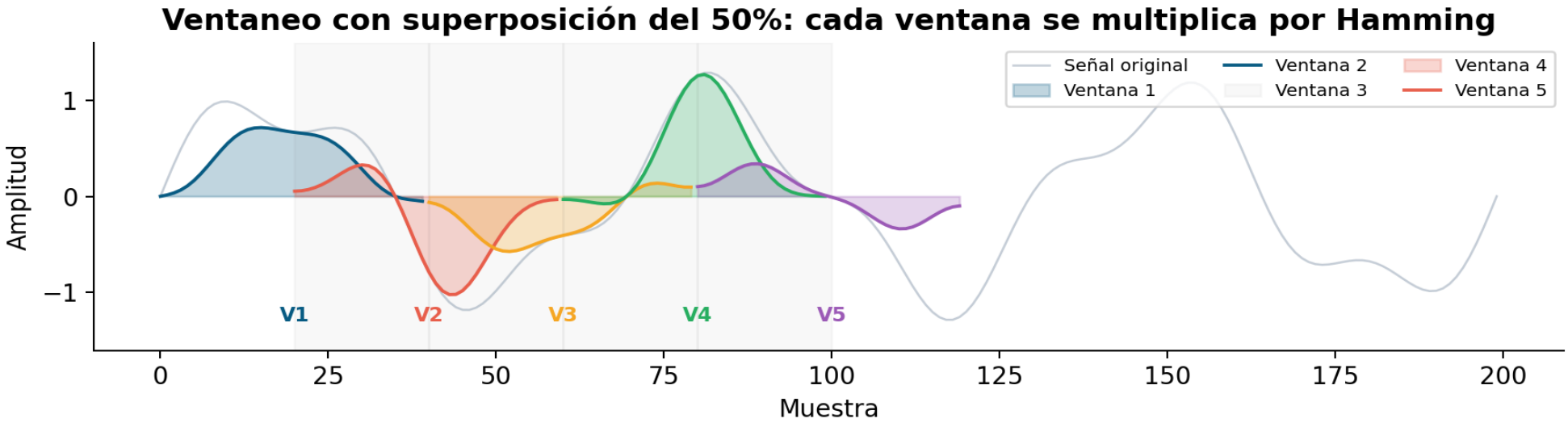
Python

```python
# Implementación vectorizada (sin bucle)
def preemphasis(signal, alpha=0.97):
    return np.append(
        signal[0],
        signal[1:] - alpha * signal[:-1]
    )

# Aplicar a todo el dataset
for audio in audios:
    audio_pe = preemphasis(audio)
```

Paso 2: Ventaneo — Segmentación Temporal

La señal de voz es no estacionaria: sus propiedades espectrales cambian con el tiempo. Para analizarla con la FFT (que asume periodicidad), dividimos la señal en segmentos cortos (20–40 ms) donde podemos asumir cuasi-estacionariedad.



Parámetros del ventaneo (nuestra configuración)

N_señal = 32,000 muestras | N_ventana = 512 muestras (32 ms) | N_hop = 256 muestras (16 ms, 50% overlap)

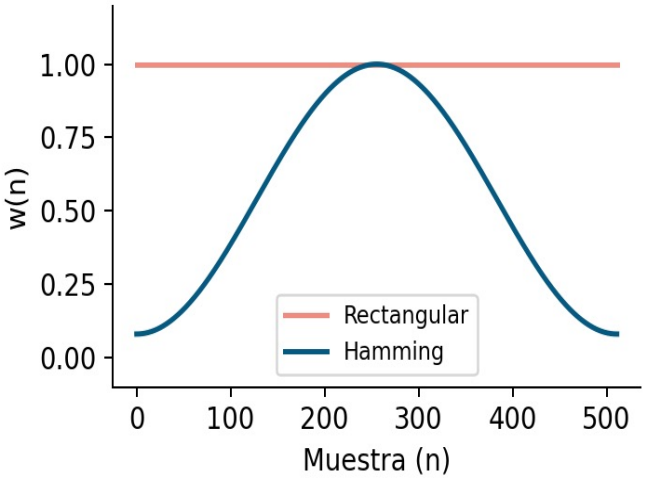
Número total de ventanas = ⌊(N_señal − N_ventana) / N_hop⌋ + 1 = ⌊(32,000 − 512) / 256⌋ + 1 = 123 + 1 = 124 frames

Superposición del 50%: cada muestra de la señal contribuye a exactamente 2 ventanas (excepto bordes), garantizando que ninguna porción de la señal quede sin representación.

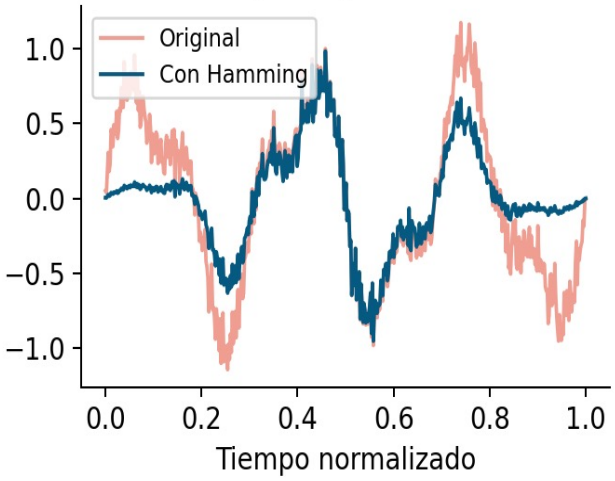
Ventana de Hamming: Ecuaciones y Efecto

$$w[n] = 0.54 - 0.46 \cdot \cos\left(\frac{2\pi n}{N-1}\right) \qquad \text{para } n = 0, 1, \ldots, N-1$$

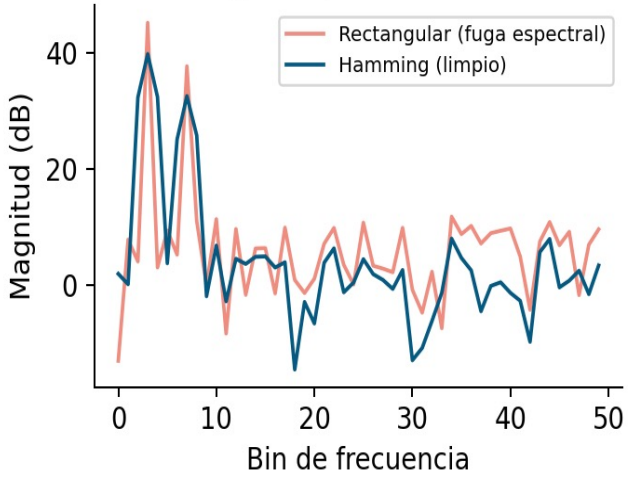
Forma de las ventanas



Señal antes / después de Hamming



FFT: fuga espectral reducida



¿Por qué Hamming y no otra ventana?

La Hamming tiene lóbulos laterales 43 dB debajo del principal (mejor que Hann: 31 dB). Su primer lóbulo lateral residual es lo suficientemente pequeño para el procesamiento de voz estándar (Vergin, 1995).

Fuga espectral (spectral leakage)

Sin ventana, los bordes abruptos de la señal truncada generan componentes de frecuencia espurias en la FFT. La Hamming suaviza los bordes → $w(0) \approx w(N-1) \approx 0.08$, eliminando la discontinuidad.

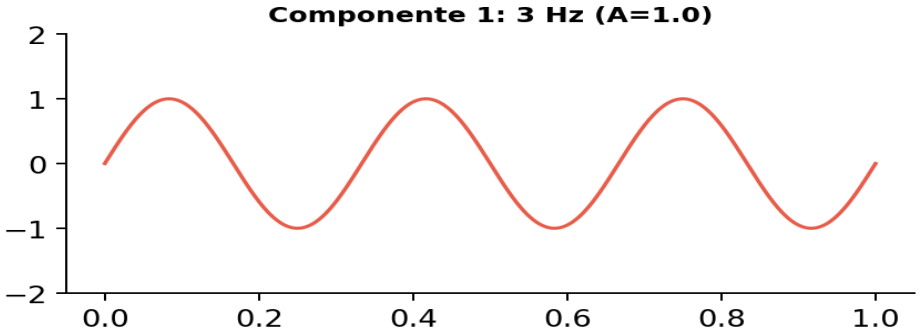
Paso 3: Transformada de Fourier Rápida (FFT)

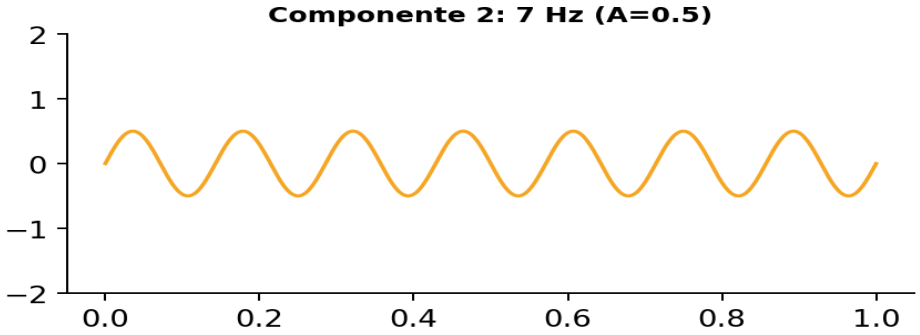
Transformada Discreta de Fourier (DFT)

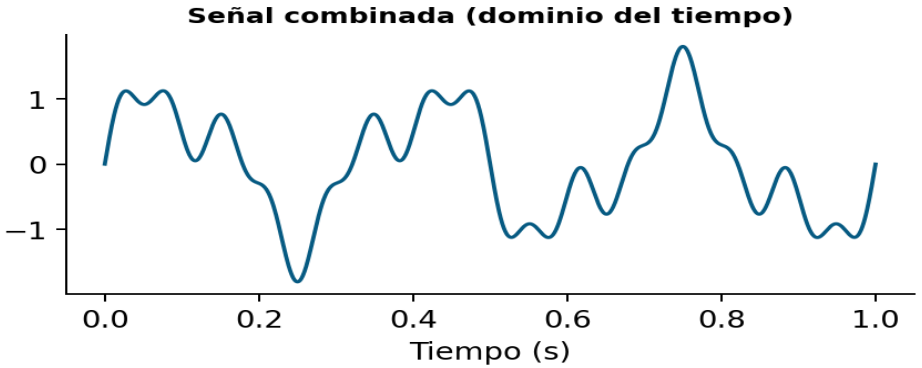
$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot e^{(-j2\pi kn/N)} \qquad k = 0, 1, \ldots, N-1$$

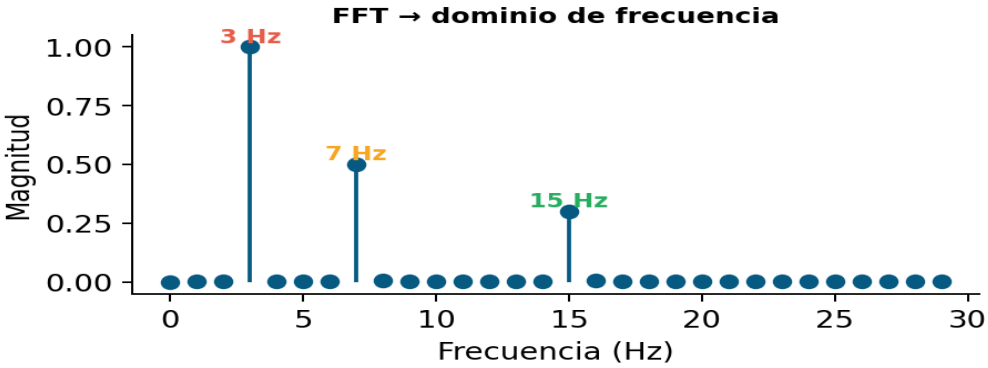
La FFT (Cooley-Tukey, 1965) calcula la DFT en O(N log N) vs O(N^2) operaciones.

Para N=512: DFT ≈ 262K ops, FFT ≈ 4.6K ops → 57× más rápido.









Del FFT al Espectrograma

Construcción del espectrograma

1. Aplicar la FFT a cada una de las 124 ventanas
2. De cada FFT de N=512 puntos, tomar solo N/2+1 = 257
   (por simetría conjugada: X[k] = X*[N−k])
3. Calcular la magnitud: |X[k]| = √(Re² + Im²)
4. Apilar las 124 columnas → Espectrograma 257 × 124

Cada columna: distribución de energía por frecuencia

Cada fila: evolución temporal de una frecuencia

El color indica la energía espectral (log scale → dB).

¿Por qué 257 y no 512?

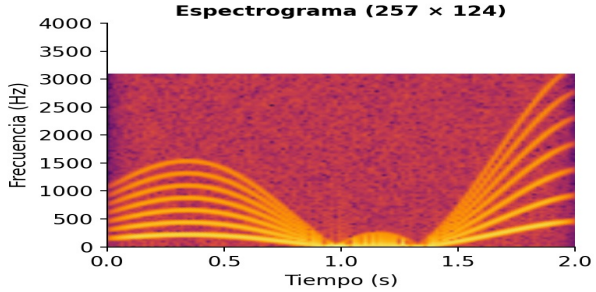
La FFT de una señal REAL tiene simetría conjugada: X[k] = X*[N−k].

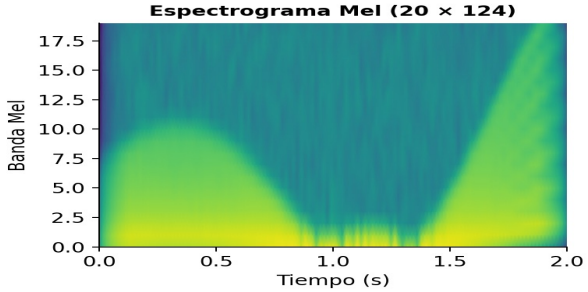
Solo la primera mitad (N/2+1 valores) contiene información independiente.

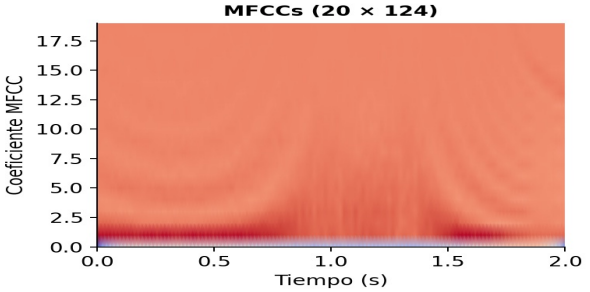
La segunda mitad es redundante. Esto significa: 512/2 + 1 = 257 bins.

Resolución frecuencial: Δf = fs / N = 16,000 / 512 = 31.25 Hz por bin

Bin 0 = 0 Hz (DC) | Bin 128 = 4,000 Hz | Bin 256 = 8,000 Hz (Nyquist)







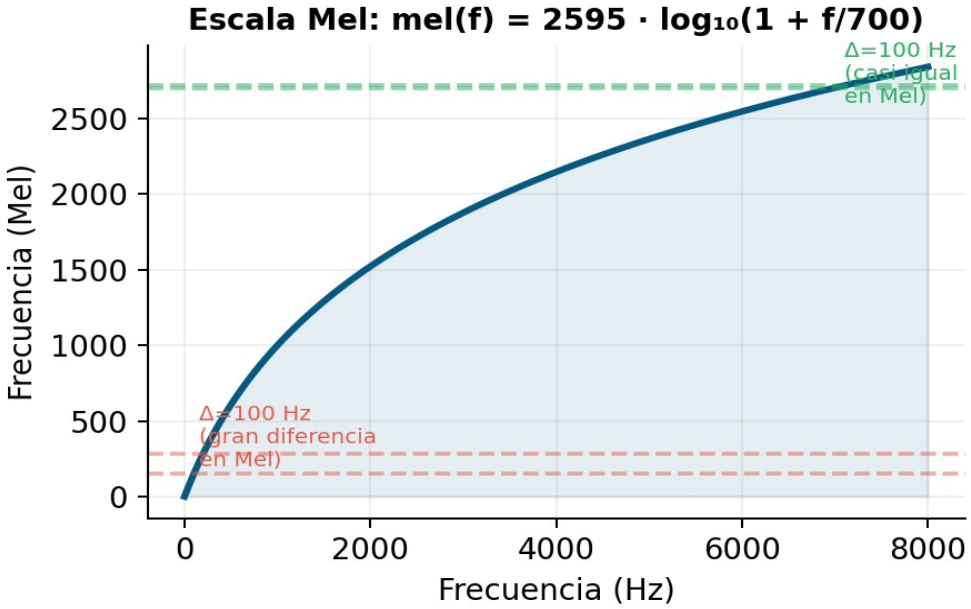
Paso 4a: Escala Mel — Percepción No Lineal

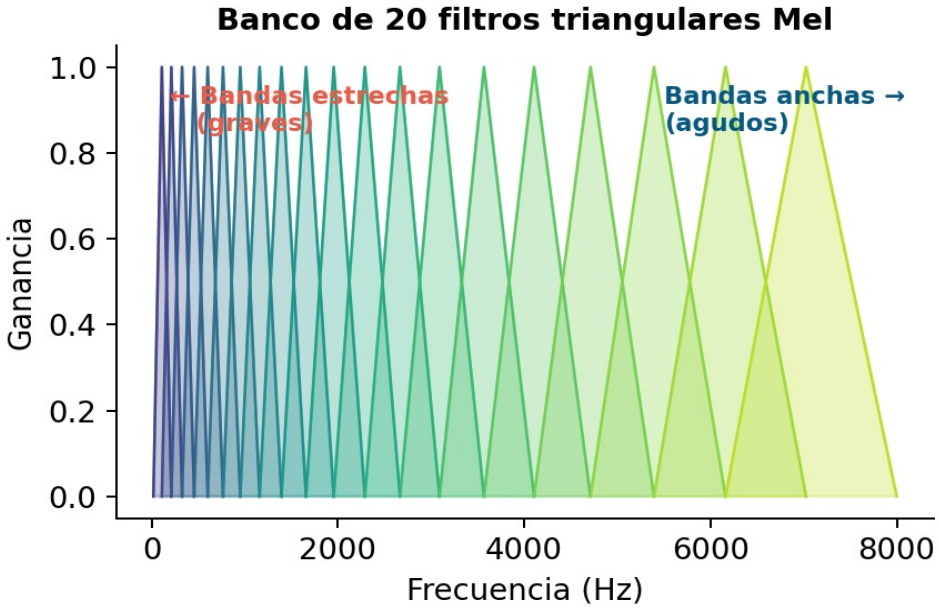
Hz → Mel: $m = 2595 \cdot \log_{10}(1 + f/700)$

Mel → Hz: $f = 700 \cdot (10^{(m/2595)} - 1)$

(O'Shaughnessy, 1987)

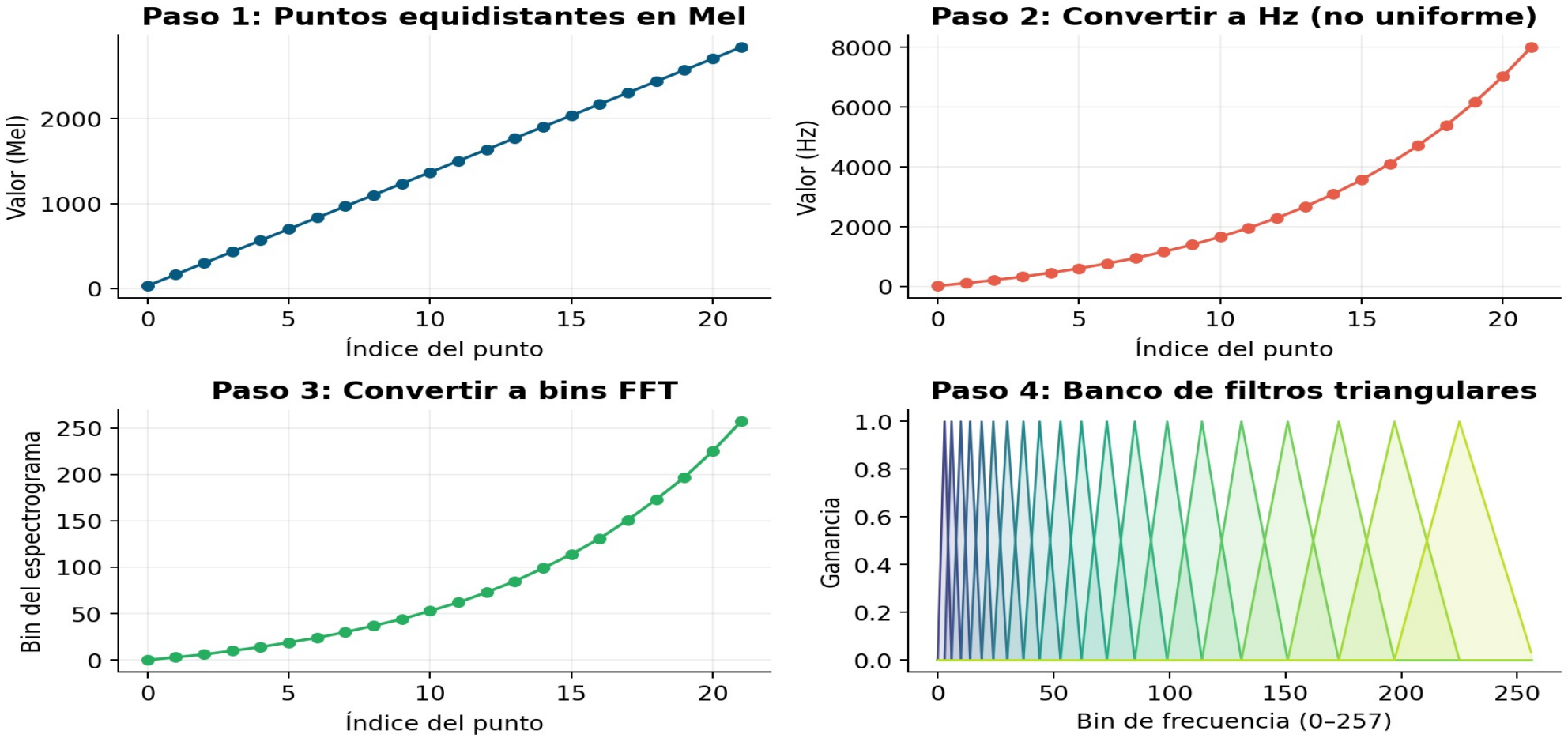
La escala Mel modela la percepción humana: distancias iguales en Mel se perciben como iguales en tono.





Clave biológica: la cóclea del oído interno tiene células ciliadas más densas en regiones de frecuencia baja, actuando como un banco de filtros no lineales. Los filtros Mel emulan esta distribución.

Paso 4b: Construcción del Banco de Filtros



Paso 4c: Filtros Triangulares — Ecuaciones

Ecuación del filtro triangular k

$$H_k(f) = \begin{cases} 0 & \text{si } f < f(k-1) \\ \dfrac{(f - f(k-1))}{(f(k) - f(k-1))} & \text{si } f(k-1) \le f \le f(k) \\ \dfrac{(f(k+1) - f)}{(f(k+1) - f(k))} & \text{si } f(k) < f \le f(k+1) \\ 0 & \text{si } f > f(k+1) \end{cases}$$

Cálculo paso a paso

1. Elegir fmin=20 Hz, fmax=fs/2=8000 Hz, M=20 bandas

2. Convertir a Mel:
   mel_min = $2595 \cdot \log_{10}(1+20/700) = 31.75$
   mel_max = $2595 \cdot \log_{10}(1+8000/700) = 2840.02$

3. Crear M+2 = 22 puntos equidistantes en Mel:
   Δmel = (2840.02−31.75)/21 = 133.73
   mel_points = [31.75, 165.48, 299.21, ...]

4. Convertir cada punto a Hz:
   hz_points = $700 \cdot (10^{(mel/2595)} - 1)$

5. Convertir Hz a bins del espectrograma:
   bin = $\lfloor (N\_FFT+1) \cdot hz / fs \rfloor$

6. Construir filtro triangular k con puntos:

Aplicación al espectrograma — multiplicación matricial

S_mel = H · S donde H ∈ ℝ^(20×257) es la matriz de filtros y S ∈ ℝ^(257×124) es el espectrograma

Resultado: S_mel ∈ ℝ^(20×124) — el espectrograma Mel. Cada fila representa la energía en una banda Mel a lo largo del tiempo.

Luego se aplica el logaritmo: S_mel_log = log(S_mel + ε) con $\varepsilon \approx 10^{-10}$ para evitar log(0)

Paso 5: Logaritmo + DCT → Coeficientes MFCC

DCT Tipo II (ortonormal)

$$C[k] = \sqrt{(2/M)} \cdot \sum_{m=0}^{M-1} \log(S\_mel[m]) \cdot \cos\left(\pi \cdot k \cdot (2m+1) / (2M) \right) \qquad k = 0, \ldots, M-1$$

¿Por qué el logaritmo?

1. **Percepción humana: el oído percibe la intensidad logarítmicamente (ley de Weber-Fechner). Los decibeles son una escala logarítmica: $dB = 10 \cdot \log_{10}(P/P_0)$**

2. **Compresión del rango dinámico:**
   Energía Mel varía de 10^{-3} a 10^{6}
   Después de log: de −6.9 a 13.8
   → Rango manejable para gradiente descendente

3. **Separación fuente-filtro: en escala log, la convolución del espectro glótico × respuesta del tracto vocal se convierte en suma → la DCT puede separarlos.**

¿Por qué la DCT?

1. Decorrelación: las bandas Mel adyacentes están naturalmente correlacionadas. La DCT produce coeficientes estadísticamente independientes.

2. Compactación de energía: los primeros coeficientes concentran la mayor parte de la información:
   - C0: energía total (promedio del log-espectro)
   - C1-C4: envolvente espectral gruesa (formantes)
   - C5-12: detalles finos
   - C13+: ruido (se pueden descartar)

3. Equivalente a PCA sobre el espectro Mel log, pero con bases fijas (cosenos) → sin necesidad de calcular eigenvectores.

Resultado: MFCCs ∈ ℝ^(20×124). Luego recorte central a 63 frames → MFCC final ∈ ℝ^(20×63) = 1,260 valores.

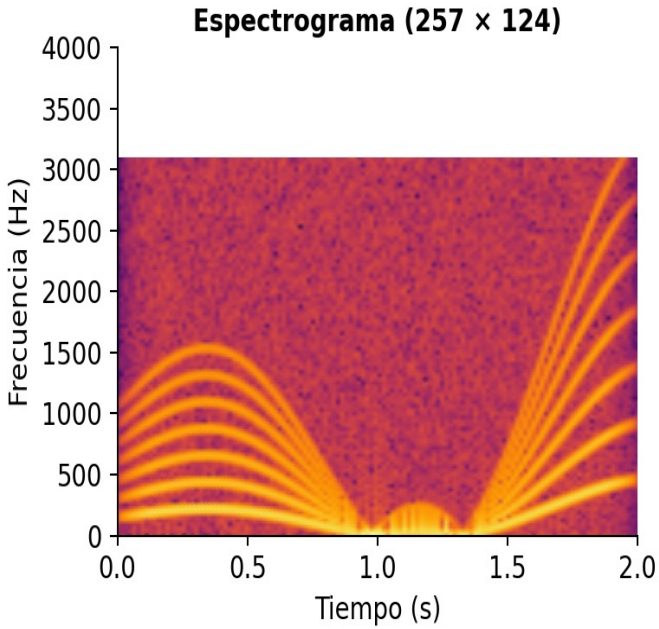
Pipeline MFCC Completo — Resumen con Ecuaciones

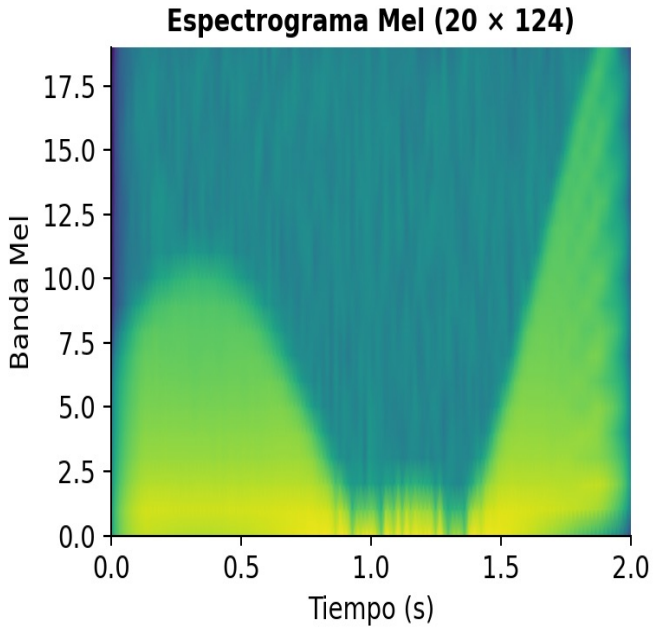
#	Paso	Ecuación / Operación	Propósito	Dim.
0	Señal cruda	$x[n],\ n=0..N-1,\ N=fs \cdot T$	Audio digital capturado	32,000
1	Pre-énfasis	$y[n] = x[n] - 0.97 \cdot x[n-1]$	Compensar sesgo glótico	32,000
2	Ventaneo	$xw[n] = y[n] \cdot w[n],\ w = \text{Hamming}$	Segmentación temporal	512×124
3	FFT	$X[k] = \Sigma\ xw[n] \cdot e^{(-j2\pi kn/N)}$	Tiempo → frecuencia	257×124
4a	Filtros Mel	$S_{mel} = H \cdot \lvert X\rvert^2,\ H \in \mathbb{R}^{(20\times257)}$	Escala perceptual	20×124
4b	Logaritmo	$L = \log(S_{mel} + \varepsilon)$	Compresión dinámica	20×124
5	DCT	$C[k] = \Sigma\ L[m] \cdot \cos(\pi k(2m+1)/2M)$	Decorrelación cepstral	20×124
6	Recorte	$C_{crop} = C[:, \text{center}-31:\text{center}+32]$	Eliminar silencios	20×63

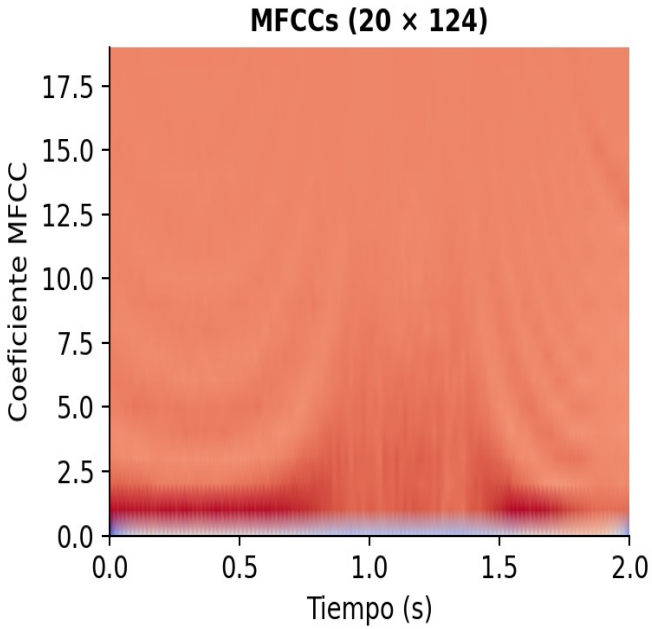
32,000 muestras → 20 × 63 = 1,260 coeficientes MFCC (compresión 25.4×)

Visualización: Espectrograma → Mel → MFCCs

Cada transformación comprime y reorganiza la información, preservando solo lo que el oído humano percibe como relevante para la clasificación emocional.







257 frecuencias lineales
(0 Hz → fs/2)
Resolución: 31.25 Hz/bin

20 bandas Mel (no lineales)
Más detalle en graves
Compresión 12.8×

20 coeficientes decorrelados
Primeros = forma general
Últimos = detalle fino

Preprocesamiento: Eliminación de Ruido

Ruido electrónico (DC offset)

El micrófono produce un offset DC en las muestras crudas. El pre-énfasis (paso 1) actúa como filtro paso alto que elimina esta componente continua automáticamente, ya que atenúa la frecuencia 0 Hz en ~30 dB.

Adicional: normalización de amplitud al rango [-1, 1] (Floating-Point PCM) para coincidir con el formato del ESD.

Ruido ambiental (no humano)

Fuentes: ventiladores, tráfico, electrodomésticos. Se aplica un filtro pasa-banda digital que deja pasar solo el rango de la voz humana (85–8,000 Hz), atenuando todo lo que queda fuera.

Los filtros Mel (paso 4) inherentemente atenúan frecuencias fuera del rango 20–8,000 Hz, proporcionando una segunda capa de filtrado.

Detección de Actividad de Voz (VAD)

No todo el audio contiene voz. Se calcula la energía RMS por ventana del espectrograma:

$$E[i] = 10 \cdot \log_{10}(\Sigma \, |X[k,i]|^2 / N) \quad [\text{en dB}]$$

Si $E[i+1] - E[i] \geq 5$ dB entre ventanas consecutivas → inicio de actividad vocal. Si no se detecta voz, el sistema no ejecuta inferencia, ahorrando energía y evitando falsos positivos.

Aumento de datos con ruido gaussiano

Se agrega ruido blanco de baja amplitud a cada audio de entrenamiento como data augmentation:

$$x_{aug}[n] = x[n] + N(\mu_x, \sigma_x/50)$$

Donde μ_x y σ_x son la media y desviación estándar de la señal original. Esto mejora la robustez de la red en entornos ruidosos reales sin distorsionar la señal significativamente.

Dataset: Emotional Speech Database (ESD)

Zhou et al. (2022) — Repositorio público de habla emocional con variabilidad léxica, lingüística y de hablantes.

Estadísticas del ESD

Total enunciados	**35,000**
Hablantes	**20 (10 inglés + 10 chino)**
Emociones	**5: Angry, Happy, Neutral, Sad, Surprise**
Enunciados/hablante/emoción	**350**
Frecuencia muestreo	**16 kHz (mono, WAV)**
SNR grabación	**> 20 dB (interior controlado)**
Duración promedio	**2.76 s (inglés), 3.22 s (chino)**
Edad hablantes	**25–35 años**
Distribución género	**50/50 (equilibrado)**

Splits oficiales (por hablante, por emoción)

Entrenamiento	300 enunciados	→ 3,000/emoción/idioma
Test (prueba)	30 enunciados	→ 300/emoción/idioma
Evaluación	20 enunciados	→ 200/emoción/idioma

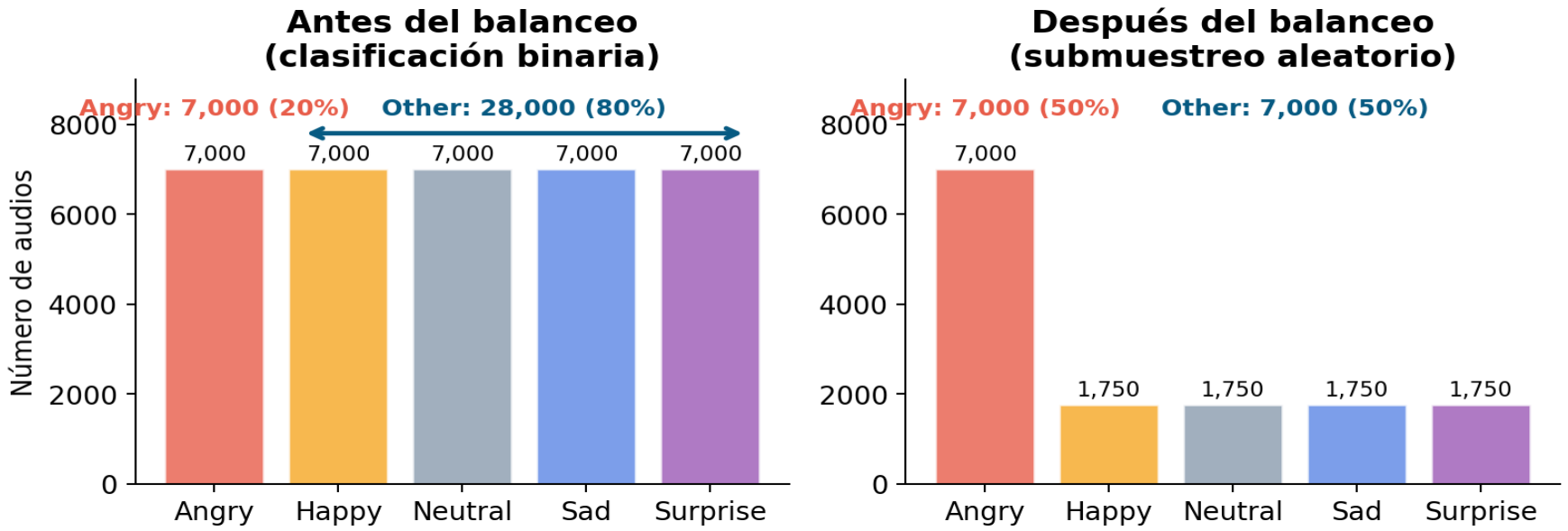
Controles de calidad del ESD

- Variabilidad léxica: 350 frases/emoción
- Variabilidad lingüística: inglés + chino
- Variabilidad de hablantes: 20 personas
- Control de confusión: sin risas, suspiros
- Entorno controlado: interior, SNR > 20 dB
- Validación: LSTM baseline 92%/89%

Tarea de clasificación binaria: Angry (1) vs. Other (0)

Las 4 emociones restantes (Happy, Neutral, Sad, Surprise) se agrupan como "Other". Esto simplifica la tarea pero crea un desbalance 1:4 que debe corregirse antes del entrenamiento.

Balanceo de Clases: El Problema del 80%



El accuracy paradox

Sin balancear: Angry=7,000 (20%) vs Other=28,000 (80%). Un modelo trivial que SIEMPRE predice "Other" obtiene 80% accuracy sin haber aprendido nada sobre el enojo. Este es el accuracy paradox en datasets desbalanceados.

Solución: submuestreo aleatorio (random undersampling). Se toman 7,000/4 = 1,750 muestras de cada emoción "Other", manteniendo equilibrio tanto entre clases como entre sub-clases. Se usa np.random.choice(n, size, replace=False) para selección sin reemplazo.

Resultado final balanceado: Train=12,000 (6K+6K) | Test=1,200 (600+600) | Eval=800 (400+400)

Balanceo de Clases: El Problema del 80%

El accuracy paradox

Sin balancear: Angry=7,000 (20%) vs Other=28,000 (80%). Un modelo trivial que SIEMPRE predice "Other" obtiene 80% accuracy sin haber aprendido nada sobre el enojo. Este es el accuracy paradox en datasets desbalanceados.

Solución: submuestreo aleatorio (random undersampling). Se toman 7,000/4 = 1,750 muestras de cada emoción "Other", manteniendo equilibrio tanto entre clases como entre sub-clases. Se usa np.random.choice(n, size, replace=False) para selección sin reemplazo.

Resultado final balanceado: Train=12,000 (6K+6K) | Test=1,200 (600+600) | Eval=800 (400+400)

Balanceo: Implementación y Carga de Datos

Dimensiones finales después de balanceo y cálculo de MFCCs

	Entrenamiento	Prueba (Test)	Evaluación
Audio crudo	(12,000 × 32,000)	(1,200 × 32,000)	(800 × 32,000)
Etiquetas	(12,000,)	(1,200,)	(800,)
MFCCs	(12,000 × 20 × 63)	(1,200 × 20 × 63)	(800 × 20 × 63)

Flujo de Datos Completo: del ESD al Entrenamiento

1	**Cargar archivos WAV**	30 carpetas (5 emociones × 2 idiomas × 3 splits) librosa.load(path, sr=16000, duration=2)
2	**Etiquetar (binario)**	Angry → label = 1 Happy, Neutral, Sad, Surprise → label = 0
3	**Balancear clases**	Submuestreo aleatorio: 1,750 muestras por emoción "Other" Resultado: 50/50 Angry vs Other
4	**Agregar ruido**	Ruido gaussiano N(μ, σ/50) + zero-padding a 32,000 Data augmentation implícito
5	**Concatenar y mezclar**	np.concatenate → vector único por split Shuffle para romper orden secuencial
6	**Calcular MFCCs**	Aplicar pipeline completo (7 pasos) a cada audio Forma final: (N_audios, 20, 63)
7	**Alimentar la CNN**	Input: MFCC (20×63×1) + Label (0/1) Entrenamiento con binary_crossentropy

Números Clave del Pipeline

16 kHz Frecuencia de muestreo	**32,000** Muestras por audio (2 s)	**512** Muestras por ventana	**124** Ventanas por audio
257 Bins FFT por ventana	**20** Filtros Mel triangulares	**63** Frames después del recorte	**1,260** Coeficientes MFCC totales

Resumen

El pipeline MFCC transforma una señal de voz cruda de 32,000 muestras en una representación compacta de 1,260 coeficientes que captura las propiedades perceptuales relevantes del habla humana.

Cada paso tiene una justificación precisa:

- Pre-énfasis → Compensar el sesgo espectral glótico (caída de 6 dB/octava)
- Ventaneo → Asumir cuasi-estacionariedad en segmentos de 32 ms
- FFT → Pasar del dominio temporal al espectral (O(N log N))
- Filtros Mel → Emular la resolución no lineal de la cóclea humana
- Logaritmo → Modelar la percepción logarítmica de la intensidad
- DCT → Decorrelacionar y compactar la información espectral

El preprocesamiento (filtrado de ruido, VAD, balanceo de clases, augmentación con ruido gaussiano) garantiza que la CNN reciba datos limpios, representativos y equilibrados.

Esta representación es la entrada directa a la red neuronal convolucional para la detección de enojo en tiempo real.