Fine Tuning FastPitch TTS a Español

Matías Di Bernardo - 2023/2024

Informe técnico
Intercambios Transorgánicos

Resumen

Este documento describe el proceso de fine-tuning del modelo Fast Pitch multispeaker en español, basado en las experiencias registradas durante varias iteraciones de ensayo y error. Se detallan la configuración del entorno, la preparación de datos, las adaptaciones realizadas al modelo, los principales obstáculos encontrados, las soluciones aplicadas y los resultados obtenidos en cuatro pruebas distintas.

---

1. Introducción

El objetivo principal de este proyecto fue adaptar y afinar un modelo de síntesis de voz Fast Pitch originalmente entrenado en inglés para que funcionase de manera óptima con datos en español. Se eligió la versión multispeaker disponible en NVIDIA NeMo, dada su capacidad de incorporar múltiples voces en un único modelo y su robustez potencial frente a variaciones de hablantes. Luego de varias iteraciones se logra ejecutar el algoritmo y se realizan diferentes pruebas para evaluar el funcionamiento. En el documento se detallan los avances, dificultades técnicas y decisiones estratégicas.

---

2. Estructura de Directorios y Configuración Inicial

La implementación del proyecto es en Google Colab por lo tanto se tienen que adaptar las carpetas para que funcione en ese entorno.

1. **Estructura objetivo**
    - Repositorio principal con subcarpetas:
        - `conf/` (archivos YAML de configuración en español)

        - `tts_data_file/` (scripts de procesado de texto/plain text)
        - `fastpitch_sup_data/` (cálculo automático de estadísticas de espectrogramas)
        - `checkpoints/{modelo}/{fecha}/` (almacenamiento de pesos de entrenamiento)

2. **Adaptaciones en Colab**
    - Uso de checkpoints precargados en español de NeMo
    - Montaje de Google Drive para persistencia de checkpoints
    - Reemplazo de configuraciones en inglés por sus equivalentes en `conf/es/`

---

3. Preparación del Conjunto de Datos

Los datos tiene que seguir un formato específico para que el pipeline funcione:

- **Origen de datos**: Archivos comprimidos (`.rar`) con carpeta `audios/` (WAVs) y metadatos en JSON. El metadao en JSON contiene el nombre y correspondiente transcripción de los audios.
- **Formato de metadatos**: Cada entrada JSON incluyó campos estándar (`audio_filepath`, `duration`, `text`) y se añadió manualmente la clave (`"speaker": 0`) o el número de speaker según corresponda.
- **Particionado**: División en subconjuntos `train` y `val` mediante scripts propios, con copia local de la carpeta `audios/`.
- **Normalización de texto**: Se validó la compatibilidad con inputs en texto directo; la fonemización se dejó como mejora futura.

---

4. Configuración del Modelo y Adaptaciones Específicas

Los siguientes ajustes son específicos para el modelo en español. El código se puede encontrar en el siguiente link.

1. **Adaptación de YAML**
    - Se empleó el archivo `fastpitch_align_44100_ipa_multi.yaml` como plantilla, modificando rutas y parámetros de frecuencia de muestreo.
    - Se ajustó el número de **speakers** a 174, acorde al modelo multispeaker en español.
2. **Diccionario de fonemas**
    - Se constató incompatibilidad dimensional entre el diccionario preexistente y el requerido para fine-tuning.
    - Solución provisional: recorte manual del diccionario para que coincidieran dimensiones de embedding.
3. **Pitch statistics**
    - Descarga y uso de `pitch_stats.json` para proporcionar valores de media y desviación estándar de pitch.
    - Se exploró la automatización de este cálculo, sin éxito definitivo.

---

5. Procedimiento de Fine-Tuning

1. **Inicialización**
    - Carga de checkpoint base en español multispeaker.
    - Verificación de compatibilidad de archivos YAML y diccionario de fonemas.
2. **Entrenamiento**
    - Scheduler de aprendizaje con tasa inicial (`LR`) variable según prueba.
    - Cada experimento configuró manualmente los parámetros de número de epochs y tamaño de batch.
3. **Persistencia**

- Guardado automático de checkpoints intermedios y del mejor modelo en Drive.

---

6. Ajuste de Métricas de Pitch

Se ejecutó un script en Colab para calcular la **media** y **desviación estándar** del pitch sobre el conjunto de entrenamiento. Aunque genera automáticamente `pitch_stats.json`, la inyección de estas variables en el pipeline requirió intervención manual debido a limitaciones en la transmisión de variables entre celdas de consola y configuraciones internas de NeMo.

---

7. Problemas Principales y Soluciones

Problema	Solución aplicada
Falta de modelo single-speaker en español precargado	Intento de fine-tuning sobre modelo multispeaker; evaluación de entrenamiento desde cero como opción.
Incompatibilidad entre diccionarios de fonemas	Recorte manual del diccionario y ajuste dimensional de embeddings.
Error por mezcla de frecuencias de muestreo (22050 vs. 44100 Hz)	Mantener frecuencia uniforme en todo el pipeline.
Limitaciones en la automatización de pitch mean/std	Ajuste manual de `pitch_stats.json` y propuesta de revisión futura para script paramétrico.

---

8. Resultados de las Pruebas

Se llevaron a cabo cuatro experimentos con distintas configuraciones de datos y procesamiento:

8.1. Primera Prueba

- **Dataset**: `MatíasVer3.zip` (23,92 min)

- **Preprocesado**: sin denoise
- **LR**: 2e-4
- **Duración de entrenamiento**: 70 min
- **Best Val Loss**: 1.21 (epoch 300)
- **Observaciones**: La dicción apresurada del hablante influyó negativamente en la calidad percibida.

8.2. Segunda Prueba

- **Dataset**: `MatíasVer4.zip` (7,81 min)
- **Preprocesado**: sin denoise
- **LR**: 2e-4
- **Duración de entrenamiento**: 37 min
- **Best Val Loss**: 1.59 (epoch 500)
- **Observaciones**: Ruido de “click” en grabaciones; su persistencia afecta inferencia.

8.3. Tercera Prueba

- **Dataset**: `MatíasVer3-Denoised.zip` (23,92 min)
- **Preprocesado**: con denoise
- **LR**: 2e-4
- **Duración de entrenamiento**: 110 min
- **Best Val Loss**: 1.44 (epoch 250)
- **Observaciones**: Eliminación de clicks sacrificó calidad de la voz; se recomienda explorar otros valores de LR y formatos de input.

8.4. Cuarta Prueba

- **Dataset**: `crowdsource_arg_dataset.zip` (145,13 min)

- **Preprocesado**: sin denoise
- **LR**: 2e-5
- **Duración de entrenamiento**: 80 min
- **Best Val Loss**: 1.58 (epoch 50)
- **Observaciones**: Pese a la elevada cantidad de datos, la validación decrece rápidamente; se sugiere revisar complejidad del modelo o calidad de anotaciones.

En la siguiente carpeta se encuentran los resultados (demos en wav) y los pesos del modelo para poder cargarlos en el colab de inferencia y generar nuevos samples.

9. Código

El resultado de este proceso son 2 Google Colab, uno en el cual se puede entrenar nuevas voces en español, y otro para poder utilizar el modelo (inferencia) con los checkpoints generados en el primer colab.

9.1. Colab Training

Link colab training

9.2. Colab Inferencia

Link colab inferencia

---

10. Discusión

Los resultados indican que la calidad y dicción del hablante influyen decisivamente en las métricas de validación, más allá de la simple cantidad de datos. Las pruebas con dataset limpio (denoise) no mejoraron consistentemente la val loss, lo que sugiere que la normalización de audio debe complementarse con ajustes de aprendizaje y arquitectura. Asimismo, el uso de un modelo multispeaker presenta ventajas de robustez, aunque complica la extensión a nuevos hablantes sin un single-speaker base.

---

11. Conclusiones y Trabajo Futuro

- **Conclusiones**:

    1. El fine-tuning de un modelo multispeaker en español es factible tras adaptar diccionarios de fonemas y métricas de pitch.

    2. La calidad de grabación y dicción del dataset original es crítica para obtener síntesis de alta fidelidad.

    3. La frecuencia de muestreo uniforme y la consistencia de configuraciones YAML aseguran estabilidad durante el entrenamiento.

- **Líneas de trabajo futuro**:

    1. Automatizar completamente el cálculo e inyección de `pitch_stats.json`.

    2. Explorar modelos single-speaker en español, ya sea transfer learning desde inglés o desde cero con dataset argentino.

    3. Evaluar métodos alternativos de vocoder (Griffin-Lim, HiFi-GAN entrenado en español multispeaker).

    4. Investigar la posibilidad de agilizar el proceso de entrenamiento y si se podría implementar un modelo Zero-Shot en base a la arquitectura transformer.

Referencias

- Documentación interna y registros de actualizaciones detallados. Link documento
- Referencia de estado del arte de modelos TTS. Link paper
- Clase sobre el modelo FastPitch y la arquitectura transformer. Link video