Revista de investigación de sistemas e informática
14(1): 83-88 (2021)
DOI: https://doi.org/10.15381/risi.v14i1.21864

ISSN: 1815-0268 (Impreso) / ISSN: 1816-3823 (Electrónico)
Facultad de Ingeniería de Sistemas e Informática - UNMSM

STT: Un sistema de apoyo a la transcripción de audiencias fiscales usando Vosk

STT: A support system for transcribing tax hearings using Vosk

Diego Vera
https://orcid.org/0000-0003-3246-6836
diegovera@unmsm.edu.pe
Ministerio Público, Fiscalía de la Nacion, Lima, Perú

Ángel Espezua
aespezua@mpfn.gob.pe
Ministerio Público, Fiscalía de la Nacion, Lima, Perú

Recibido: 02/08/2021 - Aceptado: 24/10/2021 - Publicado: 28/12/2021

RESUMEN

Las audiencias son de suma importancia dentro del sistema penal peruano y la información que es tratada aquí es importante para la resolución de un caso. Muchas veces se requiere de esta información a corto plazo, pero la transcripción manual de estas audiencias puede llevar bastante tiempo debido a que estas son de muchas horas de duración. En la actualidad existen modelos de transcripción pre entrenados que pueden realizar este trabajo en unos minutos, con esto ahorrar mucho tiempo y tener la información requerida casi al instante, pero no están implementados en un sistema de libre uso ni están disponibles en español. Se implementó un sistema transcriptor de audio a texto mediante metodología SCRUM donde semanalmente se desarrolló y probó funcionalidades nuevas, los modelos de transcripción que se implementó son Vosk, Speech to text de Google y DeepSpeech. En los resultados experimentales, este último tuvo resultados negativos en transcripción, mientras que los mejores resultados los tuvo Speech to text de Google, pero el más rápido de estos fue Vosk. Finalmente se escogió Vosk como modelo transcriptor del sistema debido a su rapidez y eficiencia en la transcripción.

Palabras clave: Voz a texto; reconocimiento de voz; aplicación de software; Ley gubernamental; Fiscalía.

ABSTRACT

Hearings are of utmost importance within the Peruvian criminal system and the information that is dealt with here is important for the resolution of a case. This information is often required in the short term, but manual transcription of these hearings can be time consuming as they last many hours. Currently there are pre-trained transcription models that can perform this work in a few minutes, thereby saving a lot of time and having the required information almost instantly, but they are not implemented in a free-to-use system nor are they available in Spanish. An audio-to-text transcription system was implemented using SCRUM methodology where new functionalities were developed and tested weekly, the transcription models that were implemented are Vosk, Google's Speech to text and DeepSpeech. In the experimental results, the latter had negative results in transcription, while the best results were Google's Speech to text, but the fastest of these was Vosk. Finally, Vosk was chosen as the transcriptional model of the system due to its speed and efficiency in transcription.

Keywords: Speech to text; speech recognition; application software; Government law; Prosecutor's office.

I. INTRODUCCIÓN

En el sistema penal peruano las audiencias constituyen la etapa de preparación y realización del juicio oral dentro de un caso penal, estas finalizan el caso con una sentencia (Menacho, 2017). En estas participan dos posiciones contrarias que debaten con pruebas para convencer al juzgador en la culpabilidad o inocencia del acusado. La información compartida en la reunión es de suma importancia, la transcripción (pasar el audio a texto) de audiencias fiscales es una tarea crucial en el desarrollo de un caso fiscal pero también muy laboriosa y demandante de tiempo debido a que se hace de manera manual con ayuda de unos asistentes de un área de audio y video dentro de la fiscalía y a que estas audiencias suelen demorar en promedio 1 hora (Belan, 2014). Teniendo en cuenta que la información de una audiencia es crítica para un caso, es necesario automatizar este proceso de transcripción para obtener resultados más rápidos a comparación de un proceso manual.

En la actualidad ya existen sistemas de transcripción de audio a texto usados ampliamente en distintas aplicaciones de propósito general (Reddy, 2013), (Shakhovska, 2019), (Sari, 2020), en específico, en audiencias penales, se ha desarrollado modelos de transcripción (Prasad, 2002) en donde toman en cuenta el ruido dentro de la sala de audiencia y la reverberación ocasionada por los múltiples micrófonos en esta sala para el desarrollo de su modelo, la desventaja de estos es que no están enfocados en el idioma español, pero sí existen modelos pre entrenados de inteligencia artificial (Kumar, 2018) que realizan esta tarea de transcripción logrando un buen performance. Los modelos seleccionados fueron Vosk (Shmyrev, 2021), Speech to text de Google (Zhang, 2017) y DeepSpeech (Hannun, 2014). En este trabajo se presenta un sistema de audio a texto de audiencias fiscales usando un modelo de transcripción, este sistema sirve como herramienta de apoyo a los encargados de audio y video en el proceso de transcripción de una audiencia.

El presente artículo se divide en las siguientes secciones: introducción donde se indica el problema y solución, materiales y métodos donde se explican tecnologías y algoritmos utilizados para la solución, el aporte donde se explica cómo funciona la solución y su proceso para la transcripción y finalmente las conclusiones.

II. OBJETIVOS

- Implementar un modelo transcriptor que reconozca el idioma español.
- Diseñar e implementar un sistema de transcripción de audio a texto para audiencias fiscales, las cuales son de larga duración, en español y muy costosas en almacenamiento.

III. FUNDAMENTACIÓN TEÓRICA

- Resumable uploads: En español llamada carga reanudable, es un algoritmo que permite enviar archivos pesados a través de varias peticiones HTTP, estas peticiones envían un fragmento del audio (audio chunk), puede ser de 1MB por petición, estos fragmentos son almacenados en el servidor y poco a poco se van uniendo para consolidar el archivo completo. Todas las peticiones, exceptuando el último, retorna un código de estatus 206, como señal que el fragmento del archivo fue almacenado correctamente, si algo sale mal, te retorna un código de status 500 mostrando el error sucedido.
- Async task (Celery): Tareas asíncronas en español, celery es una cola de tareas (Celery, 2021), su función es almacenar las tareas que le son asignadas y las va realizando una a una. Esta cola de tareas es necesaria debido a que las peticiones HTTP solo tienen un tiempo límite de 1 minuto como procesamiento, pero el procesamiento de las transcripciones en los audios puede llegar a tomar varios minutos, dependiendo del algoritmo que se use. Como beneficio adicional del uso de la cola de tareas, el procesamiento de la transcripción es ejecutada en segundo plano mientras se puede usar otras APIs.

  Chunks: Son fragmentos de un audio, estas se dividen en fragmentos de un tiempo específico establecido, en este caso se usó tiempo de 3 minutos por fragmento. En el proceso son usados los fragmentos de audios de 3 minutos debido a la carga computacional que ejerce el proceso de transcripción en la memoria RAM, con audios más pesados es mucho más grande el espacio requerido en RAM.
- **Transcripción:** La transcripción es la acción de transformar audio a texto, en el proceso construído para el sistema se usó el modelo de transcripción Vosk que también otorga

una configuración del modelo para ver cuánto es la confianza por palabra, con el objetivo de conocer cuáles son las palabras que posiblemente están erradas. Los sistemas de transcripción se pueden clasificar de 2 formas:

- Online: Es cuando el servicio de transcripción se encuentra alojado en la nube y para poder acceder a este es necesario utilizar una API pública o privada bajo algún costo que te otorgue una key para poder usarla, como es el caso de Google Speech To Text (Cloud Speech-To-Text, 2021), IBM Watson (IBM Watson, 2021) o Amazon Transcribe (Amazon Transcribe, 2021).
- Offline: Es cuando se posee el archivo del modelo pre entrenado junto al código fuente que lo ejecuta y quien es consultado a través de un cliente ya sea escritorio, web local o incluso el propio terminal. Ejemplos: Vosk (Shmyrev, 2021) o DeepSpeech (Hannun, 2014).

Algoritmos de transcripción:

- Vosk (Shmyrev, 2021): Es un modelo es soportado por distintos lenguajes como java, javascript, python, C++, entre otros, es bueno por su soporte en el lenguaje español y por su peso ligero.
- Speech to text de Google: Es el motor de transcripción ofrecido por Google, su uso tiene costo por minuto de transcripción, pero la librería Speech recognition de python (Zhang, 2017) facilita el uso de este motor de manera gratuita. Este motor de transcripción es el más exacto de los usados, pero dentro de su documentación dice que el motor de Google solo debe ser usado de manera de prueba, debido a que en cualquier momento Google puede retirar la API.
- DeepSpeech: Es un motor open source desarrollado por Mozilla, utiliza como algoritmo Recurrent neural network (RNN) (Hannun, 2014). Una ventaja de este motor es que puedes entrenarlo con tu propio dataset para hacer un modelo de transcripción.

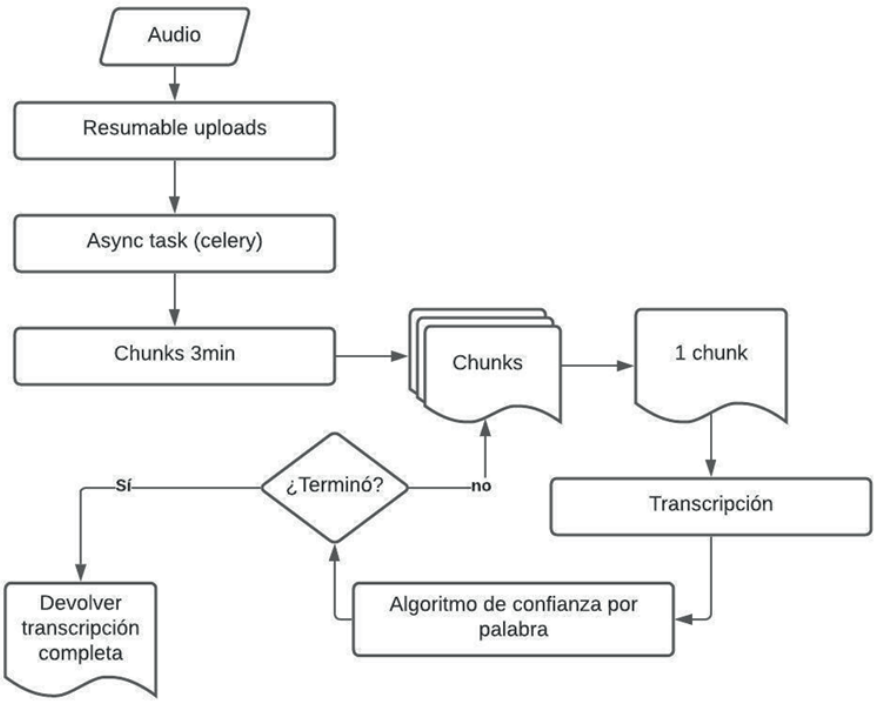
IV. APORTE

El proceso de transcripción propuesto, mostrado en la figura 1, realiza los siguientes pasos:

1. Inicia cuando se ingresa el audio desde cliente, en este caso la interfaz web desde un browser.

Figura 1

Proceso de transcripción de un audio. Fuente: Propio de los autores



2. Para guardar el audio se utiliza de resumable uploads, el audio es dividido en fragmentos de 1MB y con cada uno de estos fragmentos se hace una petición para poder ser almacenados en el servidor. El primer fragmento de 1MB es almacenado, los siguientes solo (a excepción del último) son añadidos en el primer fragmento mediante manipulación por bytes y poco a poco se van acumulando de fragmento en fragmento hasta llegar a almacenar el archivo entero.

3. Una vez el audio esté guardado en el servidor, se crea una tarea en celery (cola de tareas) a través de RabbitMQ como message broker, esto hace que el proceso de transcripción se encole y pueda ejecutarse en segundo plano de manera asíncrona.

4. Una vez el proceso es iniciado en Celery, el audio se divide en fragmentos de 3 minutos.

5. Cada uno de estos chunks se transformó en un formato aceptable para ser usados en el proceso de transcripción con el modelo Vosk, también se configuró el modelo para que otorgue la confianza por palabra y conocer cuáles son las palabras que es más probable que estén erradas.

6. Si termina el proceso de transcripción para un chunk, se pasa al siguiente chunk para que pueda transcribirse, en caso de que ya no haya más chunks para transcribir entonces ya se posee la transcripción completa y se transforma las palabras que posiblemente estén erradas de tal manera que sirvan de ayuda a los usuarios para reconocer qué palabras podría el sistema haber fallado en transcribir, luego de esto la transcripción se almacena en la base de datos.

a. Arquitectura

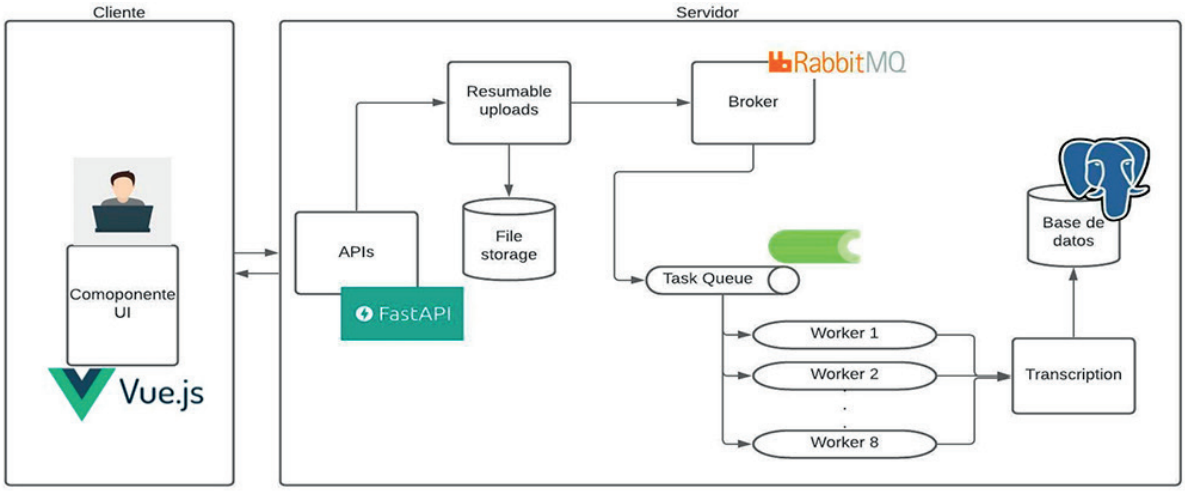
Se siguió una arquitectura cliente servidor donde en la capa del cliente se usó como framework frontend VueJS y en la capa servidor se usó de backend el framework FastAPI, RabbitMQ como message broker, Celery como cola de tareas y PostgreSQL en la base de datos, mostrado en la figura 2.

b. Componentes

- Componente UI: Es la interfaz de usuario desarrollada con el framework VueJS, hecha para la interacción con el usuario y consumir las APIs desarrolladas en backend.
- APIs: Servicios REST desarrollados con el framework FastAPI, aquí se encuentran todas las funcionalidades que el sistema posee.
- Resumable uploads: Explicado en la sección II.
- File storage: Espacio de almacenamiento del servidor otorgado para guardar los archivos de audio.
- Broker: Es un message broker, encargado de recibir las peticiones que le envíen, en este caso es notificado cuando termine de guardar un audio en el local storage, este

Figura 2

Arquitectura del sistema transcriptor de audio a texto. Fuente: Propio de los autores



envía toda la información necesaria a la cola de tareas.

- Task Queue: Es la cola de tareas, se usa celery como tecnología, almacena todas las peticiones enviadas por el broker y le asigna la tarea a un worker que esté disponible.
- Worker: Perteneciente al mismo celery, es encargado de ejecutar las tareas que se le asignen, en este caso es quien se encarga de usar el motor de transcripción para el audio.
- Transcription: Aquí se encuentra almacenado el motor de transcripción junto a las transformaciones de audio que son necesarias para poder utilizarlas en el motor.
- Base de datos: Almacena toda la información de usuarios en el sistema, se usó PostgreSQL.

c. Prueba con los distintos motores de transcripción

Se probó con distintos audios por cada motor transcriptor con el fin de conocer cuál será usado en el sistema, la tabla 1 muestra las pruebas con un audio en distintos formatos de frecuencia y con cada motor de transcripción considerado. Se puede notar que el motor transcriptor más exacto es el que otorga Speech to text de Google, seguido por Vosk que puede reconocer más palabras a comparación del anterior con la desventaja que es menos exacto en su transcripción, estos 2 motores funcionan bien en audios wav con frecuencias de 44100Hz y 16000Hz. Por otro lado el motor de DeepSpeech solo funciona con audios de frecuencia 44100Hz, pero aún así no es muy exacto con estos, siendo el motor menos exacto para la transcripción de los 3 que se posee.

V. CONCLUSIONES

Como se pudo notar en la sección III, se probaron 3 modelos de transcripción y se escogió el modelo Vosk debido a su confianza de transcripción y a su rapidez en entregar el resultado de transcripción, los resultados de transcripción más confiable son los que ofrece Speech to text de Google, pero tiene la desventaja que Google puede dar de baja su servicio en cualquier momento, por eso es recomendable solo usarlo de manera de prueba. Por esta razón fue elegido el modelo de transcripción Vosk como el principal y optativamente dentro de la API de transcripción construida se puede usar el de Google.

Los motores de transcripción no podían dar resultados eficaces sobre cualquier tipo de audio, hay algunos que no pueden ser procesados debido a que presentan ruido, el volumen de voz no es el adecuado o que aparecen sonidos que causan que lo que se está hablando no se entienda, como por ejemplo cuando dos personas están hablando al mismo tiempo. Estos problemas pueden llegar a ser solucionados con trabajos de investigación futuros, como reducción de ruido en un audio y speaker diarization.

Tabla 1

Resultados de transcripción de los distintos motores.

Motores de transcripción	Formato de audio	Transcripción
Audio real		Ya este es otro audio fabiano quiere ver la película de kimetsu no yaiba está buscando la película que tal es la película en el manga es muy buena y triste a la vez o sea es muy buena ya entonces es muy buena ya dime entonces qué vas a hacer hoy día nada como siempre y fingir prestar atención a clase, ya entonces no está prestando atención a clase las clases hasta que hora duran, hasta la 1:20 creo sí a la 1:20 y comienza a las 8:30 de lunes a sábado ya de lunes a sábado desde las 8:30 tiene que levantarse temprano ya entonces hoy día que vas a comer
Speech to text de Google	44100Hz y 16000Hz	Este es otro audio fabiano quiere ver la película de kimetsu no yaiba está buscando la película que tal es la película en el manga es muy buena y triste a la vez o sea es muy buena ya entonces es muy buena Ya dime entonces qué vas a hacer hoy día Ya Entonces no está prestando atención a la clase
Vosk	44100 y 16000Hz	este es otro audio fabián no quiere ver la película que me iba está buscando la película que tal la película en el manga es muy buena entrevista a la vez o sea es muy buena entonces es muy buena ya de inventos es que vas a hacer hoy día diciembre y ya entonces no está prestando atención a la clase las clases hasta que hora duran hasta uni veinte creo sea la uni veinte comience a las ocho y media es necesario de lunes a sábado a las ocho y media tiene que levantarse temprano entonces hoy día que vas a comer
DeepSpeech	44100Hz	y este es otro odio había no quiere ver la película del que mesonera está buscando la película que tal es la película en el mundo es muy buena y triste la vez buena entonces es muy bueno y dementes es qué vas a servidores
	16000Hz	no oestenoroeste

Fuente: Propio de los autores

VI. AGRADECIMIENTOS

Agradezco al equipo de la bandeja fiscal, Ministerio Público, por el apoyo en el trabajo, en especial al equipo de machine learning, Diana Quintanilla y Arthur Mauricio, por apoyarme en el desarrollo de este trabajo y a Jimmy Espezua por proporcionar las medidas necesarias para la presentación del artículo.

VII. REFERENCIAS

[1] H. Menacho, “AUDIENCIA DE CONTROL DE OFICIO DE LA PRISIÓN PREVENTIVA COMO HERRAMIENTA PARA EL CUMPLIMIENTO DE LAS GARANTÍAS DEL NUEVO CÓDIGO PROCESAL PENAL PERUANO”, (tesis de título de grado), Facultad de Derecho y ciencias políticas, UNIVERSIDAD NACIONAL DE ANCASH “SANTIAGO ANTÚNEZ DE MAYOLO”, Ancash, 2017, http://repositorio.unasam.edu.pe/handle/UNASAM/1830

[2] C. Belan, “CORRELACIÓN ENTRE LA VULNERACIÓN DE LOS PRINCIPIOS DE PUBLICIDAD Y CELERIDAD EN EL JUICIO PENAL Y LA PERCEPCIÓN CIUDADANA EN SU REFERENCIA. AREQUIPA, 2011.”, (tesis de maestría), ESCUELA DE POSTGRADO MAESTRÍA EN DERECHO PENAL, UNIVERSIDAD CATÓLICA DE SANTA MARÍA, AREQUIPA, 2014, https://core.ac.uk/download/pdf/198133899.pdf

[3] B. R. Reddy, E. Mahender, “Speech to Text Conversion using Android Platform”, *International Journal of Engineering Research and Applications (IJERA)*, vol. 3, nro. 1, pp. 253-258, Feb. 2013, http://ijera.com/papers/Vol3_issue1/AJ31253258.pdf

[4] R. Prasad, L. Nguyen, R. Schwartz, J. Makhoul, “MeetingLogger: Rich Transcription of Courtroom Speech”, en *Proceedings of HLT2002, Second International Conference on Human Language Technology Research*, San Francisco, 2002, pp. 303-306. https://dl.acm.org/doi/pdf/10.5555/1289189.1289216

[5] A. Kumar, S. Verma, H. Mangla, “A Survey of Deep Learning Techniques in Speech Recognition”, en *2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN)*, Greater Noida, Oct, 2018, pp. 179-185, 10.1109/ICACCCN.2018.8748399

[6] Shmyrev, N., 2021. *VOSK Models*. [online] VOSK Offline Speech Recognition API. Available at: <https://alphacephei.com/vosk/models> [Accessed 12 July 2021].

[7] Zhang, A. (2017). Speech Recognition (Version 3.8) [Software]. Available from https://github.com/Uberi/speech_recognition#readme

[8] A. Y. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, S. Satheesh, S. Sengupta, A. Coates, A. Ng, “Deep Speech: Scaling up end-to-end speech recognition”, *ArXiv*, vol. abs/1412.5567. 2014. https://arxiv.org/pdf/1412.5567

[9] N. Shakhovska, O. Basystiuk, “Development of the Speech-to-Text Chatbot Interface Based on Google API”, *MoMLeT*, 2019, ceur-ws.org/Vol-2386/paper16.pdf

[10] L. Sari, S. Thomas, M. Hasegawa-Johnson, “Training Spoken Language Understanding Systems with NonParallel Speech and Text”, *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Barcelona, 2020, pp. 8109-8113, 10.1109/ICASSP40776.2020.9054664

[11] Celery. Apr. 2021 (Referido 21/07/2021). Disponible en https://docs.celeryproject.org/en/stable/index.html

[12] Cloud Speech-To-Text. (Referido 21/07/2021). Disponible en https://cloud.google.com/speech-totext?hl=es

[13] IBM Watson STT. (Referido 21/07/2021). Disponible en https://www.ibm.com/pe-es/cloud/watson-speechto-text

[14] Amazon Transcribe. (Referido 21/07/2021). Disponible en https://aws.amazon.com/es/transcribe/

Fuentes de financiamiento:

Propias.

Conflictos de interés:

Los autores declaran no tener conflictos de interés.