Snake AI

Sebastián Escobar, Gabriel Restrepo, Luis Botero, Juan González
Ingeniería de Sistemas, ICESI
Cali, Colombia
sebastianescobar17@hotmail.com
gabrielrestrepo7@gmail.com
luis.botero2001@gmail.com
juank123ganzalez@gmail.com

***Abstract*— Este documento comprende un informe para un proyecto que hace uso de un modelo de clasificación para identificar comandos y poder así jugar a una versión del clásico juego de Snake pero usando la voz para moverse por el mapa.**

I. INTRODUCCIÓN

En este informe se presentará la forma en que se abordó un proyecto destinado a aplicar técnicas de aprendizaje supervisado y clasificación en forma de un juego clásico de Snake, haciendo uso de reconocimiento de comandos de voz para los movimientos básicos, planteando así un problema de clasificación, que fue resuelto usando una SVM para clasificar los audios una vez tratados.

II. TEORÍA

La teoría necesaria para abordar los temas referentes a este reporte son: modelos de clasificación, máquinas de soporte vectorial, análisis de componentes principales, conocimientos intermedios de comportamiento del sonido:

Para la tarea de procesar y clasificar cuatro comandos de voz, hemos elegido utilizar un modelo de Máquina de Vectores de Soporte (SVM) combinado con Análisis de Componentes Principales (PCA). A continuación, se presenta una justificación de esta elección basada en las ventajas teóricas de ambos métodos en el contexto del reconocimiento de voz.

SVM (Máquinas de Vectores de Soporte)
Capacidad Multiclase: Aunque diseñadas para clasificación binaria, las SVM se adaptan eficientemente a problemas multiclase, como distinguir entre cuatro comandos de voz, mediante técnicas como "one-vs-one" o "one-vs-all".

Maximización del Margen: Las SVM maximizan el margen entre las clases, lo que mejora la capacidad del modelo para generalizar a datos no vistos, crucial para la variabilidad en los comandos de voz.

Manejo de Alta Dimensionalidad: Las SVM funcionan bien con datos de alta dimensionalidad, como los vectores de características generados a partir de señales de voz, asegurando una clasificación precisa.

Núcleos para Separación No Lineal: Utilizando funciones núcleo (por ejemplo, RBF), las SVM pueden manejar relaciones no lineales entre características, esencial para la complejidad inherente de las señales de voz.

Control del Sobreajuste: Mediante el parámetro C, las SVM equilibran la maximización del margen y la correcta clasificación de puntos de entrenamiento, previniendo el sobreajuste y asegurando robustez en diversas condiciones.

PCA (Análisis de Componentes Principales)
Reducción de Dimensionalidad: PCA reduce la dimensionalidad de los datos de voz, eliminando redundancias y manteniendo la mayor variabilidad posible, lo que facilita el procesamiento y mejora la eficiencia del modelo.

Mejora del Rendimiento: Al reducir la dimensionalidad, PCA ayuda a minimizar el ruido y la redundancia en los datos, mejorando el rendimiento del SVM al enfocarse en las características más relevantes.

Eficiencia Computacional: La combinación de PCA y SVM reduce la carga computacional, ya que PCA disminuye el número de características que el SVM necesita procesar, acelerando tanto el entrenamiento como la predicción.

III. METODOLOGÍA

Al tener el planteamiento inicial del proyecto lo primero que se pensó fue acerca del dataset que se usaría para entrenar un modelo como el que se planteaba, pues necesitábamos audios de específicamente cuatro palabras en inglés (*up, down, left, right*), dado lo específico de la necesidad optamos por grabar nuestro propio dataset.

Tuvimos diferentes pruebas con el dataset debido a que el principal problema era el ruido de fondo de los audios usados, pues estaban condicionando los resultados, fue así como nos decantamos por una librería llamada *noisereduce* por la cual pasamos el dataset entero mejorando los resultados considerablemente.

Para el modelo de clasificación usamos una SVM a la cual le pasamos los audios tratados, primero haciéndoles el espectrograma de mel, que inicialmente funcionaba bien, pero conforme se hicieron más pruebas en diferentes equipos y con diferentes voces nos dimos cuenta de que era poco eficiente, motivo por el cual decidimos hacer ciertos cambios específicos para mejorar la precisión del modelo.

En el proceso de búsqueda de opciones de cambios que podíamos hacer al modelo, encontramos que la misma librería usada para hacer la conversión del audio al espectrograma de mel, tiene una función bastante útil para nuestro problema que es para calcular los Coeficientes Cepstrales en las Frecuencias de Mel, que nos extrae de mejor manera características del audio.

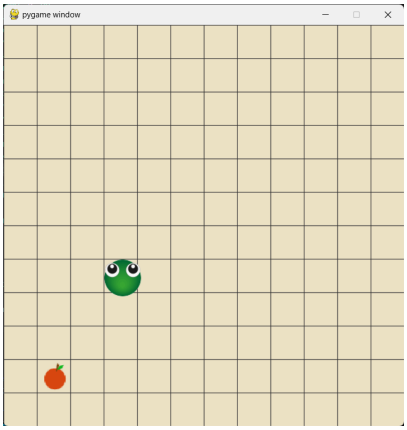
Habiendo hecho los cambios mencionados hasta el momento y, después de un análisis de las opciones de mejora del modelo de clasificación, nos fijamos en que el método de extracción de características del conjunto de audios del dataset no estaba siendo

propiamente aprovechado, en tanto que este método considera en gran medida los fonemas que componen las palabras del idioma. De esta forma, decidimos cambiar nuestra base de datos al idioma español, que, en contraste con el inglés (idioma en el que los cuatro comandos usados eran muy cortos y podrían confundirse por su pronunciación), posee palabras más largas y con más fonemas, por lo que al cambiarlo al español, los comandos (*sube, baja, derecha, izquierda*), son más diferenciables por el modelo y se puede extraer información valiosa para nuestro clasificador.

Con estos cambios aplicados vimos una notable mejoría de precisión, sin embargo aún había ciertos errores por las longitudes de la matriz y vectores resultantes, pero que, con unas sencillas funciones para evitar esto, y, manejar excepciones, fueron corregidos.

Luego de esto, tanto el modelo como el juego eran completamente funcionales, libres casi por completo de fallas en la mayoría de ambientes.

IV. RESULTADOS



Los resultados de este proceso se pueden apreciar en el siguiente material grabado para fines demostrativos.

https://youtu.be/akE-5sBQztM

Además de la evidente mejoría del comportamiento del modelo al momento de jugar, también se pueden apreciar las métricas extraídas del modelo, luego de concretar exitosamente todo el proceso de captura y limpieza de datos, dando paso al entrenamiento final del modelo.

```
[INFO] los comandos son: ['baja' 'derecha' 'izquierda' 'sube']
.....................................................................................
[INFO] hecho!!
[INFO] entrenando....
[INFO] evaluando....
[INFO] Porcentaje de prediccion usando LOOGISTICA:  0.9611111111111111
 SVM:  0.95
Precisión en cada pliegue: [0.94166667 0.95       0.975      0.95833333 0.96666667]
Precisión media: 0.9583333333333334
Fitting 5 folds for each of 8 candidates, totalling 40 fits
ACCURACY MEJOR SVM 0.95
[INFO] guardando el modelo
[INFO] hecho!!
```

V. ANÁLISIS DE RESULTADOS

A continuación, se presenta un análisis detallado de los resultados obtenidos.

El modelo SVM alcanzó una precisión promedio (accuracy) de 0.95. Este alto nivel de precisión indica que el modelo tiene un excelente desempeño en la clasificación de los comandos de voz, logrando identificar correctamente el comando en el 95% de los casos. Esta precisión se calculó mediante validación cruzada, lo que proporciona una estimación robusta y confiable del rendimiento del modelo en datos no vistos.

Para optimizar el modelo, se empleó validación cruzada con el fin de evaluar diferentes combinaciones de hiperparámetros. Se exploraron distintos valores para el parámetro de margen C y se probaron varios tipos de kernel. La validación cruzada asegura que el modelo no solo se ajuste bien a los datos de entrenamiento, sino que también generalice adecuadamente a nuevos datos. Los resultados del GridSearch indicaron que la combinación óptima era un kernel lineal con un valor de C de 0.1.

La elección de un kernel lineal sugiere que los datos de voz, después de la reducción de dimensionalidad con PCA, son linealmente separables en el espacio transformado. Esto simplifica el modelo y reduce la complejidad computacional, al mismo tiempo que mantiene una alta precisión.

C de 0.1 implica una regularización más fuerte, lo que permite un margen de separación más amplio a costa de permitir algunos errores de clasificación en los datos de entrenamiento. Esta configuración ayuda a prevenir el sobreajuste, asegurando que el modelo generalice bien en datos nuevos.

PCA redujo la dimensionalidad de los datos de voz, eliminando redundancias y ruidos, y preservando las características más importantes. Esto no solo mejoró la eficiencia computacional, sino que también ayudó a mejorar la precisión del modelo al enfocarse en los aspectos más relevantes de las señales de voz.

La reducción de dimensionalidad facilitó el trabajo del SVM, haciendo más manejable el problema de clasificación y mejorando el rendimiento general del modelo.

VII. CONCLUSIÓN Y TRABAJO A FUTURO

Del proyecto realizado podemos concluir que el manejo de audios como dataset de entrada es bastante delicado y se debe tratar con especial cuidado, pues son datos que pueden verse muy fácilmente alterados por ruido, duración, formato y demás cosas, como en este caso que hubo que rellenar espacios, cancelar ruido de fondo y filtrar mucho los audios que nos eran útiles o no para entrenar el modelo. Encontramos, además, la gran utilidad y armonía de los conceptos que, en su conjunto, permiten realizar un correcto trabajo al momento de desarrollar una solución haciendo uso de inteligencia artificial.

Para futuros trabajos, tendremos en cuenta todo el proceso que debe seguirse para la preparación de los datos, además de cada uno de los algoritmos y de las técnicas que mejoran el desempeño de los modelos de inteligencia artificial.

VIII. BIBLIOGRAFÍA

[1] S. Petrov, "Gist," https://gist.github.com/StanislavPetrovV/bb36787efbc30cd0921f0cbaa05244f1. [Accedido: 01-jun-2024]

[2] (2002) The IEEE website. [Online]. Available: http://www.ieee.org/

"Elimine el Ruido de Fondo de su Audio En Línea. VEED.IO,"

VEED.IO. [Online]. Available: https://www.veed.io/es-CO/herramientas/eliminar-ruido-de-fondo-de-un-audio. [Accessed: 01-Jun-2024].