Detección de movimientos con modelo de clasificación

Resumen

Este proyecto se centra en el desarrollo de un modelo de machine learning para el reconocimiento de actividades humanas en tiempo real, utilizando el seguimiento de puntos articulares clave a través del procesamiento de video. El sistema clasifica actividades específicas (caminar hacia la cámara, caminar hacia atrás, girar, sentarse y ponerse de pie) analizando los movimientos articulares y generando predicciones en tiempo real. Se emplearon tecnologías clave como MediaPipe para la estimación de poses, Scikit-learn para implementar modelos de clasificación supervisada, y XGBoost para lograr un rendimiento óptimo. El preprocesamiento de datos incluyó normalización, ingeniería de características utilizando deltas entre frames, y selección de características relevantes para mejorar la precisión del modelo.

Tras un ajuste de hiperparámetros mediante GridSearch, XGBoost alcanzó la mayor precisión con un 98.8%, superando a SVM y Random Forest. Los resultados destacan la efectividad de integrar el rastreo de poses en tiempo real y el machine learning para la clasificación de actividades. El trabajo futuro buscará generalizar aún más el modelo mediante la ampliación del conjunto de datos y perfeccionar su adaptabilidad a diversas condiciones ambientales, resaltando su potencial en áreas como el deporte y la medicina.

Introducción

El análisis de actividades humanas y el seguimiento de movimientos articulares representa un área de creciente interés en campos como la medicina, el deporte y la seguridad. El presente proyecto propone el desarrollo de un modelo de machine learning que a través del procesamiento de video en tiempo real logre clasificar actividades humanas específicas (caminar hacia la cámara, caminar de regreso, girar, sentarse y ponerse de pie) mediante el seguimiento de movimientos articulares clave y la clasificación de actividades en tiempo real.

El objetivo principal es crear una herramienta que combine librerías de seguimiento articular y de aprendizaje automático para ofrecer una solución eficiente y precisa, integrando técnicas de preprocesamiento, modelos de clasificación y una herramienta visual que permita a los usuarios observar la predicción en tiempo real del movimiento.

Teoría

El machine learning es un tipo de inteligencia artificial que se enfoca en hacer uso de datos y algoritmos con el objetivo de imitar la forma en que los humanos aprenden, mejorando su exactitud en tareas como la predicción y clasificación. Para este caso se usarán modelos de clasificación no supervisada, pues los datos de entrada ya están etiquetados con el movimiento que les corresponde.

En el desarrollo se hizo uso de los modelos SVM, Random Forest y XGBoost por lo que consideramos necesario abordar brevemente la lógica detrás de cada uno:

- SVM: busca el hiperplano que maximice el margen entre las distintas clases de datos. Es decir, intenta separar los puntos de diferentes clases de tal manera que la distancia entre los puntos más cercanos de cada clase y el hiperplano sea lo más grande posible.

- Random Forest: es un modelo de ensamble que utiliza múltiples árboles de decisión para realizar predicciones. Cada árbol se entrena de manera independiente con una muestra aleatoria de los datos y una selección aleatoria de características en cada nodo del árbol. El resultado final se obtiene promediando las predicciones de todos los árboles o mediante votación.

- XGBoost: es una técnica de ensamble que construye un modelo de manera secuencial, corrigiendo los errores de los modelos previos. En lugar de construir árboles completamente independientes como en Random Forest, XGBoost mejora los modelos existentes ajustando los errores residuales. Enfocándose en optimizar la función de pérdida, además de incorporar técnicas de regularización que ayudan a controlar el sobreajuste y mejora la eficiencia del modelo.

Considerando que se está abordando un problema de estimación de poses del cuerpo se hace necesario observar el siguiente diagrama que aborda los puntos comúnmente usados para lograr un rastreo adecuado y preciso de todo el cuerpo.

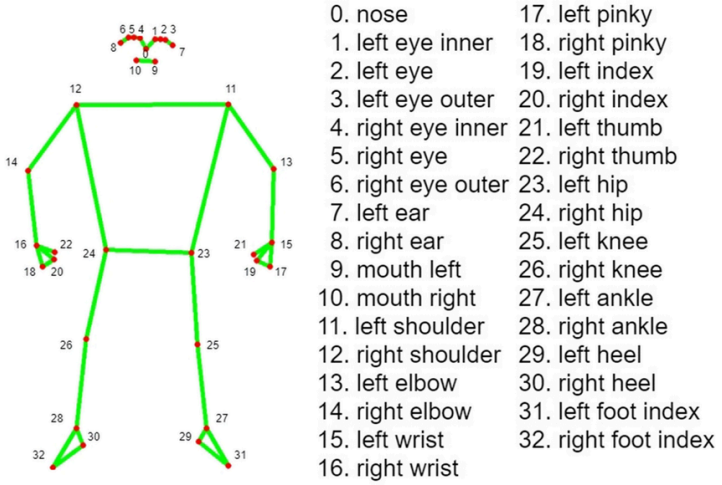

Figure - available from: Journal of Ambient Intelligence and Humanized Computing

Los videos tomados se pueden transformar a columnas con la posición de estos puntos en el plano para facilitar su procesamiento y poder usarlos en un modelo predictivo.

Metodología

Para abordar el proyecto se tomó como base la metodología CRISP-DM para organizar las etapas de trabajo, pues es adecuada para realizar iteraciones rápidas para mejorar los resultados, así como para automatizar algunas tareas y reutilizar lo que ya hayamos hecho

en cada fase ajustando el sistema de seguimiento de movimientos en tiempo real según vayamos obteniendo resultados y nuevos datos, haciendo que todo el proceso sea más ágil y eficiente.

Para la primera iteración, se analizó el enunciado entregado para entender los requerimientos del software y seleccionar las herramientas de trabajo, dentro de estas están: MediaPipe para obtener los puntos de rastreo de los movimientos; Jupyter Notebook para la creación y documentación del proyecto, permitiendo combinar código, visualizaciones y texto; Scikit-learn para implementar y evaluar modelos de aprendizaje supervisado; y cv2 para realizar la detección de movimiento en tiempo real.

Posteriormente se realizó una toma inicial de datos, a forma de prueba, para entender el formato de los datos y el tratamiento que se debía realizar sobre estos. Cada uno de los movimientos (caminar hacia adelante, caminar hacia atrás, rotar, sentarse y pararse) se tomó en videos distintos que fueron procesados por MediaPipe (obteniendo 33 marcas por cada frame) y etiquetados a mano. Con ello se realizó el análisis, limpieza y transformación de datos.

Dentro de las transformaciones, se normalizaron las marcas para que factores como la altura de las personas no afecten los resultados de los modelos. Y teniendo en cuenta que cada registro del conjunto de datos es un frame, es necesario que los datos estén en “contexto”, para que se pueda procesar el movimiento y no una posición estática. Inicialmente se tomaron grupos de 5 frames y se agruparon sus columnas en un solo registro, pero posteriormente se decidió cambiar esto por la implementación de deltas, calculando la diferencia entre las marcas cada 15 frames. Además de agregar una nueva columna para calcular la variación de la distancia de los puntos de la cadera.

Con los datos listos se implementaron los tres modelos de clasificación supervisada explicados anteriormente (SVM, Random Forest, XGBoost), y se evaluaron las siguientes métricas: accuracy, precisión, recall, y F1 Score. Estas se buscaron maximizar con el ajuste de parametros de GridSearch que otorgó los hiper parámetros óptimos para cada uno de los modelos.

Para implementar la solución en tiempo real se tomó el modelo con los mejores resultados (XGBoost) y se exportó en formato .pkl para unirlo a un programa de captura que genera los puntos de MediaPipe y los procesa con el modelo para imprimir el resultado por pantalla.

Todo el proceso se repitió en múltiples iteraciones, a medida que se obtenían nuevos datos y se realizaban ajustes al formato y procesamiento de los datos, para mejorar los resultados obtenidos.

Para el uso del modelo en un contexto real será necesario contar con una cámara enfocando un corredor de al menos 4 metros con una silla disponible. La cámara debe grabar a 30 fps alguna de las siguientes secuencias de movimientos: pararse de la silla, caminar hacia la cámara, dar la vuelta, caminar hacia la silla y sentarse. Durante la captura del video se podrá observar en pantalla la predicción más probable según el modelo.

Resultados

Como modelos de clasificación se eligieron SVM, Random Forest y XGBoost, cada uno recibió una partición 70-30 para entrenamiento y testing y la configuración inicial de cada uno fue la siguiente:

- **SVM**
  - kernel='rbf'
  - C=1
  - gamma='scale'
- **Random Forest**
  - n_estimators=100
  - max_depth=9
- **XGBoost**
  - eval_metric='mlogloss'
  - n_estimators=100
  - max_depth=6
  - scale_pos_weight=1
  - learning_rate=0.1

Ajuste de hiper parámetros

Para el ajuste hiper parámetros se desarrolló una matriz de parámetros para cada modelo y un total de 3 pliegues y nuestra variable de puntaje fue la accuracy. Para cada modelo se decidió trabajar con los siguientes campos:

- **SVM**
  - **C:** Es un parámetro de penalización para elegir entre maximizar el margen y minimizar el error de clasificación, con un C mayor se reduce el margen y con uno menor incrementa la posibilidad de un error de clasificación.
  - **kernel:** Define la función que transforma los datos en un espacio de mayor dimensión para que SVM logre encontrar el hiperplano que separa los datos.
  - **gamma:** Controla la influencia de los puntos de entrenamiento en el modelo según su distancia.
- **Random Forest**
  - **n_estimators:** Define el número de árboles que se crearán.
  - **max_depth:** Establece la profundidad de cada árbol.
  - **min_samples_split:** Indica el número mínimo de muestras para dividir un nodo
  - **min_samples_leaf:** Indica el número mínimo de muestras que debe tener cada hoja
  - **bootstrap:** Define si se usarán o no muestras de bootstrap para construir los árboles
- **XGBoost**
  - **n_estimators:** Define el número de árboles que se crearán.
  - **max_depth:** Establece la profundidad de cada árbol.
  - **scale_pos_weight:** Ajusta penalizaciones entre clases cuando nuestros datos no se encuentran balanceados.

- **learning_rate:** Controla la tasa de aprendizaje definiendo el tamaño de cada paso por iteración del modelo.

Los mejores hiper parámetros son:

- **SVM:** {'C': 100, 'gamma': 'scale', 'kernel': 'rbf'}
- **Random Forest:** {'bootstrap': False, 'max_depth': 10, 'min_samples_leaf': 2, 'min_samples_split': 2, 'n_estimators': 200}
- **XGBoost:** {'learning_rate': 0.15, 'max_depth': 9, 'n_estimators': 200, 'scale_pos_weight': 1}

Resultados obtenidos

Inicialmente se obtuvieron las siguientes métricas para los modelos:

- SVM: Accuracy del 83.5%
- Random Forest: Accuracy del 90.1%
- XGBoost: Accuracy del 98.2%

Después de aplicar Grid Search para encontrar los mejores hiper parámetros se obtuvieron los siguientes resultados:

- SVM: Accuracy del 91.8%
- Random Forest: Accuracy del 92.2%
- XGBoost: Accuracy del 98.8%

En esta ocasión, no se presentó un problema de overfitting, lo cual se debe a la incorporación de nuevos datos recopilados por compañeros y a una mejora significativa en el tratamiento de los mismos. Estos cambios impactaron notablemente los valores obtenidos por el modelo, resolviendo el problema de overfitting que anteriormente generaba valores de *accuracy* elevados.

En el caso de XGBoost, los resultados de accuracy continúan siendo altos, pero esto se debe a la precisión del modelo para clasificar y aprender gracias a sus parámetros optimizados. Además, los datos proporcionados al modelo son bastante limpios, ya que se recopilaron en distintas condiciones asegurando, al mismo tiempo, una reducción máxima del ruido.

Análisis de resultados

Tras los cambios y ajustes sobre la forma en que los datos eran entregados al modelo para procesarlos y dar una predicción evidenciamos grandes mejoras en comparación al primer .pkl. Inicialmente enfrentamos problemas de overfitting donde el modelo solo retornaba caminar y caminar hacia atrás, pero tras cambiar el enfoque de agrupamiento de frames a trabajar con deltas de posiciones, añadir la diferencia entre los puntos de la cadera, y para el rostro considerar únicamente la nariz, se obtuvo un modelo significativamente más estable, capaz de ofrecer predicciones acordes a los movimientos de la persona en cámara.

Identificamos una mejora en la generalización del modelo y una reducción en el overfitting. Sin embargo, el modelo todavía tiene un gran camino por recorrer, en especial porque ciertos movimientos del cuerpo humano se encuentran sujetas a articulaciones que dada la persona en cámara, no son estáticas o constantes, como lo son el movimiento de las manos y la distancia entre las piernas al momento de pararse y sentarse, las cuales logramos evidenciar al identificar la aparición de predicciones erróneas vinculadas a la posición de estas articulaciones, principalmente en los movimientos de pararse y sentarse. Tras esto, optamos por compartir datos con nuestros compañeros Camilo Carmona, Samuel Gutierrez, Manuel Herrera y Kevin Loachamin para lograr generalizar aún más estos dos movimientos, lo que desencadenó en mejores resultados y una mayor estabilidad. Aún así concluimos que todavía se hace necesario seguir alimentando al modelo con nuevos datos que le permitan generalizar aún más la posición de estas extremidades en los movimientos, ya que descartarlas no era una opción, pues impactó de forma significativamente negativa en la predicción de movimientos.

Conclusiones y trabajos futuros

El desarrollo de este proyecto permitió la creación de un software capaz de detectar y clasificar con precisión los movimientos de caminar hacia adelante, caminar hacia atrás, girar, sentarse y pararse, utilizando modelos de machine learning aplicados a datos procesados en tiempo real mediante MediaPipe.

El algoritmo XGBoost demostró ser el más adecuado para este tipo de problemas, logrando resultados superiores a los de SVM y Random Forest. Su capacidad para optimizar la función de pérdida y manejar datos etiquetados con precisión contribuyó a un desempeño excepcional, especialmente después de ajustar los hiperparámetros mediante GridSearch. La precisión del modelo alcanzó un **98.8%** después de la optimización, lo que refleja su capacidad para generalizar y adaptarse a nuevas situaciones.

Los puntos proporcionados por MediaPipe resultaron ser una herramienta muy importante al simplificar significativamente el procesamiento de video, evitando complejas técnicas de segmentación. La decisión de filtrar puntos irrelevantes (como los de ojos, cejas y boca) y centrarse en puntos clave como la nariz y las caderas mejoró la calidad de los datos y redujo el ruido, impactando positivamente los resultados. Además, se evidenció que los cálculos basados en deltas fueron críticos para capturar información sobre el movimiento en lugar de posiciones estáticas, siendo crucial seleccionar un número adecuado de frames para este cálculo, evitando mezclar movimientos o perder información importante.

Aunque se probaron técnicas como la reducción de dimensionalidad mediante PCA, estas no mejoraron los resultados debido al limitado número de variables y a la baja complejidad de los modelos utilizados. Por otro lado, incorporar variabilidad a los datos de prueba, al grabar en diferentes entornos y condiciones de luz, fue una estrategia que amplió la generalización del modelo y redujo problemas de overfitting, lo cual fue evidente en iteraciones iniciales donde el modelo se veía limitado por la uniformidad de los datos originales.

Este proyecto representa un paso significativo en nuestro proceso de aprendizaje hacia la implementación de soluciones prácticas y efectivas para el análisis de actividades humanas

mediante machine learning. Si bien los resultados son prometedores, se identificaron áreas para futuras mejoras, como la necesidad de continuar alimentando el modelo con más datos para fortalecer la generalización en escenarios complejos. Asimismo, la sensibilidad a las condiciones de luz y entorno sugiere oportunidades para integrar técnicas de aumento de datos o procesamiento avanzado que permitan un mejor desempeño en situaciones variables. Este trabajo demuestra el potencial de las herramientas de aprendizaje automático y procesamiento en tiempo real, ofreciendo un camino para aplicaciones futuras en áreas como el deporte y la medicina.

Referencias bibliográficas

IBM, «Machine learning», IBM, 28 de octubre de 2024. https://www.ibm.com/topics/machine-learning

«Support Vector Machine (SVM)», MATLAB & Simulink. https://la.mathworks.com/discovery/support-vector-machine.html#:~:text=Support%20vector%20machine%20(SVM)%20es,reconocimiento%20de%20im%C3%A1genes%20y%20voz

IBM, «Random Forest», IBM, 25 de octubre de 2024. https://www.ibm.com/topics/random-forest#:~:text=Random%20forest%20is%20a%20commonly,both%20classification%20and%20regression%20problems

«Cómo funciona el algoritmo XGBoost—ArcGIS Pro | Documentación». https://pro.arcgis.com/es/pro-app/latest/tool-reference/geoai/how-xgboost-works.htm#:~:text=XGBoost%20es%20la%20abreviatura%20de,aleatorio%20y%20refuerzo%20de%20gradientes