**Etapas del proyecto**: recopilación de datos, creación de la red adecuada y capacitación.

**Dataset:** Se utilizará El conjunto de datos de somnolencia en la vida real de la Universidad de Texas en Arlington (UTA-RLDD), se creó para la tarea de detección de somnolencia en varias etapas, y se enfoca no solo en casos extremos y fácilmente visibles, sino también en casos sutiles en los que las micro expresiones sutiles son los factores discriminativos. La detección de estos casos sutiles puede ser importante para detectar la somnolencia en una etapa temprana, a fin de activar los mecanismos de prevención de la somnolencia. Las micro expresiones sutiles de somnolencia tienen fuentes fisiológicas e instintivas, por lo que puede ser difícil para los actores que fingen estar somnolientos simular de manera realista tales expresiones. Nuestro conjunto de datos UTA-RLDD es el conjunto de datos de somnolencia realista más grande hasta la fecha.

El conjunto de datos RLDD consta de alrededor de 30 horas de videos RGB de 60 participantes sanos. Para cada participante obtuvimos un video para cada una de las tres clases diferentes: estado de alerta, baja vigilancia y somnolencia

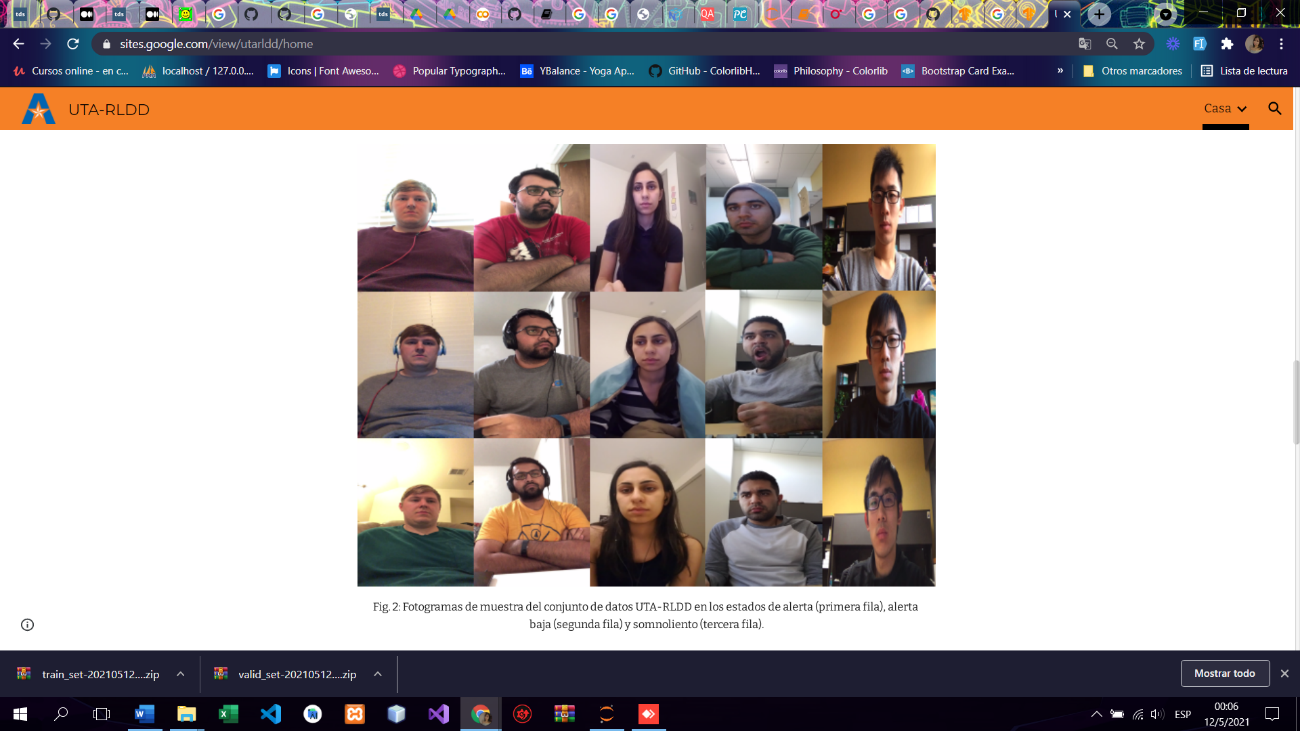
1) **Alerta:** Se les dijo a los sujetos que estar alerta significaba que estaban completamente conscientes para que pudieran conducir fácilmente durante largas horas.

2) **Vigilante bajo:** Este estado corresponde a casos sutiles en los que aparecen algunos signos de somnolencia o hay somnolencia, pero no se requiere ningún esfuerzo para mantenerse alerta. Si bien los sujetos posiblemente podrían conducir en este estado, se desaconsejaría conducir.

3) **Somnolencia:** este estado significa que el sujeto debe intentar activamente no quedarse dormido.

Teniendo en cuenta el contenido del dataset se utilizará las clases Vigilante bajo y Somnolencia.

Enlace de descarga del dataset: [http://vlm1.uta.edu/~athitsos/projects/drowsiness/](http://www.google.com/url?q=http%3A%2F%2Fvlm1.uta.edu%2F~athitsos%2Fprojects%2Fdrowsiness%2F&sa=D&sntz=1&usg=AFQjCNGvUK0KLx13-2ZLNgQ5O-kPmFq9uQ)

Ejemplo del dataset.

Manejo de los datos:

* Obtener secuencias de imágenes antes del suceso.
* Redimensionar las imágenes.
* Convertir al formato binario de numpy

**Propuesta de red.-**

El proyecto combina CNN y LSTM para predecir si un conductor al volante esta con síntomas de somnolencia o en curso de dormir utilizando una serie de imágenes momentos antes de que suceda.

Las CNN son buenas para la comprensión de imágenes, pero sin una relación de secuencia entre imágenes, perdemos mucha información temporal para predecir cómo una serie de eventos puede causar un incidente, por esta razón para realizar este proyecto se propone una red CNN-LSTM.

**Arquitectura CNN LSTM**

La CNN Long Short-Term Memory Network o CNN LSTM para abreviar es una arquitectura LSTM diseñada específicamente para problemas de predicción de secuencias con entradas espaciales, como imágenes o videos.

La arquitectura CNN LSTM implica el uso de capas de red neuronal convolucional (CNN) para la extracción de características en los datos de entrada combinados con LSTM para admitir la predicción de secuencias.

Los CNN LSTM se desarrollaron para problemas visuales de predicción de series de tiempo y la aplicación de generar descripciones textuales a partir de secuencias de imágenes (por ejemplo, videos). Específicamente, los problemas de:

* **Reconocimiento de actividad**: Generar una descripción textual de una actividad demostrada en una secuencia de imágenes.
* **Descripción de la imagen**: Generando una descripción textual de una sola imagen.
* **Descripción del video**: Generación de una descripción textual de una secuencia de imágenes.

Esta arquitectura define dos submodelos: el modelo CNN para la extracción de características y el modelo LSTM para interpretar las características a lo largo de los pasos de tiempo.

Se Requiere aplicar el modelo CNN a cada imagen de entrada y pasar la salida de cada imagen de entrada al LSTM como un solo paso de tiempo.

Podemos lograr esto envolviendo todo el modelo de entrada de CNN (una capa o más) en una capa TimeDistributed. Esta capa logra el resultado deseado de aplicar la misma capa o capas varias veces. En este caso, aplicarlo varias veces a varios pasos de tiempo de entrada y, a su vez, proporcionar una secuencia de "interpretaciones de imagen" o "características de imagen" al modelo LSTM para trabajar.

Normalmente hay 2 posibilidades:

* hacer convolución u otro cálculo neuronal antes de LSTM
* hacer el mismo tipo de trabajo después de LSTM

Para decidir qué orden elegir, debe pensar en lo que desea filtrar.

Para este proyecto, necesitamos verificar un objeto en movimiento, por lo que debemos buscar el objeto antes de detectar el movimiento. Entonces,

necesitamos hacer convoluciones antes de LSTM.

**pseudo código:**

|  |  |
| --- | --- |
|  | modelo = Secuencial()  # definir el modelo de CNN  modelo.agregar(Tiempo distribuido(Conv2D(...))  modelo.agregar(TimeDistributed (MaxPooling2D(...)))  modelo.agregar(TimeDistributed (flaten ()))  # definir el modelo LSTM  modelo.agregar(LSTM(...))  modelo.agregar(Denso(...)) |

model = Sequential()

# entrada, con 64 convoluciones para 5 imágenes   
# que tienen (224, 224, 3) forma  
model.add(TimeDistributed(Conv2D(64, (3,3), padding='same', strides=(2,2), activation='relu') input\_shape = (5, 224, 224, 3)))  
model.add(TimeDistributed(Conv2D(64, (3,3), padding='same', strides=(2,2), activation='relu')))  
model.add(TimeDistributed(MaxPooling2D((2,2), strides=(2,2))))

#segunda conv, 128  
model.add(TimeDistributed(Conv2D(128, (3,3), padding='same', strides=(2,2activation='relu')))  
model.add(TimeDistributed(Conv2D(128, (3,3), padding='same', strides=(2,2activation='relu')))  
model.add(TimeDistributed(MaxPooling2D((2,2), strides=(2,2))))

# y así sucesivamente con 512, 1024 ...   
# ...  
# entonces podemos usar Flatten para reducir la dimensión a 1   
modelo.add (Flatten ())

# y luego ... fusionar todas las salidas para   
# poder usar Dense (), y hacer predicciones ...

model.add(Dense(10))  
# ...

Bibliografía:

https://sites.google.com/view/utarldd/home

<https://www.brutalk.com/en/news/brutalk-blog/view/redes-de-memoria-a-corto-plazo-de-cnn-60471841d6ac1>

<https://medium.com/smileinnovation/how-to-work-with-time-distributed-data-in-a-neural-network-b8b39aa4ce00>

<https://medium.com/smileinnovation/training-neural-network-with-image-sequence-an-example-with-video-as-input-c3407f7a0b0f>