Explicit content

Beta

This feature is in a pre-release state and might change or have limited support. For more information, see the product launch stages (https://cloud.google.com/products/#product-launch-stages).

Explicit Content Detection (https://cloud.google.com/video-intelligence/docs/analyze-safesearch) detects adult content within a video. Adult content is content generally inappropriate for those under under 18 years of age, including but is not limited to, nudity, sexual activities, and pornography. Such content detected in cartoons or anime is also identified..

The following code sample demonstrates how to detect the presence of explicit content using the streaming client library.

JAVA NODE.JS PYTHON

/IDEO/BETA/SRC/MAIN/JAVA/COM/EXAMPLE/VIDEO/STREAMINGEXPLICITCONTENTDETECTION.JAVA)

FEEDBACK (#)

```
import com.google.api.gax.rpc.BidiStream;
import com.google.cloud.videointelligence.v1p3beta1.ExplicitContentFrame;
import com.google.cloud.videointelligence.v1p3beta1.StreamingAnnotateVideoRequest;
import com.google.cloud.videointelligence.v1p3beta1.StreamingAnnotateVideoResponse
import com.google.cloud.videointelligence.v1p3beta1.StreamingFeature;
import com.google.cloud.videointelligence.v1p3beta1.StreamingLabelDetectionConfig;
import com.google.cloud.videointelligence.v1p3beta1.StreamingVideoAnnotationResult
import com.google.cloud.videointelligence.v1p3beta1.StreamingVideoConfig;
import com.google.cloud.videointelligence.v1p3beta1.StreamingVideoIntelligenceServ
import com.google.protobuf.ByteString;
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.Arrays;
```

```
class StreamingExplicitContentDetection {

  // Perform streaming video detection for explicit content
  static void streamingExplicitContentDetection(String filePath) {
    // String filePath = "path_to_your_video_file";

    try (StreamingVideoIntelligenceServiceClient client =
        StreamingVideoIntelligenceServiceClient.create()) {

      Path path = Paths.get(filePath);
      byte[] data = Files.readAllBytes(path);
      // Set the chunk size to 5MB (recommended less than 10MB).
      int chunkSize = 5 * 1024 * 1024;
      int numChunks = (int) Math.ceil((double) data.length / chunkSize);

      StreamingLabelDetectionConfig labelConfig = StreamingLabelDetectionConfig.ne
          .setStationaryCamera(false)
          .build();

      StreamingVideoConfig streamingVideoConfig = StreamingVideoConfig.newBuilder(
          .setFeature(StreamingFeature.STREAMING_EXPLICIT_CONTENT_DETECTION)
          .setLabelDetectionConfig(labelConfig)
          .build();

      BidiStream<StreamingAnnotateVideoRequest, StreamingAnnotateVideoResponse> ca
          client.streamingAnnotateVideoCallable().call();

      // The first request must **only** contain the audio configuration:
      call.send(
          StreamingAnnotateVideoRequest.newBuilder()
              .setVideoConfig(streamingVideoConfig)
              .build());

      // Subsequent requests must **only** contain the audio data.
      // Send the requests in chunks
      for (int i = 0; i < numChunks; i++) {
        call.send(
            StreamingAnnotateVideoRequest.newBuilder()
                .setInputContent(ByteString.copyFrom(
                    Arrays.copyOfRange(data, i * chunkSize, i * chunkSize + chunkS
                .build());
      }

      // Tell the service you are done sending data
```

```
      call.closeSend();

      for (StreamingAnnotateVideoResponse response : call) {
        StreamingVideoAnnotationResults annotationResults = response.getAnnotation

        for (ExplicitContentFrame frame :
            annotationResults.getExplicitAnnotation().getFramesList()) {

          double offset = frame.getTimeOffset().getSeconds()
              + frame.getTimeOffset().getNanos() / 1e9;

          System.out.format("Offset: %f\n", offset);
          System.out.format("\tPornography: %s", frame.getPornographyLikelihood())
        }
      }
    } catch (IOException e) {
      e.printStackTrace();
    }
  }
}
```

Except as otherwise noted, the content of this page is licensed under the Creative Commons Attribution 4.0 License (https://creativecommons.org/licenses/by/4.0/), and code samples are licensed under the Apache 2.0 License (https://www.apache.org/licenses/LICENSE-2.0). For details, see our Site Policies (https://developers.google.com/terms/site-policies). Java is a registered trademark of Oracle and/or its affiliates.

Last updated December 3, 2019.