RunMo

Run Motion Capture and Analysis on the Cloud and on Edge

James Wall, Tosin Akinpelu, and Mumin Khan

Table of Contents

Problem Statement And Background

One of the most common problems that modern-day running athletes face is how to improve form and posture. Poor running form can impact running time and, even worse, lead to injury. This problem is compounded when examining the sports industry holistically. Almost every sport involves some form of running in either gameplay or as an exercise to prepare for gameplay. The problem is so pertinent that it is almost expected for an athlete who has to run a nontrivial distance on any sort of regular cadence to get a running injury at some point. Although there are some existing posture improving techniques out there, there is nothing in the market specific to running athletes. Moreover, they rely on human intelligence to access, and often come at a high enough cost that non-established athletes would have trouble accessing them. Due to the widespread nature of this problem and the lack of an existing viable solution, we elected to focus on creating a solution in this area for our final project for W251.

Existing solutions in the field

There are a host of existing products in the field that compete in this space, but there are none that do exactly what our project does. This section examines a few of the offered solutions.

Physical posture correctors

These are the most commonplace posture correctors in the market. The premise is wrapping a physical band around the area where you want to correct your posture and the band will resist your movement into non-optimal positions. Some examples can be found here: The 8 Best Posture Correctors of 2020 (verywellhealth.com)

Camera-based AI-powered workout apps

There are a host of apps that take in video feeds from smartphone cameras and use pose-recognition technology to guide the users through certain workouts. Some of the more popular examples include Kaia Health and inFiGro.

We have yet to find an app like this that specifically works for running posture.

Sensor based posture correctors

This sort of technology is the closest thing out there to what we are building. ASIC, a running shoe company, recently came out with a shoe that has pressure sensors embedded in it. A runner can run with an external camera and ASIC has technology to take input from the camera and sensors to output how the runner can improve their form: ASICS' new running shoe has an AI coach built inside: 6 things you should know (shortlist.com).

High-level Solution Proposal

Thinking about the solutions above yielded a natural niche in a user-facing application that can advise athletes of all levels on their running form to avoid injury. Moreover, if the application can run on the edge, this will democratize access further; ideally we'd like our solution to not only run on the cloud or NX Xavier, but less powerful IoT devices and smartphones.

To accomplish this, we've set an end goal of allowing a user to be able to point a web camera at themselves running and get near-immediate feedback on specific parts of their running form to improve upon. The application will be able to identify key biomechanical markers and features and make recommendations in near-real time.

Through research in the running field as well as prior knowledge, there are several features that we identified to use in this project. The main features are: elbow angle, knee angle, back angle, and stride length. We set target ranges for each of these features to be in.

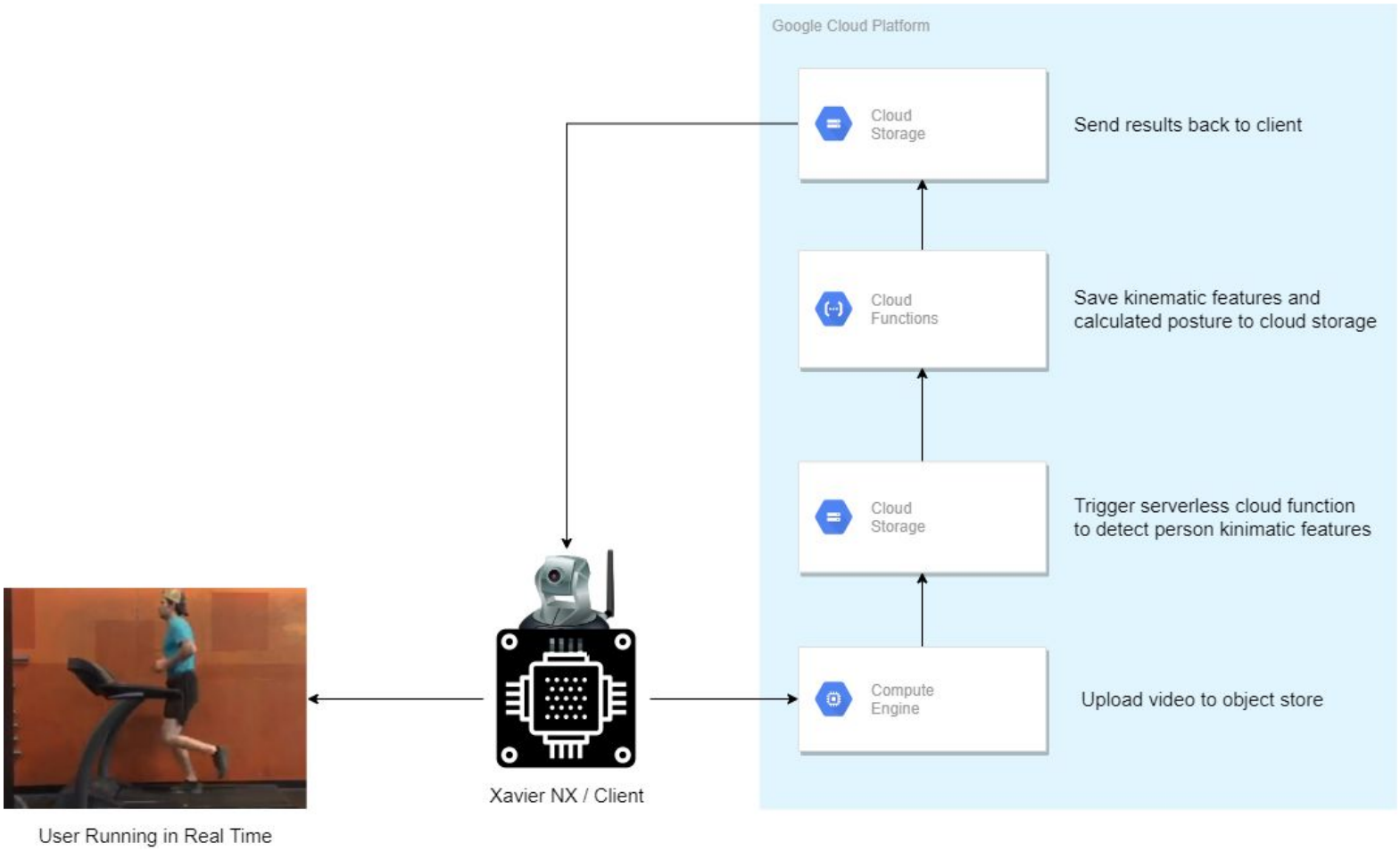
Implementation

This project was completed on Nvidia's Xavier (NX) to take in a video feed of a user running on a treadmill. The NX takes the video and feeds it to an "IoT Hub" docker container on the NX. That Docker Container then interfaces with GCP Object storage and drops the video in there.

Any new video dropped in a particular GCP Object storage notebook triggers a Google Colab notebook, which calls the Video Intelligence API. The Video Intelligence API identifies human features in the video, such as knees, ankles, and hips.

From there, we take the identified features and run some logic over them to determine features such as knee angle and stride length. We then take those values and run them through an equation we created to determine whether the user had good or bad posture at a given point in time. We can then use this data to give feedback to the user on how well they are doing running overall and give specific feedback to their running form.

Solution Architecture Diagram



Technical Discussion

We made heavy use of Google's Object Storage service and Google's Video Intelligence API, as well as Docker. The workflow relies on a Docker container that runs a Python script capturing video in real time. That Python script then set our Google Cloud Platform credentials and wrote the video file directly to GCP Object storage. We wrote a Python Google Cloud Function that is triggered anytime a new video file is placed in our GCP Object Storage bucket (much like a Lambda function triggering on an S3 bucket with AWS). This cloud function ingests the video data, runs the Google Video Intelligence API against it, and classifies whether the posture of the running individual was generally good or bad, based on the trigonometric angles we extracted. It then writes this data to a CSV. We then execute a second Python script in our NX Docker container to download the CSV results from our GCP Object Storage bucket and to display them to the user.

Aside on angle calculations

We researched proper running form and identified several features for our model to examine. The most compelling models seem to rely on extracted positional data rather than unsupervised learned features. We examined both elbow angles, knee angles, back angle, and body symmetry to determine whether a user had good posture or not. We then made a binary classification based on these input signals and wrote that classification data back to the user.

Our features were as follows:

Feature Table

Feature Num	Feature Name	Description	Target range
1	Back angle	The amount that someone is bent forward while running. Note: have to be careful not to conflate this with shoulder angle	30 degrees leaning forward
2	Elbow bend (Right and left)	Angle of elbow while running	90 degrees +/- 10 degrees
3	Stride length (Right and left)	How far (or not) someone puts their foot forwards when taking a step	1.2 * leg length
4	Is holding on to treadmill handles?	Boolean to measure whether participant is holding on to treadmill handles or not	N/A
5	Lower body symmetry/asymmetry	How similar or dissimilar is the participant's right side and left side while running?	0 asymmetry +/- 10% asymmetry
6	Upper body (arm) symmetry/asymmetry	How similar or dissimilar is the participant's right side and left side while running?	0 asymmetry +/- 10% asymmetry
7	Knee angle (right and left) while fully extended forward	What is the angle of the participant's knee(s) while the leg is in front of the runner in the step?	Bent - 70 degrees. Extended - 170
8	Knee angle (right and left) while fully extended backward	What is the angle of the participant's knee(s) while the leg is behind the runner in the step?	Bent - 70 degrees. Extended - 170

Example Output/Demo

Our goal was to create a graphical user interface (GUI) to display in real-time to the user when the app is processing and the output data. Unfortunately we ran out of time, so we settled on outputting a CSV file to the user that displays whether they had good or bad posture at a given time. The output is as follows:

TIMESTAMP	is_good_posture?
0	TRUE
0.01	FALSE
0.02	TRUE
0.03	TRUE
0.04	TRUE
0.05	TRUE
0.06	FALSE
0.07	FALSE
0.08	TRUE
0.09	TRUE
0.1	FALSE

We currently leave it to the user to match the point-in-time posture with their own video to see what they are doing. We also leave it to the user to calculate what percentage of the time they had good or bad posture. A future iteration of this project will complete these tasks and display to the user in a nice GUI.

Retrospective

“There are no such things as failures; there are only success and learning opportunities.” Although this project was extremely challenging to complete, we learned a lot on our way to our final deliverable. From the beginning, our team suffered from “analysis paralysis”. As a whole, we took too long to commit to a plan of action, often choosing to experiment and abandon rather than to follow up and make it work. We also chose a rather nascent and uncharted domain space for our project. Although sports medicine is better today than ever before, there’s still a lot of “art” to the practice.

During the project, we each made significant technical leaps. None of us had worked with any of the technology or methodologies in our application. We encountered and resolved issues in every part of the process. For instance, we learned that there are significant differences between Amazon Web Services and other cloud providers. Although many offer similar types of services, there are fundamental differences between role definitions and service interaction. We struggled initially to get a Google Cloud Object Storage bucket that would allow all of us to simultaneously work on this project. We also learned a bit about the perils of using open source technologies as well as public APIs. Our project hinged on the use of Google’s Video Intelligence API, however in the middle of the term Google released a major update to the API that caused our Python script that processed all of our NX data to not work anymore. We learned about the importance of using specific versions of public technologies so as to make our own project stable while building and testing it. We also learned that the MQTT protocol does not work very well with video. We ended up having to call GCP APIs directly to pass the video from our NX directly to GCP rather than through an IoT hub. One element that went right for us is that we were able to model our project off of several public examples of using the Video Intelligence API that Google released to the public. This example code gave us a framework to model our project off of as well as public feedback on the project.

Our team overcame many setbacks throughout this process. For one, one of our team members contracted COVID-19 and consequently was first unable to contribute, and then had to drop from the project entirely, leaving work undone. Additionally, we struggled with setting meaningful project milestones throughout the project. We all learned valuable lessons in realistic goals as well as the importance of consistent communication, and how the lack of communication can completely derail a project.

If we could do this project again, we would have assigned one person to handle the front-end GUI that would display to the user. This would make the end product a lot more polished. Moreover, we would've opted not to assign exclusive chunks of work to an individual member, as that made them a single point of failure. Instead, we would've split the project into larger portions and assigned them to subteams so that members could balance each other's schedules and commitments.

References

https://www.kaiahealth.com/

https://www.healthline.com/health/running-injuries

https://www.verywellhealth.com/best-posture-correctors-4171981

https://www.shortlist.com/news/asics-new-running-shoe-has-an-ai-coach-built-inside-6-things-you-should-know-401640

https://cloud.google.com/video-intelligence/docs/reference/rpc/google.cloud.videointelligence.v1p3beta1#google.cloud.videointelligence.v1p3beta1.VideoIntelligenceService