**How to Crowd Estimate using Crowd Sourced data**

Mohamed Abdou

As an alternative to established video-based solutions, with the consent of pedestrians they are sharing with us their spatio-temporal data which includes they latitude and longitude positions as well as the time stamp. These insights are fully known to not represent the whole population so other work needs to be made to estimate the whole crowd status. So, inferring the crowd density is needed even if only a limited set of pedestrians share their locations. One of the main challenges in doing so is accounting for standing, walking pedestrians, and bus passengers.

The density and speed of a crowd are important local characteristics to assess the criticality of a crowd situation, so mathematics will play a huge role in these by using the some means of clustering which might be the only branch of machine learning needed.

A normal heat map of most crowded places based on the data collected can be easily constructed which covers the density aspect. However, this doesn’t differentiate between passengers and other pedestrians. And to do this differentiation we need to include the aspect of velocity, which can be calculated using consecutive spatio-temporal data and their timestamps. Once this is done, grouping the data based on speed can help us create more informative real-time heat maps of the situation of crowds on streets.

A participatory sensing approach for crowd monitoring faces a major limitation: Participation is based on a voluntary base. Regardless of the incentivization strategy, it’s expected that only a small fraction of all attendees of a mass gathering is being tracked. This makes it challenging to conclude about the crowd density.

The sampling rate of the spatio-temporal data is also a factor that plays a huge role in making the whole estimation as accurate as possible, as we are monitoring bodies moving in noticeably different speeds, so a different treatment needs to be done to each based on their speeds, and since doing logic switching in the frontend is computationally expensive what we can do is set it to send those data packets after moving a certain amount or leaving a specific proximity. Or all phones send periodic data and the backend has to do some computations to skip repeated or close spatio-temporal data from the same phone id.