Meeting with Dr Benji Rosman

Monday, 18 September 2017

12:00, MSB UG09

- Our biggest problem is a lack of knowledge.
- What we've done:
    - Project is Synthetically Generated Voice (talked about what exactly the project is)
        - He said "it's quite a difficult project"
    - Told him about preliminary report plan ie. Voice conversion/feature extraction.
    - Style transfer for audio, but this plan won't really work too well.
    - Narrowed it down to feature extraction and speech recognition + speech synthesis
    - He laughed when we said we have 6 weeks -> we don't know how feasible it will be
        - He wouldn't go down the text to speech route. Feels like it's the wrong way to solve the problem.
    - We have the VCTK corpus for data and also ethical clearance
        - He said we won't have time to record voices.
        - Need to figure out how to scope down the project.
        - In machine learning -> supervised vs unsupervised learning.
            - With supervised learning you'd be doing a regression type problem.
            - Input/output pairs (being the synthetic and human recordings)
            - Imagine an x and y axis with one being human and one robot and map the points.
            - That would be the best way to go about thinking about this.
        - Get really, really, simple proof of concept rather than sophisticated approach.
    - Different ways to represent this problem, one being time series data where you could do something like a RNN, fraction of a second at a time doing the remapping. The next thing would be to flatten the signal and treat it as an image in *i* dimensional space & do some sort of signal processing. The next thing would be function approximation where you take signal & extract features and transform the features somehow and apply them to the output signal. Eg. If you had a pitch extraction function (avg pitch) and learn a function that maps it to something else (ie. Up or down) and reapply it to the whole signal.
        - In 6 weeks you're not going to get amazing results really.
        - No way you'll get something that will sound like human voices.
- So those are the 3 representational choices you could go about it. He would go with the last one
    - I.e Function approximation where you take signal & extract features and transform the features somehow and apply them to the output signal. Eg. If you had a pitch extraction function (avg pitch) and learn a function that maps it to something else (ie. Up or down) and reapply it to the whole signal.
    - Extract some stats from natural human speech and get a curve for it, extract the same stats from the robot speech & get a curve. Can you even manually (or potentially learn) the functions between the two and try transform it from the one to the other, adjust the values perhaps (in some naïve way) and see firstly as a proof of concept can you change the signal based on some training data. At this stage you're not running it through a model you're just mapping the stats of one to the other. This gives the first proof of concept, ie. Having looked at human speech and synthetic speech what's the main differences between the stats.

    - He talked about how you need to keep some data for testing ie. Don't use your testing data for training because that's stupid.
    - For now you could do a subjective evaluation.
    - From then he would try automate the process (all of this above you could do manually), could do some sort of curve fitting process with diff feature sets. That would be the easiest thing to do in the amt of time you have that might accomplish the task.
- A lot of people just jump into machine learning, but you should understand your provlem and the core properties and then use increasingly sophisticated methods to solve.
- It's nice to have cool results but you want to see if students can ask fundamental questions about the problem and map to some more abstract problem, do some POC and have an idea of how to expand it.
    - You could throw a conv net at it but how would you carry on from there? You'd just look on the internet and see what others did but that doesn't really show any insight. Not what you want to see when marking a project like this, you want to see a profound understanding of the problem.
    - Doing it this way you could say ok we need a more sophisticated mapping now, maybe look at temporal properties of the signal and take those into account the result would probs be better, if we could use less naïve mapping that would work better.
- Can do machine learning in matlab -> even doing regression (fitting a function to data) – you could do that manually but you're still doing machine learning! You want a function that maps from one input space to an output space **and you don't want it designed analytically, you want it to be derived from data.** So it is machine learning, also feature engineering (more traditional machine learning approach), nowadays with DNN feature learning is done automatically but he doesn't think this is the right way to go about this problem, should do it more manually.
- Ultimately if you had another year on this project that would be the way to go (DNN) and you could look at cool models like auto-encoders, he thinks that could work quite well but given your time constraints and what is expected of a 4th year project the most important thing is to come up with some insights and understanding of the problem and useful solid suggestions on how to improve things and some insights into what the problem is about.
- In terms of implementation
    - The best language to use is the one you're most comfortable with. Ie. MATLAB will be fine if you're more competent in it.
    - Also consider open source support.
- Essentially, we won't be needing to use neural networks. Could use something simple as linear regression, or code some transform by hand, don't do something more sophisticated than it needs to be initially. NN takes the longest to train and require the most data so you'd need maybe a month of compute time and loads of data so he wouldn't suggest using them in this problem. (At least not at our stage -> don't start with something too complicated)
- K-nearest neighbours – simple to implement.
- Do the simplest thing so that you know exactly what's going on, and at some level of abstraction it's kind of doing the same thing as a neural network.
    - You can say in future work how NN might be the right way to go.
    - Throwing a NN at the problem doesn't tell you anything about how it's solved and that's not necessarily what you want in a 4th year project.