[*Mnih et al.*, 2015] report using the Deep Q-Network to let the computer to learn how to play the Atari 2600 games. This paper has the detailed information about the structure, parameter and the way they train the algorithm. It has two key ideas, (1) use the experience replay, and iterative update that adjusts the action-values towards target values. It uses the discounted award function, which discount the future actions for the rewards. The learning essentially is a regression problem to estimate the Q-value from the observed images. The code is written all in Lua, but there is a similar project using it to train the flappyBird, which you can find it here: <https://yanpanlau.github.io/2016/07/10/FlappyBird-Keras.html>.

[*Araya-Polo et al.*, 2018] talks about their way to do tomography velocity model using deep neural network. They are still only using the synthetic data by first randomly generating velocity model and then generate the seismic waveforms, and use these as the pair to train the network. After that, they will generate more velocity model and test the model against these newly generated models. Also note that, their output needs to be processed in order to have the same demission as the velocity models.

[*Shi and Dustdar*, 2016] discusses the edge computing. An edge device is any computing or networking resource residing between data sources and cloud-based datacenters. It has many advantages over the cloud computing, i.e. reduces the communication significantly, computing on the edge device will be much smaller than the cloud counterpart, improve the battery usages at the end devices, and solves partially the privacy issue, etc. It has a lot of different applications, and this paper shows two different applications: online shopping and finding missing children. It can also apply to the disaster response, which will be really useful.

[*Rouet-leduc et al.*, n.d.] reports using machine learning algorithms (random forest) to estimate the GPS displacement rate from the seismic data. Basically, they are casting this into a regression problem. It seems to me (by using the 60-day example showing in their paper), that they first fit a linear line using lease squares over 60 day (60 data points in total, since the GPS sample is 1 per day). This slope of the linear fit (displacement rate) will be the target of the regression (smoothed by using a moving average). As for the seismic data, they use hourly or daily segments to extract the features, therefore, take the daily segments for example, they will have 60 sets of features, and then they take the average of these 60 sets to get one-set of the features. And use this one-set feature to estimate the displacement rate. Then the random forest model comes in to fit the regression problem. They also vary the window length of the seismic segments and the GPS. The best results come from using the 60 day as the time window for both GPS and seismic segments. Some comments I have for their paper:

* If the above understanding is correct, this is to say, their regression problem for the training dataset (first 4-year data) is only 24 data points (6 data points per year), which they already learned well (the performance of the next 8-year testing data looks good). I am a little surprised, and maybe the pattern of the signal is simple (the displacement rate is changing at a period of 1 year - dominant period).
* Their work reminds me of figure 2 of Rogers and Dragert 2003, where they try to relate the GPS displacement to the seismic data (they didn't go to higher time resolution, i.e. 60 days, they try to see that yearly)
* The displacement measured on the GPS is due to tremor activity, therefore, shouldn't we expect the seismic stations record that information? Then Parkfield or Alaska could potentially see the same thing.
* The Pearson correlation coefficient is a good measure of the phase of the two signal, right? Since if I have two signal phase matched well, even though their amplitudes are off, we still could have a good correlation. Therefore, correlation 0.66 (their best correlation) is good or not, I am a little hesitant to say whether the random forest performs really well or not. But clearly, there is a correlation.
* I didn't find the supplementary material, therefore, I can not see the estimation on the other GPS station. I guess the results are less ideal. But I was thinking to do something similar to use one seismic station's waveform to predict the other station's waveform (essentially learn the Green's function between the two.