**Machine Learning Engineer Nanodegree**

**Capstone Project**

David Howard

10/15/2017

**I. Definition**

**Project Overview**

SLAM (simultaneous location and mapping) is a well-known technique for using sensor data to create a map of a geographic region and also keep track of the sensors location within the constructed map. Typically this is done with either a particle filter or a Kalman filter. One application of SLAM operates on an unknown region and builds up knowledge about the region via sensor measurements. Another application uses existing knowledge, such as maps, along with sensors, and is aimed at determining the sensor's location within the known region. <https://en.wikipedia.org/wiki/Simultaneous_localization_and_mapping>

In reading about SLAM, I found it interesting that I had previously been trained to be a human SLAM algorithm when I was a helicopter pilot in the U.S. Army. We trained to navigate high speed low level flight by using only a topographic map and our eyes observing the terrain elevation features (no GPS allowed). There is a name for this technique, Nap-Of-the-Earth flight.   
<https://en.wikipedia.org/wiki/Nap-of-the-earth#Helicopter_NOE_flying>

One source of potential data for use in a SLAM system is a digital elevation model, generically referred to as DEM. A great source of such data was created by the NASA Shuttle Radar Topography Mission (SRTM). In 2000, the Space Shuttle was used to create a digital elevation map of the entire world (minus some polar regions). The NASA SRTM data is publicly available for download, in various height resolutions. [https://www2.jpl.nasa.gov/srtm/mission.htm](https://www2.jpl.nasa.gov/srtm/mission.htm))

**Problem Statement**

This project will attempt to create a simple SLAM type implementation using a convolutional neural network to determine position within a known region. The train/test data used for learning will be SRTM elevation data treated as an image. Input for evaluation will be images composed of pieces of the same data with possible modifications such as distortion, alignment and reduced resolution. Output will be a predicted position of the input images. The metric will be accuracy of the predictions over a test data set.

**Metrics**

The outputs of the model will be predictions of the probabilities of a test image location matching one of the actual locations. The outputs will be a set of probabilities, like the dog project. The 'absolute' accuracy will be the average of the highest predicted probability being correct over the set of test data. The evaluation will also attempt to evaluate if the actual location matches any one of the higher of the list of predicted probabilities, it if isn't the highest.

**II. Analysis**

*(approx. 2-4 pages)*

**Data Exploration**

SRTM data is available online at <https://dds.cr.usgs.gov/srtm>.

SRTM data is available in multiple resolutions. The data is segmented into 1 latitude/longitude degree squares. The highest publicly available resolution is 1 arc-second per elevation posting (Level 2, ~30 meters at the equator), which results in a 3600x3600 matrix of elevations. The data files are actually 3601x3601 in order to fill overlaps if multiple cells are composed together. Other resolutions are 3 arc seconds (Level 1, ~90 meters) and (Level 0, 30 arc seconds). A detailed description of the data is available at <https://dds.cr.usgs.gov/srtm/version2_1/Documentation/SRTM_Topo.pdf>

Organization of data

* Level 1
  + Binary file labeled type ‘.hgt’
  + 1201x1201 grid of height postings
  + Covers 1-degree latitude/longitude
  + Post spacing is 3 arc-seconds
  + Data type is unsigned 16 bit, big-endian
  + Rows are lower to higher
  + Columns are left to right
  + Naming convention specifies location e.g. N39W120.hgt
* Level 2
  + Binary file labeled type ‘.hgt’
  + 3601x3601 grid of height postings
  + Covers 1-degree latitude/longitude
  + Post Spacing is 1 arc-seconds
  + Data type is unsigned 16 bit, big-endian
  + Rows are lower to higher
  + Columns are left to right
  + Naming convention specifies location e.g. N39W120.hgt

**Exploratory Visualization**

In this section, you will need to provide some form of visualization that summarizes or extracts a relevant characteristic or feature about the data. The visualization should adequately support the data being used. Discuss why this visualization was chosen and how it is relevant. Questions to ask yourself when writing this section:

* *Have you visualized a relevant characteristic or feature about the dataset or input data?*
* *Is the visualization thoroughly analyzed and discussed?*
* *If a plot is provided, are the axes, title, and datum clearly defined?*

**Algorithms and Techniques**

In this section, you will need to discuss the algorithms and techniques you intend to use for solving the problem. You should justify the use of each one based on the characteristics of the problem and the problem domain. Questions to ask yourself when writing this section:

* *Are the algorithms you will use, including any default variables/parameters in the project clearly defined?*
* *Are the techniques to be used thoroughly discussed and justified?*
* *Is it made clear how the input data or datasets will be handled by the algorithms and techniques chosen?*
* **http://cs231n.github.io/**
* **Cross-validate or Train/validate/test : k-foldcross-validation less useful in CNN due to training time**
  + **Use single train/validate/test datasets**
* **Problem: Limited data set : use ImageDataGenerator**
  + **Subdivide original ‘image’**
  + **Use imagedatagenerator to generate N additional images of each subdivision**
  + **Divide data individually into train/validate/train sets**
  + **Be sure all subdivisions are represented in each set**
  + **Renormalize subdivisions before use**
* **Problem:**
  + **Size of subdivisions**
  + **Large subdivisions may generate good weights since they are too general?**
  + **Small subdivisions may work better but take longer training time?**
* **Instead of monochrome, color by height as RGB?**
  + **Possibly increases resolution of features**
  + **Increases dimensionality**
  + **Training time?**
* **‘static’ Feature transforms**
  + **Cartesian to polar**
  + **Color histogram (global color)**
  + **Histogram of oriented gradients (edge information)**
  + **Bag of words : random patches – to – encoding**
  + **CNN extracts feature transforms itself**

**Benchmark**

In this section, you will need to provide a clearly defined benchmark result or threshold for comparing across performances obtained by your solution. The reasoning behind the benchmark (in the case where it is not an established result) should be discussed. Questions to ask yourself when writing this section:

* *Has some result or value been provided that acts as a benchmark for measuring performance?*
* *Is it clear how this result or value was obtained (whether by data or by hypothesis)?*

**III. Methodology**

*(approx. 3-5 pages)*

**Data Preprocessing**

In this section, all of your preprocessing steps will need to be clearly documented, if any were necessary. From the previous section, any of the abnormalities or characteristics that you identified about the dataset will be addressed and corrected here. Questions to ask yourself when writing this section:

* *If the algorithms chosen require preprocessing steps like feature selection or feature transformations, have they been properly documented?*
* *Based on the****Data Exploration****section, if there were abnormalities or characteristics that needed to be addressed, have they been properly corrected?*
* *If no preprocessing is needed, has it been made clear why?*

**Implementation**

In this section, the process for which metrics, algorithms, and techniques that you implemented for the given data will need to be clearly documented. It should be abundantly clear how the implementation was carried out, and discussion should be made regarding any complications that occurred during this process. Questions to ask yourself when writing this section:

* *Is it made clear how the algorithms and techniques were implemented with the given datasets or input data?*
* *Were there any complications with the original metrics or techniques that required changing prior to acquiring a solution?*
* *Was there any part of the coding process (e.g., writing complicated functions) that should be documented?*

**Refinement**

In this section, you will need to discuss the process of improvement you made upon the algorithms and techniques you used in your implementation. For example, adjusting parameters for certain models to acquire improved solutions would fall under the refinement category. Your initial and final solutions should be reported, as well as any significant intermediate results as necessary. Questions to ask yourself when writing this section:

* *Has an initial solution been found and clearly reported?*
* *Is the process of improvement clearly documented, such as what techniques were used?*
* *Are intermediate and final solutions clearly reported as the process is improved?*

**IV. Results**

*(approx. 2-3 pages)*

**Model Evaluation and Validation**

In this section, the final model and any supporting qualities should be evaluated in detail. It should be clear how the final model was derived and why this model was chosen. In addition, some type of analysis should be used to validate the robustness of this model and its solution, such as manipulating the input data or environment to see how the model’s solution is affected (this is called sensitivity analysis). Questions to ask yourself when writing this section:

* *Is the final model reasonable and aligning with solution expectations? Are the final parameters of the model appropriate?*
* *Has the final model been tested with various inputs to evaluate whether the model generalizes well to unseen data?*
* *Is the model robust enough for the problem? Do small perturbations (changes) in training data or the input space greatly affect the results?*
* *Can results found from the model be trusted?*

**Justification**

In this section, your model’s final solution and its results should be compared to the benchmark you established earlier in the project using some type of statistical analysis. You should also justify whether these results and the solution are significant enough to have solved the problem posed in the project. Questions to ask yourself when writing this section:

* *Are the final results found stronger than the benchmark result reported earlier?*
* *Have you thoroughly analyzed and discussed the final solution?*
* *Is the final solution significant enough to have solved the problem?*

**V. Conclusion**

*(approx. 1-2 pages)*

**Free-Form Visualization**

In this section, you will need to provide some form of visualization that emphasizes an important quality about the project. It is much more free-form, but should reasonably support a significant result or characteristic about the problem that you want to discuss. Questions to ask yourself when writing this section:

* *Have you visualized a relevant or important quality about the problem, dataset, input data, or results?*
* *Is the visualization thoroughly analyzed and discussed?*
* *If a plot is provided, are the axes, title, and datum clearly defined?*

**Reflection**

In this section, you will summarize the entire end-to-end problem solution and discuss one or two particular aspects of the project you found interesting or difficult. You are expected to reflect on the project as a whole to show that you have a firm understanding of the entire process employed in your work. Questions to ask yourself when writing this section:

* *Have you thoroughly summarized the entire process you used for this project?*
* *Were there any interesting aspects of the project?*
* *Were there any difficult aspects of the project?*
* *Does the final model and solution fit your expectations for the problem, and should it be used in a general setting to solve these types of problems?*

**Improvement**

In this section, you will need to provide discussion as to how one aspect of the implementation you designed could be improved. As an example, consider ways your implementation can be made more general, and what would need to be modified. You do not need to make this improvement, but the potential solutions resulting from these changes are considered and compared/contrasted to your current solution. Questions to ask yourself when writing this section:

* *Are there further improvements that could be made on the algorithms or techniques you used in this project?*
* *Were there algorithms or techniques you researched that you did not know how to implement, but would consider using if you knew how?*
* *If you used your final solution as the new benchmark, do you think an even better solution exists?*