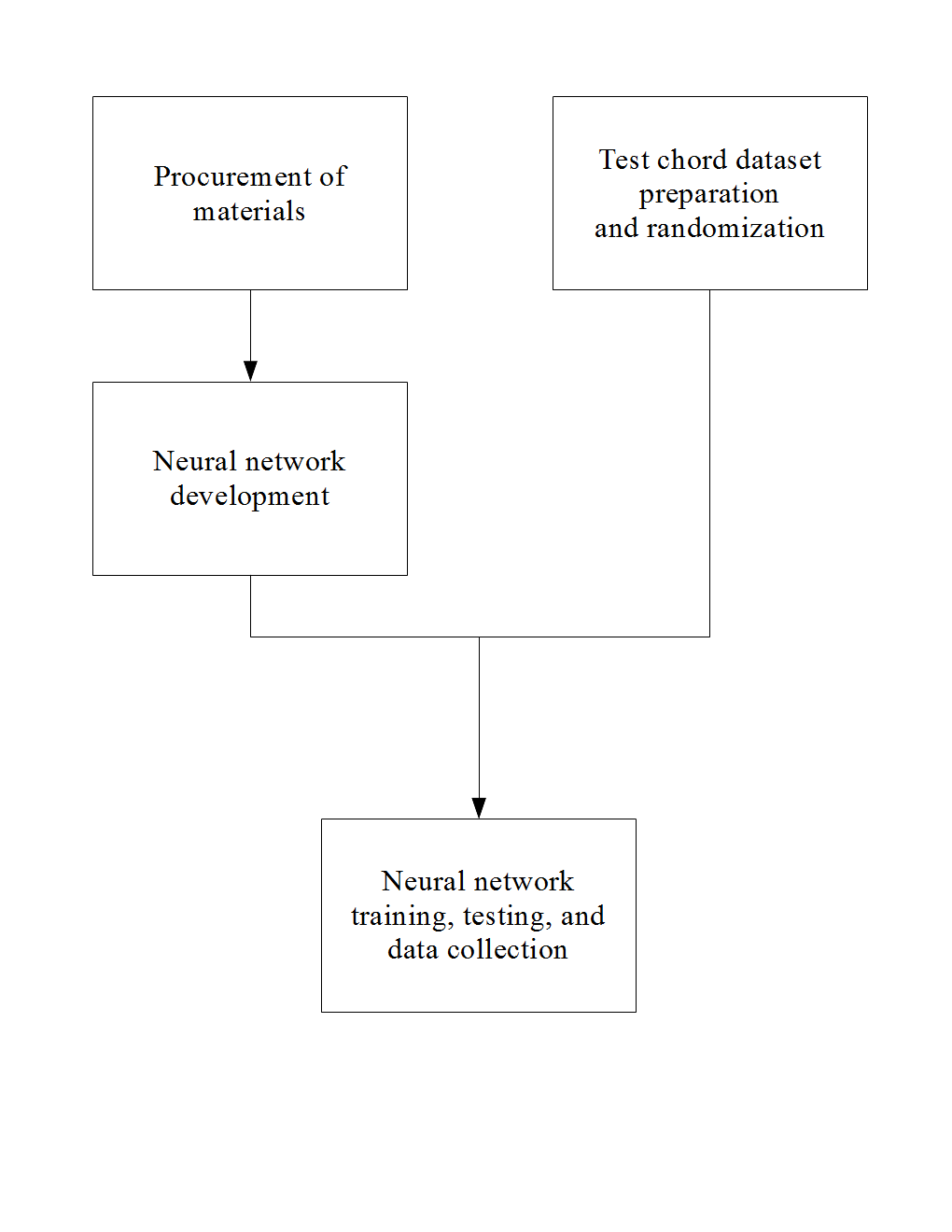
|  |  |  |
| --- | --- | --- |
| PSHS Main Campus | | Project Code: B 09 |
| Project Title: *Real-Time Identification of Common and Extended Musical Chords using Artificial Neural Networks* | | |
| Researchers: | Coronel, Lesli Natasha A. | |
| Navarro, Joachim Alfonso A. | |
| - | |

**METHODOLOGY**

**Level 0 Process Flowchart**



**Level 1 Processes**

**Procurement of materials**

1. **Acquisition of graphics processing unit** (Nickolls, Buck, Garland, & Skadron, 2008)A GPU or graphics processing unit is useful for parallel computing applications such as artificial neural network (**ANN**) simulations and training. Colina, Perez, and Paraan (2017) found that the use of a consumer GPU accelerated ANN computation by approximately 30 times when compared to a regular central processing unit. The minimum recommended GPU for research work is the **NVIDIA GeForce GTX 1070** (“GTX 1070” or “1070”) (Dettmers, 2017). This GPU is used in many neural network researches and papers (Colina, Perez, & Paraan, 2017; Kang, Hong, & Park, 2017; Kim, Hong, Nam, & Park, 2017, Zorrilla *et al*., 2017). This GPU is readily available in PC hardware stores across the country, and in prominent online stores such as Amazon, Newegg, and Lazada.
2. **Acquisition of programming tools**These are pieces of software necessary to define the structure of, train, and test the neural network. They will be installed in one of the researchers’ computers using the methods in mustgoplay (2016), which uses these components (except **e.**):
   1. Theano & Keras  
      Keras is a Python-specific neural network library that allows for swift prototyping of ANN structures. Its relative simplicity is well-suited to research. It runs on Theano, a mathematical matrix library for Python.
   2. NVIDIA CUDA & cuDNN  
      These are libraries that allow computation on an NVIDIA GPU and ANN simulation on a NVIDIA GPU, respectively.
   3. Anaconda  
      This is a specific distribution of the Python and R programming languages for scientific computing and data processing.
   4. PyCharm  
      This is a piece of software known as an *independent development environment* or IDE, which allows the user to write and run code.
   5. *pyrtmidi* library (Kidd, 2017)This library will be installed as it allows real-time MIDI signal input to be interpreted by Python code. This interpretation can be fed to the neural network for chord identification.

**Test chord dataset preparation and randomization**

1. **Determination of sample size**The appropriate sample size will be used to minimize bias and testing duration. The methods in Turmon and Fine (1994) will be used as a guide.
2. **Preparation of chord dataset**A program that makes a full list of chords from the 12 possible root notes and the chosen chord types will be written. It will then output these chords to a text file that will serve as the training dataset of the ANN. Its evaluation dataset will be the result of a randomization to be detailed in *Neural network training, testing, and data collection - Stratified random sampling of chords*.
3. **Writing of stratified random sampling program**To facilitate stratified random sampling for every pass, a random sampler will be written in Python. This sampler takes a number of chords from each chord type in the dataset at random; these will be played on the MIDI controller during each testing phase. This method dramatically speeds up the random sampling process.

**Neural network development**

1. **MIDI input-output coding**  
   An algorithm that takes a current MIDI input and converts it into a format that can be interpreted by the neural network will be written. The chords will be interpreted by the neural network as found in Perera and Kodithuwakku (2005).
2. **Neural network coding**Using the appropriate programming tools in *Procurement of materials: Acquisition of programming tools*, an artificial neural network whose goal is to identify chords will be written in Python. Its structure and configuration will be designed based on the MADALINE model in Perera and Kodithuwakku (2005), but may be modified as needed.

**Neural network training, testing, and data collection**

This entire series of steps is iterative; it proceeds once for every training epoch (one training “pass through” the dataset) of the neural network.

1. **Neural network training**The neural network will be trained using the methods in Perera and Kodithuwakku (2005). The appropriate revisions to the neural network (e.g. number of neurons) will be implemented according to the differences in input dataset size, equipment, number of chord types, etc. between the aforementioned study and this one.
2. **Stratified random sampling of chords**Due to the size of the input dataset, a stratified random sample of chords for each chord type will be selected using the program written in *Test chord dataset preparation and randomization: Preparation of chord dataset*. The chords will be grouped by (that is, the strata will be) *chord* *type*. A number (the sample size) of chords of each type will be in this sample.
3. **Testing, data logging, and neural network adjustment**After training, the neural network will be tested with the chords in the sample. The appropriate data such as error rate and response time will be recorded. After the testing phase, adjustments will be made to the neural network depending on the desired output. The network will then be trained again until a condition to be determined is satisfied. This condition may be dependent on the design of the neural network, so it is not specified here.

**References**

Colina, N. C. A., Perez, C. E., & Paraan, F. N. C. (2017). Simple techniques for improving deep neural network outcomes on commodity hardware. *AIP Conference Proceedings*, *1871*, 040001. doi:10.1063/1.4996523.

Dettmers, T. (2017). Which GPU(s) to get for deep learning: My experience and advice for using GPUs in deep learning. Retrieved from http://timdettmers.com/2017/04/09/which-gpu-for-deep-learning/.

Kang, J. K., Hong, H. G., Park, K. R. (2017). Pedestrian detection based on adaptive selection of visible light or far-infrared light camera image by fuzzy inference system and convolutional neural network-based verification. *Sensors*, *17*(7), 1598. doi:10.3390/s17071598.

Kidd, P. (2017). *pyrtmidi: Real-time MIDI I/O for Python* [GitHub repository]. Retrieved August 23, 2017, from <https://github.com/patrickkidd/pyrtmidi>.

Kim, K. W., Hong, H. G., Nam, G. P., & Park K. R. (2017). A study of deep CNN-based classification of open and closed eyes using a visible light camera sensor. *Sensors*, *17*(7), 1534. doi:10.3390/s17071534.

Nickolls, J., Buck, I., Garland, M., & Skadron, K. (2008). Scalable parallel programming with CUDA. *ACM Queue, 6*(2), 40-53.

mustgoplay (2016). Anaconda Python, CUDA, cuDNN and Theano with PyCharm install for CPU and GPU on Windows 10 [Online forum comment.] Retrieved from https://github.com/Theano/Theano/issues/5348.

Perera, N., & Kodithuwakku, S. R. (2005, December 15). Music chord recognition using artificial neural networks. *1st Proceedings of the International Conference on Information and Automation*, 304-308.

Turmon, M. J. & Fine, T. L. (1994). Sample size requirements for feedforward neural networks. In G. Tesauro, D. S. Touretzky & T. K. Leen (Eds.), *Advances in Neural Information Processing Systems* (pp. 327-334).Boston, MA: MIT Press.

Zorrilla, A. L., de Velasco-Vázquez, Serradilla-Casado, O., Roa-Barco, L., Graña, M., Chyzhyk, D., & Price, C. C. (2017). Brain white matter lesion segmentation with 2D/3D CNN. *International Work-Conference on the Interplay Between Natural and Artificial Computation 2017*, *10334*, 394-403. doi:10.1007/978-3-319-59740-9\_39.