1 Evaluating Performance

Before making predictions on unlabeled data such as the Chaotic Synthesizers, we must determine that our model performs reasonably on data that it has never interacted with. The most common practice is to divide a full data set into a subset of *training* samples, and *testing* samples. The exact ratio of sample volume between these subsets varies depending on the task [3, 2, 11]. We as the names imply, the training subset is used to fit the model, and we use the labeled testing data set to evaluate. Since the testing data is labeled, we can compare the classifiers predictions to the *ground truth* labels.

1.1 Cross Validation

Performance evaluations are most commonly implemented in the form of *K-Fold Cross-Validation* (Also called X-val). This process involves taking a data set containing N unique samples and dividing it into K non-overlapping subsets. 1 subset is reserved to evaluate the model, and $K-1$ are used to train the model. This belongs a larger family of statisical validations called *Resampling methods* [4]. Below, we detail pseudo-code for a K-Fold Cross Validation algorithm.

Algorithm 1 A K-Fold Cross Validation program.

```
Require: Untrained Network or related learning algorithm, $F^*$
Require: A full data set of $N$ samples. $X^{(i)}$, $i \in 0, 1, 2, ..., N-1$
Require: Number of splits in Cross validation, $K$
Require: Performance metric function(s), $P$
  Divide Data into $K$ non-overlapping subsets $x_i$, each with roughly $K/N$ samples
  $X \to \{x_0, x_1, x_2, ..., x_{K-2}, x_{K-1}\}$
  Performance History $\leftarrow \{\}$
  for $j = 0, 1, 2, 3, ..., K-2, K-1$ do
    Reset all parameters in $F^*$ to a random "untrained" state
    Set aside testing data subset
    $X_{test} = x_j$
    Concatenate the remaining subsets into training data set
    $X_{train} = x_{j \neq k}$
    Train the model, $F^*$ with the $X_{train}$ data set
    Evaluate the trained model with metric function(s) $P$
    Store Performance $P$ in Performance History array
  end for
  Compare performance results, and adjust model, parameters as needed, and repeat if
  desired.
```

By using cross-validation, we gain a deeper understanding of the model to ensure that the model can be properly trained consistently. This also helps counteract the possibility that the model got very lucky or unlucky with a particular set of initial conditions [1, 4].

1.2 Performance Metrics

In the case of the multi-category classifier, it is also important that we choose the appropriate metrics to evaluate our performance. While the neural network itself uses the cost function as it's sole performance metric to optimize, we also require a set of more human-readable functions. We introduce a set of functions and metrics than enable a more tangible interpretation of model performance. To evaluate a performance metric, we require a set of samples with ground truth labels, y, and a model's prediction for those labels, y^*.

1.2.1 Confusion Matrix

The confusion matrix (also called a confusion table) is a very quick, often graphical model that can be used to show how a model performs over a subset of predictions. The general idea of this object is count the number of times class A is predicted to be in class B, and vice-versa [1]. If we see that these classes are being repeatedly *confused* in the model's prediction process, then we must modify the model or features to account for it.

For a k-classes classifier, a confusion matrix will have shape $k \times k$. Each row represents the actual labeled class and each column represents a predicted class. Thus for a confusion matrix, C, we can say that:

$$C_{i,j} = \text{Number of samples that belong to class } i\text{, and were predicted to be in class } j \tag{1}$$

Thus, $i = j$, represents a correct prediction, and $i \neq k$ indicates an incorrect prediction. A confusion matrix with a strong main diagonal indicates a model that consistently predicts correct labels [1].

Images of Confusion Matrices Go Here!

Figure 1: Example Confusion Matrices for k-classes tasks

We can also use a confusion matrix to define four useful quantities relating a prediction to it's label. Suppose a prediction, $y^* = p$ and a truth, $y = q$.

1. **True-Positive**
   A sample is a true-positive if $p = q$
2. **True-Negative**
3. **False-Positive**
4. **False-Negative**

finish this!

1.2.2 Precision Score

Precision score (also called *specificity*) offers a more concise performance metric than a confusion metric. For a classifier with $k = 2$ unique classes, we define the precision score of a model as:

$$\text{Prec} = \frac{TP}{TP + FP} \tag{2}$$

Where TP is the number of *true-positive*, and FP is the number of false-positives predictions. For a k-classes confusion matrix, C, the precision score is given by the entry $C_{j,j}$ divided by the sum of column j:

$$\text{Prec}_j = \frac{C_{j,j}}{\sum_{i=0}^{k-1} C_{i,j}} \tag{3}$$

For multi-class problems, the precision score is usually used as an average over all classes.

Physical Significance of Precision

1.2.3 Recall Score

Recall score (also called *sensitivity*) also offers a more concise performance metric than a confusion metric. For a classifier with $k = 2$ unique classes, we define the recall score of a model as:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{4}$$

Where TP is the number of *true-positive*, and FN is the number of false-negative predictions. For a k-classes confusion matrix, C, the precision score is given by the entry $C_{j,j}$ divided by the sum of row j:

$$\text{Recall}j = \frac{C_{j,j}}{\sum_{i=0}^{k-1} C_{j,i}} \tag{5}$$

For multi-class problems, the recall score is also used as an average over all classes.

Physical Significance of Recall

References

[1] Geron, Aurelien. *Hands-on Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems.* O'Reilly, 2017.

[2] Geron, Aurelien. *Hands-on Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems.* 2nd ed., O'Reilly, 2019.

[3] Goodfellow, Ian, et al.*Deep Learning.* MIT Press, 2017.

[4] James, Gareth, et al. *An Introduction to Statistical Learning with Applications in R.* Springer, 2017.

[5] Khan, M. Kashif Saeed, and Wasfi G. Al-Khatib. "Machine-Learning Based Classification of Speech and Music." Multimedia Systems, vol. 12, no. 1, 2006, pp. 55–67., doi:10.1007/s00530-006-0034-0.

[6] Levine, Daniel S. *Introduction to Neural and Cognitive Modeling.* 3rd ed., Routledge, 2019.

[7] Liu, Zhu, et al. "Audio Feature Extraction and Analysis for Scene Segmentation and Classification." Journal of VLSI Signal Processing, vol. 20, 1998, pp. 61–79.

[8] Loy, James , *Neural Network Projects with Python.* Packt Publishing, 2019

[9] McCulloch, Warren S., and Walter Pitts. "A Logical Calculus of the Ideas Immanent in Nervous Activity." *The Bulletin of Mathematical Biophysics*, vol. 5, no. 4, 1943, pp. 115–133.

[10] Mierswa, Ingo, and Katharina Morik. "Automatic Feature Extraction for Classifying Audio Data." *Machine Learning*, vol. 58, no. 2-3, 2005, pp. 127–149., doi:10.1007/s10994-005-5824-7.

[11] Mitchell, Tom Michael. Machine Learning. 1st ed., McGraw-Hill, 1997.

[12] Olson, Harry E. *Music, Physics and Engineering.* 2nd ed., Dover Publications, 1967.

[13] Peatross, Justin, and Michael Ware. *Physics of Light and Optics.* Brigham Young University, Department of Physics, 2015.

[14] Petrik, Marek. "Introduction to Deep Learning." Machine Learning. 20 April. 2020, Durham, New Hampshire.

[15] Short, K. and Garcia R.A. 2006. "Signal Analysis Using the Complex Spectral Phase Evolution (CSPE) Method." AES: *Audio Engineering Society Convention Paper.*

[16] Virtanen, Tuomas, et al. *Computational Analysis of Sound Scenes and Events.* Springer, 2018.

[17] White, Harvey Elliott, and Donald H. White. *Physics and Music: the Science of Musical Sound*. Dover Publications, Inc., 2019.

[18] Zhang, Tong, and C.-C. Jay Kuo. "Content-Based Classification and Retrieval of Audio." *Advanced Signal Processing Algorithms, Architectures, and Implementations VIII*, 2 Oct. 1998, pp. 432–443., doi:10.1117/12.325703.