MACHINE LEARNING APPLICATIONS IN DIGITAL PATHOLOGY

TESTING METHODS

Leo Berman
Albert Bulik
Yuan Nghiem

1 Introduction

We are training a machine learning (ML) model, the purpose of which is to mimic the behavior of a trained breast cancer pathologist. The trained model, which will annotate breast cancer biopsy slides, must be tested against annotations provided by a human pathologist. The model must also communicate its results to the user by means of an intuitive graphical user interface. The full list of functional requirements and design criteria is replicated in §2.

In this document, we will review each criterion and requirement, and describe the testing methodology used to determine whether the criterion or requirement has been satisfied. The document is organized into three sections: statistical testing methods (§3), qualitative testing methods (§4), and quantitative (non-statistical) testing methods (§5).

2 Functional Requirements and Design Criteria

Table 1: Functional Requirements

Priority	Requirement	Metric	Target Value/s
Non-negotiable	Inter-Rater Reliability	Cohen's kappa coefficient $\kappa \in [-1,1]$	$\kappa \geq 0.6$
Non-negotiable	Class-Dependent Performance	F_1 score $\in [0,1]$	$F_1 \geq 0.9$
Non-negotiable	Whole-slide image (WSI) accuracy for each slide	{Pass,Fail}	Pass
Non-negotiable	Semi-Supervised Model	Utilization of unlabeled images in final model (%)	100%
Non-negotiable	Graphical user interface (GUI) shows reference annotations	{Pass,Fail}	Pass
Non-negotiable	GUI displays malignancy probability	{Pass,Fail}	Pass
Non-negotiable	GUI displays reference annotations	{Pass,Fail}	Pass
Negotiable	Time to train model	T_{train} (days)	$T_{train} < 3$ days

Table 2: Design Criteria

Priority	Constraint	Metric	Target/Limit
Non-negotiable	Programming language must be Python	{Pass,Fail}	Pass
Non-negotiable	Data availability	Available training data (bytes)	1.2 Terabytes
Non-Negotiable	Adherence to ISIP standard for Python code formatting	{Pass,Fail}	Pass
Negotiable	Adherence to PEP8 Python standard for code formatting	{Pass,Fail}	Pass

3 Statistical Testing Methods

Rows 1 and 2 in Table 1 concern statistical methods. For each requirement (inter-rater reliability and class-dependent performance), model outputs are compared against the control group of pathologist annotations, observations are aggregated into a single statistic and each statistic is validated against the respective target value.

3.1 Inter-Rater Reliability

Inter-rater reliability is a correlation coefficient describing agreement between two raters. Raters are independent parties who sort, or assign numbers or labels, to data. In our case, the two raters are the trained ML model and the human pathologists who annotate biopsy slides at the Fox Chase Cancer Center pathology lab.

Inter-rater reliability is analogous to the population correlation coefficient

$$\rho_{xy} = \frac{\sigma_{xy}}{\sigma_x \sigma_y} \in [-1,1].$$

Equation 1: Population correlation coefficient, where σ_{xy} is the covariance between populations x and y, σ_x is the standard deviation of x and σ_y is the standard deviation of y

Indeed, if raters were assigning numbers to data, then we would use the Pearson correlation coefficient

$$r_{xy} = \frac{s_{xy}}{s_x s_y} \in [-1,1].$$

Equation 2: Pearson/sample correlation coefficient, for sample covariance s_{xy}, and sample standard deviations s_x and s_y.

However, because we are assigning unordered nominal values (i.e., *labels*) to frames of biopsy slides, which have no immediate numeric interpretation, we use Cohen's kappa coefficient κ (Equation 3). ρ, r and κ all map to real numbers $\in [-1,1]$, where a value of 1 indicates perfect agreement between populations/samples/raters, -1 indicates perfect disagreement, and 0 indicates no correlation.

$$\kappa = \frac{p_o - p_e}{1 - p_e} \in [-1,1]$$

Equation 3: Cohen's kappa coefficient calculation.

The quantities p_o and p_e in Equation 3 are, respectively, the probability of *observed* agreement between raters, and the probability of *expected* agreement between raters. Both p_o and p_e are calculated via relative frequency. To illustrate how they're calculated, we'll use a simple 2-class example of patients diagnosed as either *healthy* or *sick* by two different doctors.

Out of 30 patients, doctor 1 diagnoses 17 as sick and 13 as healthy, and doctor 2 diagnoses 15 as sick and 15 as healthy. Both agree that 13 of the patients are sick and 11 are healthy, but disagree among the remainder of patients. The confusion matrix for their diagnoses is displayed in Table 3.

Table 3: Confusion matrix for 2-class example; green suggests agreement between the raters and red suggests disagreement. Rows correspond to diagnoses made by doctor 1 and columns correspond to diagnoses made by doctor 2.

		Doctor 2		
		Sick	Healthy	Sums across rows (doctor 1's diagnoses)
Doctor 1	Sick	13	4	17
	Healthy	2	11	13
	Sums across columns (doctor 2's diagnoses)	15	15	30

Note that diagonals in Table 3 indicate instances where both doctors agreed on a diagnosis. The observed probability of agreement between doctors 1 and 2, p_o, is the relative frequency of the sum along diagonals (Equation 4).

$$p_o = \frac{(13 + 11)\ [patients]}{30\ [total\ patients]} = \frac{24}{30} = 0.8.$$

Equation 4: Observed probability of agreement p_o.

In contrast, if we considered both doctors' diagnoses to be independent of one other (if, for example, we didn't know which patients received which diagnosis), then our expected probability of agreement p_e is the sum of products in Equation 5.

$$\begin{aligned} p_e &= p(both\ doctors\ agree|sick\ patient) + p(both\ doctors\ agree|healthy\ patient) \\ &= p(sick\ diagnosis|doctor_1) \cdot p(sick\ diagnosis|doctor_2) \\ &\quad + p(healthy\ diagnosis|doctor_1) \cdot p(healthy\ diagnosis|doctor_2) \\ &= \left(\frac{17}{30}\right)\left(\frac{15}{30}\right) + \left(\frac{13}{30}\right)\left(\frac{15}{30}\right) = 0.283 + 0.217 = 0.5 \end{aligned}$$

Equation 5: Expected probability of agreement p_e.

Therefore, the inter-rater reliability between doctors 1 and 2, according to Cohen's kappa coefficient, is calculated as follows (Equation 6).

$$\kappa = \frac{p_o - p_e}{1 - p_e} = \frac{0.8 - 0.5}{1 - 0.5} = \frac{0.3}{0.5} = 0.6 = 60\%.$$

Equation 6: Inter-rater reliability calculated by Cohen's kappa coefficient for 2-class example.

This implies there is 60% correlation between diagnoses between doctors 1 and 2 (not to be interpreted that they agree 60% of the time, but that in our observations, they agreed 60% more than random chance; Equation 6 applies min-max normalization to the probability of agreement p).

When calculating Cohen's kappa coefficient for our model, we apply a similar methodology. First, a biopsy slide is input to the trained model; the slide is segmented into frames and the model assigns labels to each frame from the list of cancer classifications (unlabeled, background, normal, null, artifact, non-neoplastic tissue, inflammation, suspicious tissue, ductal carcinoma in situ, and invasive ductal carcinoma; 10 possible labels in all).

Secondly, the model label is compared to the human pathologist's label for tissue in the frame area. For instance, our model may believe a biopsy slide frame to contain non-neoplastic tissue, but the annotating pathologist may have indicated invasive ductal carcinoma tissue in the same frame area. The confusion matrix is constructed by summing every instance of every permutation of labels between the model and annotating pathologist. That is, a matrix similar to Table 3 is constructed, containing 10 columns and 10 rows for each label, instead of a 2x2 matrix.

$$p_o = \frac{1}{N}\sum_{k=0}^{N} n_{kk}\,, \qquad p_e = \frac{1}{N^2}\sum_{k=0}^{N}\left[\left(\sum_{j=0}^{N} n_{k,j}\right)\left(\sum_{i=0}^{N} n_{i,k}\right)\right]$$

Equation 7: General form of probabilities in Cohen's kappa coefficient (Equation 3), where $n_{i,j}$ is the element n at index i, j of the confusion matrix, and N is the number of labels or classes.

Lastly, we apply the same calculations in Equation 4, Equation 5, and Equation 6 to determine our model's inter-rater reliability. (The general form of p_o and p_e, for any size confusion matrix, is provided by Equation 7.) Our target value is $\kappa \geq 0.6$. This target was extracted from a similar study on whole-slide image classification [1]. The study used deep learning models and whole-slide image data augmented with molecular markers. The experiment's three trials yielded kappa coefficients of 0.54, 0.61, and 0.58.

3.2 Class-Dependent Performance

An F-score is a measure of the predictive ability of a machine learning model. An F_1 score is the harmonic mean of precision and recall (Equation 8).

$$F_1 = 2\frac{precision \cdot recall}{precision + recall}$$

Equation 8: F_1 score calculation.

To understand why an F_1 score is useful, we need to understand the meaning of precision and recall.

Consider a 2-class example where a model is used to predict whether a patient is healthy or sick. *Recall* is the ratio of true positives to the sum of true positives + false negatives. In our 2-class example, this would be the ratio

$$Recall = \frac{true\ positives}{true\ positives + false\ negatives} = \frac{true\ sick\ cases}{true\ sick\ cases + false\ healthy\ cases}.$$

Equation 9: Recall in a 2-class example.

Recall indicates the percentage of cases correctly identified as sick, with respect to all sick cases in the dataset.

In contrast, precision is the ratio of true positives to the sum of true positives + false positives. In our 2-class example, this would be the ratio

$$Precision = \frac{true\ positives}{true\ positives + false\ positives} = \frac{true\ sick\ cases}{true\ sick\ cases + false\ sick\ cases}.$$

Equation 10: Precision in a 2-class example.

Precision indicates the percentage of cases correctly identified as sick, with respect to all cases believed by the model to be sick. Precision and recall for the 2-class example are illustrated in Figure 1.

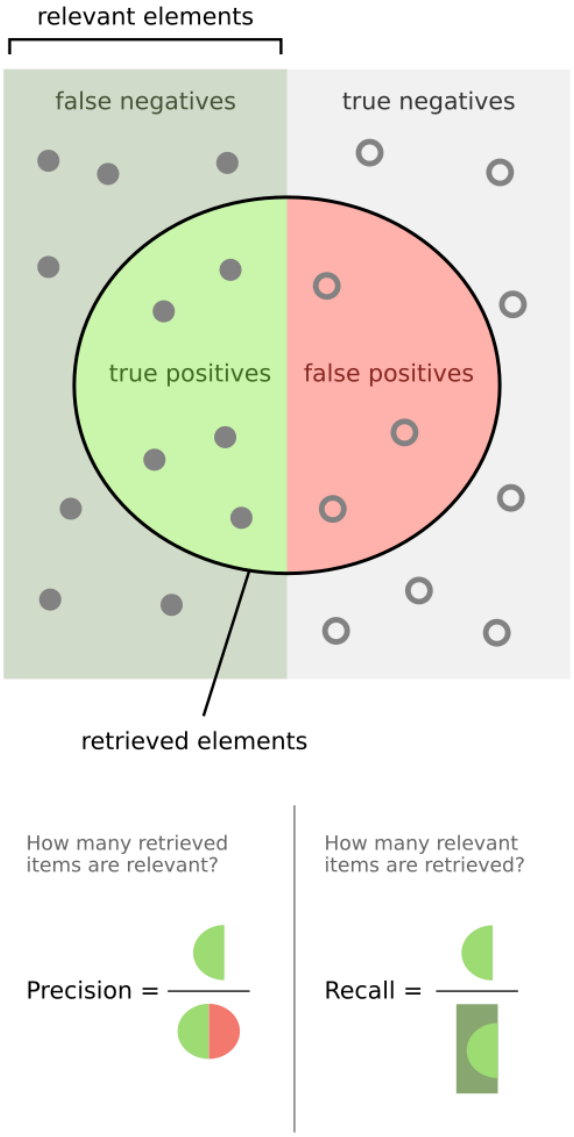

Figure 1: Precision and recall in Venn diagrams.

The two metrics, precision and recall, complement each other. If a model only predicted patients to be sick (i.e., a model that never predicts a patient to be healthy), then the model would have *high recall* but *low precision*. Conversely, if a model correctly predicts only one patient to be sick, but makes no other sick diagnoses (i.e., predicts all other patients to be healthy, regardless if they're healthy or sick), then the model would have *low recall* but *high precision*. Recall concerns the span of the model's 'true'

predictions over the input dataset, and precision concerns the model's accuracy *within* its 'true' predictions.

An F_1 score balances precision and recall. Both quantities must be significant for an F_1 score to be significant. If either precision or recall approaches 0 then the corresponding F_1 score also approaches 0.

In the prior example, we used two classes (sick and healthy). For multi-class cases, we calculate precision and recall for each class, and subsequently average F_1 scores over all classes [2]. For example, if we were using the non-neoplastic (NNeo) label as reference, the true positive rate would be the number of correctly classified NNeo frames, and the false negative rate would be the sum of NNeo frames incorrectly classified by another label. These would be used to calculate recall for NNeo, which would in turn be used to calculate the F_1 score for NNeo, which would be averaged with F_1 scores for other classes.

We can derive true positive (TP), false positive (FP), false negative (FN), and true negative (TN) statistics for every class from the confusion matrix of all classes (Table 4).

Table 4: Example confusion matrix.

		Pathologist (True)										
	Frame Labels	Unlab	Bckg	Norm	Null	Artf	Nneo	Infl	Susp	Indc	Dcis	Sums across rows
Model (Predicted)	Unlab	20	1	4	1	7	6	7	0	4	6	56
	Bckg	6	30	1	3	5	2	4	7	3	3	64
	Norm	3	2	29	6	0	1	1	3	1	2	48
	Null	3	0	4	24	3	4	6	3	5	0	52
	Artf	5	6	4	7	29	3	6	2	3	3	68
	Nneo	1	6	6	1	3	21	5	3	3	5	54
	Infl	6	3	7	4	0	6	26	7	1	2	62
	Susp	6	4	6	4	5	3	6	22	0	6	62
	Indc	5	5	2	6	0	3	6	0	28	0	55
	Dcis	5	0	6	6	7	7	4	7	5	24	71
	Sums across columns	60	57	69	62	59	56	71	54	53	51	592

To illustrate, suppose we were deriving statistics for the NNeo class. The TP rate of NNeo is its position on the diagonal of the confusion matrix, highlighted in green in Table 4; the FN rate of NNeo is the sum of all elements highlighted in orange; the FP rate of NNeo is the sum of all elements highlighted in red; and the TN rate of NNeo is the sum of all elements highlighted in blue. Table 4 simplifies to Table 5.

Table 5: Simplified confusion matrix for nneo case. Green indicates TP; red indicates FP; orange indicates FN; blue indicates TN.

		Pathologist (True)		
	Frame Labels	NNeo	Not NNeo	Sums across rows
Model (Predicted)	NNeo	21	33	TP+FP = 54
	Not NNeo	35	503	FN+TN = 538
	Sums across columns	TP+FN = 56	FP+TN = 536	592

The example NNeo F_1 score for Table 5 is calculated as follows (Equation 11).

$$Precision = \frac{true\ positives}{true\ positives + false\ positives} = \frac{21}{21 + 33} = 0.389$$

$$Recall = \frac{true\ positives}{true\ positives + false\ negatives} = \frac{21}{21 + 35} = 0.375$$

$$F_1 = 2\frac{precision \cdot recall}{precision + recall} = 2\frac{(0.389)(0.375)}{0.389 + 0.375} = 0.382$$

Equation 11: NNeo F_1 score calculation.

Our target F_1 score is 0.90. This target value was extracted from a similar 2019 study employing deep-learning models to classify whole-slide images [3]. The F_1 score yielded by the study was 0.90.

4 Qualitative Testing Methods

4.1 Whole Slide Image (WSI) Accuracy

The purpose of WSI accuracy is to allow us to compare our model to general machine learning models in industry. Since we are mostly focusing on detecting location and precision of patches of cancer, WSI accuracy isn't the most important metric, but it is important to establish a baseline in comparison with other systems with different areas of focus. While the implementation of this feature satisfies the pass/fail requirement, we will be cross-referencing it with industry standards to benchmark our system later in the process which falls outside the scope of this project. The pass/fail metric for this functional requirement will be the ability to generate predictions for whole slides based on patch level classifiers. We will take a series of slides with annotations from our model and by categorizing these annotations graphically we will be able to ascertain whether our system successfully processes WSI predictions.

4.2 GUI Requirements

The three GUI requirements will be pass/fail based on their presence on our graphical representation of the report. The reference and hypothesis annotations will be graphically evaluated, and we will compare the confidences to spreadsheets containing our information. Overall, the estimate of our GUIs functionality will be based on the ease of access of our program and the ability of a person unfamiliar with our program to easily parse through and understand the data [4].

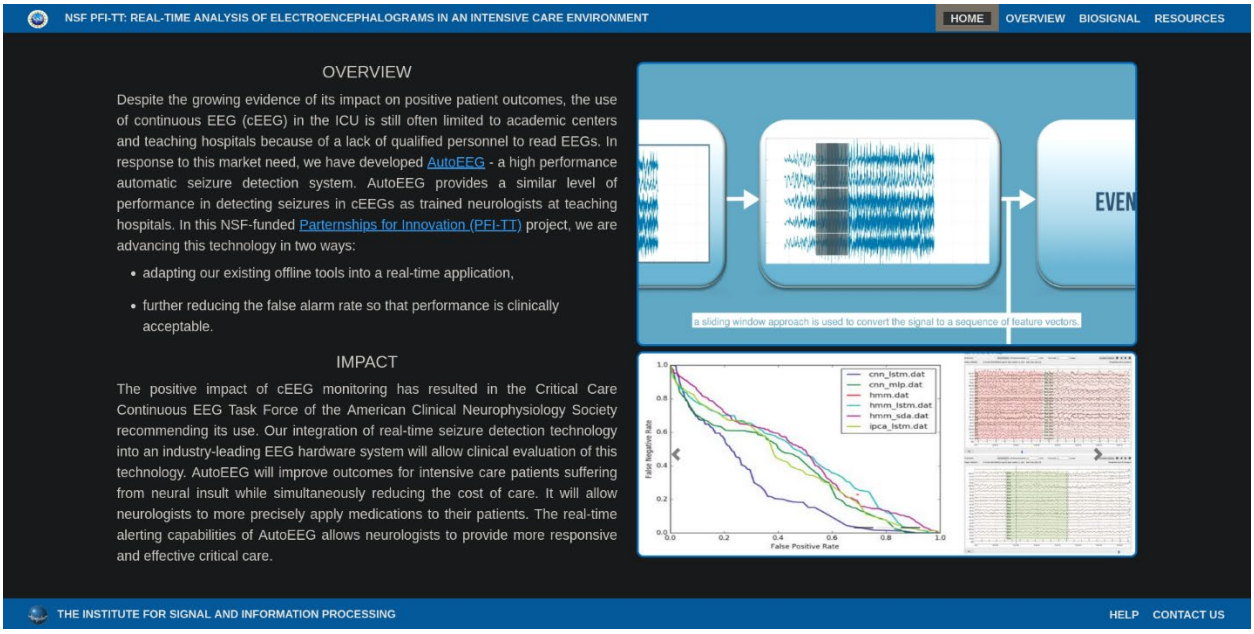

Figure 2: Example Demo

4.3 Code Standards and Language Requirements

The formatting of the code will be evaluated by Dr. Picone in combination with a graduate student, Claudia Dumitrescu. They will be using the ISIP bluebook standard [5] and existing sample code. Alongside this, we will be evaluating naming conventions, commenting style, and overall code style by cross referencing with the most recent1 PEP8 style guide [6] and other example code.

```
# Wrong:

# Arguments on first line forbidden when not using vertical alignment.
foo = long_function_name(var_one, var_two,
  var_three, var_four)

# Further indentation required as indentation is not distinguishable.
def long_function_name(
  var_one, var_two, var_three,
  var_four):
  print(var_one)
```

```
# Correct:

# Aligned with opening delimiter.
foo = long_function_name(var_one, var_two,
                         var_three, var_four)

# Add 4 spaces (an extra level of indentation) to distinguish arguments from the rest.
def long_function_name(
    var_one, var_two, var_three,
    var_four):
  print(var_one)

# Hanging indents should add a level.
foo = long_function_name(
  var_one, var_two,
  var_three, var_four)
```

Figure 3: PEP8 Code Example

5 Quantitative Non-Statistical Testing Methods

5.1 Unsupervised Learning Model

Currently we utilize a supervised learning model which leverages data that has been annotated by a pathologist. This creates a bottleneck where we are limited by the amount of data that has been processed by pathologists. While our model does take time to process training data, the time it takes for a pathologist to annotate these slides is much greater. This is where an unsupervised model comes in. An unsupervised model takes slides that have not been annotated and extracts features that it thinks are important by leveraging techniques based on the variance of the images. Once it's categorized these features, it will utilize a smaller subset of labeled data to classify the different subsets of these features. This will allow us to utilize all the images we have even if a pathologist has not touched the data. It's possible that by using a larger amount of information we can

build a more generalized model. While we will be testing these results to see if there is an improvement, it's likely that extensive testing will fall outside the scope of this project, but we will be testing our ability to utilize this data by measuring the total amount of utilization of the data. We utilize catching exceptions and logs to see which data was used successfully and which failed.

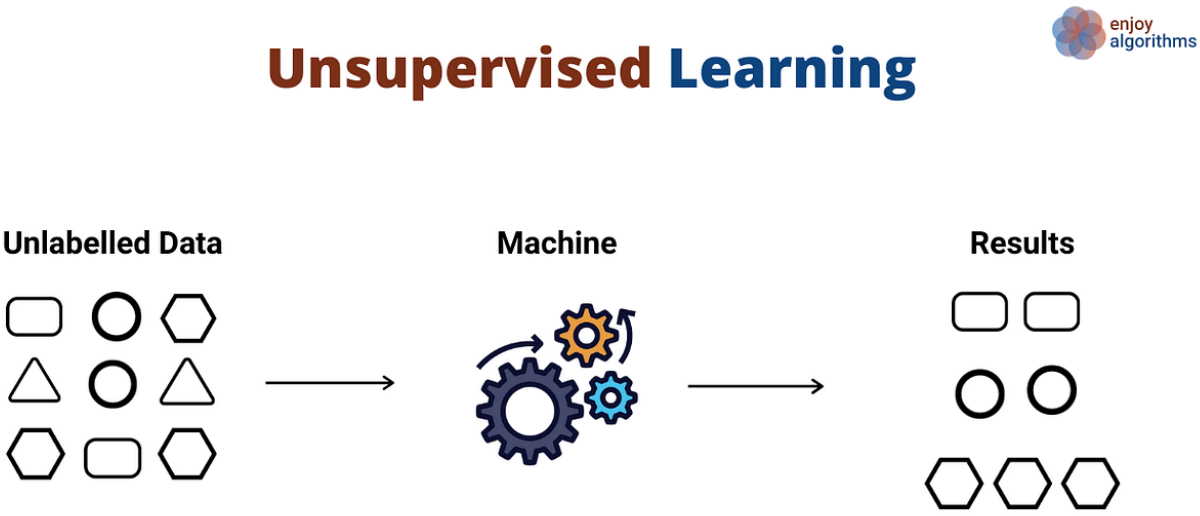

Figure 4: Unsupervised Learning Example

5.2 Time to Train & Data Limitations

Time to Train is a self-imposed limitation of less than 3 days due to the scope of our project. To run proper testing trials to refine our model we must limit the amount of time our trials take. With our data limitation of 1.2 Terabytes of annotated data [7], we need to limit our trial time to less than 3 days which gives us enough headroom to find a sweet spot where the model is accurate enough that the trials are significant but have a short enough trial time that we can try different parameters out. While we can run many trials in parallel due to the high performance compute (HPC) cluster [8], we will eventually be limited since this compute cluster is being utilized for other classes and other projects. We will, of course, measure this by timing our models by using the output of results from Slurm, a job manager on the cluster that covers metrics like CPU, GPU, and real time [9].

6 References

[1] V. M. T. d. J. N. S. M. O. G. M. H. E. D. J. P. W. P. P. J. v. D. &. M. V. Suzanne C. Wetstein, "Deep learning-based breast cancer grading and survival analysis on whole-slide histopathology images," Nature, 6 September 2022. [Online]. Available: https://www.nature.com/articles/s41598-022-19112-9. [Accessed 20 October 2024].

[2] baeldung, "https://www.baeldung.com/cs/multi-class-f1-score," F-1 Score for Multi-Class Classification, 18 March 2024. [Online]. Available: https://www.baeldung.com/cs/multi-class-f1-score. [Accessed 20 October 2024].

[3] "Multi-Level Batch Normalization In Deep Networks For Invasive Ductal Carcinoma Cell Discrimination In Histopathology Images," arxiv, 11 January 2019 . [Online]. Available: https://arxiv.org/abs/1901.03684. [Accessed 20 October 2024].

[4] "NSF PFI-TT: REAL-TIME ANALYSIS OF ELECTROENCEPHALOGRAMS IN AN INTENSIVE CARE ENVIRONMENT," ISIP, [Online]. Available: https://isip.piconepress.com/projects/nsf_pfi_tt/. [Accessed 20 October2 2024].

[5] P. Dr, "ISIP Standards," ISIP, [Online]. Available: https://isip.piconepress.com/projects/speech/software/tutorials/general/. [Accessed 20 October 2024].

[6] B. W. <. a. p. A. C. <. a. g. Guido van Rossum <guido at python.org>, "PEP 8 – Style Guide for Python Code," python.org, 01 August 2013. [Online]. Available: https://peps.python.org/pep-0008/. [Accessed 10 October 2024].

[7] Z. W. B. D. I. O. J. P. J. Simons, "The Temple University Hospital DPATH Corpus:," Temple University Hospital DPATH Corpus, 15 January 2021. [Online]. Available: https://view.officeapps.live.com/op/view.aspx?src=https%3A%2F%2Fisip.piconepress.com%2Fpublications%2Freports%2F2021%2Ftuh_dpath%2Fannotations%2Fannotation_guidelines_v10.docx&wdOrigin=BROWSELINK. [Accessed 1 October 2024].

[8] "Introduction to Parallelism," 2018. [Online]. Available: https://cwant.github.io/hpc-beyond/21-introduction-to-parallelism/index.html. [Accessed 20 October 2024].

[9] Slurm Workload Manager, 6 August 2021. [Online]. Available: https://slurm.schedmd.com/overview.html . [Accessed 20 October 2024].