Weak supervision team project
Due Friday, December 8th 2023

Introduction

In this assignment, you will investigate weakly supervised methods for analyzing pathology images to predict two-year survival of patients diagnosed with brain tumors.

Teams of two will work together to tune the provided baseline networks. Each member will select one of the topics described below to investigate. Teams will submit a single jointly written report describing their methods, results, and insights from their analysis.

Background

This project addressed the prediction of patient clinical outcomes from whole-slide pathology images (WSIs). Predicting patient-level labels like diagnosis or clinical outcome is difficult as WSIs are very large, with each image typically containing several billion pixels. As we discussed in class, weakly-supervised and multi-instance learning are popular approaches to addressing this problem [1, 2]. These methods treat a WSI as a collection of tiles or *instances* and use various mechanisms to identify the most salient instances and aggregate their contributions for patient-level prediction.

The aim of this project is to predict the two-year survival of diffuse glioma brain tumors using whole-slide images from the Lower Grade Glioma (LGG) and Glioblastoma (GBM) projects of The Cancer Genome Atlas (TCGA). Diffuse gliomas are a hetergeneous disease encompassing tumors with widely varying molecular characteristics, histologic patterns, and clinical outcomes. Multiple studies have examined the relationship between histology and clinical outcomes using machine-learning approaches [3]. In this project, you will perform binary classification of those subjects who survive beyond two years, and those who expire prior to two years. Patients who are right-censored at less than 2 years are not present in the dataset. See the lecture from November 6th for a review of this topic.

Data

The data archive contains train, validation, and test folders containing images from 417, 53, and 53 patients respectively. Each patient is represented by a single TensorFlow record (.tfr) file that contains extracted features, clinical outcome labels, and other patient and image metadata. Features were extracted by tiling slides using 224×224 pixel non-overlapping tiles at 100X magnification. An EfficientNetV2S model was used to extract 1280-dimensional features from each tile [4]. A label value of 1 indicates survival beyond two years.

The data can be downloaded here: tcga_glioma.zip.

Move this file to your Google Drive, and replace the path in the dataset creation cell within the provided Jupyter notebook.

Software

The provided code contains functions for generating and reading datasets, and for measuring the performance of models. ***Note:*** for automated downloading of the Google Drive hosted data, use dataset_hosted.py (change this filename to dataset.py). For working with a downloaded .zip archive, use dataset.py.

- **/code/dataset.py** unpack locally saved data .zip archive and prepare dataset objects.
- **/code/dataset_hosted.py** download data .zip archive and prepare dataset objects.
- **/code/metrics.py** performance metrics.
- **/code/reader.py** used by dataset.py for reading .tfr files.
- **/code/transforms.py** used by reader.py for generating spatial format (see topics below).

Step 1: Tune the baseline models

The notebook defines two baseline models: 1. A simple model based on global pooling and 2. An attention model based on the paper presented in class [1]. The notebook settings currently train these models for just a few epochs to demonstrate how to train them. In this task, you will use the training and validation datasets to explore the convergence of these models and to generate frozen baselines for testing and comparison below. **Work together and use only one set of baseline models throughout the report.**

1. Use the validation and training datasets to explore the convergence of each baseline model. Identify convergence criteria for each model (such as a manually selected number of epochs, or successive change in loss). Apply these criteria and save the models at the convergence point you defined. In this step, you are free to adjust the gradient optimization parameters.
2. For each baseline model, generate a convergence plot that shows training and validation loss versus epoch. Indicate the convergence point on the plot. Document your convergence criteria in the written report.
3. Apply the frozen models to the test set using your chosen performance metrics. Include a justification for the selection of metrics in your written report.

Step 2: Topic investigation

Each team member should select one topic from the list below to investigate. (clearly indicate which topics are selected in the written report). Topics are organized by difficulty. For each topic, follow the topic-specific reporting requirements including comparing to the relevant baseline on the testing data.

Uncertainty (basic difficulty). Use Monte Carlo Dropout to explore the relationship between uncertainty and accuracy. Following the analysis from [5], reproduce Figure 1d showing the uncertainty distribution for correct and incorrect predictions, and Figure 3b showing the accuracy when analyzing the least uncertain to most uncertain samples. For each baseline model, ensure the terminal dense layer has dropout (10% dropout rate) and train to convergence. To estimate uncertainty σ_i for the test samples, apply the trained model 100 times (use training mode to apply dropout i.e. one set of predictions for the entire test dataset is `[model(x, train=True) for (x,y) in test]`). To generate the predictions, apply the model once to each sample with `train=False`. Replicate Figure 1d by plotting the histograms of σ_i for the correct (blue) and incorrect (green) test predictions. To replicate Figure 3b, first sort the test samples by increasing uncertainty σ_i. Calculate classification performance for datasets in increments of 5, starting with the 5 most certain samples, then the 10 most certain samples, until the entire dataset is included. Display the performance metric on the y-axis, and dataset size on the x-axis. It is not necessary to replicate the confidence intervals or the random referral line. Repeat this with both baseline models.

Augmentation (moderate difficulty). Augmentation is typically applied to images where operations like rotation, mirror flip, contrast, and brightness adjustment can be applied. In most weakly supervised models, images are transformed to *features* as a pre-processing step, and augmentation is not used in training since feature extraction is time-intensive. In this topic, you will explore the effectiveness of applying augmentations in feature space. First, implement augmentation operations that can be applied to the features, and use these in training the provided simple and attention models. Suggestions for augmentations include adding noise (proportional to feature variance) or dropping a fraction of instances. Measure the performance of models trained with augmentations on your testing data. Second, use your trained models and augmentations to apply test time augmentation on the testing data. Compare performance to the baselines. Provide a clear description of the augmentations you implemented, how you selected augmentation parameters, and offer an interpretation of the results.

Submission

Each team will submit 1 written report consisting of a single document in PDF format. This written portion of the report should not exceed 2 pages, excluding figures. All performance data should be presented as tables within the written report. An appendix should be appended to the end of the report including in this order: 1. Figures 2. Code. All figures and tables must be referenced in the written report. The report must include the names of both team members and their selected topics. **There will be a 3% deduction for each instance where these directions are not followed.**

References

[1] Ming Y Lu et al. “Data-efficient and weakly supervised computational pathology on whole-slide images”. In: *Nature biomedical engineering* 5.6 (2021), pp. 555–570.

[2] Gabriele Campanella et al. “Clinical-grade computational pathology using weakly supervised deep learning on whole slide images”. In: *Nature medicine* 25.8 (2019), pp. 1301–1309.

[3] Pooya Mobadersany et al. “Predicting cancer outcomes from histology and genomics using convolutional networks”. In: *Proceedings of the National Academy of Sciences* 115.13 (2018), E2970–E2979.

[4] Mingxing Tan and Quoc V. Le. “EfficientNetV2: Smaller Models and Faster Training”. In: *CoRR* abs/2104.00298 (2021). arXiv: 2104.00298. URL: https://arxiv.org/abs/2104.00298.

[5] Christian Leibig et al. “Leveraging uncertainty information from deep neural networks for disease detection”. In: *Scientific reports* 7.1 (2017), p. 17816.