CMSC 35440 Machine Learning in Biology and Medicine

Homework 2: Cancer Site Classification from Histopathology Features

Released: Jan 26,2026

Due: Feb 2,2026 at 11:59 PM Chicago Time on Gradescope

In this homework, you'll explore embeddings of histopathology image features for three different cancer types.

This assignment is inspired by the Cancer Site Classification project by Renyu Zhang, which demonstrates how deep learning can be used to classify cancer types from histopathology whole slide images (WSI). The original project uses Inception-V3 to extract features from image tiles and trains a classifier to distinguish between colorectal adenocarcinoma (COAD) and uterine corpus endometrial carcinoma (UCEC). However, that approach requires GPUs, downloading large slide files, and significant computational resources.

In this homework, we adapt the same core concept—cancer site classification from histopathology—but make it accessible for homework by using **pre-extracted features** from a modern vision transformer (UNI2-h). This allows us to focus on the downstream analysis: understanding how different cancer types cluster in feature space, visualizing embeddings, and analyzing model behavior—all without requiring GPUs.

We'll work with three distinct cancer types:

- **Diffuse large B-cell lymphoma (DLBC)**: The most common type of non-Hodgkin lymphoma, originating from B-cells in the lymphatic system.
- **Cholangiocarcinoma (CHOL)**: A rare but aggressive cancer that originates in the bile ducts, which carry bile from the liver to the small intestine.
- **Uveal melanoma (UVM)**: A rare cancer that develops in the uvea (the middle layer of the eye), which includes the iris, ciliary body, and choroid.

These three cancers affect very different organ systems (lymphatic, digestive, and ocular), making them an interesting test case for whether histopathology features can capture organ-specific morphological patterns. Histopathology slides from all three can present diagnostic challenges, especially in cases with unusual morphological features or when tissue samples are limited.

In this homework, you'll work with a third party embedding model for whole slide images (WSI) using UNI2-h, a state-of-the-art vision transformer model trained on histopathology images. UNI2-h was trained using self-supervised learning on a large collection of histopathology images and can extract meaningful representations from tissue slides. You may want to think of these as pre-extracted features. The embeddings we will use are pre-computed embeddings from the MahmoodLab/UNI2-h-features dataset on Hugging Face. By using precomputed embeddings, we avoid the need for GPUs and can focus on the downstream analysis: clustering, visualization, and understanding model behavior. This mirrors real-world scenarios where embeddings might be done once and reused for multiple analyses. Last, you'll practice a vital step for biomedical machine learning: expert review. Before these models can ever be deployed in real patient settings, the- must undergo rigorous review. In the US, any medical products intended for patient usage must be approved by the FDA. An excellent historical case of demonstrating why we need such review is Thalidomide in the late 1950s. It was originally marketed in Europe as a treatment for morning sickness, especially during pregnancy. However, the drug was blocked in the US by an expert reviewer at the FDA, Dr. Frances Kelsey (a UChicago MD/PhD alum!), who was concerned over the lack of evidence concerning the drug's safety. She was of course right to be concerned, as Thalidomide was shown to cause severe birth defects, leading to its removal from European markets. Suffice to say, expert review is crucial to patient safety, especially as we dive into this new age of AI/ML in medicine.

Instructions

1. Download and open this starter notebook. **No need for any GPUs for this homework** - we'll use pre-extracted features.
2. Download the pre-extracted features from Hugging Face. We'll use features from three TCGA projects: `TCGA-DLBC` (diffuse large B-cell lymphoma), `TCGA-CHOL` (cholangiocarcinoma), and `TCGA-UVM` (uveal melanoma).

- The features are available at: https://huggingface.co/datasets/MahmoodLab/UNI2-h-features/tree/main/TCGA
- You'll need to download:
  - `TCGA-DLBC.tar.gz` (~1.5 GB)
  - `TCGA-CHOL.tar.gz` (~4.5 GB)
  - `TCGA-UVM.tar.gz` (~4.5 GB)
- **Total download size: ~10.5 GB** - These are smaller datasets suitable for homework assignments.
- **IMPORTANT**: This dataset is gated. You must:
  1. Create a Hugging Face account with your institutional email (@uchicago.edu) at https://huggingface.co/join
  2. Set your institutional email as your primary email and verify it

    3. Accept the dataset terms at https://huggingface.co/datasets/MahmoodLab/UNI2-h-features (CC-BY-NC-ND 4.0 license for non-commercial research use)
    4. Get your access token from https://huggingface.co/settings/tokens
    5. Login via command line: `huggingface-cli login` (paste your token when prompted) OR use `login(token="your_token")` in Python
- After extracting, each tar.gz contains HDF5 (`.h5`) files for individual slides. Each `.h5` file contains:
    - `features` : 1 x num_patches x 1536 (UNI2-h feature vectors for all patches in the slide)
    - `coords` : 1 x num_patches x 2 (coordinates of each patch)

1. Load and aggregate features per patient. Each patient may have multiple slides/tiles, so we need to create one embedding per patient.

- **The exact aggregation method is up to you** (mean, max, or other pooling strategies).
- Include in your writeup a brief description and justification of the aggregation method you used.
- Tips:
    - The feature files are typically organized by case/sample ID. You may need to parse filenames or use metadata to group features by patient.
    - Each .h5 file contains patch-level features (1536-dimensional vectors) for all patches in a single slide. You should aggregate patch-level features to slide-level (e.g., mean pooling), then aggregate slides to patient-level.
    - The TCGA case ID format is typically `TCGA-XX-XXXX` where the first part identifies the patient.

1. Cluster your patient embeddings to 3 clusters. We're trying to derive a model which can distinguish the different cancer types. Simple `KMeans` from `sklearn` is fine for this. Derive cluster IDs for each patient.

- **Note**: KMeans assigns arbitrary cluster IDs (0, 1, 2). To evaluate performance, you'll need to determine which cluster corresponds to which cancer type (e.g., by majority voting within each cluster).

1. Use PCA to reduce your embeddings to 2 dimensions for visualization.
2. Create multiple visualizations:

- **PCA Visualization**: A scatter plot where the color of the points differ by the TCGA project (`DLBC` , `CHOL` , `UVM`) and the shape of the points differ by the cluster ID assigned by clustering.
- **t-SNE or UMAP Visualization**: Create an alternative 2D embedding using t-SNE or UMAP (from `sklearn.manifold` or `umap` library) and compare it to PCA. How do the two visualizations differ?

- **Confusion Matrix**: Create a confusion matrix showing how well the clusters align with the true cancer type labels.
- **Cluster Analysis**: Create bar plots or other visualizations showing the distribution of cancer types within each cluster.

1. Consider these questions (you should probably address some of these in your writeup):

- In the 2D projection of the embeddings, is there a natural decision boundary you would be able to draw to classify the different cancer types?
- Are there patients which are misclassified by this decision boundary?
- How close is clustering to this decision boundary?
- Are there patients which are misclassified by clustering?
- With three cancer types, how well does 3-cluster KMeans separate them? Are some cancer types easier to distinguish than others?
- Are misclassifications by the model (either clustering or the imagined decision boundary) actually mislabeled data? For example, if a patient is labeled as lymphoma but the model thinks it's cholangiocarcinoma, is the model right or is the label right?
- How do PCA and t-SNE/UMAP visualizations compare? Which one better separates the cancer types?
- **If you achieve perfect (100%) clustering accuracy**: Why do you think the model achieved perfect separation? What properties of these three cancer types make them easy to distinguish? Would you expect similar results with more similar cancer types (e.g., different subtypes of lung cancer)?

1. Review histopathology images:

- We'll use another data modality to understand what the model is "seeing". Histopathology slides for TCGA cases can be accessed through the **GDC Data Portal** (https://portal.gdc.cancer.gov).
- Histopathology slides show tissue structure at the microscopic level. DLBC typically shows large B-cells with characteristic nuclear features, cholangiocarcinoma shows glandular structures in bile duct tissue, and uveal melanoma shows melanocytic cells with characteristic pigmentation patterns.
- **To access slide images**:
  1. Go to https://portal.gdc.cancer.gov
  2. Click "Repository" and filter by "Data Type" → "Slide Image"
  3. Filter by project (TCGA-DLBC, TCGA-CHOL, or TCGA-UVM)
  4. You can also search for specific case IDs in the search bar
  5. Download the slide image (SVS format) and view with a slide viewer like QuPath (free, https://qupath.github.io/) or OpenSlide
- **Alternative (easier)**: Search Google Images for example histopathology images of each cancer type (e.g., "DLBC histopathology", "cholangiocarcinoma histology",

"uveal melanoma pathology") to understand the morphological differences.

- **If you have misclassifications**: Consider whether the tissue morphology of misclassified cases might explain the model's confusion.
- **If you have no misclassifications (100% accuracy)**: Discuss why these three cancer types are morphologically distinct. Example cases in your dataset:
    - DLBC: `TCGA-RQ-A68N`, `TCGA-GS-A9TQ`
    - CHOL: `TCGA-W5-AA2G`, `TCGA-ZH-A8Y6`
    - UVM: `TCGA-V4-A9EX`, `TCGA-YZ-A985`
- In your writeup, briefly describe the morphological differences between the cancer types and discuss why expert review remains important even when models achieve high accuracy.

1. Writeup your work, your writeup should be 1 to 2 pages long, excluding figures. 12pt font, single space, 1 inch margins, letter size paper. Please submit either a PDF or a Word document. Make sure to include the following:

- Your embedding visualizations (PCA, t-SNE/UMAP, confusion matrix, cluster analysis).
- A brief justification for the aggregation method you used.
- A discussion of some of the above questions regarding clustering performance and visualization analysis.
- A discussion of the morphological differences between cancer types (from your image review or research).
- A discussion on the question: Why is review by an expert important? Phrased another way, why should someone with domain knowledge review models?

1. Submit your homework. Make sure to include:
2. Your writeup containing a figure with your embedding visualization.
3. Your notebook with your code.

Required Python Packages

The following packages are needed for this homework:

```
huggingface_hub, h5py, numpy, pandas, scikit-learn,
matplotlib, seaborn, tqdm
```

You can install them with: `pip install huggingface_hub h5py numpy pandas scikit-learn matplotlib seaborn tqdm`

- Office Hour: right after Tuesday and Thursday Class
- Late submission will get points deducted. 1 point for being one day late, meaning even 1 minute late from the deadline would still be considered as a day late. 1 point will be deducted per day.

- Do not generate you report fully LLMs with the typical markdown format. It is easy to detect LLM usage on a written report, and we cannot give you any grade for it.

Your Code

Start your implementation below. You may add as many code and markdown cells as you need.

In [1]:
```
# Install required packages (run this cell first)
%pip install huggingface_hub h5py numpy pandas scikit-learn matplotlib seabo
```

```
Requirement already satisfied: huggingface_hub in /opt/homebrew/lib/python3.
10/site-packages (1.3.2)
Requirement already satisfied: h5py in /opt/homebrew/lib/python3.10/site-pac
kages (3.13.0)
Requirement already satisfied: numpy in /opt/homebrew/lib/python3.10/site-pa
ckages (2.0.2)
Requirement already satisfied: pandas in /opt/homebrew/lib/python3.10/site-p
ackages (2.2.3)
Requirement already satisfied: scikit-learn in /opt/homebrew/lib/python3.10/
site-packages (1.6.1)
Requirement already satisfied: matplotlib in /opt/homebrew/lib/python3.10/si
te-packages (3.10.1)
Requirement already satisfied: seaborn in /opt/homebrew/lib/python3.10/site-
packages (0.13.2)
Requirement already satisfied: tqdm in /opt/homebrew/lib/python3.10/site-pac
kages (4.67.1)
Requirement already satisfied: filelock in /opt/homebrew/lib/python3.10/site
-packages (from huggingface_hub) (3.20.3)
Requirement already satisfied: fsspec>=2023.5.0 in /opt/homebrew/lib/python
3.10/site-packages (from huggingface_hub) (2025.10.0)
Requirement already satisfied: hf-xet<2.0.0,>=1.2.0 in /opt/homebrew/lib/pyt
hon3.10/site-packages (from huggingface_hub) (1.2.0)
Requirement already satisfied: httpx<1,>=0.23.0 in /opt/homebrew/lib/python
3.10/site-packages (from huggingface_hub) (0.28.1)
Requirement already satisfied: packaging>=20.9 in /Users/summerann/Library/P
ython/3.10/lib/python/site-packages (from huggingface_hub) (24.2)
Requirement already satisfied: pyyaml>=5.1 in /opt/homebrew/lib/python3.10/s
ite-packages (from huggingface_hub) (6.0.3)
Requirement already satisfied: shellingham in /opt/homebrew/lib/python3.10/s
ite-packages (from huggingface_hub) (1.5.4)
Requirement already satisfied: typer-slim in /opt/homebrew/lib/python3.10/si
te-packages (from huggingface_hub) (0.21.1)
Requirement already satisfied: typing-extensions>=4.1.0 in /Users/summerann/
Library/Python/3.10/lib/python/site-packages (from huggingface_hub) (4.12.2)
Requirement already satisfied: python-dateutil>=2.8.2 in /Users/summerann/Li
brary/Python/3.10/lib/python/site-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in /opt/homebrew/lib/python3.10/
site-packages (from pandas) (2025.1)
Requirement already satisfied: tzdata>=2022.7 in /opt/homebrew/lib/python3.1
0/site-packages (from pandas) (2025.1)
Requirement already satisfied: scipy>=1.6.0 in /opt/homebrew/lib/python3.10/
site-packages (from scikit-learn) (1.15.2)
Requirement already satisfied: joblib>=1.2.0 in /opt/homebrew/lib/python3.1
0/site-packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in /opt/homebrew/lib/pyt
hon3.10/site-packages (from scikit-learn) (3.5.0)
Requirement already satisfied: contourpy>=1.0.1 in /opt/homebrew/lib/python
3.10/site-packages (from matplotlib) (1.3.1)
Requirement already satisfied: cycler>=0.10 in /opt/homebrew/lib/python3.10/
site-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /opt/homebrew/lib/python
3.10/site-packages (from matplotlib) (4.56.0)
Requirement already satisfied: kiwisolver>=1.3.1 in /opt/homebrew/lib/python
3.10/site-packages (from matplotlib) (1.4.8)
Requirement already satisfied: pillow>=8 in /opt/homebrew/lib/python3.10/sit
e-packages (from matplotlib) (11.1.0)
```

```
Requirement already satisfied: pyparsing>=2.3.1 in /opt/homebrew/lib/python
3.10/site-packages (from matplotlib) (3.2.2)
Requirement already satisfied: anyio in /opt/homebrew/lib/python3.10/site-pa
ckages (from httpx<1,>=0.23.0->huggingface_hub) (4.12.1)
Requirement already satisfied: certifi in /opt/homebrew/lib/python3.10/site-
packages (from httpx<1,>=0.23.0->huggingface_hub) (2025.1.31)
Requirement already satisfied: httpcore==1.* in /opt/homebrew/lib/python3.1
0/site-packages (from httpx<1,>=0.23.0->huggingface_hub) (1.0.9)
Requirement already satisfied: idna in /opt/homebrew/lib/python3.10/site-pac
kages (from httpx<1,>=0.23.0->huggingface_hub) (3.10)
Requirement already satisfied: h11>=0.16 in /opt/homebrew/lib/python3.10/sit
e-packages (from httpcore==1.*->httpx<1,>=0.23.0->huggingface_hub) (0.16.0)
Requirement already satisfied: six>=1.5 in /Users/summerann/Library/Python/
3.10/lib/python/site-packages (from python-dateutil>=2.8.2->pandas) (1.16.0)
Requirement already satisfied: click>=8.0.0 in /opt/homebrew/lib/python3.10/
site-packages (from typer-slim->huggingface_hub) (8.3.1)
Requirement already satisfied: exceptiongroup>=1.0.2 in /Users/summerann/Lib
rary/Python/3.10/lib/python/site-packages (from anyio->httpx<1,>=0.23.0->hug
gingface_hub) (1.2.2)

[notice] A new release of pip is available: 24.3.1 -> 25.3
[notice] To update, run: python3.10 -m pip install --upgrade pip
Note: you may need to restart the kernel to use updated packages.
```