

CS 2731

Introduction to Natural Language Processing

Session 27: Final project presentations

December 11, 2024



University of
Pittsburgh

School of Computing and Information

Course logistics

- Final project reports due **tomorrow, Thu Dec 12, 11:59pm**
- *Thanks for a great semester!*

Instructions

- Plan for **5 min max** presentations + a brief Q&A
- Cover at least these key points
 - Project motivation (briefly)
 - Data
 - Methods, or annotation/collection approach for dataset projects
 - Results
- Put your slides in this presentation after your project name slide by **class session, 2:30pm on Wed Dec 11**

Schedule

1. Joel, Jiyang
2. Dilip, Anveshika, Akshat
3. Yansheng, Xiaoyan, Shijai
4. John, Rojin, Xianglong
5. Yushui, Yifang, Zhuochun
6. Maanya, Jerry, Alex
7. Geonyeong, Kiran, Hugh, Carolina

1. Joel, Jiyang

Project motivation

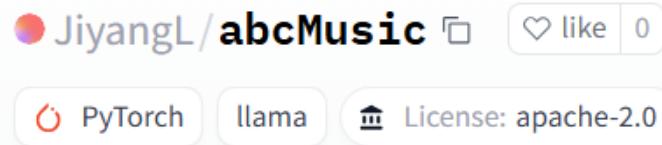
- Prominent LLM availability enables potential symbolic music generation via text-based music notation
- A new (2024) LLM benchmark for evaluation of musical understanding shows that GPT-4 outperforms music PhDs
- Including at music generation with ABC notation
- ABC notation is old, newly relevant, and has room for exploration with modern NLP techniques

Data

- Dataset of 270,000 tunes from "Interacting with GPT-2 to Generate Controlled and Believable Musical Sequences in ABC Notation"
 - Removed whitespace
 - Removed invalid characters
 - Created custom
 - Separated data into 80/20 split

Methods

The model is uploaded to huggingface JiyangL/abcMusic.



Model basics:

- Based on llama2 fine-tuning.
- Use abc character music tokenization.
- Use attention mechanism to capture dependencies between note sequences.
- The input is X (Reference Number), L (Default Note Length), M (Meter), K (Key), and a piece of song. The output is the entire song.

Training Resources

- GPU: L4 (22.5GB RAM)
- CPU: 53GB RAM
- Time: 40 hours (20 hours/epoch)
- Using DeepSpeed, FP16 precision.
- CUDA 11.8
- Python 3.10

Hyperparameters

Parameter Name	Value
BATCH_SIZE	8
NUM_EPOCHS	2
Learning Rate	1e-4
Weight Decay	0.1
Mode	llama
Train Type	lora
LoRA Dimension (lora_dim)	64
LoRA Scaling Factor (lora_alpha)	16
LoRA Dropout	0.1

Demo 1

```
Construct melodies by blending the designated musical pattern wi  
th the supplied motif.  
X:10 L:1/8 M:9/8 K:Emin [ :B2EG2EF3 B2EG2EFED B2EG2EF3] </s>  
input_text: Human: Construct melodies by blending the designated  
musical pattern with the supplied motif. </s> Assistant:  
2024-12-10 15:54:35.501189: E external/local_xla/xla/stream_exec  
utor/cuda/cuda_fft.cc:485] Unable to register cuFFT factory: At  
tempting to register factory for plugin cuFFT when one has alread  
y been registered  
2024-12-10 15:54:35.523833: E external/local_xla/xla/stream_exec  
utor/cuda/cuda_dnn.cc:8454] Unable to register cuDNN factory: At  
tempting to register factory for plugin cuDNN when one has alrea  
dy been registered  
2024-12-10 15:54:35.532768: E external/local_xla/xla/stream_exec  
utor/cuda/cuda_blas.cc:1452] Unable to register cuBLAS factory:  
Attempting to register factory for plugin cuBLAS when one has al  
ready been registered  
2024-12-10 15:54:36.788368: W tensorflow/compiler/tf2tensorrt/ut  
ils/py_utils.cc:38] TF-TRT Warning: Could not find TensorRT  
-----
```

Response:

X:1

L:1/8

M:6/8

K:G

|:D |G2BABc|dge~d3|cBA~B3|AGBAGE| G2BABc|dge~d3|cBA~B3|AGFG2:||
:f|gfgage|deddcB|cBA~B3|AGBAGE| gfgage|deddcB|cBA~B3|AGFG2:|||

Input:



Demo 1

ABC Editor

```
X:1  
L:1/8  
M:6/8  
K:G  
|:D |G2BABC|dge~d3|cBA~B3|AGBAGE|  
G2BABC|dge~d3|cBA~B3|AGFG2|  
gfgage|deddcB|cBA~B3|AGBAGE|  
gfgage|deddcB|cBA~B3|AGFG2 ||
```

Paste

Tune Search via TheSession.org

Search

The image shows four staves of musical notation. Each staff begins with a treble clef and a key signature of one sharp (F#). The time signature is 6/8. The notation consists of various note heads (solid black, open, and cross) and rests, separated by vertical bar lines. The notes are mostly eighth notes, with some sixteenth-note patterns and rests. There are also some longer sustained notes indicated by a horizontal line under the note head.

No errors

Transpose

G G# A Bb B C C# D Eb E F F#

0:00 100 % (120 BPM)

Demo 2

Response:

X:1

L:1/8

M:6/8

K:G

"^A" d | "G" G2 G BAG | "G" d2 d"C" efg | "G" dBG GAB | "
Em" E3"D7" DEF | "G" G2 G BAG | "G" d2 d efg |
"G" dBG"D7" ABA | "G" G3- G2 || "B" g | "G" g2 g gab | "
C" e2 d efg | "G" dBG GAB | "Em" E3"D7" DEF |
"G" G2 G BAG | "G" d2 d efg | "G" dBG"D7" ABA | "G" G3-
G2 ||

Demo 2

abc.rectangled.com

My Pitt | All Camp... 部件 - Liu, Jiyang ... 邮件 - Liu, Jiyang ... Homepage Dashboard Google 翻译 CS 2731 Introduc... Speech and Langu... YouTube New chat PDFdo.com 在...

ABC Player and Editor 2.0

ABC Editor

X:1
L:1/8
M:6/8
K:G
"A" d | "G" G2 G BAG | "G" d2 d"C" efg | "G" dBG GAB | "Em" E3"D7" DEF | "G"
G2 G BAG | "G" d2 d efg |
"G" dBG"D7" ABA | "G" G3- G2 || "B" g | "G" g2 g gab | "C" e2 d efg | "G" dBG GAB
| "Em" E3"D7" DEF |
"G" G2 G BAG | "G" d2 d efg | "G" dBG"D7" ABA | "G" G3- G2 ||

No errors

Paste

Tune Search via TheSession.org

Enter tune name Search

Transpose

G G# A Bb B C C# D E Eb F F#

0:00 100 % (120 BPM)

Results

- Not all prompts returned valid submissions
- Based on a filtered sample of successes, we generated a ROUGE score (Recall-Oriented Understudy for Gisting Evaluation)
 - Average: 0.4883

The screenshot shows a digital interface for creating musical compositions. At the top is a toolbar with various icons. Below it is a text area displaying musical metadata:
X:1
L:1/8
M:6/8
K:G
The main text area contains a sequence of musical notes and rests represented by letters and numbers, separated by vertical bars:
D | G2 BBAG | A2 c c2 B | A3 d3 | D3 z2 D | G2
BBAG | A2 c c2 B | A3 d3 | D3 z2 d | dcddcB |
c2 e e2 c | cBBdBA | A2 d d2 A | ABcdcB | B2 A A2
B | G3 B3 | G3 z2 |]

Generated Composition

Below the generated composition text is a set of player controls consisting of arrows for navigation, a play/pause button, and a slider indicating the current time as 0:00.



The screenshot shows a digital interface for creating musical compositions, similar to the one on the left. At the top is a toolbar with various icons. Below it is a text area displaying musical metadata:
X:1
L:1/8
M:6/8
K:G
The main text area contains a sequence of musical notes and rests represented by letters and numbers, separated by vertical bars:
D | GG GG2 B | dd B G3 | AAA A3 |
FFE D2D | GGG G2B | ddB G3 | A3 D3 | G6 ||

Target Notation

Below the target notation text is a set of player controls consisting of arrows for navigation, a play/pause button, and a slider indicating the current time as 0:00.



2. Dilip, Anveshika, Akshat

Natural Language Processing

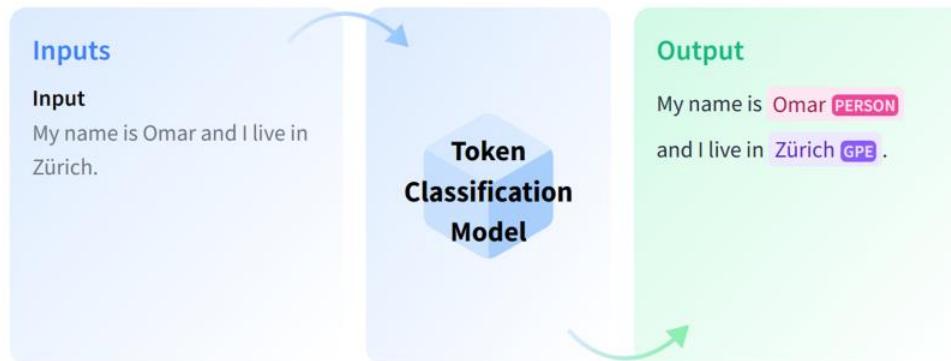
Deidentification of PHI in Electronic Health Records

Anveshika Kamble, Dilip Teja, Akshat Dhamale

12/11/2024

Project Motivation

- Named Entity Recognition (NER) is the task of identifying and extracting named entities from text.
- Clinical notes are particularly useful for NER, as they contain unstructured text that is rich in medical terminology and often reflects the patient's medical history, symptoms etc.
- The goal is to accurately identify and extract the relevant named entities from the text, in order to facilitate downstream tasks such as information retrieval and clinical decision tasks.



Data set

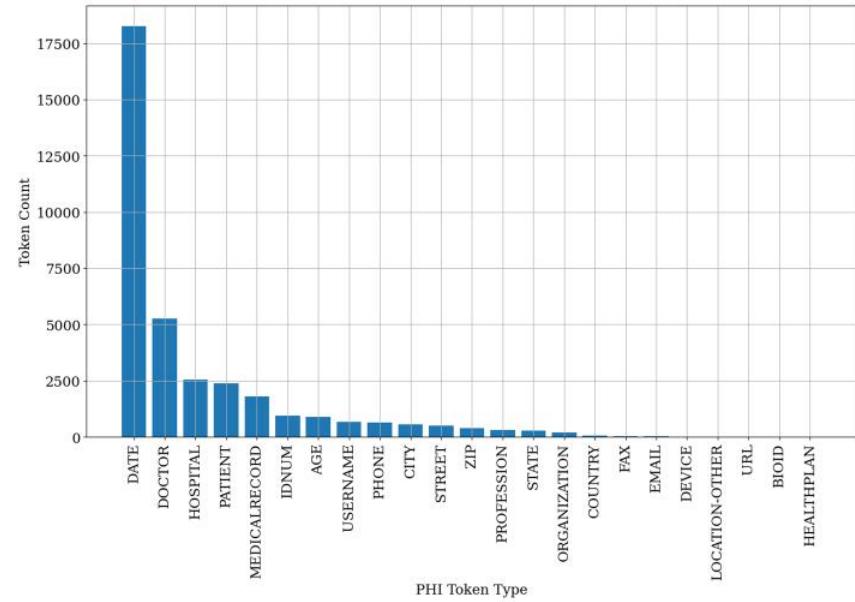
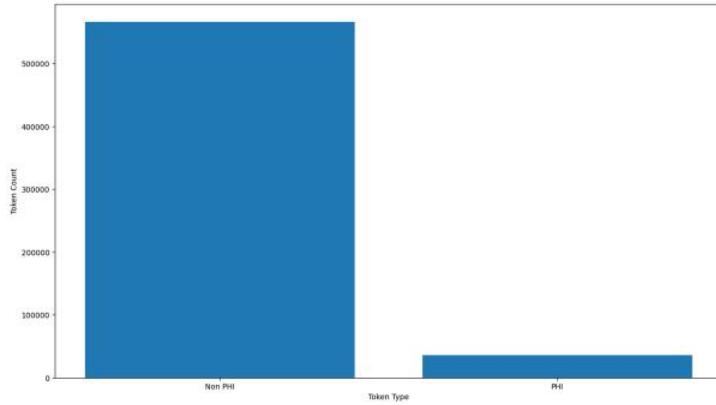
- The i2b2/UTHealth 2014 corpus. clinical records from 296 patients and 1304 clinical notes
- Manually annotated by trained professionals for PHI categories such as names, dates, locations etc.
- 7 main categories totalling 23 with sub-categories
- The data split is 70% percent for training and 30% for testing

RAW MEDICAL NOTE

Physician Discharge Summary Admit date: 10/12/1982 Discharge date: 10/22/1982 Patient Information Jack Reacher, 54 y.o. male (DOB = 1/21/1928). Home Address: 123 Park Drive, San Diego, CA, 03245. Home Phone: 202-555-0199 (home). Hospital Care Team Service: Orthopedics Inpatient Attending: Roger C Kelly, MD Attending phys phone: (634)743-5135 Discharge Unit: HCS843 Primary Care Physician: Hassan V Kim, MD 512-832-5025.



Data set



Challenges with Data set

Imbalances within data set

- Huge difference between frequency of PHI tokens (<5%) and non-PHI tokens

Imbalances within PHI tokens

- Few PHI tokens are highly frequent than many others
- Might affect the predictions on lower frequent PHI
- 23 Total PHI categories (with subcategories)



University of
Pittsburgh.

Data Preparation

- Centers on converting raw Clinical files to data format that the models can be fed to

Step -1 : Parsing

- Clinical files are in XML format
- Parsing the XML for attributes TEXT and TAG
- TEXT tag for unstructured clinical note into list of sentences
- TAG tag for annotate Tags
- <DATE id="P0" start="16" end="26" text="2067-05-03" TYPE="DATE" comment="" />
- <AGE id="P1" start="50" end="52" text="55" TYPE="AGE" comment="" />
- <NAME id="P2" start="290" end="296" text="Oakley" TYPE="DOCTOR" comment="" />

Data Preparation

Step -2 : Tokenize

- Tokenize the sentence using tokenizer
- Tokenize the tag word
- Encode the tokenized sentence with correct labels

	Sentence	Word	Label	Category
0	Record date: 2080-11-30	2080-11-30	DATE	DATE
1	Owen is a 63 y/o male here for evaluation of t...	Owen	PATIENT	NAME
2	Owen is a 63 y/o male here for evaluation of t...	63	AGE	AGE
3	SKIN ULCER-DR Esposito	Esposito	DOCTOR	NAME
5	Sees Dr Esposito for chr ulcer.	Esposito	DOCTOR	NAME

Data Preparation

Step -3 : BIO Encoding

- "B" (Beginning): The first token of an entity is tagged with "B" to indicate the beginning of the entity.
- "I" (Inside): Tokens subsequent to the first token of an entity are tagged with "I" to indicate they are inside the entity.
- "O" (Outside): Tokens that are not part of any named entity are tagged with "O" to indicate they are outside any entity.

```
[ 'SK', '##IN', 'U', '##LC', '##ER', '-', 'DR', 'E', '##sp', '##os', '##ito' ]  
[ 'O', 'O', 'O', 'O', 'O', 'O', 'B-DOCTOR', 'I-DOCTOR', 'I-DOCTOR', 'I-DOCTOR' ]
```

- Padding and truncating the data to have all the data points with same size and remove irregularities
- Adding special tokens



University of
Pittsburgh.

Explored Models

- BERT uncased
 - Baseline model trained for multiple classes (24)
 - Pre-trained for NER tasks
 - Bidirectional context-awareness helps in disambiguating medical entities in text
- Bio BERT
 - Pre trained on medical data
 - Expected to understand medical entities better than conventional bert
 - class weights were added to handle class imbalance in token classification. Rare labels are assigned higher weights to ensure balanced model learning.
- RoBERTa
 - Pre Trained on NER dataset containing 10 classes
 - Fine-tuned on tasks that use the whole sentence (potentially masked) to make decisions
- ELECTRA
 - Relatively lower compute model pre trained on NER dataset
 - Trained to distinguish "real" input tokens vs "fake" input tokens generated by another neural network, similar to the discriminator of a GAN
- Llama 3.2 1-B
 - Pre trained on Various NER tasks



Evaluation Metric

Precision: Measures the percentage of predicted entities that are correct. High precision means fewer false positives.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

Recall: Measures the percentage of actual entities that are correctly predicted. High recall means fewer false negatives.

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

F1 - score: Balances precision and Recall

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$



Hyper Parameters

	BERT-Uncased	BioBERT	RoBERTa	Electra	LLama 3.2-1B
learning rate	2e-05	5e-05	2e-05	2e-05	5e-05
train batch size	16	16	16	16	64
eval batch size	16	16	16	16	64
optimizer	Adam with betas=(0.9,0.999) and epsilon=1e-08				
lr scheduler type	linear	linear	cosine	linear	linear
lr scheduler warmup ratio	0.1	0.1	0.1	0.1	0.1
num epochs	5	5	5	5	5

Hyper Parameters

	BERT-Uncased	BioBERT	RoBERTa	Electra	LLama 3.2-1B
learning rate	2e-05	5e-05	2e-05	2e-05	5e-05
train batch size	16	16	16	16	64
eval batch size	16	16	16	16	64
optimizer	Adam with betas=(0.9,0.999) and epsilon=1e-08				
lr scheduler type	linear	linear	cosine	linear	linear
lr scheduler warmup ratio	0.1	0.1	0.1	0.1	0.1
num epochs	5	5	5	5	5

Results

	BERT-Uncased			BioBERT			RoBERTa			Electra			LLama 1-B		
	Precision	Recall	F1-Score	Precision	Recall	F1-Score	Precision	Recall	F1-Score	Precision	Recall	F1-Score	Precision	Recall	F1-Score
O	0.98	1	0.99	0.98	1	0.99	0.98	1	0.99	0.98	1	0.99	0.98	1	0.99
DATE	0	0	0	0.13	0.11	0.18	0.14	0.11	0.14	0.19	0.12	0.14	0.46	0.39	0.43
DOCTOR	0	0	0	0.13	0.15	0.12	0.13	0.14	0.19	0.11	0.13	0.18	0.44	0.32	0.47
HOSPITAL	0	0	0	0.1	0.15	0.15	0.16	0.14	0.16	0.13	0.11	0.15	0.38	0.4	0.38
PATIENT	0	0	0	0.15	0.14	0.15	0.1	0.11	0.15	0.18	0.2	0.17	0.42	0.43	0.3
AGE	0	0	0	0.11	0.1	0.19	0.2	0.12	0.1	0.19	0.14	0.16	0.38	0.31	0.36
COUNTRY	0	0	0	0.1	0.15	0.14	0.17	0.15	0.17	0.18	0.2	0.11	0.46	0.3	0.49
CITY	0	0	0	0.12	0.1	0.17	0.12	0.14	0.1	0.16	0.18	0.1	0.3	0.42	0.45
STATE	0	0	0	0.11	0.13	0.17	0.19	0.13	0.1	0.13	0.16	0.16	0.47	0.46	0.34
PHONE	0	0	0	0.16	0.14	0.11	0.12	0.13	0.12	0.14	0.16	0.17	0.41	0.48	0.43
Avg	0.96	0.96	0.96	0.98	0.97	0.97	0.98	0.97	0.96	0.98	0.97	0.97	0.98	0.98	0.98

Discussion and Conclusions

- Overall, pre-trained models showcase better performance.
- Balancing weights has significantly improved the class-wise accuracies.
- Entity classes like STATE, PHONE, and HOSPITAL are likely underrepresented in the dataset. Models struggle to identify these entities, resulting in near-zero precision and recall in many cases(BERT).
- Amongst all models, LLama performed comparatively better.



Future work

- Hyper Parameter tuning
 - Currently set to default .
 - Training for more number of epochs (around 20)
- Addressing the imbalance.
 - Currently addressed using data removal techniques . Can have more innovative approaches to handle the problem
 - Provide more examples of underrepresented classes to improve model generalization.
- Try and run other LLM's
 - Can explore bigger models like GPT, if they really improve the evaluation metric
 - Anonymization(For Semantic Transformation): MIXTRAL



Thank You.

3. Yansheng, Xiaoyan, Shijai

Project Motivation

- Problem Statement: Movie plot summaries are crucial for understanding and recommending films, but manually creating them is time-consuming.
- Value of Work: Automating movie summaries can save time and provide structured data for recommendation systems and review tools.
- Summary: The project aims to automatically generate movie summaries from subtitles to enhance content consumption experiences.

Dataset

- Data Used: CMU Movie Summary Corpus and Opensubtitles subtitle data.

- Dataset Size: Contains 1,322 movie subtitles with an average of 1,255 dialogues per movie.

- Data Processing: Metadata removal (timestamps, subtitle numbers) and cleaning of stop words and punctuation.

Data Preparation

```
▶ |from datasets import load_dataset  
  
# Download the dataset and use the default cache path  
open_subtitles = load_dataset("open_subtitles", lang1="en", lang2="hi", split="train[:1%]")  
  
# View the dataset  
print(open_subtitles)  
  
▶ import re  
from datasets import load_dataset, concatenate_datasets  
  
# Load the local dataset in text mode and convert it into a Dataset object  
cmu_dataset = load_dataset("text", data_files="movie_summary_corpus.txt", split="train")  
I  
  
# Cleaning and reconstruction of CMU data  
def preprocess_cmu(examples):  
    cleaned = []  
    for line in examples["text"]:  
        line = line.strip().lower()  
        line = re.sub(r"[a-z0-9\.,!?\.;:\\"(\)-]", "", line)  
        cleaned.append(line)  
    return {"input_text": cleaned, "target_text": cleaned}  
  
cmu_dataset = cmu_dataset.map(preprocess_cmu, batched=True)  
# Remove the original "text" column  
cmu_dataset = cmu_dataset.remove_columns(["text"])  
  
print(f"CMU dataset size: {len(cmu_dataset)}")  
  
# Loading the OpenSubtitles dataset  
open_subtitles = load_dataset("open_subtitles", lang1="en", lang2="hi", split="train[:1%]")  
  
def preprocess_opensub(examples):  
    en_texts = []  
    for trans in examples["translation"]:  
        subtitle = trans["en"].lower()  
        subtitle = re.sub(r"[a-z0-9\.,!?\.;:\\"(\)-]", "", subtitle)  
        en_texts.append(subtitle)  
    return {"input_text": en_texts, "target_text": en_texts}  
  
opensub_dataset = open_subtitles.map(preprocess_opensub, batched=True)  
opensub_dataset = opensub_dataset.remove_columns(["translation"])  
  
print(f"OpenSubtitles dataset size: {len(opensub_dataset)}")  
  
# Merge Datasets  
combined_dataset = concatenate_datasets([cmu_dataset, opensub_dataset])  
print(f"Combined dataset size: {len(combined_dataset)})")
```

Train tokenizer

```
from transformers import PegasusTokenizer, PegasusForConditionalGeneration
import re
import os

# Loading PEGASUS tokenizer
tokenizer = PegasusTokenizer.from_pretrained("google/pegasus-large")

# Read and parse the CMU movie summary dataset
summaries_data = {}
with open('movie_summary_corpus.txt', 'r', encoding='utf-8') as f:
    cmu_summaries = f.readlines()

# Parse the movie ID and summary content of each line and perform preprocessing
for line in cmu_summaries:
    parts = line.strip().split('\t')
    if len(parts) > 1:
        movie_id = parts[0]
        summary = parts[1].lower()
        summary = re.sub(r'[a-zA-Z\s.,!?:;\\"()-]', '', summary)
        summaries_data[movie_id] = summary

# Check the IDs and summaries of the first 5 movies
print("Summary of the first 5 movies: ", list(summaries_data.items())[:5])

# Split the dataset into multiple chunks and load them in chunks
# Define the chunking function
def split_dataset(data, chunk_size, output_dir):
    os.makedirs(output_dir, exist_ok=True)
    for i in range(0, len(data), chunk_size):
        chunk = data[i:i + chunk_size]
        with open(f"{output_dir}/data_chunk_{i // chunk_size}.txt", "w") as f:
            f.write('\n'.join(chunk))

# Save data in chunks
split_dataset(list(summaries_data.values()), chunk_size=500, output_dir=".//data_chunks")

# Inspecting the generated blocks
print("Data block file: ", os.listdir("./data_chunks"))
```

Train Model

```
from transformers import PegasusForConditionalGeneration, Trainer, TrainingArguments, DataCollatorForSeq2Seq
from datasets import Dataset

# Loading the PEGASUS model
model = PegasusForConditionalGeneration.from_pretrained("google/pegasus-large")

# Check the equipment
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
model = model.to(device)
print("Using device:", device)

# Get all data block files
data_files = os.listdir("./data_chunks")

# Define training parameters
training_args = TrainingArguments(
    output_dir="./pegasus_movie_summarization",
    overwrite_output_dir=True,
    num_train_epochs=1,
    per_device_train_batch_size=1,
    gradient_accumulation_steps=128, # Simulate larger batch sizes to avoid running out of video memory. Accumulated
    save_steps=500,
    save_total_limit=2,
    prediction_loss_only=True,
    logging_steps=100,
    report_to="none",
    learning_rate=5e-5,
    lr_scheduler_type="linear",
    fp16=True # Enable mixed precision to reduce video memory usage
)

# Defining Data Collaborators
data_collator = DataCollatorForSeq2Seq(tokenizer, model=model)

# Iterate through each data block
for data_file in data_files:
    print(f"Processing chunks: {data_file}")
    # Load each piece of data
    with open(f"./data_chunks/{data_file}", "r") as f:
        train_text = f.read().splitlines()

    # Constructing the dataset
    data = [{"input_text": line, "target_text": line} for line in train_text]
    dataset = Dataset.from_list(data)

    # Encode the dataset
    def preprocess_function(examples):
        inputs = tokenizer(examples['input_text'], max_length=128, truncation=True, padding="max_length")
        targets = tokenizer(examples['target_text'], max_length=16, truncation=True, padding="max_length")
        inputs['labels'] = targets['input_ids']
        return inputs

    encoded_dataset = dataset.map(preprocess_function, batched=True)

    # Setting up the Trainer
    trainer = Trainer(
        model=model,
        args=training_args,
        train_dataset=encoded_dataset,
        data_collator=data_collator,
    )

    # Train the current data block
    trainer.train()

    # Save the intermediate model
    trainer.save_model(f"./pegasus_movie_summarization_chunk_{data_file.split('_')[-1]}")
    tokenizer.save_pretrained(f"./pegasus_movie_summarization_chunk_{data_file.split('_')[-1]}")

print("Training completed!")
```

Test

```
● from transformers import PegasusTokenizer, PegasusForConditionalGeneration
from datasets import load_metric
import torch
import re

# Loading the model and tokenizer
final_model_dir = "./pegasus_movie_summarization_chunk_10"
tokenizer = PegasusTokenizer.from_pretrained(final_model_dir)
model = PegasusForConditionalGeneration.from_pretrained(final_model_dir)

# Device Setup
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
model = model.to(device)

# Load subtitle file
subtitle_file = "movie_full_subtitles.txt"
with open(subtitle_file, "r", encoding="utf-8") as f:
    full_subtitles = f.read()

# Process subtitles in chunks
def split_text(text, chunk_size=1024):
    return [text[i:i+chunk_size] for i in range(0, len(text), chunk_size)]

subtitle_chunks = split_text(full_subtitles, chunk_size=1024)

# Generate summary
summaries = []
for chunk in subtitle_chunks:
    inputs = tokenizer(chunk, max_length=1024, truncation=True, return_tensors="pt", padding="longest").to(device)
    summary_ids = model.generate(
        inputs.input_ids,
        max_length=300,
        num_beams=8,
        temperature=0.7,
        early_stopping=True
    )
    summaries.append(tokenizer.decode(summary_ids[0], skip_special_tokens=True, clean_up_tokenization_spaces=True))

# Merge summary of all chunks
generated_summary = " ".join(summaries)

# Load Reference Summary
reference_summary_file = "movie_reference_summary.txt"
with open(reference_summary_file, "r", encoding="utf-8") as f:
    reference_summary = f.read()

# Loading evaluation indicators
rouge = load_metric("rouge")
bleu = load_metric("bleu")

# Calculate ROUGE
rouge_result = rouge.compute(
    predictions=[generated_summary],
    references=[reference_summary]
)

# Calculating BLEU
bleu_result = bleu.compute(
    predictions=[[generated_summary.split()]],
    references=[[reference_summary.split()]]
)

# Printing Results
print("Generated Summary:\n", generated_summary)
print("ROUGE Results:\n", rouge_result)
print("BLEU Results:\n", bleu_result)
```

Result

```
training_args = TrainingArguments(
    output_dir='./pegasus_movie_summarization',
    overwrite_output_dir=True,
    num_train_epochs=1,
    per_device_train_batch_size=1,
    gradient_accumulation_steps=128, # Simulate larger batch sizes to
    save_steps=500,
    save_total_limit=2,
    prediction_loss_only=True,
    logging_steps=100,
    report_to="none",
    learning_rate=5e-5,
    lr_scheduler_type="linear",
    fp16=True # Enable mixed precision to reduce video memory usage
)
```

Evaluation

ROUGE

BLEU

Q&A

4. John, Rojin, Xianglong

Mask Out: A Novel Regularization Method for Code-Switching Language Identification

Xianglong Xu, John Evan Bowen, Rojin Taheri
{xix110, jeb386, rot64}@pitt.edu

Project Motivation

- What is code-switching?
 - Why is it important to study?
 - Applications like media content analysis, VOD's (voice operated devices)
 - Despite the applications, research on code-switching tasks have only recently gained substantial attention



Our Solution

- Titled Mask Out
 - Regularization technique for code-switching language identification
 - How does it work?
 - What is the focus?
 - Obscuring language-specific cues for better generalization
 - Why?
 - Learn to rely on broader contextual patterns for language identification



Datasets

- Focused on two primary datasets for this project from the Linguistic Code-switching Evaluation benchmark:
 - Spanish-English Dataset
 - Used for training the model
 - Apply masking technique
 - Nepali-English Dataset
 - Used to evaluate the models performance and generalization capacity across different contexts



Methods

- This project is built on mBERT
 - Multilingual Bidirectional Encoder Representations from Transformers
 - A pre-trained LLM optimized for multiple languages, which is particularly well-suited for handling code-switched text
 - We apply a probabilistic token masking scheme during the training phase
 - Using a predefined probability p
 - This is meant to enhance word-level language identification in code-switched data



Methods Cont.

- By obscuring certain words with a MASK token, we compel the model to develop a more robust and context-aware understanding of language switching patterns
 - This then challenges the model to rely on broader contextual cues rather than overfitting to surface-level linguistic markers
 - We also implemented an early stopping method
 - Done by monitoring the model's validation performance and halts training if the validation loss does not improve over a predefined number of epochs

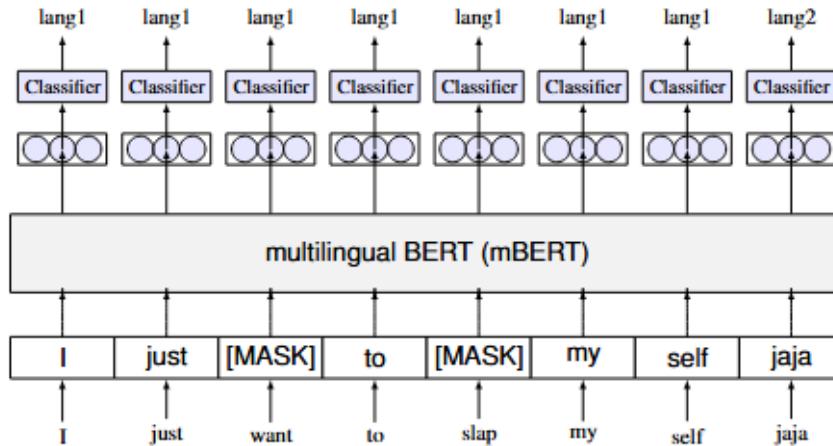
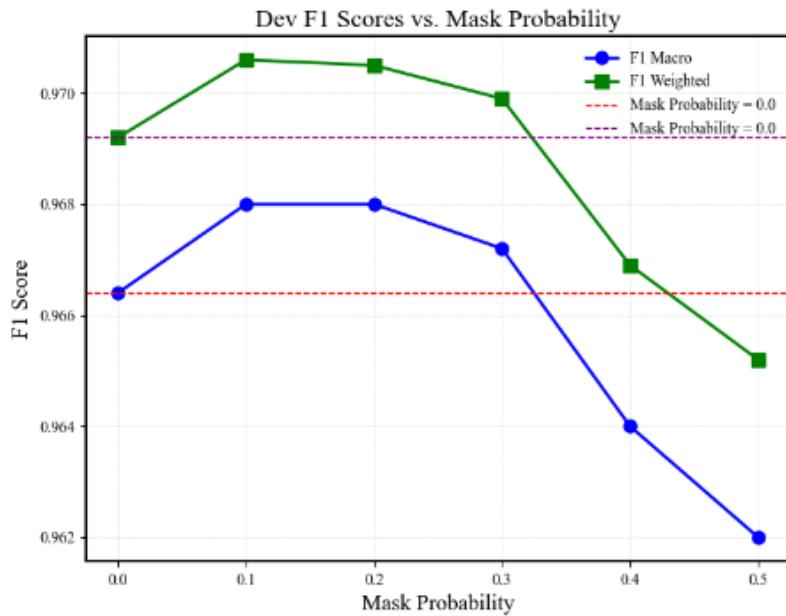
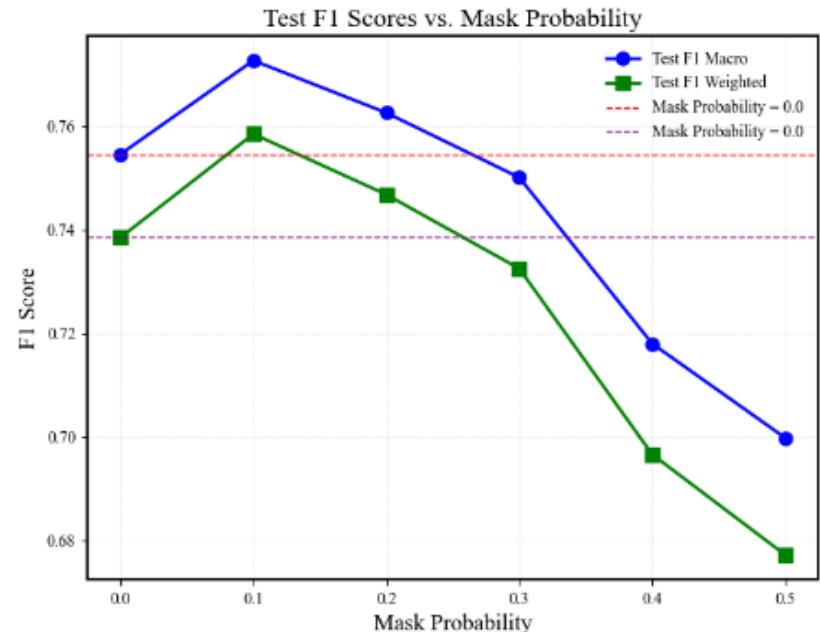


Figure 2: Visualization of mBERT's implementation (sentence, number 99, from the (SPA-ENG) training dataset).

Results



(a) F1 score on the dev dataset



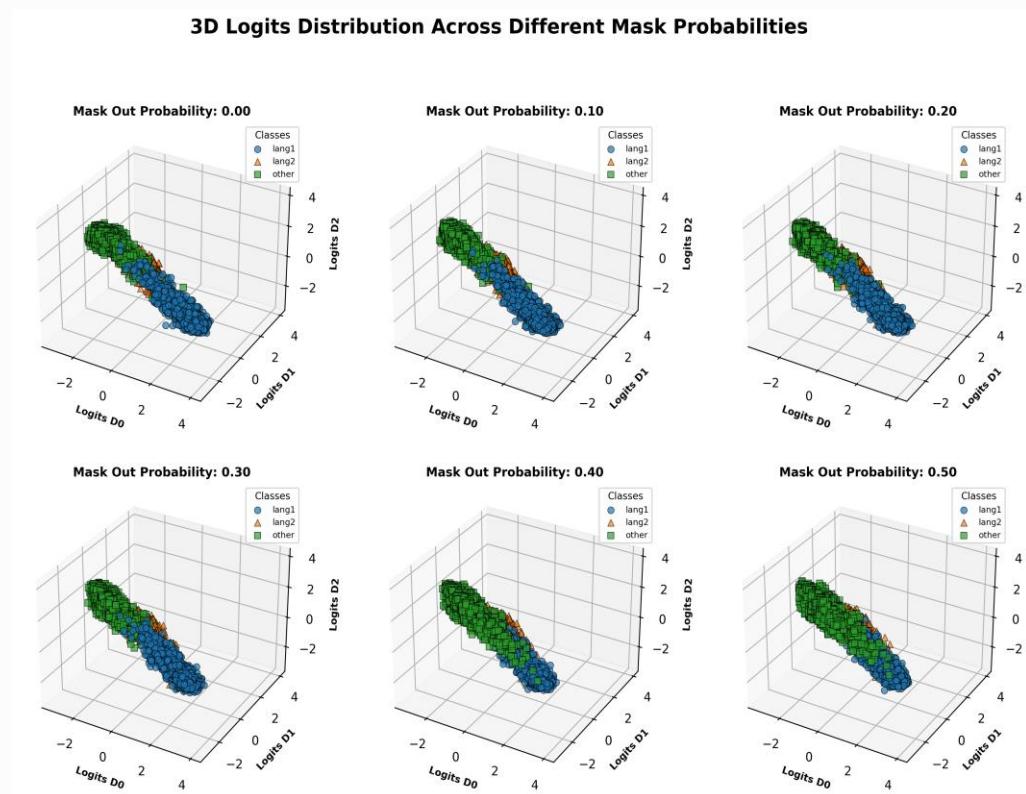
(b) F1 score on the test dataset

Results Overview

Mask Out Probability	Spanish-English (SPA-ENG)						Nepali-English (NEP-ENG)	
	Dev Loss	F1 Macro	F1 Weighted	Precision Macro	Recall Macro	Accuracy	F1 Macro	F1 Weighted
0.0	0.0268	0.9664	0.9692	0.9666	0.9662	0.9692	0.7546	0.7385
0.1	0.0253	0.9680	0.9706	0.9687	0.9674	0.9706	0.7727	0.7585
0.2	0.0257	0.9680	0.9705	0.9689	0.9671	0.9705	0.7626	0.7468
0.3	0.0265	0.9672	0.9699	0.9684	0.9661	0.9699	0.7501	0.7325
0.4	0.0286	0.9640	0.9669	0.9653	0.9628	0.9669	0.7180	0.6966
0.5	0.0308	0.9620	0.9652	0.9635	0.9605	0.9652	0.6998	0.6772

Table 1: Performance comparison of models with different training datasets.

More results



Questions/Comments

5. Yushui, Yifang, Zhuochun

Exploring Cultural Bias in Language Models Through Word Grouping Games

Motivation

- Large language models (LLMs) exhibit cultural bias, reinforcing stereotypes.
- Current evaluations lack emphasis on reasoning using cultural nuances.
- Our study introduces the Word Grouping Game (WGG), a novel approach to assess cultural reasoning in LLMs.

Dataset

- Word Grouping Game(WWG)
 - Task: Form groups of four words based on a shared cultural topic.
 - Focus: Chinese culture, with data in native Chinese and English translation.
- Data Creation:
 - 80 word-groups based on Chinese culture.
 - Topics tagged into three categories:
 - Everyday (e.g., Study Subjects)
 - Pop Culture (e.g., Video Games)
 - Linguistic (e.g., Poetry Classifications).
- Game Datasets:
 - Sample sizes: 2, 3, and 4 groups per game.
 - Created balanced dev/test (100 games) splits for reliable evaluation.

Dataset

- One game sample in the Chinese and translated English WGG dataset with 4 groups.

Word 1	Word 2	Word 3	Word 4	Group Name
古体诗	近体诗	词	曲	中国诗歌形式分类
青鱼	草鱼	鲢鱼	鳙鱼	四大家鱼
树皮	树枝	根	干	树的组成
梳子	齿轮	锯子	拉链	具有齿的物体

Word 1	Word 2	Word 3	Word 4	Group Name
ancient poetry	modern poetry	word	song	Chinese poetry forms
herring	grass carp	silver carp	bighead carp	Four major fish kinds
bark	branches	root	trunk	Tree composition
comb	gear	saw	zipper	Objects with gear

Methods

- Evaluation Metrics:
 - F1 Score: Measures correctness in grouping.
 - BERT Score (0-1) and GPT Topic Similarity (1-5): Topic similarity evaluation.
- Models:
 - Closed-source: GPT-3.5 Turbo.
 - Open-source: LLaMA2-7B, Mistral-7B.
- Setup:
 - Zero-shot prompting, specifying game rules and expected output.
 - Models tested on both native Chinese and English-translated datasets.

Results

- Translated English culture-related groupings has higher F1 score.
- As game size increased, performance steadily decreased.
- Performance:
 - Mistral-7B: comparable or even better performance than GPT-3.5-Turbo
 - LLaMA2-7B: limited reasoning ability

Models	2-Group Game Results					
	F1 Score		GTS		BERT Score	
	C	TE	C	TE	C	TE
GPT-3.5-Turbo	0.891	0.952	2.922	3.102	0.668	0.314
LLaMA2-7B	0.004	0.222	0.510	1.095	0.011	0.140
Mistral-7B	0.877	0.935	2.265	2.855	0.622	0.584

Models	3-Group Game Results					
	F1 Score		GTS		BERT Score	
	C	TE	C	TE	C	TE
GPT-3.5-Turbo	0.851	0.925	2.783	3.072	0.516	0.269
LLaMA2-7B	0.003	0.054	0.347	0.500	0.003	0.036
Mistral-7B	0.873	0.910	2.547	2.900	0.647	0.585

Models	4-Group Game Results					
	F1 Score		GTS		BERT Score	
	C	TE	C	TE	C	TE
GPT-3.5-Turbo	0.809	0.883	2.851	2.998	0.508	0.289
LLaMA2-7B	0.037	0.077	0.290	0.525	0.029	0.063
Mistral-7B	0.822	0.915	2.283	2.880	0.626	0.593

C" and "TE" denote Chinese and translated English datasets. "GTS" is defined as GPT Topic Similarity.

Discussions

- Cultural Bias in LLMs:
 - All LLMs performed worse with native Chinese data compared to English translations.
 - This may be caused by the dominant position of English in today's world languages, which leads to the majority of pre-training datasets being only available in English.
- Complexity vs. Performance:
 - All models showed predictable declines with increasing game complexity.
 - This phenomenon matches our previous hypothesis that the game difficulty will also increase as the number of groups increases.

Conclusion

- WGG effectively evaluates cultural reasoning in LLMs.
- Clear biases observed via the evaluation, highlighting disparities between different languages and cultures among today's LLMs.

Future Work

- Expand WGG to include additional cultures and datasets.
- Explore fine-tuning open-source models with culturally representative data.
- Incorporate interactive evaluations with human feedback.

Task and Roles Assigned to Group Members

- Yushui Han: Chinese WGG development, English translation, GPT model evaluation, presentation preparation.
- Yifang He: Chinese WGG development, English translation, GPT model evaluation, presentation preparation.
- Zhuochun Li: Chinese WGG development, two open-source models evaluation, report writing.

6. Maanya, Jerry, Alex

Dissecting Omnibus Bills

By: Maanya, Jerry and
Alex

Introduction

- Our project involved the Legiscan dataset a real-time legislative tracking service.
- Legiscan contains information on all legislation that is passed at a state level in the United States
- We planned to use the dataset to dissect and investigate “omnibus bills”



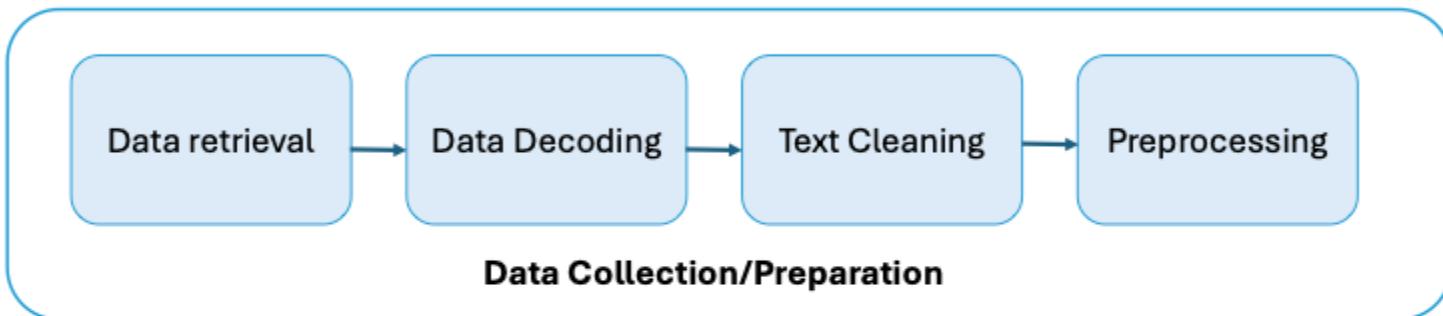
Things change...

- We initially planned to...
 - Look at 4 different states for analysis
 - Create a summary for each bill to group bills into different categories and aid in the task of determining the content of the legislation
 - Establish similarity scores between bills to determine the progression for a bill over time.
 - Determine the extent to which “omnibus” bills are similar to pieces of legislation proposed in the previous year(s).
- But we ended up modifying our plan
 - Changed to just one state (dang api keys)
 - Shifted focus to predicting bill passage using bill text and other metadata
 - Still did a little bit of omnibus stuff!

Data Pre-Processing

- **Data Retrieval:**
 - Extract bill IDs from CSV and fetch encoded text via API.
- **Decoding:**
 - Handle Base64-encoded formats:
 - **HTML:** Parse with BeautifulSoup, remove tags/scripts, clean further.
 - **PDF:** Extract text using pypdf.PdfReader and clean.
- **Text Cleaning:**
 - Remove unwanted patterns (e.g., extra spaces, numeric sequences).
 - Standardize text formatting (e.g., lowercase, punctuation removal).
- **Preprocessing:**
 - Tokenize text, remove stopwords using NLTK.
 - Apply regex to clean odd characters and finalize text.

Predictive Algorithm



Data Collection/Preparation



Feature Engineering

Models to Predict

Identifying Omnibus Bills

- We quickly realized that identifying omnibus legislation is not so easy...
- In general each states legislative branch operates differently
 - Different calendars
 - Different party control
 - Different schedules in terms of how they enact legislation
 - Sometimes bills will self identify as “omnibus”
- All of this makes it difficult to identify which bills are “omnibus bills”
 - But we did do this for a few states!



Initial Results

	Accuracy	Precision	Recall	F1
Naïve Bayes	0.9814	0.9632	0.9814	0.9722
Support Vector Machine				
K-nearest neighbor				

Where to go from here?

- Investigate similarity scores between omnibus bills and individual bills
 - Use a cosine similarity and other metrics
- Incorporate more features!
 - So far we only have the bill text, a brief description, and the title
 - How does which party introduce the bill affect what gets passed?
 - How does which committee the bill is coming from affect it getting passed or not?
- Tie it all together
 - Is there a way to look and see automatically highlight where a bill shows up in the omnibus bill?
 - What features can lead to the bill being incorporated to an omnibus bill?

Conclusion

- There is some indication that the content and language of a bill can be used to predict what action is taken.
- Initial results that can potentially show that cosine similarity can be used to see if any of the bills correlate
- Good data takes time and a lot of effort to compile and manage!

7. Geonyeong, Kiran, Hugh, Carolina

Automated Extraction of Cellular Niches from Scientific Literature

12/11/2024

Hugh Galloway (hug18@pitt.edu),
Kiran Shridhar Alva (kiranshridhar@pitt.edu),
Geonyeong Choi (gec108@pitt.edu)

Overview

- The goal of this project is to compile a large set of papers which analyze spatially resolved single-cell data
- After the creation of this dataset we want to automatically extract cellular neighborhoods (CNs), also referred to as 'niches' or 'recurrent cellular neighborhoods' from this set of papers
- CNs are recurring patterns of cells that group together in tissue and can be used to predict important clinical outcomes like patient response to therapy
- There are many papers which measure single-cell gene expression and colocalization of cell types but to date no one has compared findings across these papers. Our automated meta-analysis would allow us to immediately understand trends in the results across this incredibly hot field.

Project Deliverables

- The project deliverables are the following:
 1. Extract a dataset of relevant spatial single-cell papers which contain cellular niches (search for papers in biorxiv/PubMed, extract using abstract and title, extract the papers as pdfs)
 2. Develop a method for extracting cellular neighborhoods and their associated biology from the papers in an automated manner (evaluate performance on a small hand-labeled subset)
 3. Run the aforementioned method on all papers in our extracted collection and create a table or database that matches each paper to its cellular niches, associated biology and important data like disease types and tissue of origin

Technology Background

- Spatial Transcriptomics is a relatively new and exceedingly popular method for analyzing tissue biology
- Basic idea is that now we have gene expression within individual cells as well as the coordinates of those cells
- Named Nature's method of the year in 2020 [1]



[1] states that RNA-seq is like a smoothie (gives us broken down components of tissue), single-cell RNA-seq is like a fruit salad (we know what the individual components are now), and spatial transcriptomics is like a fruit tart (we know the individual components and their arrangement)

Paper Extraction

- For the paper extraction we used the Entrez API in BioPython [2] and searched for relevant papers with a variety of prompts (shown on the right).
- Some of these are technology specific terms: for example: CODEX and CyCIF are popular multiplex imaging methods
- The more specific the technology term, the better the results tended to be.
- More general terms like spatial single-cell cellular neighborhoods pulled more papers but also pulled more review papers and opinion papers that discuss spatial single-cell technologies and not actual discovery papers that have the results we are looking for.
- Overall, we pulled 134 papers.

Prompts used for paper searching:
[“multiplex immunofluorescence cellular neighborhoods”, “CODEX cellular neighborhoods”, “spatial transcriptomics cellular neighborhoods”, “spatial single-cell cellular neighborhoods”, “CyCIF cellular neighborhoods”]

Paper Annotation

- 18 papers that contained cellular neighborhoods were manually annotated for the project.
- The following fields were included in the annotations:
 - Disease Name: Name of the disease that is analyzed in this paper. Some of the papers were healthy tissue atlases so there wasn't necessarily a disease for every paper.
 - Clinical Variable: When cellular neighborhoods are collected they are almost always compared across patient groups. The variable that we use to split the patients is referred to as a clinical variable. Most often these are survival (long vs short), recurrence, and response to therapy.
 - Neighborhood Names: The set of unique neighborhood names present in the paper.
 - Neighborhood Associations: Some of the neighborhoods have significant associations with clinical variables. Here we assign 'Pos' for association with a positive outcome (like successful response to therapy), 'Neg' for association with a negative outcome, and 'None' for interactions that are not significant.
- For disease name we simply search for a description of the tissue samples, and that gives you the correct disease.
- For clinical variable we look for the discussion of the cellular neighborhoods and find the variable that they are compared against. This gives us the neighborhood associations as well.
- For the neighborhood names we look for the unique neighborhood names that are present in the text. We only look to extract neighborhoods that are in the main text body.

Model Configurations

- We use the GPT-4o model to extract the cellular neighborhood information from the papers.
- We consider 4 model configurations:
 - ‘pure_baseline’: This configuration has a prompt for extracting the disease name, clinical variable and unique neighborhood names. There is a second prompt which has the ground truth neighborhood names and the ground truth clinical variable, and it asks the model to extract the associations between the listed neighborhood names and the clinical variable. Both prompts include a text representation of the entire paper as context.
 - ‘one_shot_CoT’: In this scenario there are separate prompts for all of the items we want to extract. Each prompt contains a relevant text excerpt from a paper not in the labeled dataset and an answer that shows what the correct output would look like for that text excerpt. We attempt to structure the answer in a chain-of-thought (CoT) format. All prompts include the entire paper as context.
 - ‘rag_baseline’: This is the same as pure baseline, but instead of using the entire paper, we chunk the paper and find the most relevant chunks via RAG. We use chunk size of 2000 and pull 5 chunks.
 - ‘rag_one_shot’: Same as ‘one_shot_CoT’ except we use RAG to find the most relevant chunks. Again we use 5 chunks per prompt.

Baseline Model Prompt for CN Extraction

This prompt contains parts of an academic paper represented as a string as context. This paper should describe cellular neighborhoods (CNs). The paper describes a spatial single-cell experiment. The paper should be focusing on a specific disease, however, it could analyze healthy tissue. You must report the disease type discussed in the paper and return a string for that disease type. If no disease is described, then return 'None'. You must select a disease type from the following list: {disease_string}. For example, non-small cell lung cancer would be mapped to 'lung_cancer'.

You must extract the names of all of the cellular neighborhoods. Examples of names could be 'Immune enriched', 'tumor stroma boundary', 'follicle', 'vasculature', etc. Do note that cellular neighborhoods are also often referred to as cellular niches or recurrent cellular neighborhoods. The neighborhood names must be returned as a list of strings. I do not have specific neighborhood names that you must match.

This paper should compare the neighborhoods across groups of patients, to see if they are enriched in specific patient groups. We refer to the variable that splits groups of patients as a clinical variable. Often, they are survival or response to immunotherapy (or any other type of treatment) or a disease type/subtype. I need you to tell me what clinical variable is associated with the cellular neighborhoods from this paper. Your assigned clinical variable MUST match one of the ones in this list: {clinical_var_string}. For example, disease-free survival (DFS) would be mapped to survival because that is in the list. If the clinical variable is not in the list, then put 'None', but in no circumstance should you return a clinical variable prediction that is not included in the provided list. Also, note that not every study will use a clinical variable, so in that case, please assign 'None' as the clinical variable.

One-Shot Chain of Thought Prompt – Clinical Variable Example

This prompt contains parts of an academic paper represented as a string as context. These papers tend to perform spatial single-cell analysis, and we are specifically interested in papers that analyze cellular neighborhoods (CNs). Most of the time, when a cellular neighborhood analysis is performed, researchers will split the patients into groups and compare the enrichment of different cellular neighborhoods within these groups. We refer to the variable that splits the patients into groups as a 'clinical_variable'.

You will need to analyze the text representation of a paper in this message and decide what the clinical variable in the paper is. Note that the clinical variable you assign MUST match one of the clinical variables in this list: {clinical_var_string}. For example, progression-free survival (PFS) would be mapped to 'survival' because that is in the list. Disease-free survival (DFS) would also be mapped to 'survival' because 'survival' is in the list of allowed outputs. Sometimes there is not a clinical variable, and in this case, you would assign 'None'.

I have included an example of relevant text excerpts and an example answer based on those excerpts.

Excerpt:

'We compared the proportion of cells representing each CN within a given tumor sample and found that LTS tumors had significantly higher representation of macrophage-enriched CN7 than STS tumors (Fig. 3d). Moreover, using this refined cohort, we confirmed the association between CN7 and improved survival (Fig. 3e and Extended Data Fig. 9g). This aligned with our neighborhood analysis using variable numbers of interacting cells, where CNs enriched in M1-like MDMs were associated with prolonged survival (Extended Data Fig. 7). Notably, CN2 and CN9 (pan-immune hotspots) were also associated with improved survival (Fig. 3e);'

Answer:

We can see that they are evaluating associations between cellular neighborhoods (CNs) and different tumors. Because the authors associated cellular neighborhoods with survival, we can confirm that the clinical variable in this case should be: 'survival'.

One-Shot Chain of Thought Prompt – Neighborhood Name

Example

This prompt contains parts of an academic paper represented as a string as context. This paper should describe cellular neighborhoods (CNs). You must extract the names of all of the cellular neighborhoods. Examples of names could be 'immune enriched', 'tumor stroma boundary', 'follicle', 'vasculature', etc. Do note that cellular neighborhoods are also often referred to as cellular types or even tissue compartments. The neighborhood names must be returned as a list of strings. I do not have specific neighborhood names, you can just match.

Below is an example of a relevant excerpt from an academic paper and a correct assignment of cellular neighborhood names:

Excerpt:

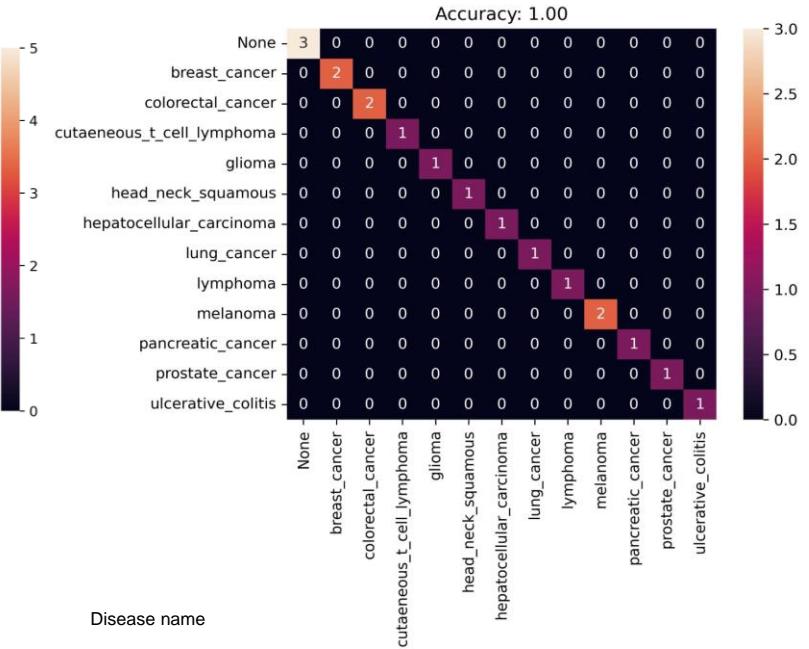
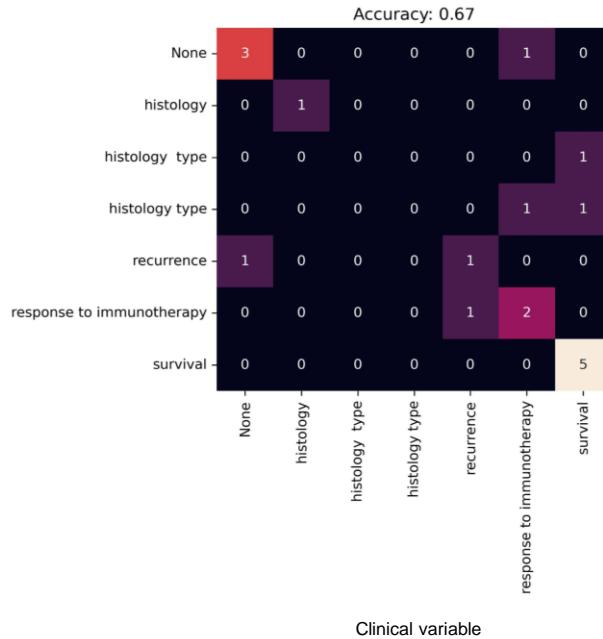
'We next compared multicellular interactions between glioblastoma and BrM. Using $N = 10$ nearest neighbors (the midpoint of our model and similar to other studies¹³), we identified 9 CNs across glioblastoma and BrM images (Fig. 3a,b and Extended Data Fig. 9a,b). The cellular composition of CNs recapitulated known tissue features, including the tumor boundary (CN1) or tumor compartment (CN8), two pan-immune hotspots with either high levels of all immune populations (CN2) or deficiencies in select subsets (CN9), high (CN3) or low (CN4) astrocytes, vascular niche (CN6), macrophage-enriched (CN7), and a neighborhood largely represented by cells undefined by our panel (CN5) (Fig. 3b). As expected, glioblastoma was dominated by CN3 and CN4 (astrocyte-enriched), whereas BrM-cores were enriched for CN8 (tumor compartment), reflecting the infiltrative'

Answer:

Here we can see a collection of names that refer to tissue structures (like tumor boundary) and cell groups (like pan-immune hotspot). We can see that all of these structures are marked by cellular neighborhood IDs like CN1, which will not always be the case for every paper but is useful to look for. Note that there is a mention of two pan-immune hotspots, so our final list should contain: 'pan-immune hotspot 1, pan-immune hotspot 2'. Using the information just described, we can assign the following list as the correct output:

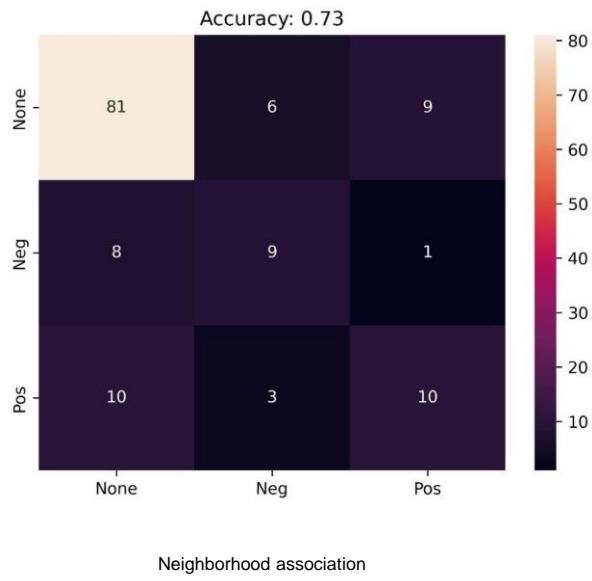
['tumor boundary', 'tumor compartment', 'pan-immune hotspot 1', 'pan-immune hotspot 2', 'astrocyte high', 'astrocyte low', 'vascular niche', 'macrophage enriched', 'undefined']

Pure Baseline



	rouge2_pr	rouge2_re	rouge2_fn	bertscore_bertscore	bertscore_bertscore	
0.689655	0.714286	0.701754	0.917667	0.925667	0.921649	
0.181818	0.095238	0.125	0.888138	0.867692	0.877796	
0	0	0	0.824859	0.776646	0.800026	
0.08	0.125	0.097561	0.863584	0.891812	0.877471	
0.62	0.596154	0.607843	0.890867	0.898802	0.894817	
0.166667	0.3	0.214286	0.844108	0.889732	0.86632	
1	1	1	0.965801	0.98066	0.973174	
0.55	0.55	0.55	0.943204	0.949728	0.946455	
0.08	0.181818	0.111111	0.815866	0.85366	0.834335	
0.178571	0.333333	0.232558	0.857271	0.883388	0.870134	
0.071429	0.285714	0.114286	0.812705	0.89518	0.851951	
0.684211	0.590909	0.634146	0.970098	0.957845	0.963933	
1	1	1	0.951399	0.970004	0.960611	
0.793103	0.92	0.851852	0.928945	0.955296	0.941936	
0.090909	0.037037	0.052632	0.8283	0.826826	0.827562	
0.619048	0.684211	0.65	0.910892	0.941266	0.92583	
0.866667	0.928571	0.896552	0.953324	0.971786	0.962466	
0.777778	0.777778	0.777778	0.983241	0.989411	0.986316	

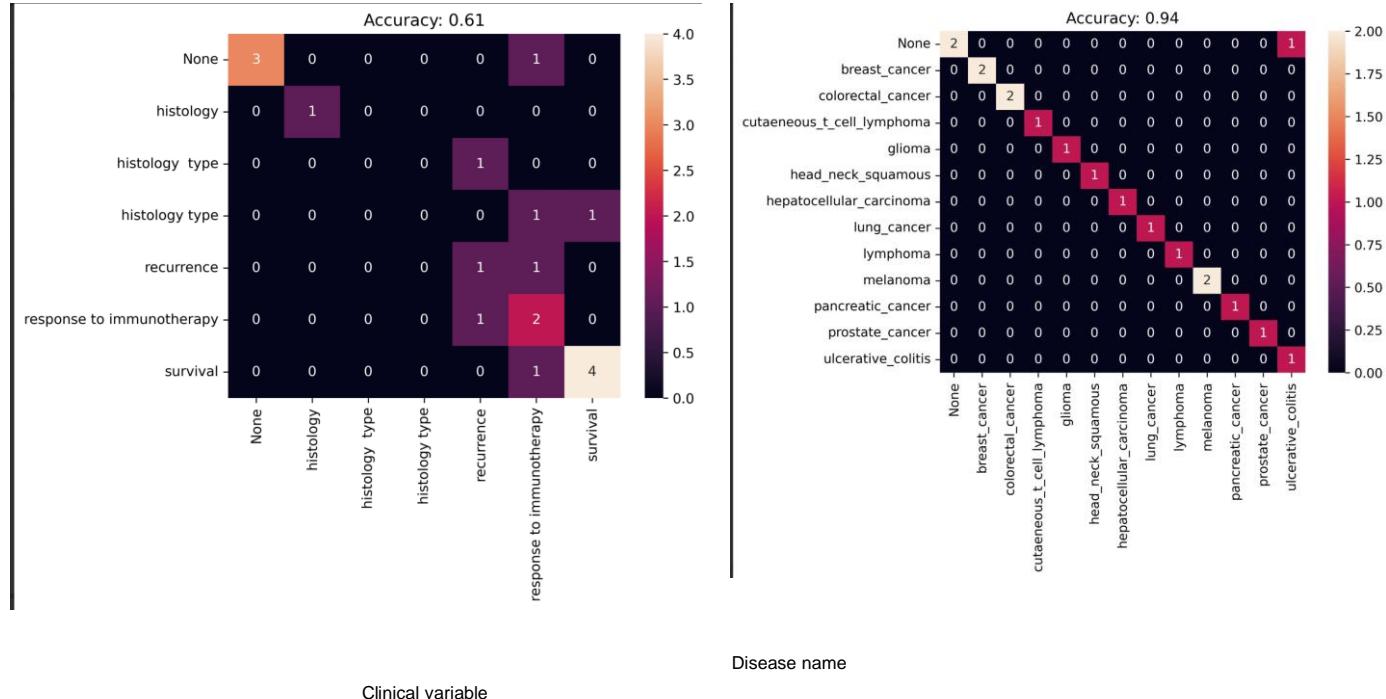
Neighborhood Names



Result

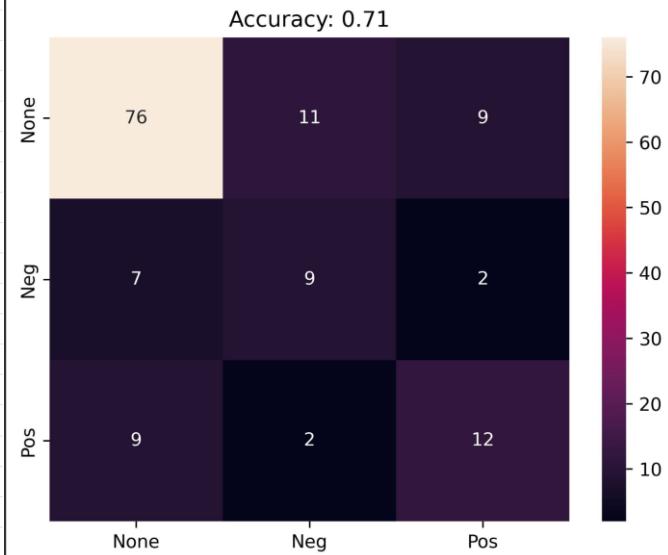
A	B	C	D	E	F	G	H	I
paper_names	disease_preds	clinical_variable_preds	neighborhood_name_neighborhood_assoc_preds					
ak_prostate	prostate_cancer	histology	AR+ non-immune stro	['None', 'None', 'None', 'None', 'None', 'None', 'None', 'None', 'None', 'None']				
blise_head_and_neck	head_neck_squamous	survival	compartmentalized, m	['Neg', 'Pos', 'Pos', 'Neg', 'Pos', 'None', 'None']				
blise_pancreatic	pancreatic_cancer	survival	immune aggregates, ti	['CD44+ CD8+ CD4+ T cells - Pos', 'proliferative t cells - None', 'Gr2B- T cells - None']				
eng_breast	breast_cancer	survival	quiescent stroma, tum	['macrophages vimentin+ fibroblasts cd4 t cells: None', 'tumor and t cell: Pos', 'macrophage and tumor: P				
hickey_intestine	None	None	Paneth cell enriched,	(['None', 'None', 'None'])				
hickey_melanoma	melanoma	response to immunotherapy	Proliferating Tumor, R	['None', 'None', 'None', 'None', 'None']				
jin_lymphoma	lymphoma	None	CD8+ T cell-enriched, t	['CD8+ T cell enriched - Pos', 'tumor cell enriched - Neg']				
karimi_brain	glioma	survival	tumour boundary, par	['None', 'Neg', 'Pos', 'None', 'Pos', 'Neg', 'Pos', 'Pos', 'Neg']				
lake_kidney	None	None	Nephron segments, In	['None', 'None', 'None', 'None', 'None']				
lemaire_hepatocellular	hepatocellular_carcinoma	survival	M2-macrophage imm	['Neg', 'Pos', 'None', 'None', 'Neg', 'Neg']				
maus_melanoma	melanoma	recurrence	tumor core, regional c	['None', 'None', 'Pos']				
mayer_colitis	ulcerative_colitis	response to immunotherapy	B cell follicle, lymphoic	(['None', 'None', 'None'])				
mi_breast	breast_cancer	survival	Treg and Tex-enriched	(['Treg and Tex enriched - None', 'tumor compartment - Neg', 'CD57+ enriched - None', 'tumor boundary - epithelium, immune-ir'])				
phillips_lymphoma	cutaeneous_t_cell_lymphoma	response to immunotherapy		['None', 'None', 'None', 'None', 'Pos', 'Pos', 'None', 'Neg']				
shovik_marrow	None	None	Adipocyte Niche, HSPC	['None', 'None', 'None']				
shurch_crc	colorectal_cancer	survival	T cell enriched, macro	(['Follicle: Pos', 'T cell enriched: None', 'macrophage enriched: None', 'granulocyte enriched: Pos', 'bulk tu'])				
sorin_lung	lung_cancer	response to immunotherapy	tumor stroma, vascula	(['tumor stroma: None', 'vascular: Pos', 'pan-immune 1: None', 'pan-immune 2: None', 'tumor core: None'])				
su_crc	colorectal_cancer	recurrence	P53+ tumor, stromal, t	(['Neg', 'Neg', 'Neg', 'Neg', 'None', 'Neg', 'None', 'Neg'])				

One Shot Chain of Thought



rouge2_pr	rouge2_re	rouge2_fn	bertscore_bertscore	bertscore_fmeasure	
0.395349	0.607143	0.478873	0.886703	0.906281	0.896385
0.108108	0.190476	0.137931	0.851983	0.891009	0.871059
0	0	0	0.808208	0.767624	0.787393
0.066667	0.125	0.086957	0.84278	0.87217	0.857223
0.52	0.25	0.337662	0.937782	0.885835	0.911068
0.2	0.4	0.266667	0.888656	0.919009	0.903578
0.625	0.833333	0.714286	0.940546	0.967603	0.953882
0.9	0.9	0.9	0.97978	0.982575	0.981176
0.033898	0.181818	0.057143	0.801155	0.860375	0.82971
0.333333	0.466667	0.388889	0.885033	0.889935	0.887478
0.032967	0.428571	0.061224	0.788363	0.865707	0.825227
0.75	0.681818	0.714286	0.96654	0.95959	0.963052
0.875	0.875	0.875	0.927684	0.943882	0.935713
0.793103	0.92	0.851852	0.934266	0.95757	0.945774
0	0	0	0.837067	0.788406	0.812008
0.257143	0.473684	0.333333	0.855733	0.917948	0.885749
0.866667	0.928571	0.896552	0.953324	0.971786	0.962466
0.722222	0.722222	0.722222	0.981558	0.987717	0.984628

Neighborhood Names

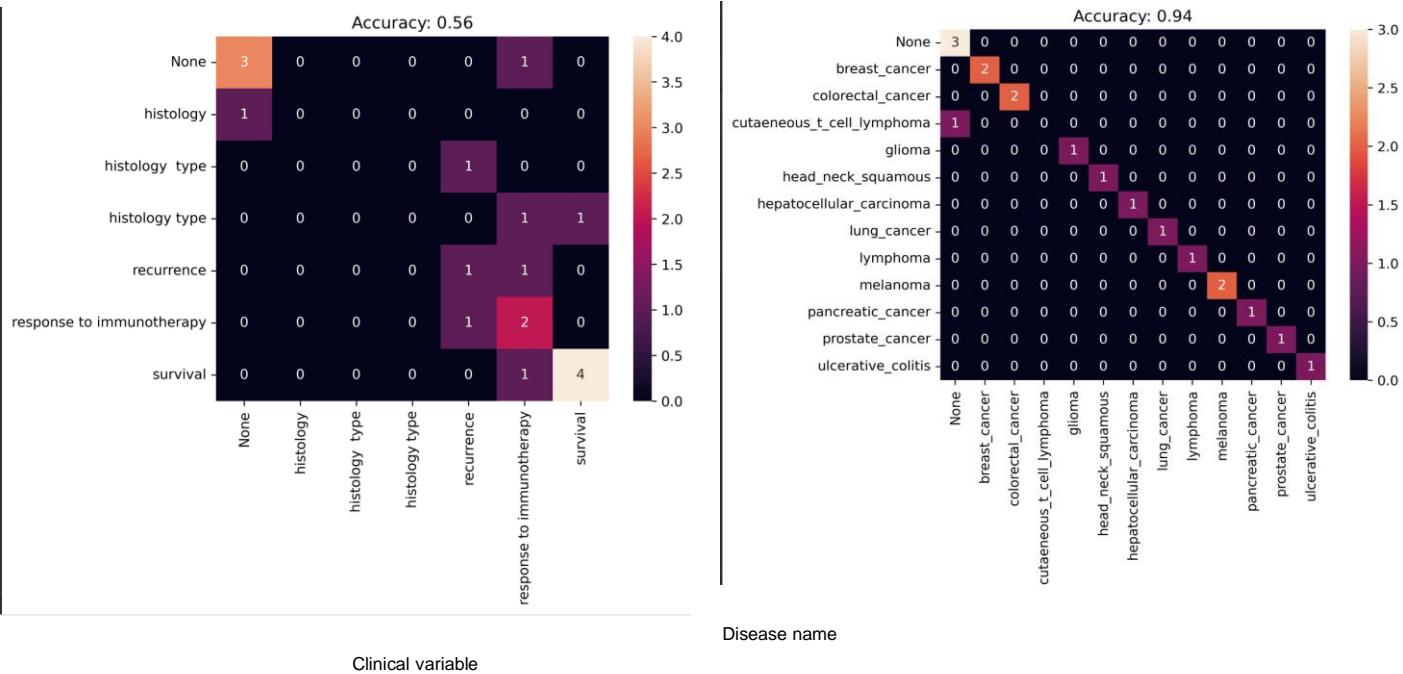


Neighborhood association

Result

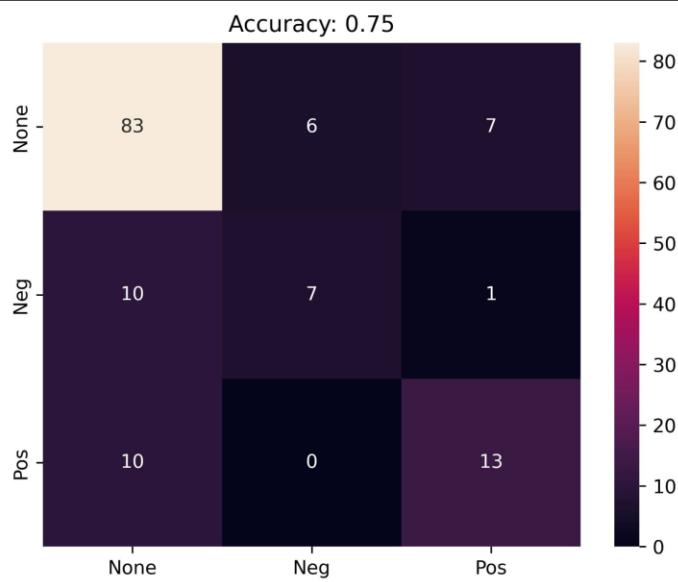
paper_nar disease_preds	clinical_variable_preds	neighborhood_name_preds	neighborhood_assoc_preds
ak_prostate prostate_cancer	histology	mast cell and M2 macrophage, mast cell and Treg	['None', 'None', 'None', 'None', 'None', 'None', 'None', 'None', 'None', 'None']
blise_head_neck_squamous	survival	neoplastic tumor-immune compartment, \pm SMA	['Neg', 'None', 'Pos', 'Neg', 'Pos', 'Pos', 'None']
blise_pancreatic_cancer	survival	immune aggregate, tumor-adjacent stroma, nor	['Pos', 'None', 'None']
eng_breast breast_cancer	survival	quiescent stromal proximal to tumor, mixed fibr	['None', 'None', 'Pos', 'Neg', 'None']
hickey_int ulcerative_colitis	None	microvasculature, macrovasculature, innervated	['None', 'None', 'None', 'None', 'None', 'None', 'None', 'None', 'None', 'None']
hickey_me melanoma	response to immunotherapy	immune infiltrate, inflamed tumor, productive T	['None', 'None', 'None', 'None', 'None', 'None']
jin_lymph lymphoma	response to immunotherapy	CD8+ T cell-enriched CN, tumor cell-enriched CM	['Pos', 'Neg']
karimi_bra glioma	survival	tumor boundary, tumor compartment, pan-immr	['None', 'None', 'Pos', 'None', 'Pos', 'Pos', 'None', 'Neg']
lake_kidn None	None	renal corpuscle, juxtaglomerular apparatus, vas	['None', 'None', 'None', 'None', 'None', 'None']
lemaire_t hepatocellular_carcinoma	recurrence	M2-macrophage immune, T cell immune, CK19+	['Neg', 'Neg', 'Pos', 'Pos', 'Neg', 'None']
maus_mel melanoma	recurrence	tumor and dendritic cells, dendritic cells in cytot	['None', 'None', 'Pos']
mayer_col ulcerative_colitis	response to immunotherapy	B cell follicle, lymphoid aggregate, mixed immur	['None', 'None', 'Neg', 'None', 'None', 'Pos', 'Neg', 'Neg', 'None', 'None']
mi_breast breast_cancer	response to immunotherapy	B cell-enriched, fibroblast-enriched, tumor comp	['Neg', 'Neg', 'None', 'None', 'Pos', 'Pos', 'None', 'None', 'None', 'None']
phillip_ly cutaneous_t_cell_lymphoma	response to immunotherapy	epithelium, vasculature, immune-infiltrated strc	['Neg', 'None', 'Pos', 'None', 'Pos', 'None', 'Neg']
shovik_m None	None	arterio-endosteal neighborhood, adipocyte nich	['None', 'None', 'None', 'None', 'None', 'None', 'None', 'None', 'None', 'None']
shurch_cr colorectal_cancer	survival	granulocyte-enriched neighborhood, T cell-enrich	['Neg', 'Pos', 'Neg', 'None', 'None', 'None', 'None', 'None', 'None', 'None']
sorin_lung lung_cancer	response to immunotherapy	tumor stroma, vascular niche, pan-immune-1, p	['Neg', 'Pos', 'Pos', 'None', 'Pos', 'Pos', 'Pos']
su_crc colorectal_cancer	recurrence	TP53+ tumor, stromal, bulk tumor, immune-enriched	['Neg', 'Neg', 'None', 'None', 'Neg', 'None', 'None']

RAG Baseline



rouge2_p	rouge2_r	rouge2_f	bertscore_bertscore	bertscore_fmeasure	
0	0	0	0.824079	0.771342	0.796839
0.352941	0.285714	0.315789	0.931081	0.914856	0.922897
0	0	0	0.8257	0.784209	0.80442
0.038462	0.0625	0.047619	0.838728	0.858803	0.848647
0.489796	0.461538	0.475248	0.891187	0.905429	0.898252
0.214286	0.3	0.25	0.847524	0.87251	0.859836
0.625	0.833333	0.714286	0.940546	0.967603	0.953882
0.55	0.55	0.55	0.943204	0.949728	0.946455
0	0	0	0	0	0
0.172414	0.333333	0.227273	0.855233	0.883236	0.869009
0	0	0	0.808441	0.837284	0.82261
0.75	0.681818	0.714286	0.96654	0.95959	0.963052
1	1	1	0.947181	0.9681	0.957526
0.733333	0.44	0.55	0.912493	0.917207	0.914844
0.090909	0.037037	0.052632	0.833409	0.829029	0.831213
0.285714	0.315789	0.3	0.8529	0.890676	0.871379
0.866667	0.928571	0.896552	0.953324	0.971786	0.962466
0.777778	0.777778	0.777778	0.983241	0.989411	0.986316

Neighborhood Names

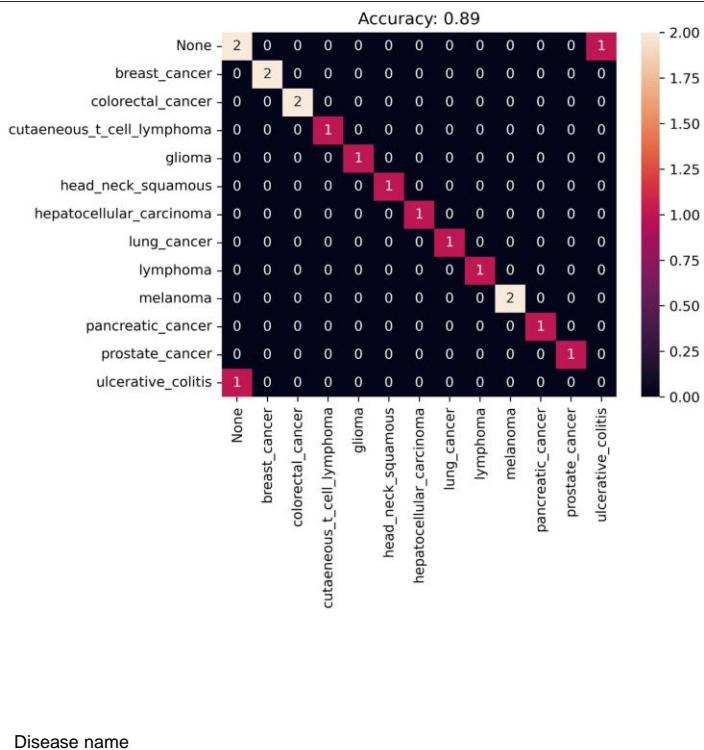
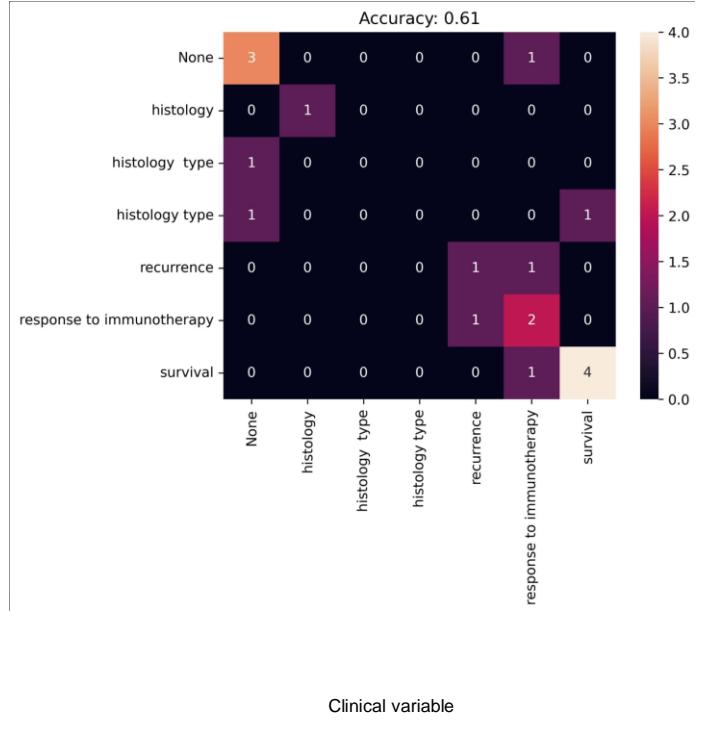


Neighborhood association

Result

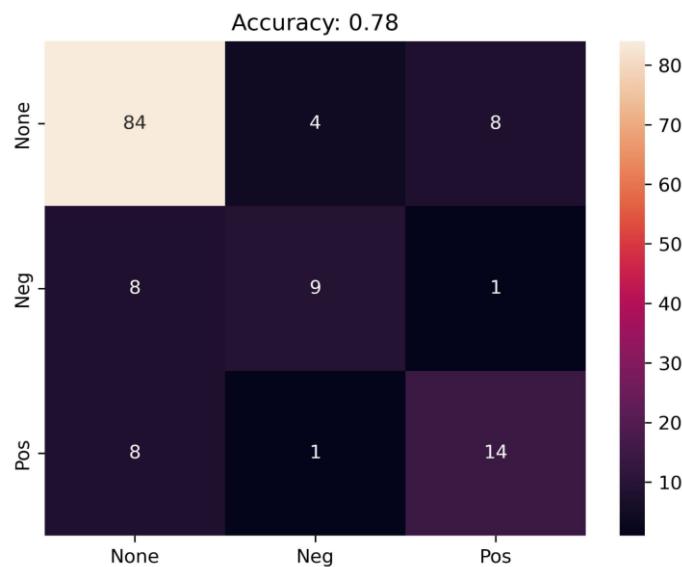
paper_nar	disease_p	clinical_va	neighborh	neighborhood_assoc_preds
ak_prostal	prostate_c	None	Immune	e ['None', 'None', 'None', 'Neg', 'None', 'None', 'Pos', 'None', 'None', 'None']
blise_head	head_necl	survival	†±SMA+	ce ['Pos', 'Pos', 'Neg', 'None', 'Pos', 'Pos', 'Pos', 'Neg']
blise_panc	pancreatic	survival	RCN1, RCN	['Pos', 'None', 'Neg']
eng_breas	breast_cai	survival	tumor ma	['Neg', 'None', 'None', 'Neg', 'None']
hickey_int	None	None	Inner Foll	i ['None', 'None', 'None']
hickey_me	melanom	response	t Immune Ir	['None', 'None', 'None', 'None', 'None']
jin_lymph	lymphom	response	t CD8+ T cel	['Pos', 'Neg']
karimi_bra	glioma	survival	tumour bc	['None', 'None', 'Pos', 'Pos', 'None', 'None', 'None', 'Pos', 'Neg']
lake_kidne	None	None		['None', 'None', 'None', 'None', 'None']
lemaire_l	hepatocell	recurrence	M2-macro	['None', 'None', 'None', 'None', 'Neg', 'Neg']
maus_mel	melanom	recurrence	tumor-cen	['None', 'None', 'Pos']
mayer_col	ulcerative	response	t B cell follic	['None', 'None', 'None', 'Pos', 'None', 'None', 'None', 'None', 'None', 'Pos', 'Pos']
mi_breast	breast_car	response	t Treg and T	['None', 'Neg', 'None', 'None', 'Pos', 'Pos', 'None', 'None', 'None', 'None']
phillips_ly	None	response	t epithelium	['None', 'None', 'None', 'None', 'None', 'None', 'None', 'None', 'None', 'Neg']
shovik_me	None	None	Adipocyte	['None', 'None', 'None', 'None', 'None', 'None', 'None', 'None', 'None', 'None']
shurchn_cr	colorectal	survival	T cell enric	['None', 'None', 'None', 'Pos', 'None', 'Neg', 'Pos', 'None', 'None']
sorin_lung	lung_canc	response	t tumor strc	['None', 'Pos', 'None', 'None', 'None', 'Pos', 'None']
su_crc	colorectal	recurrence	P53+ tumc	['None', 'None', 'None', 'None', 'Pos', 'None', 'None', 'None']

RAG One shot



	rouge2_p	rouge2_re	rouge2_fn	bertscore	bertscore_fmeasure	
0.733333	0.392857	0.511628	0.950177	0.912802	0.931115	
0.304348	0.333333	0.318182	0.901172	0.897422	0.899293	
0	0	0	0.819484	0.782472	0.80055	
0.038462	0.0625	0.047619	0.833387	0.861235	0.847082	
0.408163	0.384615	0.39604	0.846342	0.878747	0.86224	
0.263158	0.5	0.344828	0.834062	0.893137	0.862589	
0.625	0.833333	0.714286	0.940546	0.967603	0.953882	
0.9	0.9	0.9	0.987858	0.987361	0.987609	
0.117647	0.181818	0.142857	0.849895	0.852935	0.851412	
0.105263	0.266667	0.150943	0.839518	0.874641	0.85672	
0	0	0	0.793898	0.845663	0.818964	
0.619048	0.590909	0.604651	0.931875	0.934913	0.933391	
1	1	1	0.947181	0.9681	0.957526	
0.793103	0.92	0.851852	0.926849	0.947497	0.937059	
0	0	0	0.802094	0.783533	0.792705	
0.882353	0.789474	0.833333	0.955956	0.943305	0.949588	
0.866667	0.928571	0.896552	0.953324	0.971786	0.962466	
0.538462	0.777778	0.636364	0.927204	0.967496	0.946922	

Neighborhood Names



Neighborhood association

Result

paper_nar	disease_p	clinical_va	neighborh	neighborhood_assoc_preds
ak_prostat	prostate_c	histology	neuroend	['None', 'None', 'None', 'None', 'None', 'None', 'None', 'None', 'None', 'None']
blise_head	head_necl	survival	†SMA+ ce	['Pos', 'None', 'Neg', 'None', 'Pos', 'Pos', 'Neg']
blise_pan	pancreatic	survival	germinal c	['Pos', 'Pos', 'None']
eng_breas	breast_car	survival	tumor mai	['None', 'None', 'Neg', 'None', 'None']
hickey_int	ulcerative	None	Neuroend	['None', 'None', 'None', 'None', 'None', 'None', 'None', 'None', 'None', 'None']
hickey_me	melanoma	response	t Proliferati	['None', 'None', 'None', 'None', 'None']
jin_lymph	lymphoma	response	t CD8+ T cel	['Pos', 'Neg']
karimi_br	glioma	survival	tumor bol	['None', 'Pos', 'Pos', 'Pos', 'None', 'None', 'None', 'Pos', 'Neg']
lake_kidne	None	None	fibrotic nic	['None', 'None', 'None', 'None', 'None']
lemaire_f	hepatocell	None	CK19+ tun	['Neg', 'Pos', 'None', 'None', 'Neg', 'None']
maus_mel	melanoma	recurrence	tumor-cen	['Pos', 'None', 'Neg']
mayer_col	None	None	epitheliun	['None', 'None', 'Neg', 'Pos', 'Neg', 'None', 'None', 'None', 'None', 'Pos']
mi_breast	breast_car	response	t Treg and T	['None', 'Neg', 'None', 'None', 'Pos', 'Pos', 'None', 'None', 'None', 'None']
phillips_ly	cutaeneou	response	t epitheliun	['None', 'None', 'Pos', 'None', 'None', 'Pos', 'None', 'Neg']
shovik_ma	None	None	Adipocyte	['None', 'None', 'None', 'None', 'None', 'None', 'None', 'None', 'None', 'None']
shuruch_cr	colorectal	survival	T cell enric	['None', 'Pos', 'Neg', 'None', 'None', 'Pos', 'None', 'None', 'None', 'None']
sorin_lung	lung_canc	response	t tumor strc	['Neg', 'Pos', 'None', 'None', 'None', 'None', 'Pos']
su_crc	colorectal	recurrence	P53+ tumc	['None', 'None', 'None', 'None', 'Pos', 'None', 'None', 'None']

Analysis

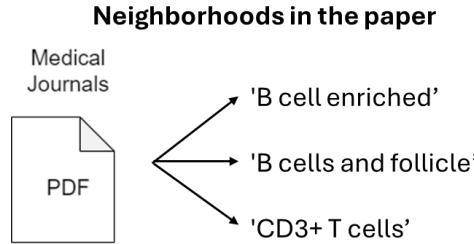
Immune cell topography predicts response to PD-1 blockade in cutaneous T cell lymphoma

Darci Phillips^{1,2,3,9}, Magdalena Matusiak^{3,9}, Belén Rivero Gutierrez³, Salil S. Bhate^{1,3,4}, Graham L. Barlow^{1,3},

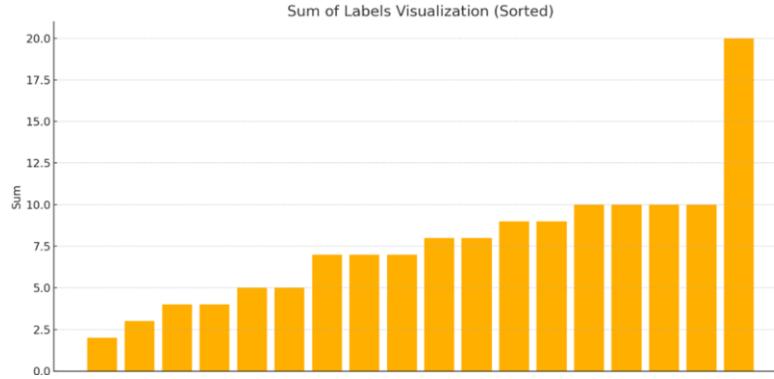
Processing PDF: /content/data/phillips_lymphoma.pdf
Extracted Title: Immune cell topography predicts response to PD-1 blockade in cutaneous T cell lymphoma
Extracted for paper_name: phillips_lymphoma.pdf
Extracted for paper_title: Automated mapping of phenotype space with single-cell data.
Extracted for collection_technology: CODEX
Extracted for tissue_type: Skin
Extracted for disease_name: Lymphoma
Extracted for clinical_variable: response to pembrolizumab, immunotherapy response, disease outcome, progression, prognosis, survival, disease-free survival, clinical grade
Extracted for unique_neighborhoods: ['Epithelium', 'Immune-infiltrated stroma', 'Vasculation', 'Vascularized stroma', 'Tumor and dendritic cells', 'Lymphatic enriched strom
Extracted for clinical_variable_association: ['positively associated with Ki-67 expression', 'positively associated with IDO expression', 'positively associated with PD-1 e
Extracted for association_summary: The clinical variable association summary was determined by modeling differences between responders and nonresponders, as well as pretrea

Prediction neighborhoods with fine tuning

- Multiclass Classification: Each paper has multiple labels



Number of labels in each paper



Prediction neighborhoods with fine tuning

- Problem 1: Sparsity of the label
- Number of neighborhoods: 138
- Unique neighborhoods: 127

• Prediction result

Loss	Accuracy	Correct Positive Predictions	Precision	Sensitivity
0.5396	0.8863	8/131	0.0684	0.0611

Prediction neighborhoods with fine tuning

- Problem 2: Complicated label names

- 'B cell enriched'
- 'B cells and follicle'
- 'CD3+ T cells'
- 'CD44+ CD8+ CD4+ T cells'
- 'CD57+ enriched'

- Solution: Label Categorization

- Ex:
- 'B cell enriched', 'B cells and follicle' => **B cells**
- 'CD3+ T cells', 'CD44+ CD8+ CD4+ T cells' => **T cells**

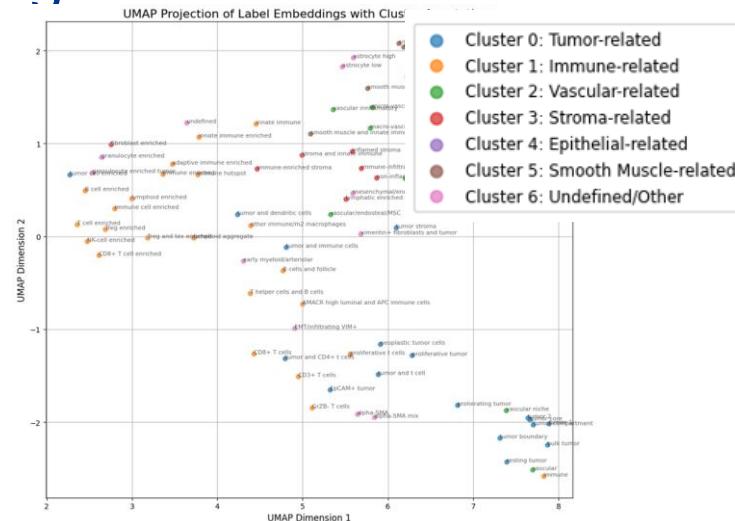
Prediction neighborhoods with fine tuning

- Solution: Label Categorization
 - 1. User defined labels- Manual categorization
 - Assign the cluster to the neighborhood names
 - Advantages

•- Proper categorization

• Weakness

- # • - Manual Categorizing



Prediction neighborhoods with fine tuning

- Automatic categorization
 - - Neighborhood names to vector using BioBert
 - - Assign the cluster to the neighborhood names

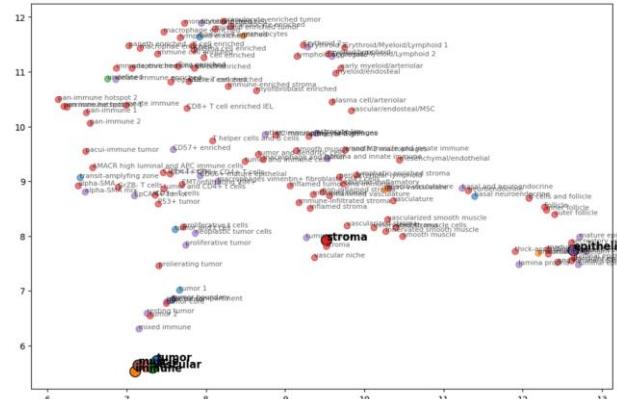
• Advantages

- ## • - Automatic categorizing

• - Customization

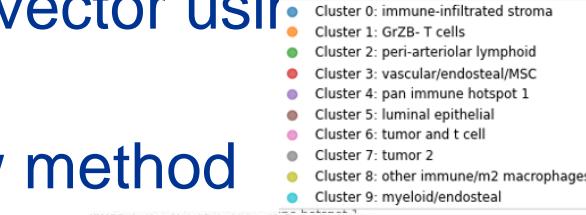
- **Weakness**

Accuracy	0.7471
Precision	0.7444
Sensitivity	0.7701
 - - Fixed location of pinpoints



Prediction neighborhoods with fine tuning

- 3. Using K-means and representative cell types
- - Neighborhood names to vector using BioBert
- - K - means with the elbow method
- - Assign the cluster to the neighborhood
- | Accuracy | Precision | Sensitivity |
|----------|-----------|-------------|
| 0.7412 | 0.7263 | 0.7931 |
- - Dynamic location of pinpoints (Centroids)



University of
Pittsburgh

References

- [1] Marx, V. Method of the Year: spatially resolved transcriptomics. *Nat Methods* 18, 9–14 (2021). doi: <https://doi.org/10.1038/s41592-021-01065-y>
- [2] Biopython: freely available Python tools for computational molecular biology and bioinformatics | Bioinformatics | Oxford Academic. <https://academic.oup.com/bioinformatics/article/25/11/1422/330687>.