EXPLORATORY TASK #4:

complete all initial

Attention Filters

&

start LLM Automation

Summary of Currently Implemented Patterns

Attention Patterns implemented so far:

- Linguistic Roles
  - Positional Patterns:
    - Near Token Attention ✓
    - Punctuation Attention ✓
    - Token Repetition Attention ✓
  - Linguistic Role Alignment:
    - Part-of-Speech Alignment ✓
    - Dependency Alignment ✓
  - Semi-structured Evaluation:
    - Chain of Thought Evaluation ✓

Methodology to **automate** filtering for patterns was started.

Scoring Methodologies:

Sum of Absolute Errors

```
score = np.abs(att - pred_att).sum()
```

Pros include simplicity & interpretability of method. It can also be extended to use the Mean Absolute Error instead. Con: Difficult to understand in practice

Jensen-Shannon Distance

```
for row_att, row_out in zip(att, pred_att):
   jsd_dist.append(sqrt(js_diverg(row_att,row_out))
score = np.mean(jsd_dist)
```

In practice, the normalized value from 0 to 1 is easier to understand / makes for a better score. However the equations aren't as intuitive.

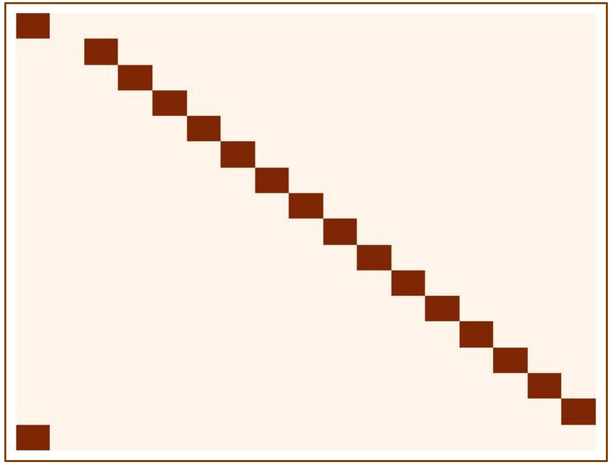
Reminder of Objective:

1. Classify attention heads into meaningful linguistic patterns or categories.
2. Determine whether a method to automate this categorization is possible and / or effective.

“Optimal” Pattern Heads Visualized

SECTION 1/3 | Linguistic Role: **Positional Patterns**

Near Token Attention



Sentence: “When it was time to go home, Beep knew he needed more fuel.”

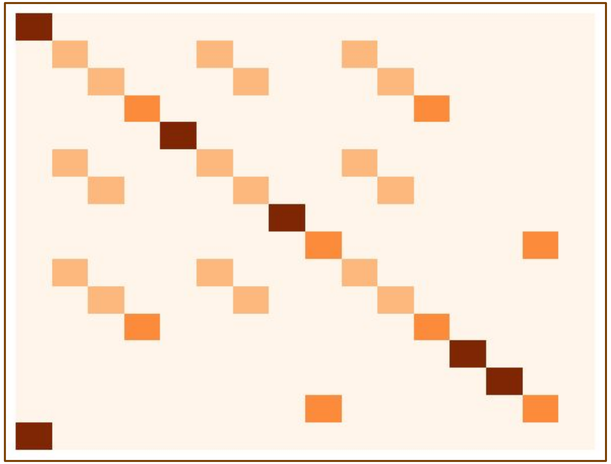
Def: Assigns 100% Attention to next / same / prev token

Punctuation Attention

Sentence: “Hi. How are you? I'm fine! Thanks. Bye, see you tomorrow.”

Def: Assigns Uniform Attention to all upcoming punctuation tokens

Repetition Attention



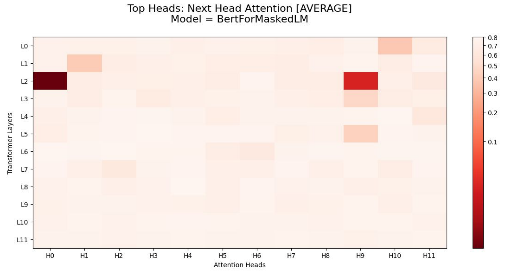
Sentence: "I like apples and I like bananas. I like apples more though."

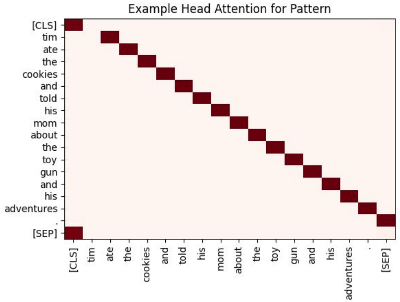
Def: Assigns Uniform Attention to all repeated tokens

TEST Examples in tiny-stories DATASET

SECTION **1/3** | Linguistic Role: **Positional Patterns**

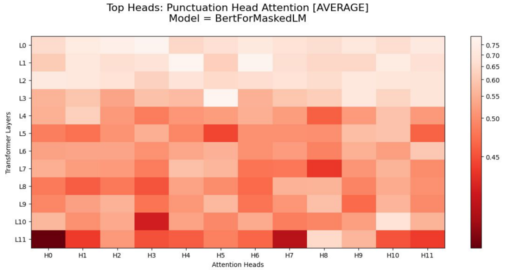
Next Token Attention

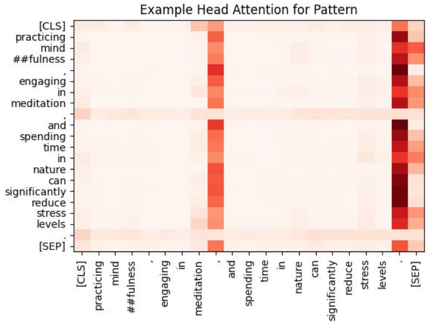




BERT Top Heads:
2.0*, 2.9, 0.10

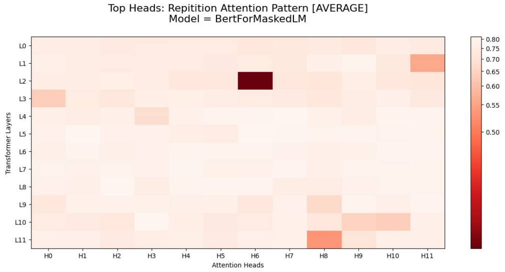
Punctuation Attention

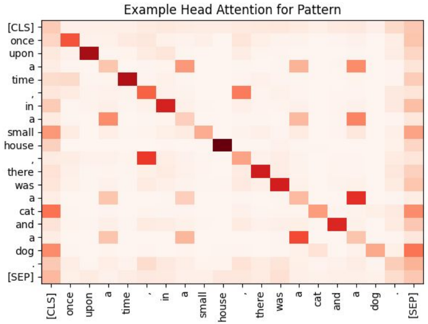




BERT Top Heads:
11.0, 10.3, 11.7*

Repetition Attention



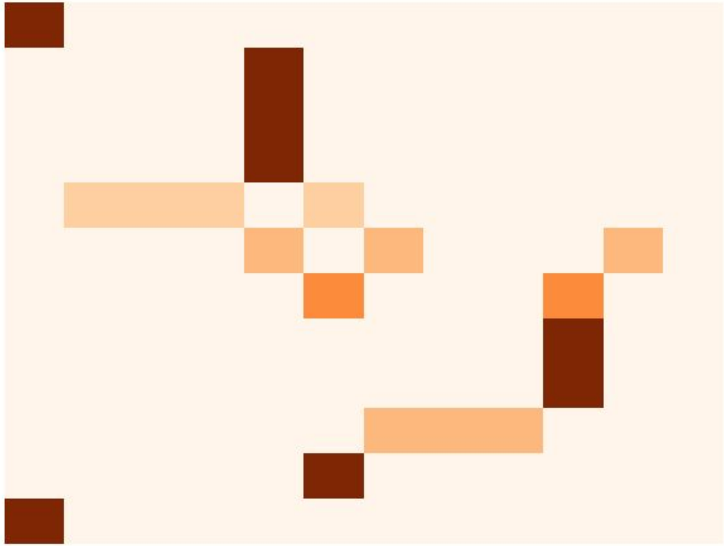


BERT Top Heads:
2.6*, 11.8, 1.11

“Optimal” Pattern Heads Visualized

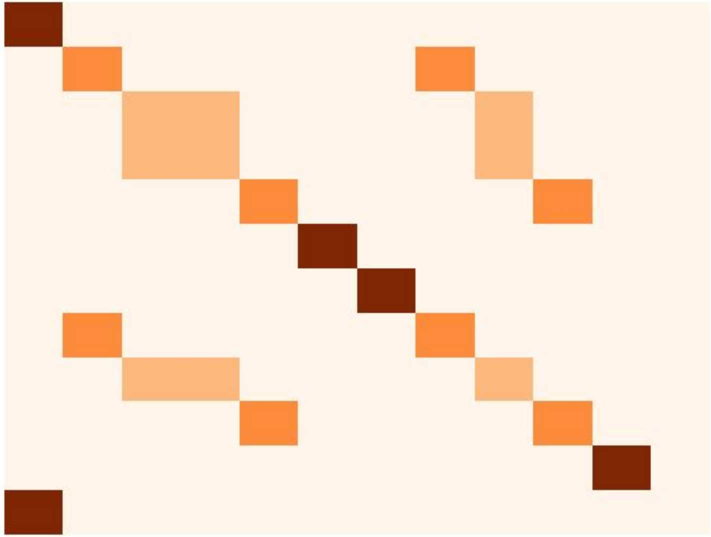
SECTION 2/3 | Linguistic Role: **Linguistic Role Alignment**

Dependency Alignment



Def: Uses spaCy to generate dependency tree.
Assigns uniform and bidirectional attention
between each word and its syntactic children.

Part-of-Speech Alignment



Def: Uses spaCy to generate POS tags for ∇
words.
Assigns uniform attention to all other tokens of the
same POS tag type. CLS and EOS attend to self.

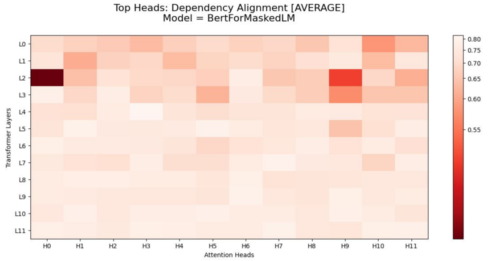
Sentence:

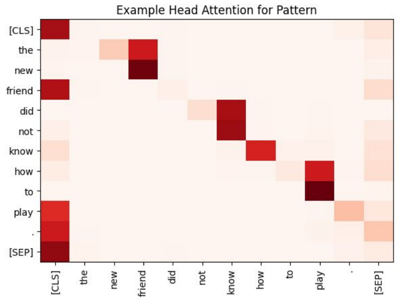
The quick brown fox jumps over the lazy dog.

TEST Examples in tiny-stories DATASET

SECTION **2/3** | Linguistic Role: **Linguistic Role Alignment**

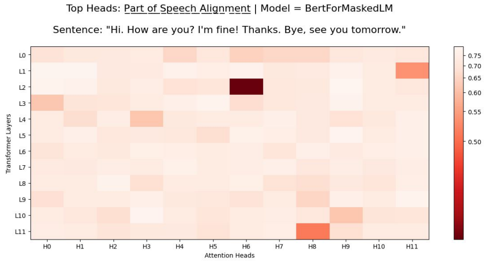
Dependency Alignment

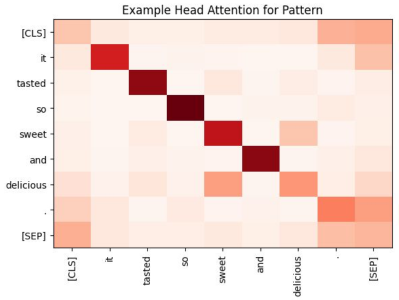




BERT Top Heads: 2.0, 3.9*, 2.9

Part-of-Speech Alignment





BERT Top Heads: 2.6*, 11.8, 0.6

“Optimal” Pattern Heads Visualized

SECTION **3/3** | Linguistic Role: **Semi-structured Evaluation**

Chain-of-Thought Evaluation

~~**MODEL:** H100 Llama 8B~~
[slurm difficulties, h100 down]

MODEL:
DATA: Abstract Algebra

True Att: After Hint
Pred Att: Before Hint

* Data was filtered: only sentences LLM changes its answer for if given hint *

Def: Assumes optimal attention is the one which results in the correct answer. Scores this compared to attention that gives correct answer after normalizing and column comparison

TEST Examples in algorithmic DATASET

SECTION **3/3** | Linguistic Role: **Semi-structured Evaluation**

Chain-of-Thought Evaluation

examples of high
CoT pattern heads
coming soon

Logic written,
CoT examples coming soon

PATTERN RECOGNITION via LLM AUTOMATION

1. Generate examples of attention head patterns

⬇

2. Transform heads into a parseable dataset for LLM

⬇

3. Have LLM hypothesize head function from data

⬇

4. Have LLM write filtering pattern using hypothesis

⬇

5. Use generated pattern code to test LLM hypothesis

Algorithm Particularities:

Yellow Portions are currently done by hand
Blue Portions are currently semi-automated

Additional Notes -

For Step 1, In practice I generated examples of similar attention heads by looping through heads on text data for a small number of sentences to see whether the activations had similarities.

For Step 2, In practice I transformed results of step 1 into an easier format for an LLM to understand by converting most common token activation patterns to JSON with keys

PATTERN RECOGNITION via LLM AUTOMATION

Results from Manual Walkthrough:

1. Started with Head 7.1 in BERT which seemed to cluster attention around verbs
2. Applied full automation algorithm from previous slide and visualized agreement

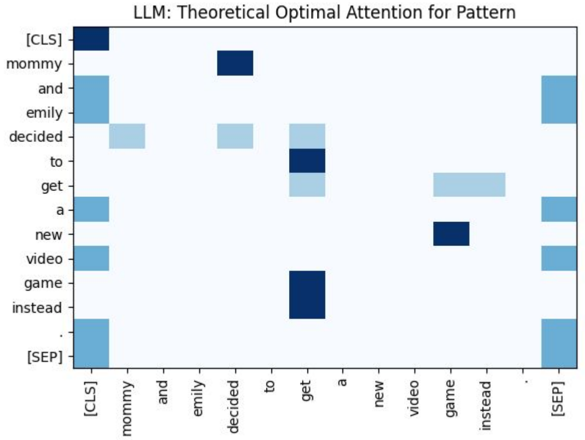
LLM Prompt Notes [our chat]

- Provided Gemini with:
  1. model_name, layer #, head #
  2. JSON with top 10% activations / sentence
  3. examples: dependency & part of speech

- Specified that it should
  1. generate 3 hypotheses first
  2. pick the most-fitting hypothesis
  3. validate this hypothesis with evidence
  4. create an example text matrix
  5. implement the Python function
  6. Check matrix is valid before returning

Gemini 2.5 Generated Filter:

verb_phrase_modifier_attention()



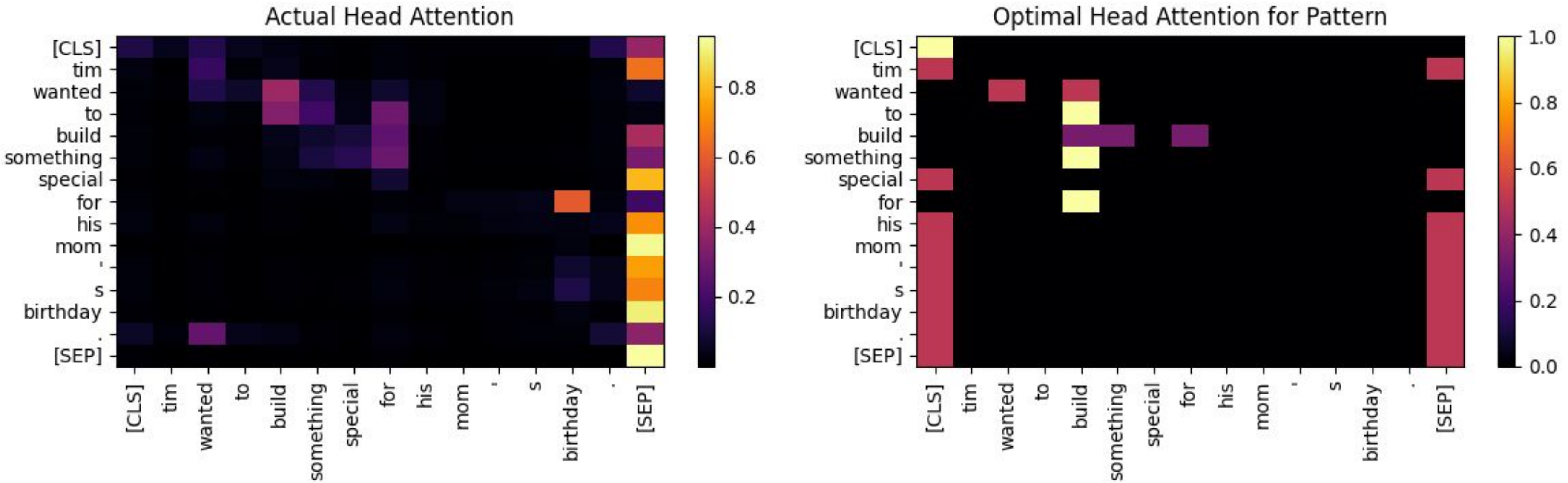
LLM Prediction: Verb <-> Obj, Verb <->Adpos

PATTERN RECOGNITION via LLM AUTOMATION

Example 2:

Results: Verb-Related Phrase and Modifier Focus @ L7,H1 | Raw Score = 0.55

Sentence: "Tim wanted to build something special for his mom's birthday."



Generally Interesting Notes from Analysis

[**CURRENT CODE**]

1. Simple patterns are present in more complex patterns
   → Next Token Attention Heads seem to be building blocks
   for repetition-oriented and dependency-oriented heads

2. The middle layers of BERT are underrepresented in initial analysis
   → Could be from theory that middle layers of large language
   models and neural networks in general are less interpretable

3. Looks like some patterns are binary, where they switch all attention
   to the sep & last token if pattern is not relevant to sentence
   → Punctuation pattern did this when there were no punctuations

Thoughts, Questions, Next Steps

Thoughts.

1. Some patterns should be improved: symmetrical -> unidirectional
2. Additional experimental validation of Patterns could be conducted.

Questions.

1. Chain-of-Thought Evaluation: good dataset & implementation?
2. Automation: best prompt / data structure?

Next Steps [non-exhaustive list]:

1. Continue Automation of Pattern Filters (create automation pipeline)
2. Improve implementation of existing patterns & extend to other LLM’s

MSRP Notes:

Fall Extension form open 7/17, Poster Draft due 7/27

Planned MSRP Poster Organization

NLP Background & Problem Statement

Interesting img of NLP interp task

Description of filters implemented (second slide)

Description of automation process

Results & Conclusions

Impact & Future Work (mention python library)

Citations