Multimodal Model for Diagram Question Answering

GWU Capstone Fall 2022 - DATS 6501_80
Final Presentation
Alexis Kaldany, Joshua Ting
GitHub Repo

Background

A Diagram Is Worth A Dozen Images

Kembhavi et al. (2016)[3]

- **Interpretation and reasoning of scientific diagrams from elementary school science textbooks**

- Visual Question and Answering Task with complex feature engineering

- Question/image pairing is sufficiently complex that normal VQA models don’t work

- Paper solves this problem by **creating a new network architecture called *DQA-Net***

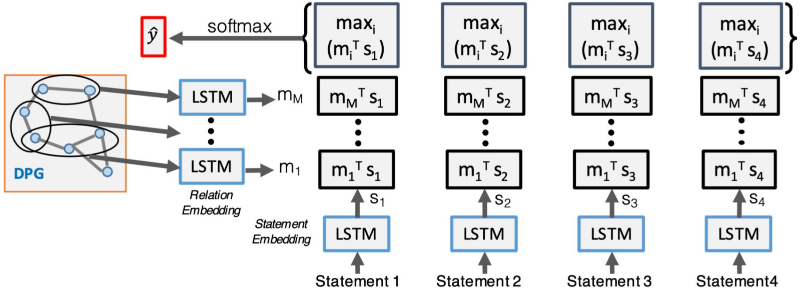


Fig. 3. An overview of the DQA-NET solution to diagram question answering. The network encodes the DPG into a set of facts, learns to attend on the most relevant fact, given a question and then answers the question.

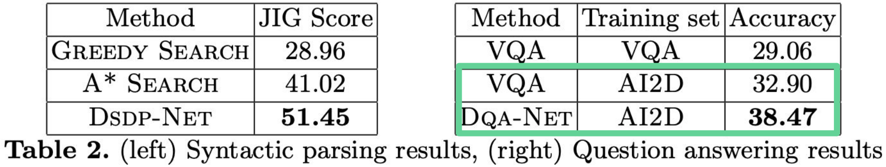
Figure 1: DQA-Net architecture[3]

Outcome of A Diagram Is Worth A Dozen Images

Kembhavi et al. (2016)[3]

- New Dataset:
  - **Diagrams with exhaustive annotations** of constituents and relationships

- Results:
  - Their best model reached **38.47% accuracy** with their DQA-NET model.
  - Pure VQA model reached **32.90% accuracy**



Method	JIG Score
Greedy Search	28.96
A* Search	41.02
Dsdp-Net	**51.45**

Method	Training set	Accuracy
VQA	VQA	29.06
VQA	AI2D	32.90
Dqa-Net	AI2D	**38.47**

Table 2. (left) Syntactic parsing results, (right) Question answering results

Figure 4: Benchmark results[3]

Contribution Goals to Original Paper

1. **Transformers and/or Auto Encoder-Decoder** architectures with pretraining
   a. Achieved impressive results in complex domains (NLP, CV, Speech)

2. **Reduce the complexity** during featuring engineering
   a. No separate model to create Diagram Parse Graphs (DPGs)

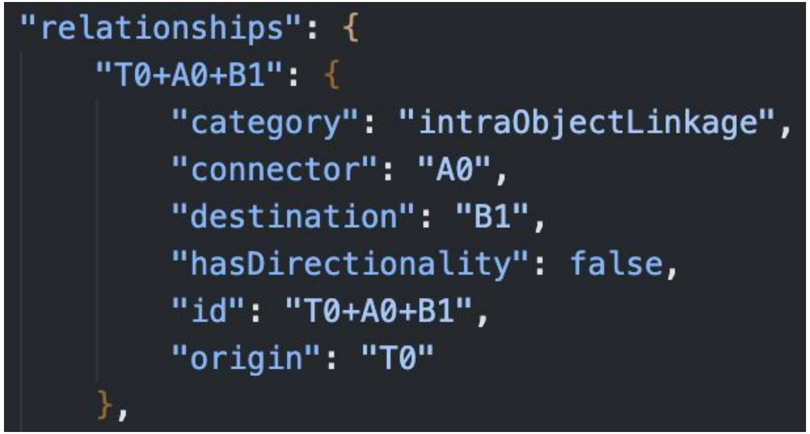
3. Use a pretrained model and try **Transfer Learning**

4. Try to achieve **comparable results**

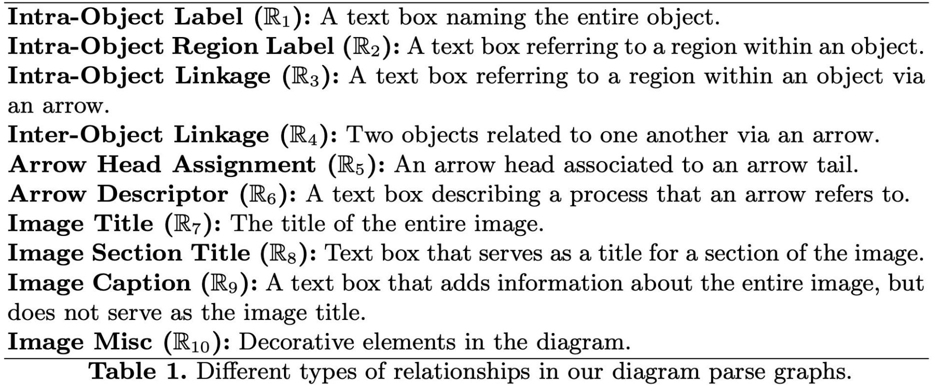
Data

Dataset Overview

- 5,000 **diagrams**, **annotations**
- 15,000 **questions and answers**
- Annotation json contains coordinates and linkages between detected objects
- Question json contains question - answer pairings



```
"relationships": {
    "T0+A0+B1": {
        "category": "intraObjectLinkage",
        "connector": "A0",
        "destination": "B1",
        "hasDirectionality": false,
        "id": "T0+A0+B1",
        "origin": "T0"
    },
```

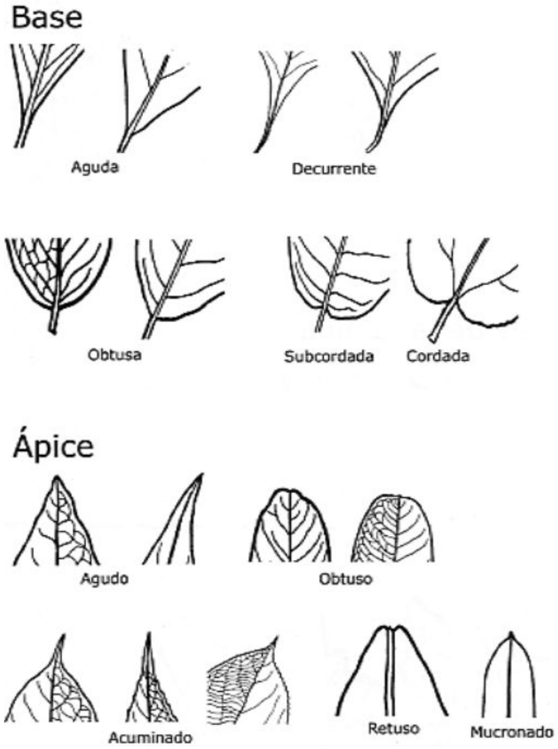


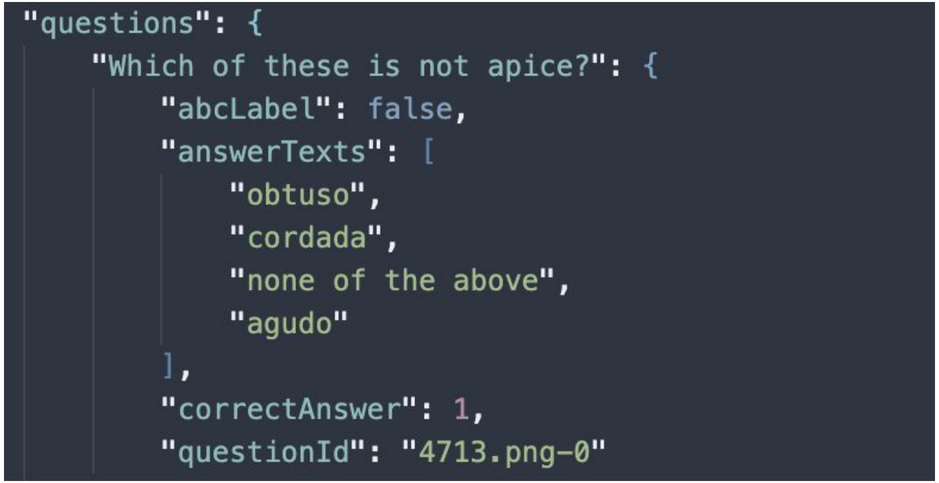
Intra-Object Label ($\mathbb{R}_1$): A text box naming the entire object.
Intra-Object Region Label ($\mathbb{R}_2$): A text box referring to a region within an object.
Intra-Object Linkage ($\mathbb{R}_3$): A text box referring to a region within an object via an arrow.
Inter-Object Linkage ($\mathbb{R}_4$): Two objects related to one another via an arrow.
Arrow Head Assignment ($\mathbb{R}_5$): An arrow head associated to an arrow tail.
Arrow Descriptor ($\mathbb{R}_6$): A text box describing a process that an arrow refers to.
Image Title ($\mathbb{R}_7$): The title of the entire image.
Image Section Title ($\mathbb{R}_8$): Text box that serves as a title for a section of the image.
Image Caption ($\mathbb{R}_9$): A text box that adds information about the entire image, but does not serve as the image title.
Image Misc ($\mathbb{R}_{10}$): Decorative elements in the diagram.

Table 1. Different types of relationships in our diagram parse graphs.

Figure 5: Annotations[2]

Example: Question - Image Pairs

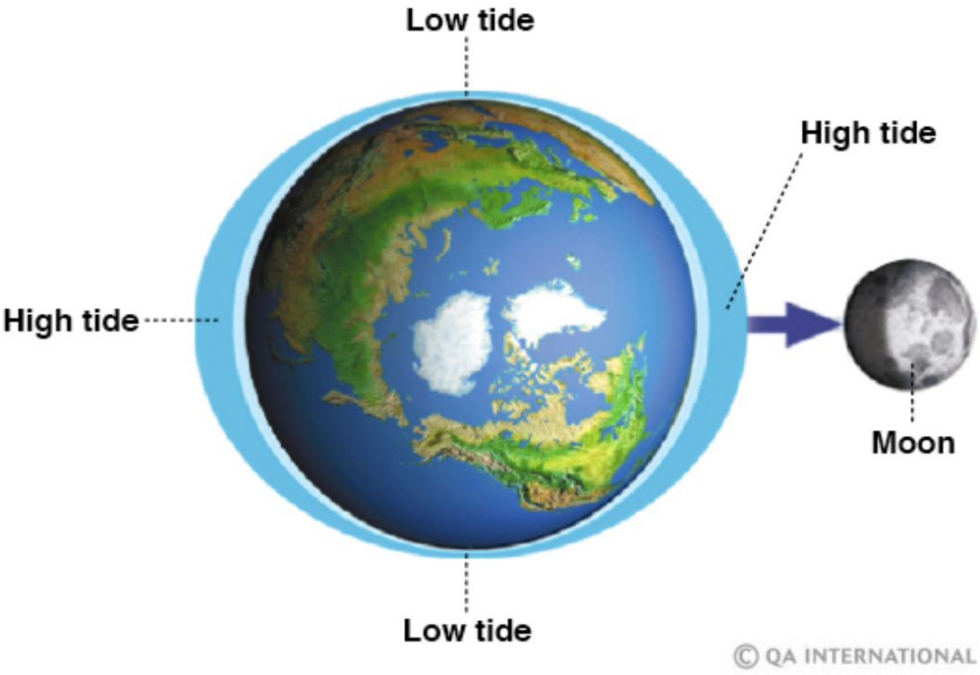




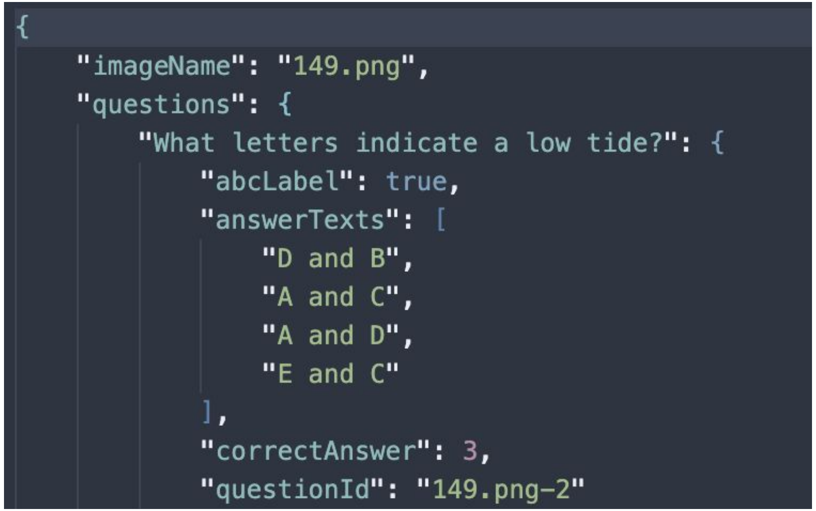
```
"questions": {
    "Which of these is not apice?": {
        "abcLabel": false,
        "answerTexts": [
            "obtuso",
            "cordada",
            "none of the above",
            "agudo"
        ],
        "correctAnswer": 1,
        "questionId": "4713.png-0"
```

Figure 6: Data sample[2]

Example II: Question - Image Pairs

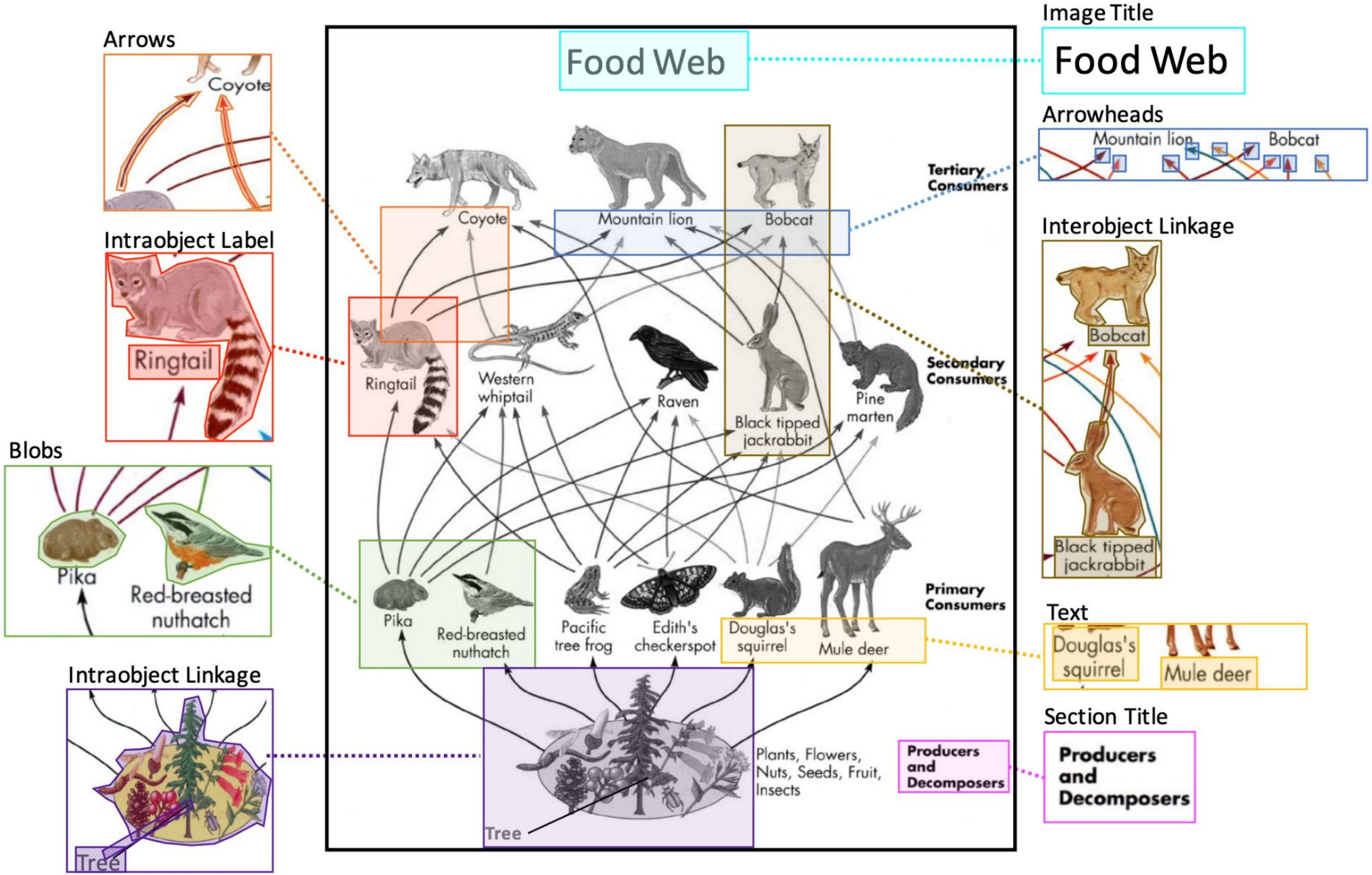


© QA INTERNATIONAL



```
{
    "imageName": "149.png",
    "questions": {
        "What letters indicate a low tide?": {
            "abcLabel": true,
            "answerTexts": [
                "D and B",
                "A and C",
                "A and D",
                "E and C"
            ],
            "correctAnswer": 3,
            "questionId": "149.png-2"
```

Figure 7: Data sample[2]



Multiple Choice Question: From the above food web diagram, what will lead to an increase in the population of deer? a) increase in lion b) decrease in plants **c) decrease in lion** d) increase in pika

Figure 3: Sample data example with diagram, annotations, and graph relationships[3]

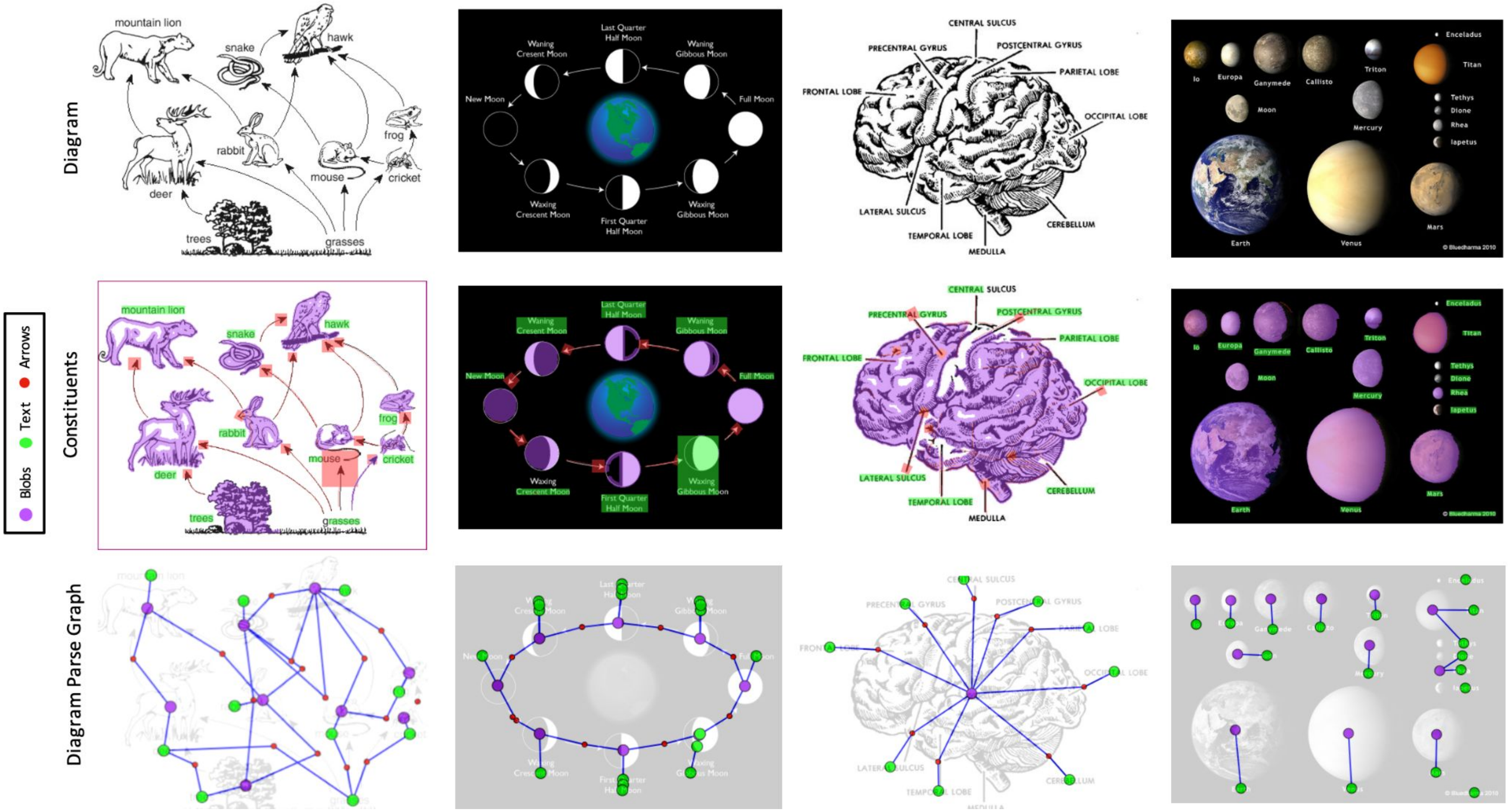


Figure 2: Diagram Graph Parser result examples[3]

Data Acquisition and EDA

Data Acquisition

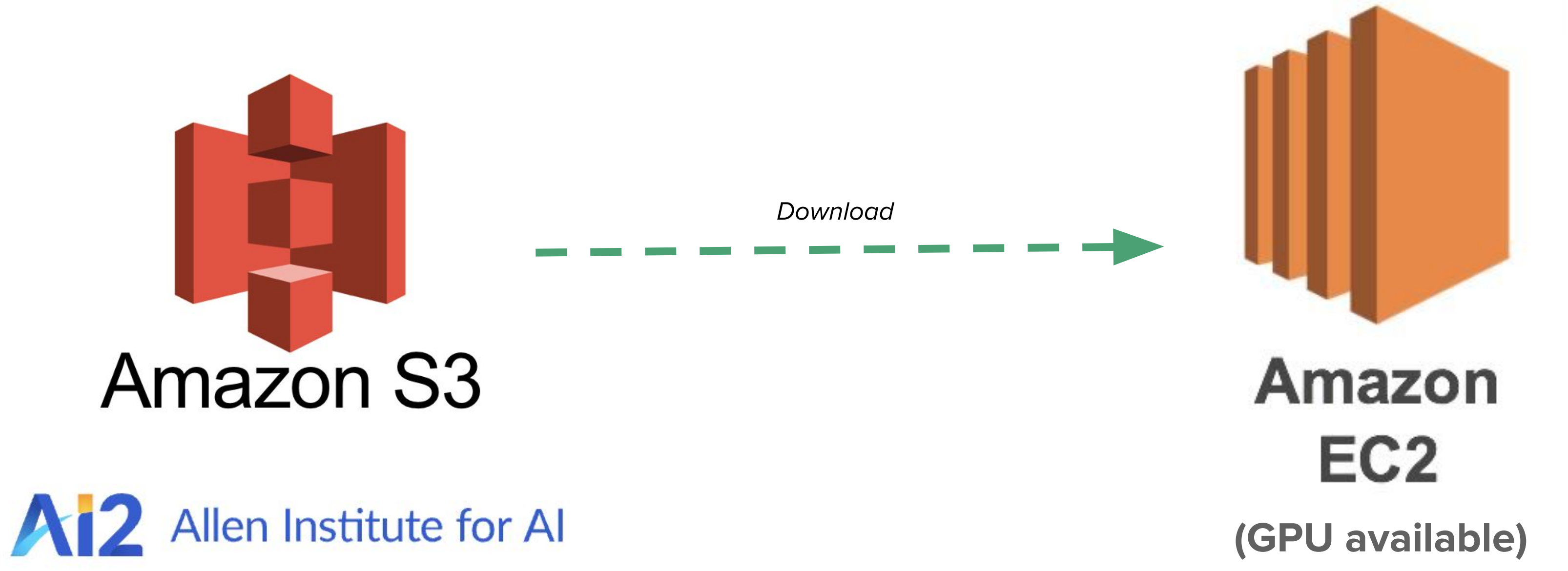


Figure 8: Data Acquisition Diagram

Data Split

Dataset Split	Choice A	Choice B	Choice C	Choice D	TOTAL
Training	**25.3%** (2,831)	**25.5%** (2,845)	**25.6%** (2,856)	**23.6%** (2,639)	**72.1%** (11,171)
Validation	**23.8%** (296)	**26.5%** (329)	**27.7%** (345)	**21.9%** (272)	**8.0%** (1,242)
Testing	**24.5%** (758)	**26.0%** (802)	**26.9%** (832)	**22.5%** (696)	**19.9%** (3,088)
TOTAL	**25.1%** (3,885)	**25.6%** (3,976)	**26.0%** (4,033)	**23.3%** (3,607)	15,501

Figure 9: Train/Test/Validation splits

Model Selection and Background

Model Selection

Name	Type	Pre-Training Objectives	Key Heads
VisualBERT[5]	Vision-Language Transformer Model	1. Bidirectional masked language model 2. Sentence-image prediction on caption data	1. Multiple Choice 2. Question Answering 3. Visual Reasoning
ViLT[6]	V-L Transformer Model without Convolution for Visual Embedding	1. Masked language model with whole word masking 2. Image text matching 3. Word patch alignment: predict masked image patches of a text word	1. Question Answering
LayoutMV3[7]	Document Transformer Model	1. Bidirectional masked language model 2. Masked image model 3. Word patch alignment	1. Question Answering

Figure 10: Model selection considerations

VisualBERT[5]

- Has **head for Multiple-Choice downstream task**
- BERT with visual input component
- Pre-Training Objectives:
  - **Masked Language Model with Image**
    - Text tokens are masked but image vectors are not
  - **Sentence Image Prediction**
    - Provide a text segment consisting of two captions, one describes the image, the other has 50% to describe the image or be randomly drawn caption
- Trained on Common Object in Context (COCO) dataset[18]

VisualBERT Model Architecture

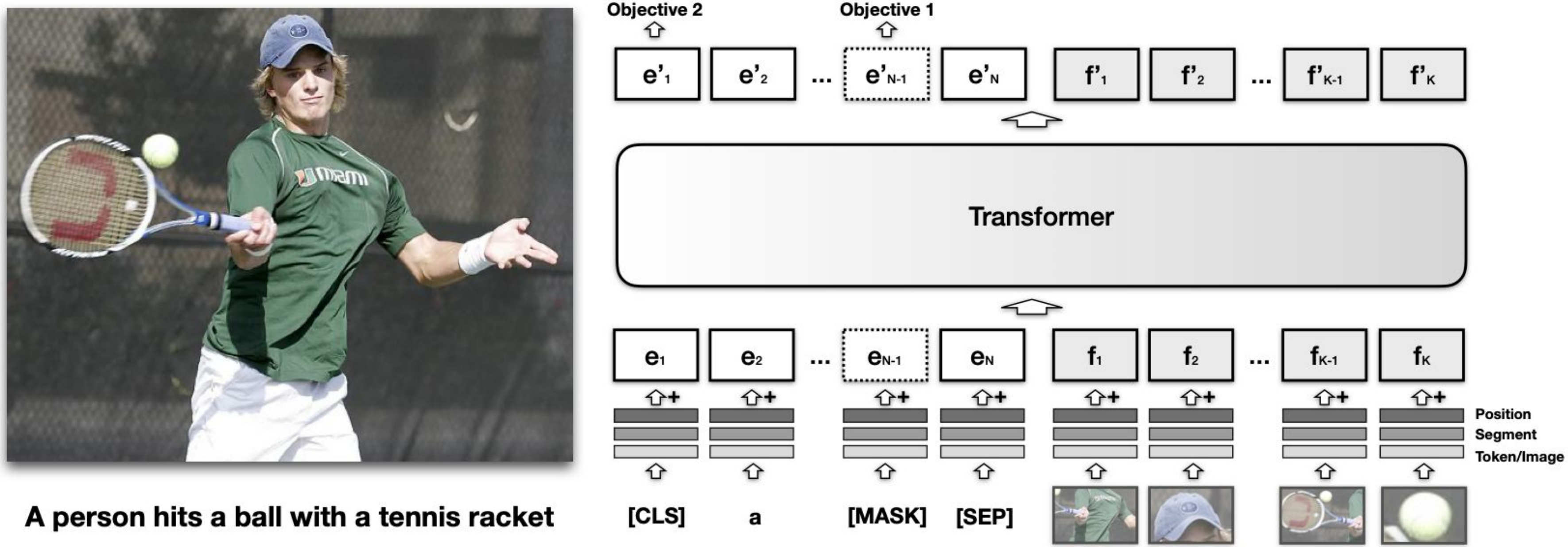

A person hits a ball with a tennis racket
Objective 2
Objective 1
e'_1 e'_2 ... e'_{N-1} e'_N f'_1 f'_2 ... f'_{K-1} f'_K
Transformer
e_1 e_2 ... e_{N-1} e_N f_1 f_2 ... f_{K-1} f_K
Position
Segment
Token/Image
[CLS]
a
[MASK]
[SEP]

Figure 11: VisualBERT architecture[5]

Metrics Selection

Metric
*Accuracy
F1 Score
Recall
Precision
Specificity

**Main metric*

Data Processing

Types of Input Scenarios

1 No Annotations used in Inputs

Inputs:

1. **Q-A pairs**: **text embeddings**
2. **Diagram images**: **visual embeddings**
3. **Annotations**: **not used**

2 Draw Annotations on Diagram

Inputs:

1. **Q-A pairs**: **text embeddings**
2. **Diagram images**: **visual embeddings**
3. **Annotations**: drawn on diagram images and part of **visual embeddings**

3 Embedding Annotations via Strings

Inputs:

1. **Q-A pairs**: **text embeddings**
2. **Diagram images with annotations**: **visual embeddings**
3. **Annotations**: as strings and concatenated with Q-A pairs and part of **text embeddings**

Embeddings

Text Embedding

- BERT Tokenizer[19] for contextual text embeddings

Visual Embedding

- Resize diagram images to 224x224
- Forward pass through ResNET18 and get the ‘avgpool’ feature maps layer

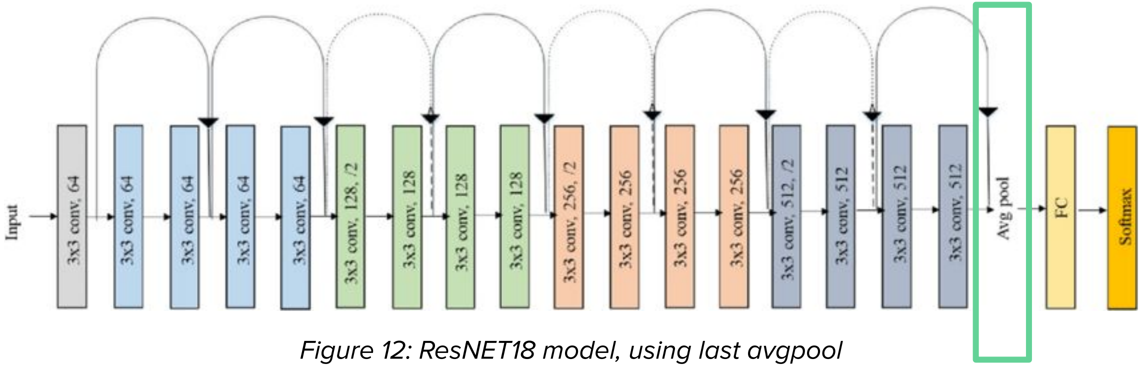
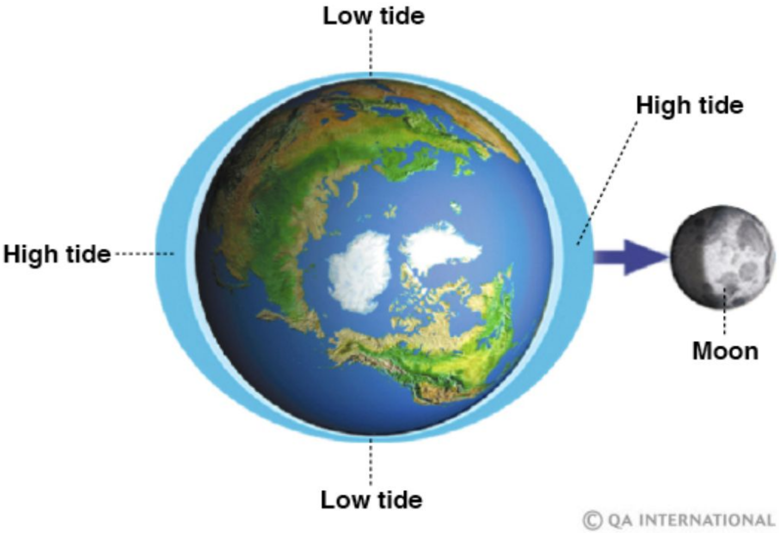


Figure 12: ResNET18 model, using last avgpool layer[20]

Image Annotator

Before



After

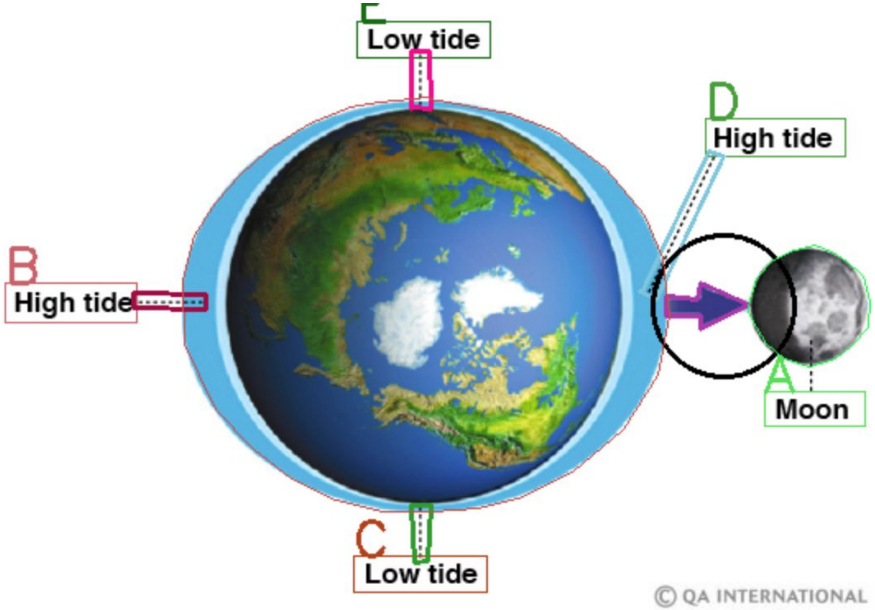


Figure 13: Diagram with visual annotations applied

Embedding Annotations Via Strings

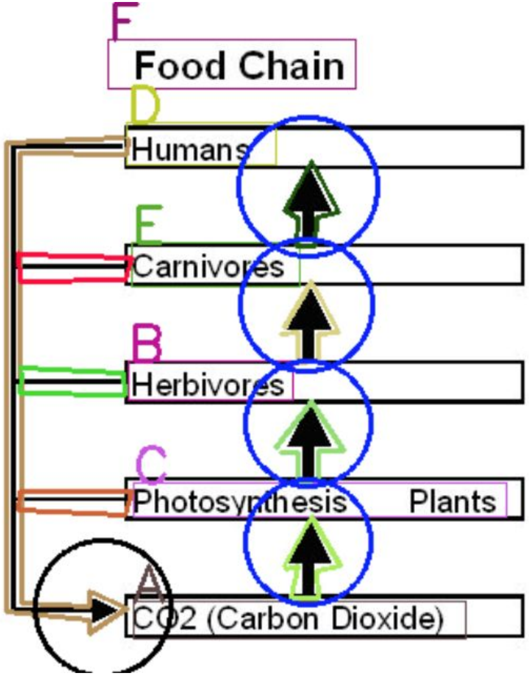


Figure 14: Data sample[2]

Question: Which organism is both predator and prey in the above food chain?

Annotations: The title of the image is F. D object links to A. E object links to D. B object links to E. C object links to B. A object links to C.

Model Training and Results

VisualBERT Fine-Tuning

- Trained for **16 epochs** for each of the 3 input scenarios
- Used **PyTorch**
- Hyperparameters
  - Optimizer: AdamW()[16]
  - Criterion: torch.nn.CrossEntropy()[17]

Final Test Evaluation Accuracy vs Benchmarks

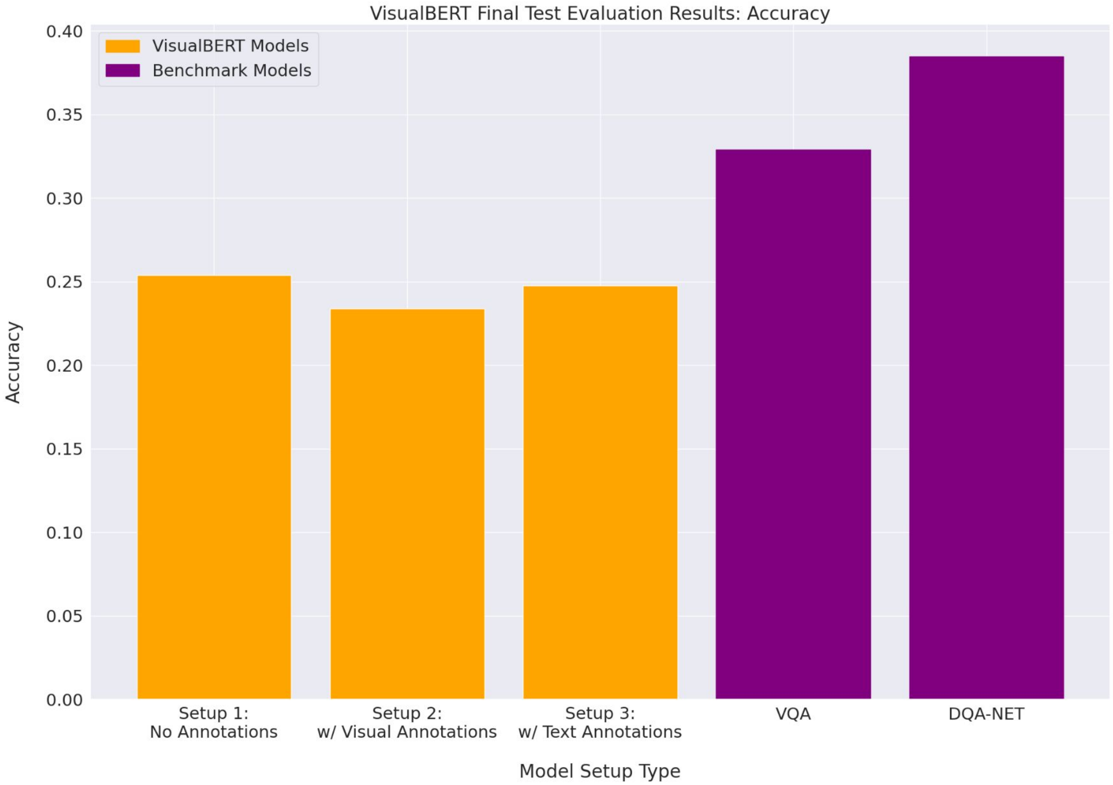


Figure 15: VisualBERT vs Benchmarked Model accuracy on same testing dataset

Interpreting Results: Complexity of Dataset

- Diagrams are **not like regular natural images**
  - Arrows, Relationships between objects, Text
- Annotations
  - How do we **structure this as an input**?
  - Difficult to process
- Lots of signals between the different data modes but **difficult to ensure the model can appropriately learn a good joint representation** of all those inputs

Final Words

Lessons Learned and Future Improvements

Contributions:

- Proved that large **pretrained transformer models may not always perform better** than more data-customized architectures, at least for diagram QA

Possible Future Improvements:

- Try ViLT or LayoutMV3 models
  a. Models with **masked image pretraining or word patch alignment** may have performed better in this task
- Try more techniques to **embed the annotations**
- **Better visual embedding** techniques
- Try **data-customized architectures**

References

1. Github Repo
2. *AI2 Diagram Dataset (AI2D)*
   AI2 Diagram Dataset (AI2D) was accessed on 9/5/2022 from https://registry.opendata.aws/allenai-diagrams.
3. *A Diagram is Worth a Thousand Words*
   @article{Kembhavi2016ADI,
   title={A Diagram is Worth a Dozen Images},
   author={Aniruddha Kembhavi and Michael Salvato and Eric Kolve and Minjoon Seo and Hannaneh Hajishirzi and Ali Farhadi},
   journal={ArXiv},
   year={2016},
   volume={abs/1603.07396}
4. Paper code:
5. VISUALBERT: A SIMPLE AND PERFORMANT BASELINE FOR VISION AND LANGUAGE, Li et al., 2019
6. ViLT: Vision-and-Language Transformer Without Convolution or Region Supervision. Kim et al., 2021
7. LayoutLMv3: Pre-training for Document AI with Unified Text and Image Masking. Huang et al., 2022
8. VisualBERT for Multiple Choice Hugging Face
9. VisualBERT for Question Answering Hugging Face
10. VILT for Question Answering Hugging Face
11. LayoutMV3 Hugging Face
12. VisualBERT Demo
13. BERT Multiple Choice Sample
14. Fine Tuning on Multiple Choice Task
15. Hugging Face
16. PyTorch AdamW Optimizer
17. PyTorch Cross Entropy
18. Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dolla´r, and C Lawrence Zitnick. Microsoft COCO captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*, 2015.
19. BERT Tokenizer
20. Ramzan, Farheen & Khan, Muhammad Usman & Rehmat, Asim & Iqbal, Sajid & Saba, Tanzila & Rehman, Amjad & Mehmood, Zahid. (2019). A Deep Learning Approach for Automated Diagnosis and Multi-Class Classification of Alzheimer's Disease Stages Using Resting-State fMRI and Residual Neural Networks. Journal of Medical Systems. 44. 10.1007/s10916-019-1475-2.

Appendix: Additional Slides Not Used

Project Inspiration

- Work on a dataset or model type we had **never encountered before**

- We have worked with images, time series, text, and tabular data but all **separately**

- Use this as an opportunity to **combine different modes** of data and techniques into one model

DQA-Net

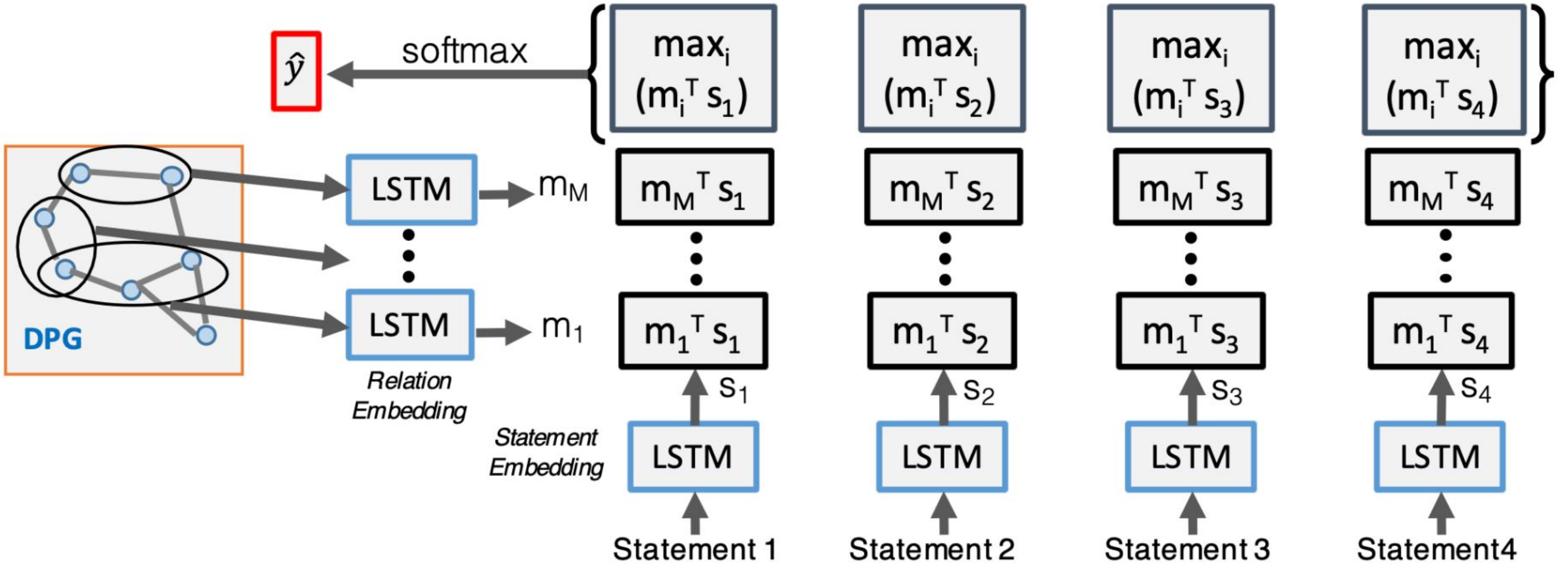


Fig. 3. An overview of the DQA-Net solution to diagram question answering. The network encodes the DPG into a set of facts, learns to attend on the most relevant fact, given a question and then answers the question.

Figure X: DQA-Net architecture[3]

DSDP-NET

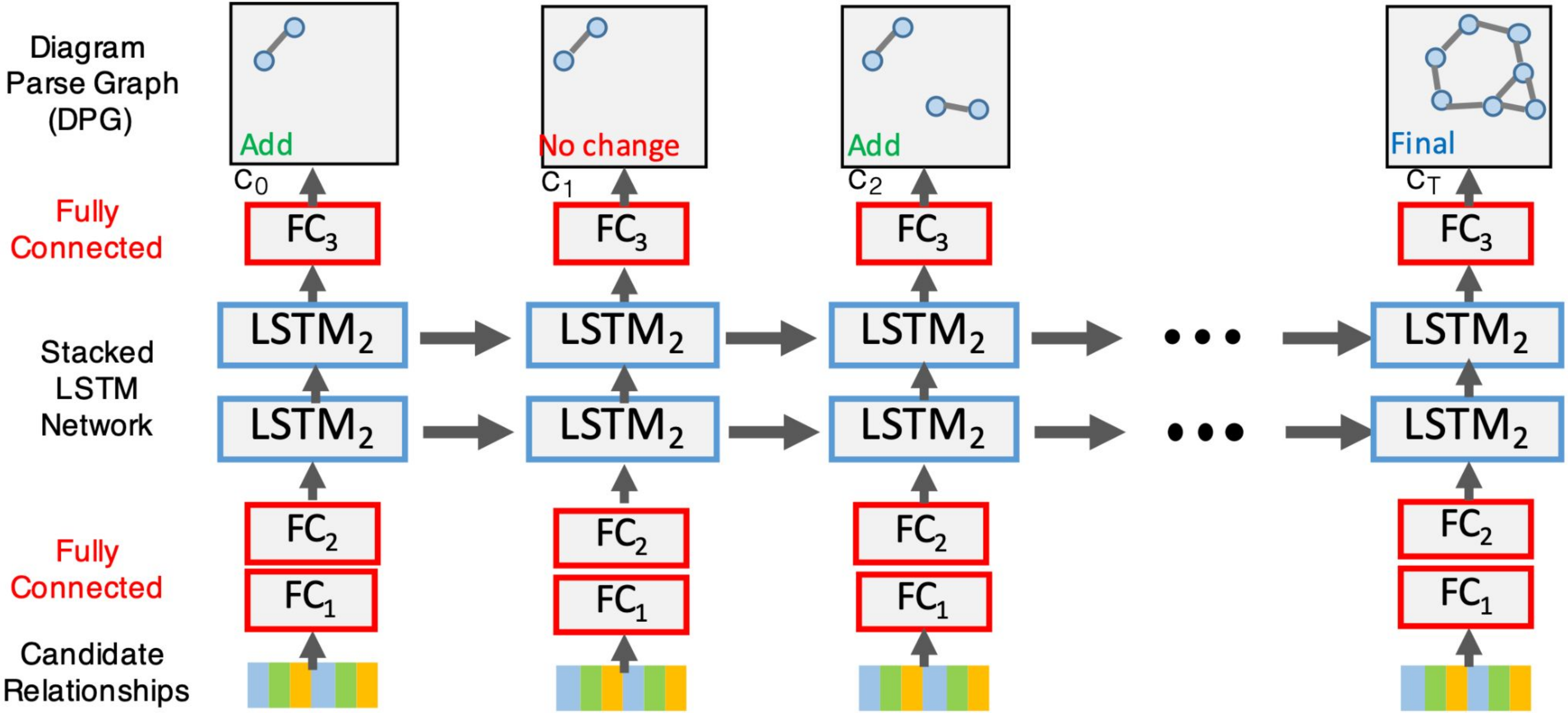


Figure X: DSDP-Net architecture[3]

Contribution Goals to Original Paper

1. **Transformers and/or Auto Encoder-Decoder** architectures with pretraining
   a. Achieved impressive results in complex domains (NLP, CV, Speech)

2. **Reduce the complexity** during featuring engineering
   a. No separate model to create Diagram Parse Graphs (DPGs)

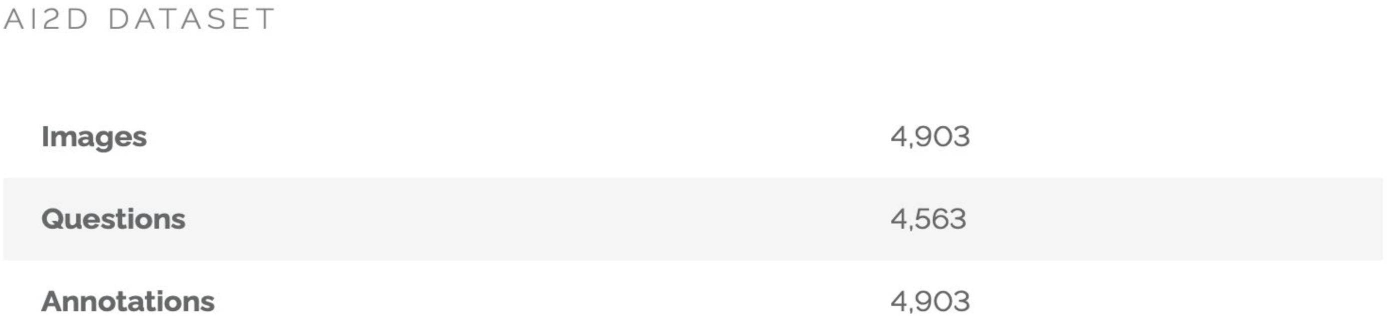
3. Use a pretrained model and try **Transfer Learning**

4. Create a more streamlined, more **end-to-end model**

5. Try to achieve **comparable results**

Dataset Overview

- AI2D dataset[2]
    - Free and open source access, a **Creative Commons license**
    - On the AWS Data Marketplace within a **S3 bucket**
    - Original images folder = 1.07 GB

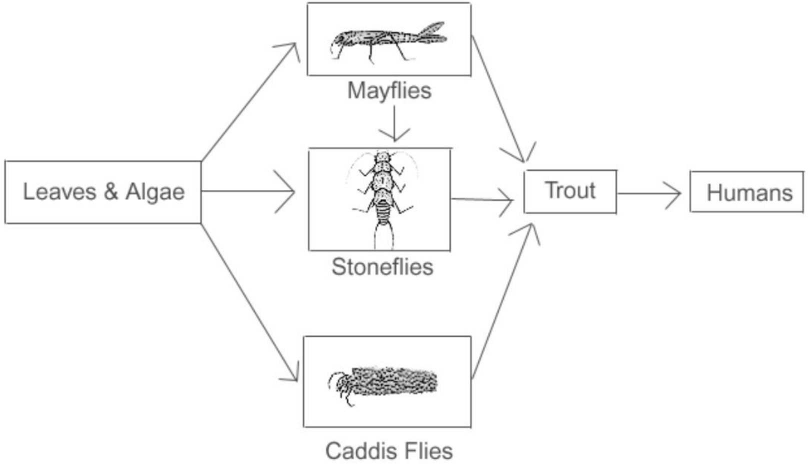


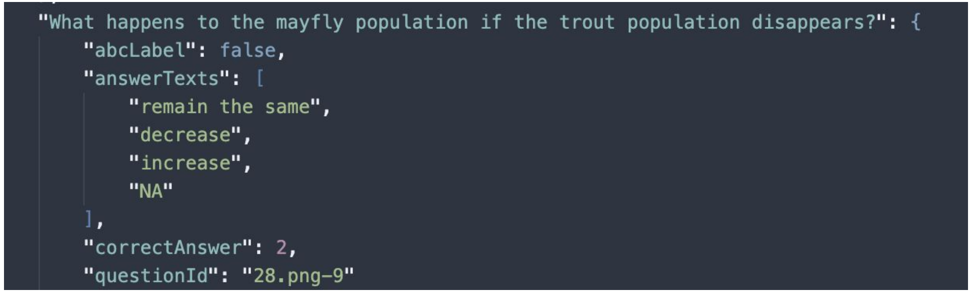
AI2D DATASET

Images	4,903
Questions	4,563
Annotations	4,903

Figure X: A12D Dataset[2]

Example II: Question - Image Pairs



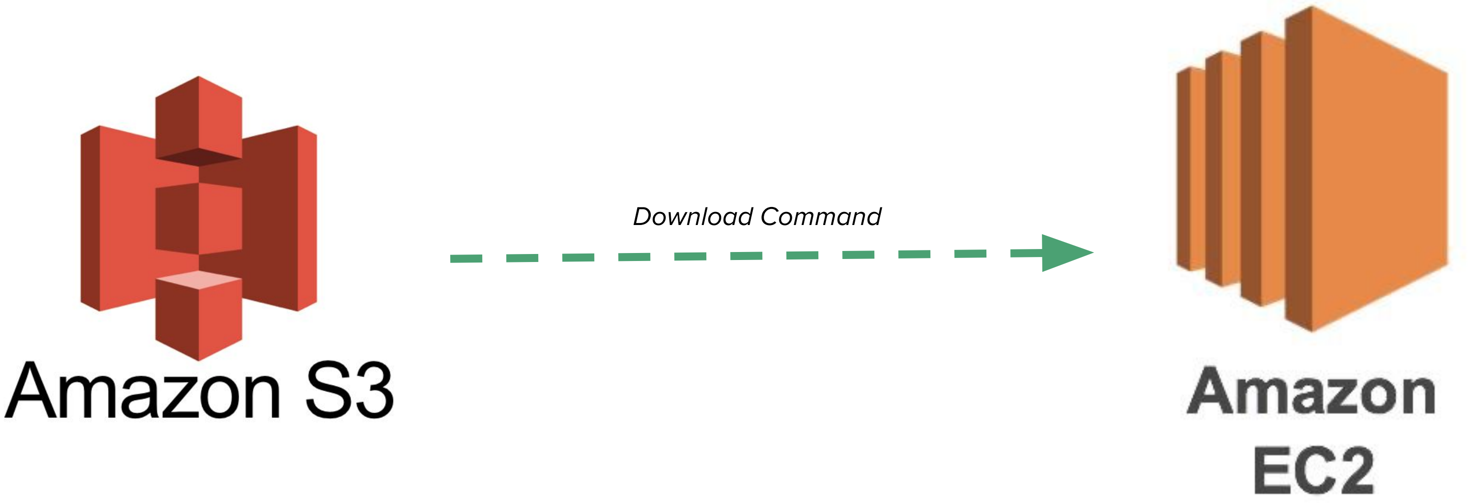


```
"What happens to the mayfly population if the trout population disappears?": {
    "abcLabel": false,
    "answerTexts": [
        "remain the same",
        "decrease",
        "increase",
        "NA"
    ],
    "correctAnswer": 2,
    "questionId": "28.png-9"
```

Figure X: Data sample[2]

Cloud Environment

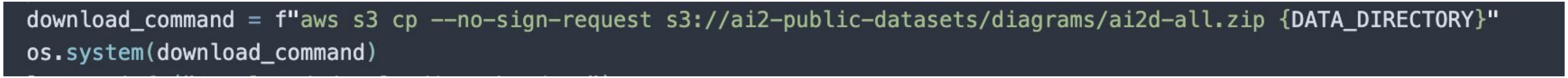
Data Acquisition



Download Command

(GPU available)

Download Command:



```
download_command = f"aws s3 cp --no-sign-request s3://ai2-public-datasets/diagrams/ai2d-all.zip {DATA_DIRECTORY}"
os.system(download_command)
```

Figure X: Data Acquisition Diagram

Cloud Environment and Software Used

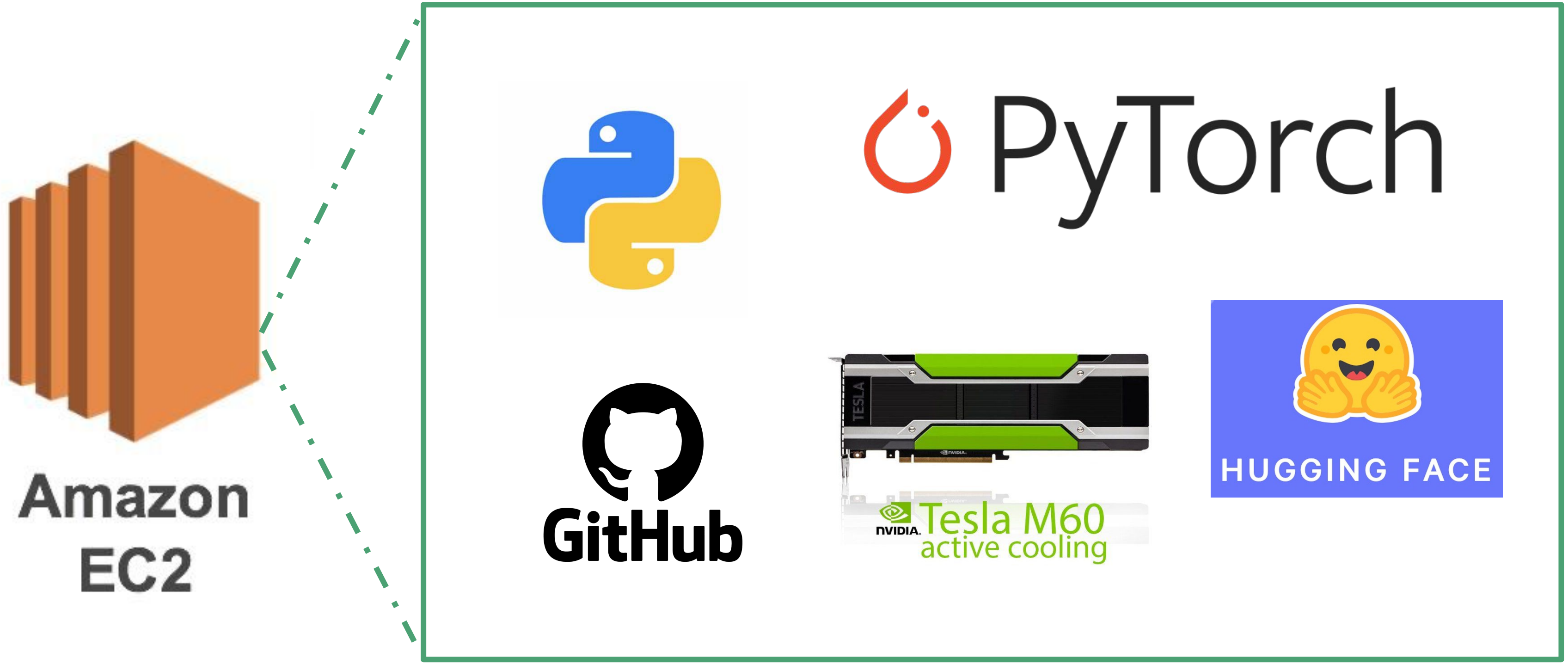


Figure X: Cloud setup and key software used

VisualBERT[5]

- Task specific pre-training
  - Has **head for Multiple-Choice downstream task**
- **BERT with visual input component**
  - 12 layers
  - 768 hidden size
  - 12 self-attention heads
- Pre-Training Objectives:
  - **Masked Language Model with Image**
    - Text tokens are masked but image vectors are not
  - **Sentence Image Prediction**
    - Provide a text segment consisting of two captions, one describes the image, the other has 50% to describe the image or be randomly drawn caption
- Trained on Common Object in Context (COCO) dataset[18]

Modules

1. **Data Downloader**: Downloads data into workspace
2. **Data Preprocessor**: Performs preprocessing of images, q-a pairs, annotations into one dataframe
3. **Train/Val/Test Splitter**: Splits train/test/validation datasets
4. **Image Annotator**: Annotates regions of diagram image from the annotations file
5. **Visual Embedder**: Generate visual embeddings using a Resnet18 model
6. **Text Embedder**: Tokenize our inputs with BERT tokenizer to generate text embeddings
7. **Annotations-Strings Embedder**: Turns annotations into text embeddings
8. **Data Loader**: Yield batch of data while performing each of the preprocessing modules above to output text and visual embeddings
9. **Model**: Includes training loop and support functions
10. **Testing**: Test and generate results
11. **Plotter**: Plot diagrams and results

Embedding Annotations Via Strings

Problem

1. VisualBERT takes text and visual embeddings as inputs
2. Nowhere to place the annotations in any format “out of the box”, except drawing them onto the image

Solution

1. Encode the annotations into a string, add to the question string before tokenizer
2. Ensures model absorbs in some way annotation data

3 Embedding Annotations Via Strings

Problem

1. VisualBERT take question and the visual embeddings of the diagram as inputs
2. Nowhere to place the annotations in any format “out of the box”, except drawing them onto the image.

Solution

1. Encode the annotations into a string, add to the question string before tokenizer
2. Ensures model absorbs in some way annotation data

Outcome

1. No change whatsoever in metrics :(

Test Data Evaluation Results

Test Evaluation Results

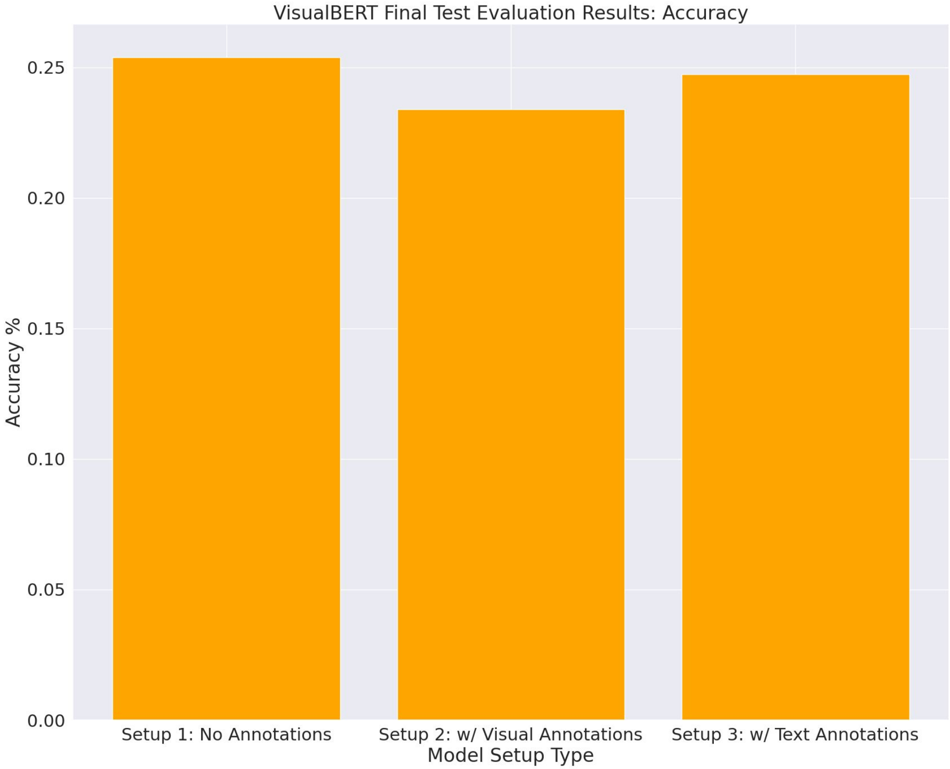


Figure X: VisualBERT final accuracy on testing data evaluation

Setup 1 Results

Metrics: Setup 1

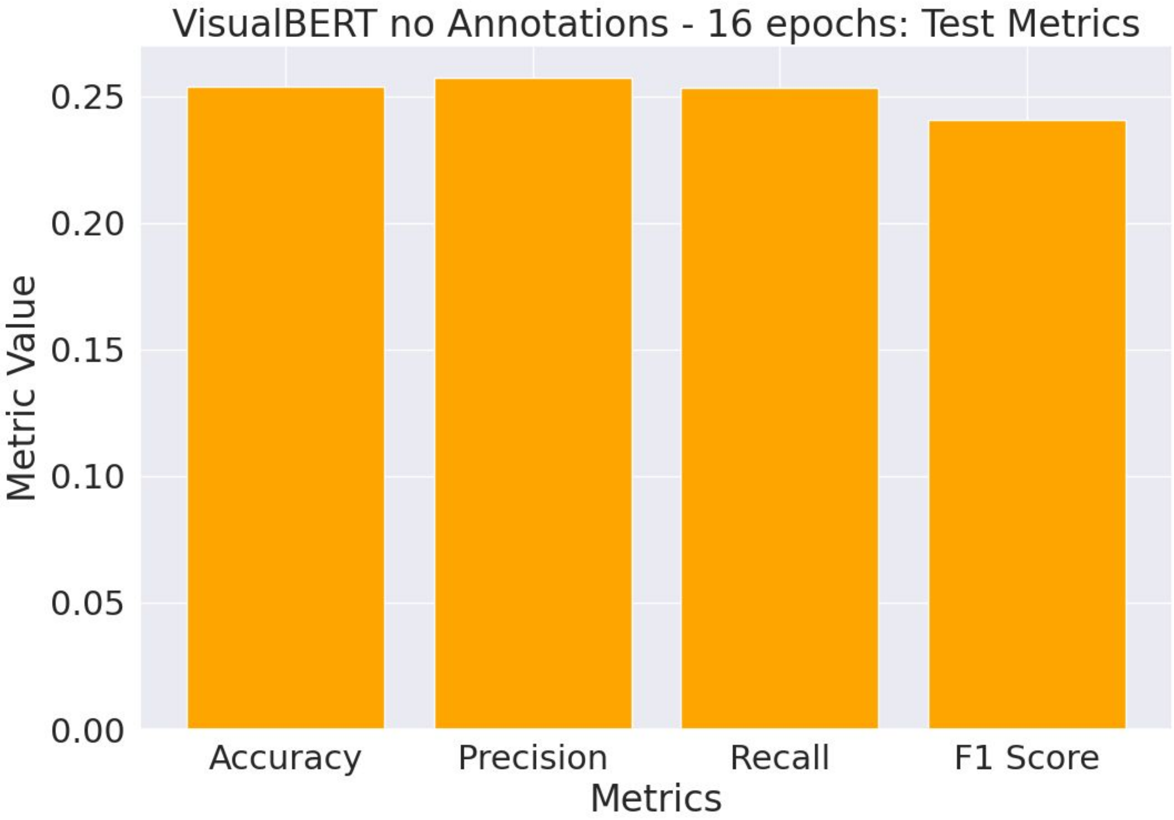


Figure X: VisualBERT Metrics - Setup 1: No Annotations

Class Metrics: Setup 1

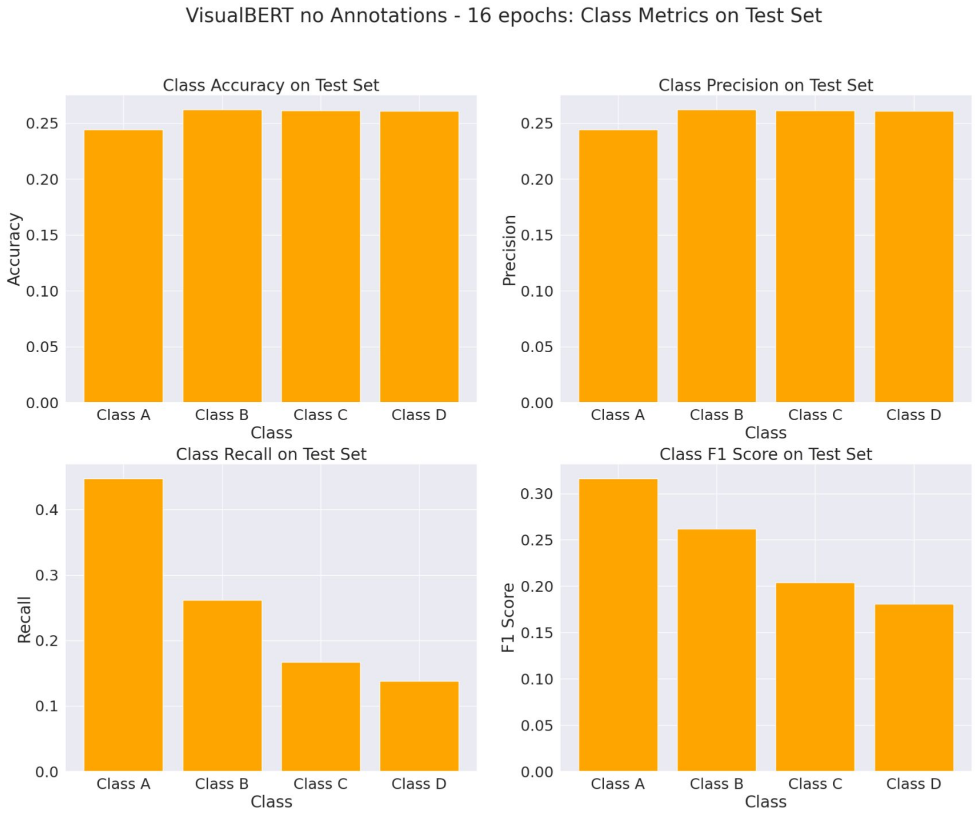
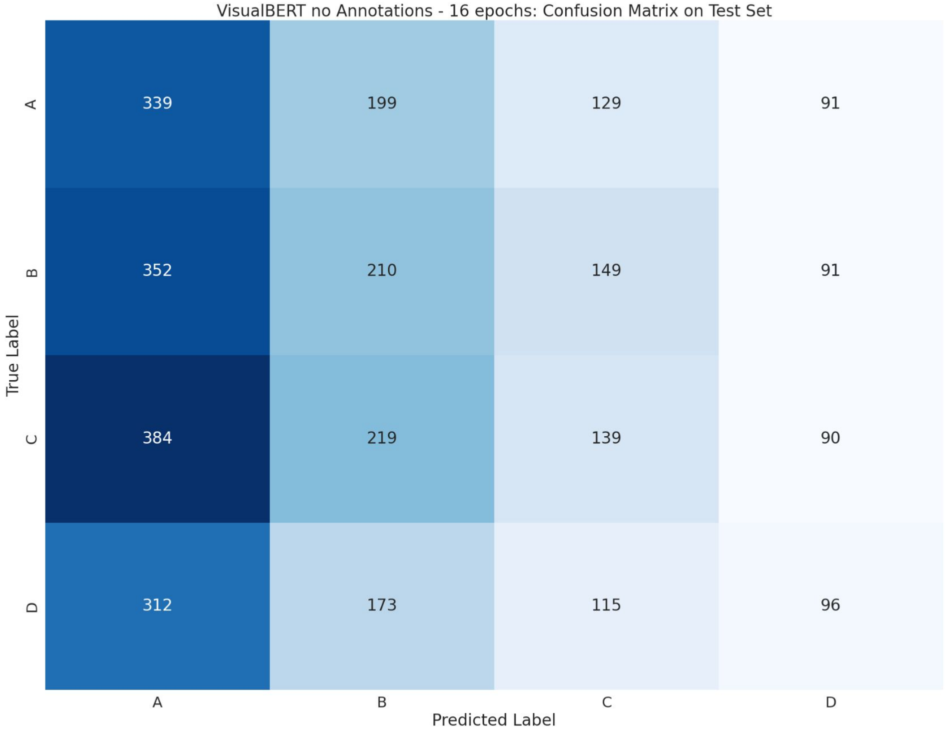


Figure X: VisualBERT Class Metrics - Setup 1: No Annotations

Confusion Matrix: Setup 1



VisualBERT no Annotations - 16 epochs: Confusion Matrix on Test Set

True Label \ Predicted Label	A	B	C	D
A	339	199	129	91
B	352	210	149	91
C	384	219	139	90
D	312	173	115	96

Figure X: VisualBERT Confusion Matrix - Setup 1: No Annotations

Training Results: Setup 1

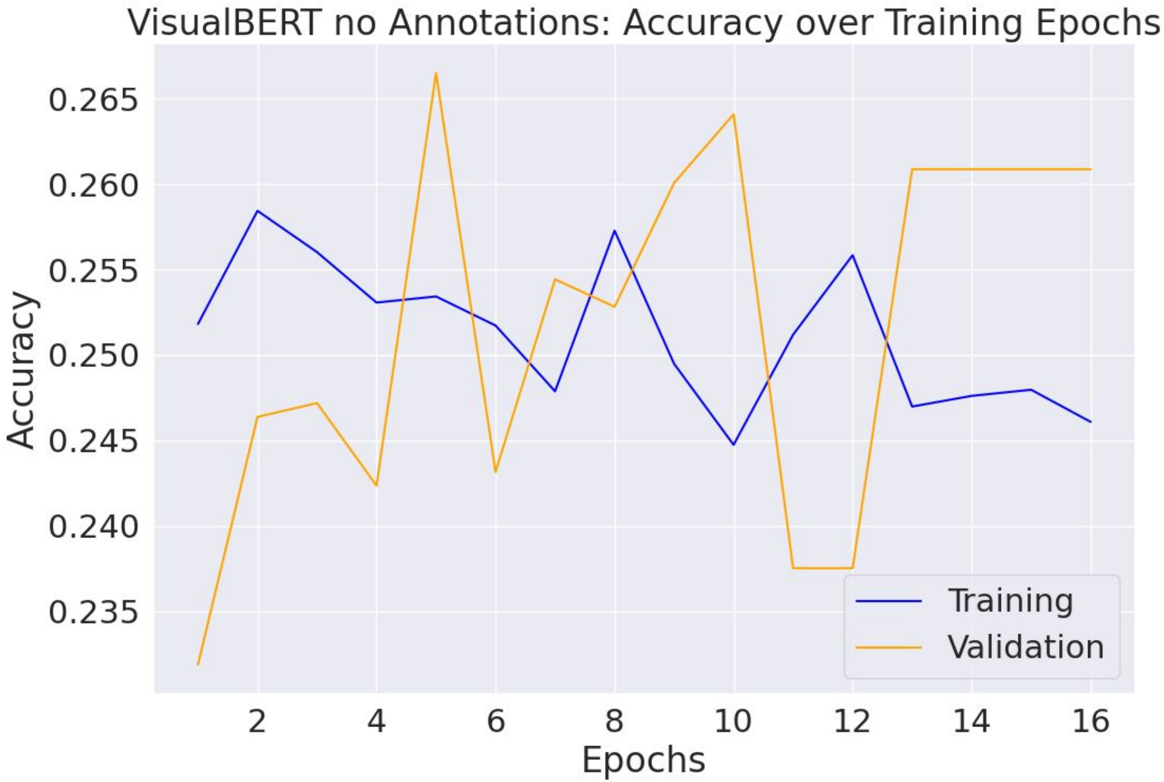


Figure X: VisualBERT Accuracy over Training Epochs - Setup 1: No Annotations

Setup 2 Results

Metrics: Setup 2

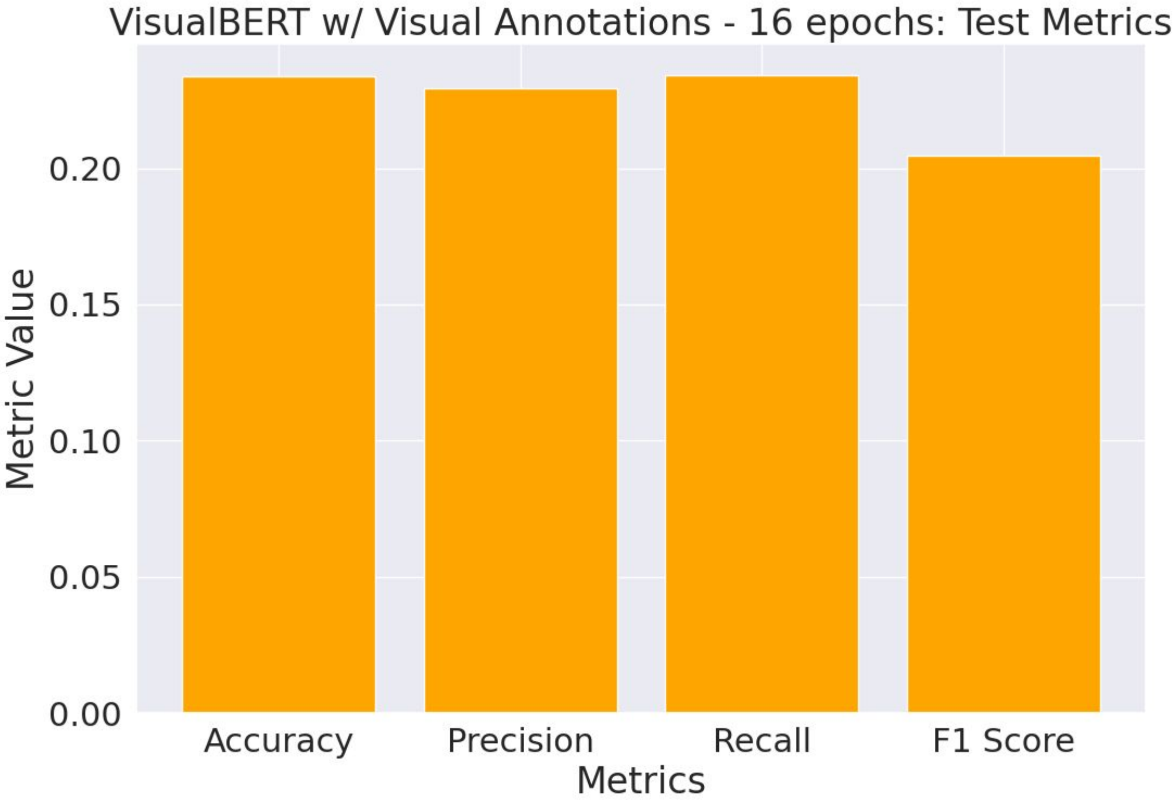


Figure X: VisualBERT Metrics - Setup 2: Visual Annotations

Class Metrics: Setup 2

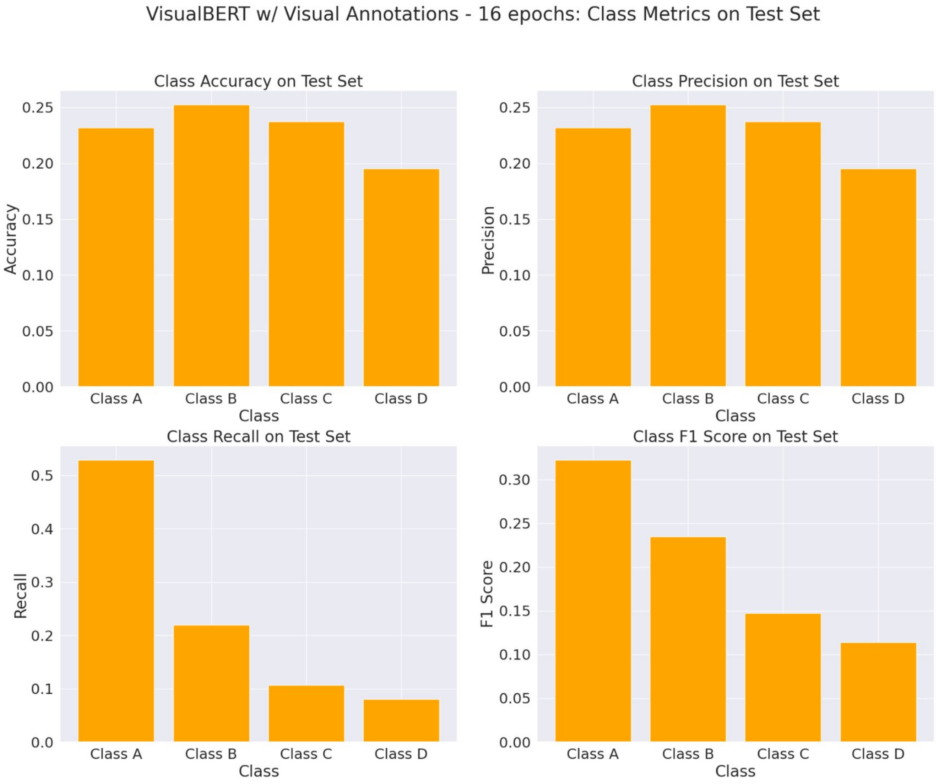
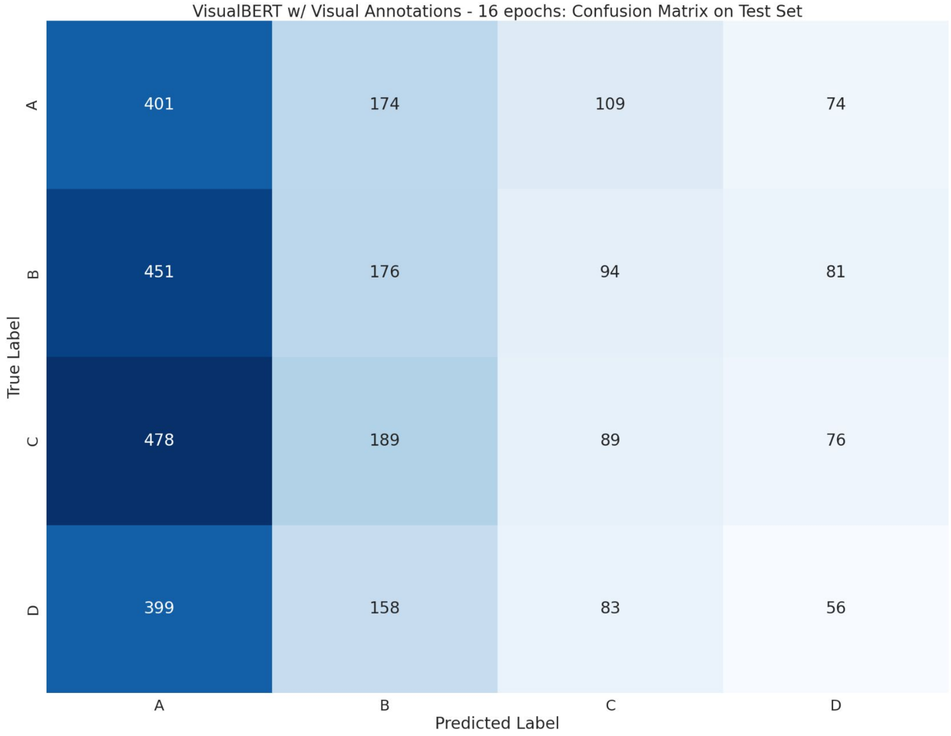


Figure X: VisualBERT Class Metrics - Setup 2: Visual Annotations

Confusion Matrix: Setup 2



VisualBERT w/ Visual Annotations - 16 epochs: Confusion Matrix on Test Set

True Label \ Predicted Label	A	B	C	D
A	401	174	109	74
B	451	176	94	81
C	478	189	89	76
D	399	158	83	56

Figure X: VisualBERT Confusion Matrix - Setup 2: Visual Annotations

Training Results: Setup 2

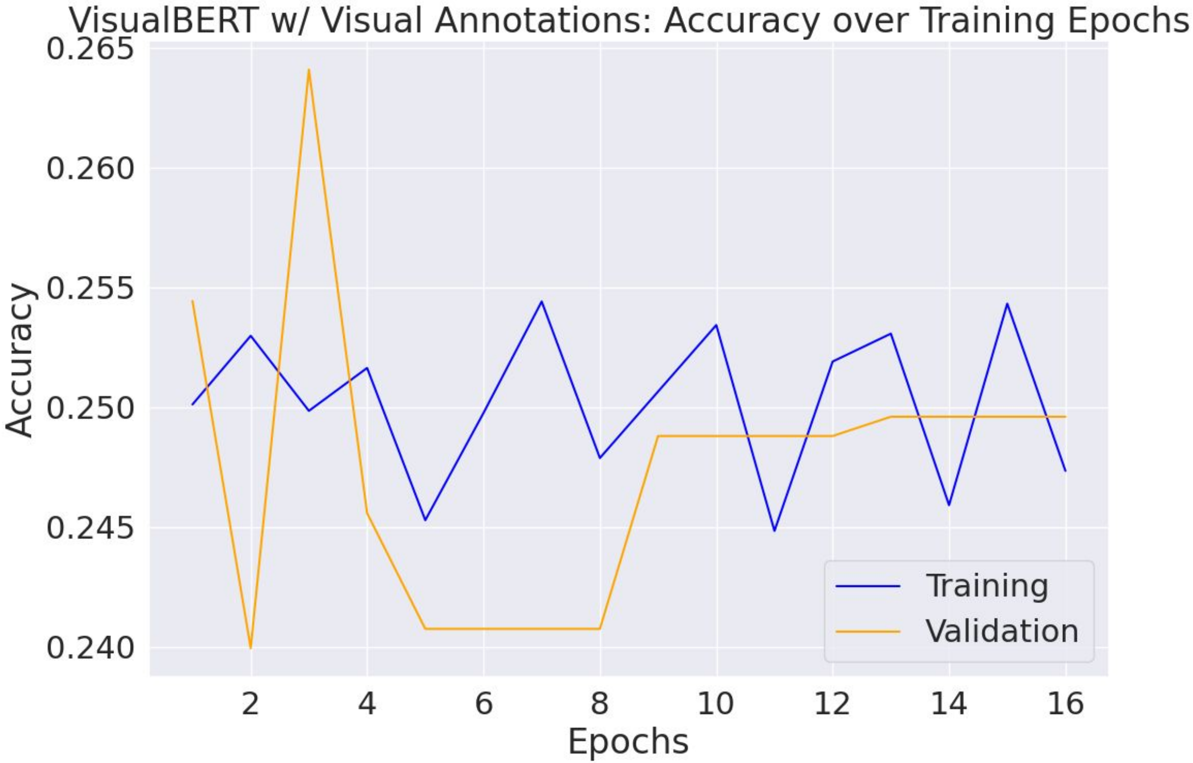


Figure X: VisualBERT Accuracy over Training Epochs - Setup 2: Visual Annotations

Setup 3 Results

Metrics: Setup 3

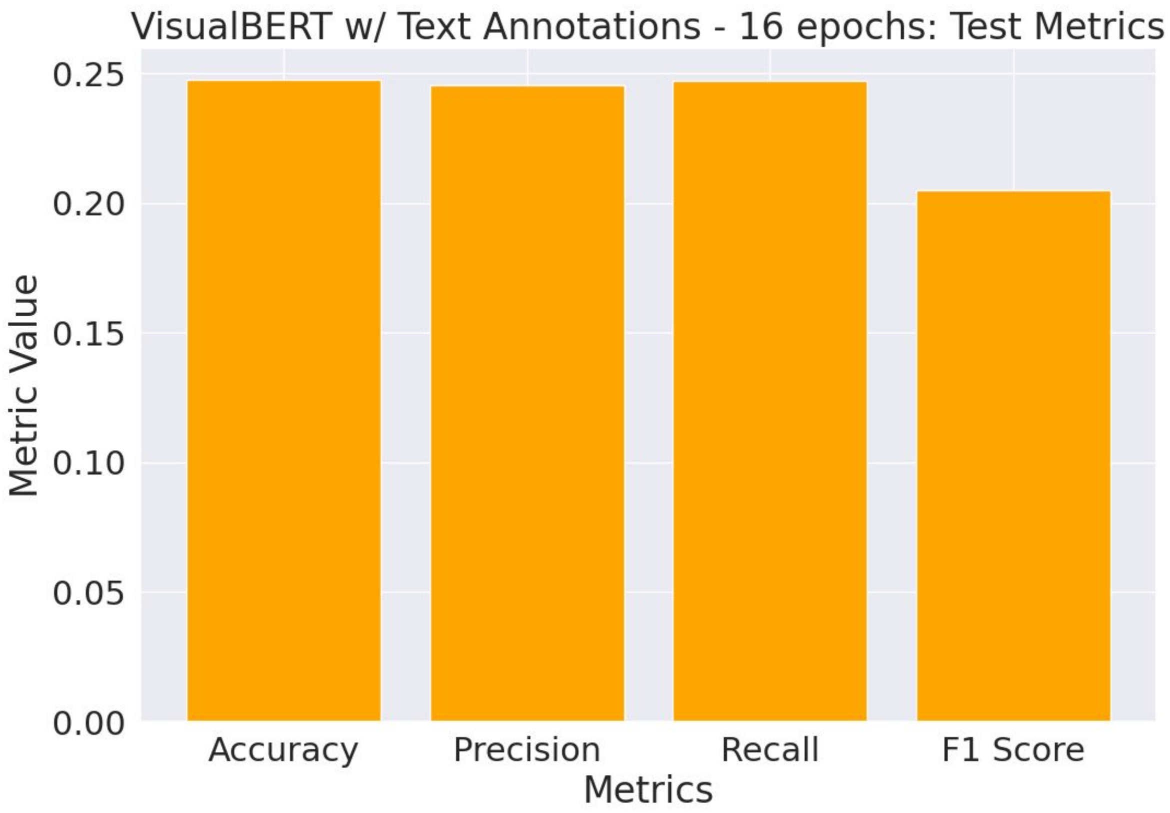


Figure X: VisualBERT Metrics - Setup 3: Text Annotations

Class Metrics: Setup 3

VisualBERT w/ Text Annotations - 16 epochs: Class Metrics on Test Set

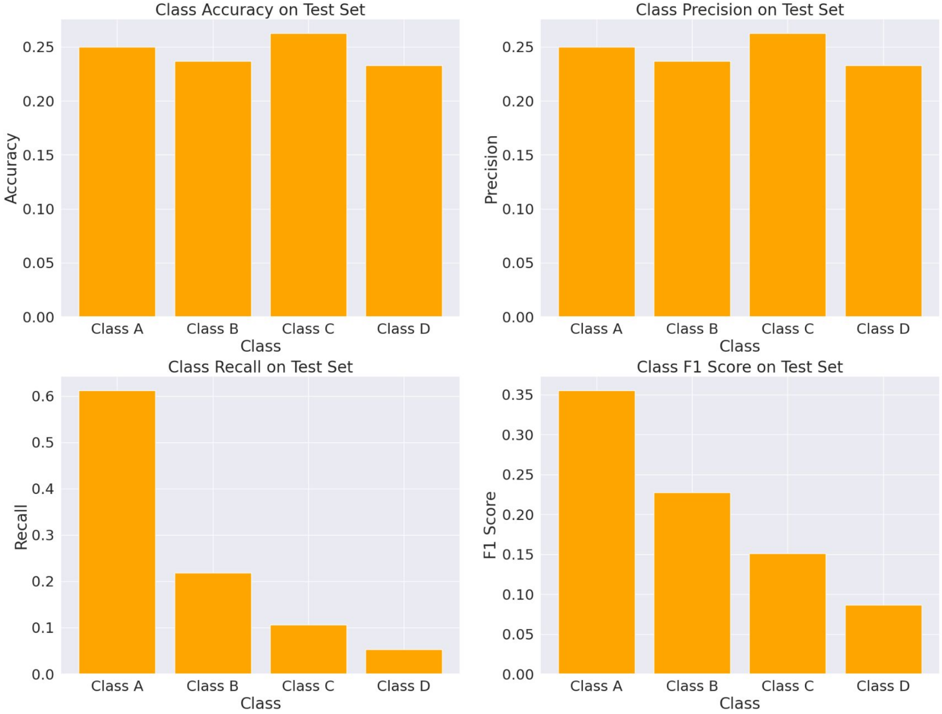
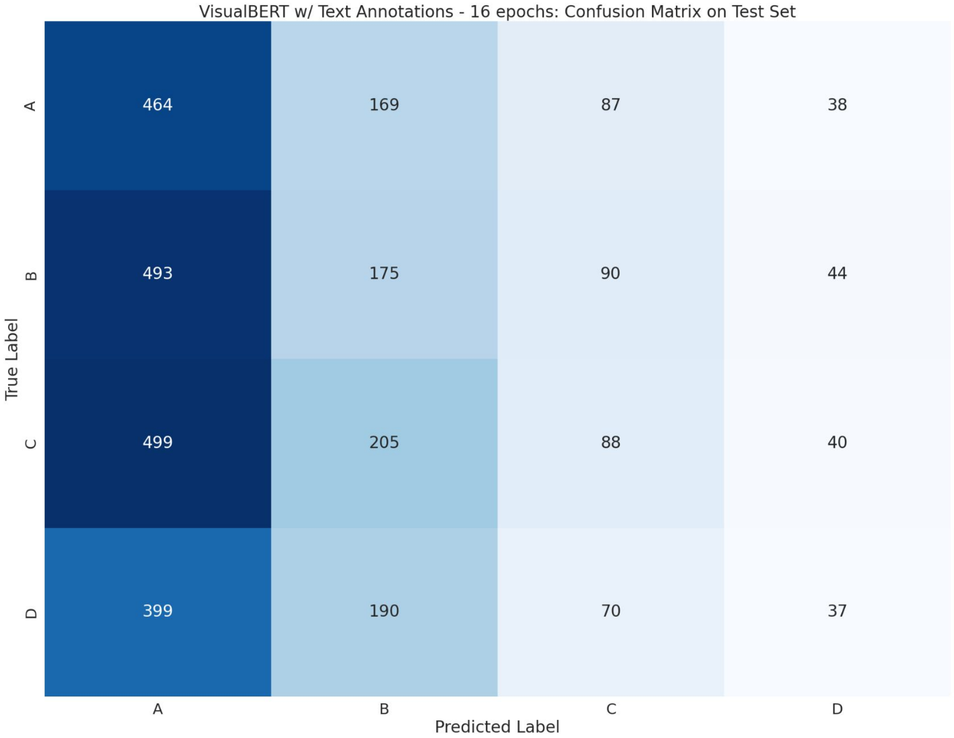


Figure X: VisualBERT Class Metrics - Setup 3: Text Annotations

Confusion Matrix: Setup 3



True Label \ Predicted Label	A	B	C	D
A	464	169	87	38
B	493	175	90	44
C	499	205	88	40
D	399	190	70	37

Figure X: VisualBERT Confusion Matrix - Setup 3: Text Annotations

Training Results: Setup 3

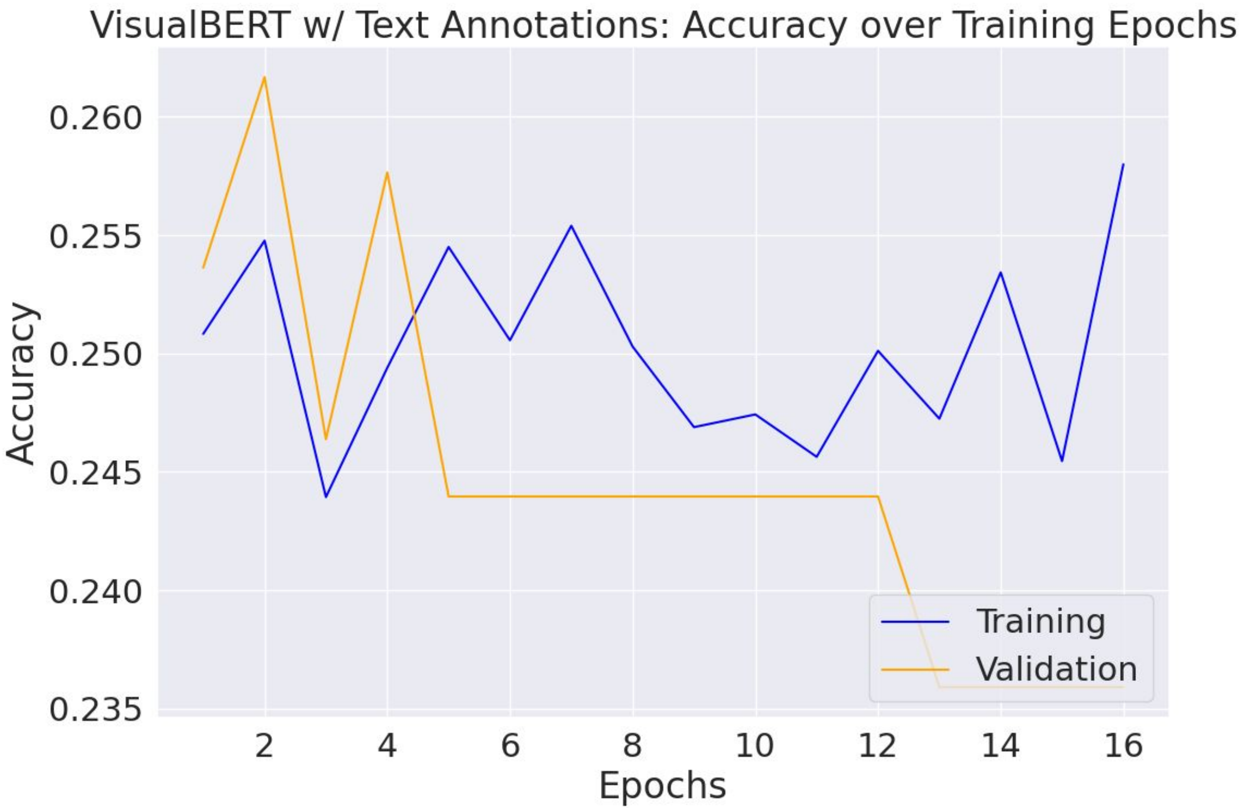


Figure X: VisualBERT Accuracy over Training Epochs - Setup 3: Visual Annotations

Diagram Object

Intra-Object Label (R1): A text box naming the entire object.

Intra-Object Region Label (R2): A text box referring to a region within an object.

Intra-Object Linkage (R3): A text box referring to a region within an object via an arrow.

Inter-Object Linkage (R4): Two objects related to one another via an arrow.

Arrow Head Assignment (R5): An arrow head associated to an arrow tail.

Arrow Descriptor (R6): A text box describing a process that an arrow refers to.

Image Title (R7): The title of the entire image.

Image Section Title (R8): Text box that serves as a title for a section of the image. Image Caption (R9): A text box that adds information about the entire image, but does not serve as the image title.

Image Misc (R10): Decorative elements in the diagram.