Setting the Scene: Understanding Text in Images for VQA (Team 5)

Sweta Kotha, Meredith Riggs, Naoki Tsuda, Sean Zhang

Abstract

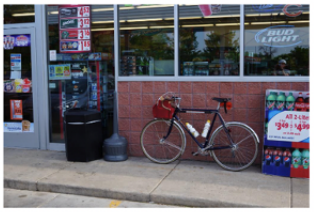
Recent work in Visual Question Answering (VQA) has shown that state-of-the-art models fall short in detecting and semantically understanding "scene text," or text present in images that is required to understand the image fully. Though there has been work in OCR, object detection, and VQA, most VQA work has not directly focused on the detection and understanding of scene text. In this paper, we use an M4C model to implement three main approaches to better solve this problem: stop token prediction, a global multimodal transformer to learn latent graphical representations between modalities, and additional pre-training and fine-tuning using a larger dataset, OCR-VQA. With these approaches, we find a 0.01% improvement in accuracy and average ANLS, improvements in counting objects, and increased OCR performance.

1. Introduction

Visual question answering (VQA) is a task within multimodal machine learning that uses information from an image to answer a natural language question about that image. This task typically takes two inputs from an image– visual information and textual information. The textual information in natural scene images is called "scene text," and scene text appears in about 50% of images in large VQA datasets such as MS Common Objects (Biten et al., 2019).

Crucially, the questions in VQA do not merely ask for a repetition of detected scene text; rather, they rely on a semantic understanding of the objects in an image in conjunction with the text (Biten et al., 2019). A 2018 VizWiz study found that 21% of VQA questions from the CVPR VQA challenge relied on semantic understanding of scene text, yet none of the top entries in the challenge answered these types of questions correctly (Singh et al., 2019).

Using two VQA datasets, we implement multiple approaches to increase performance on scene text visual question answering, and we work with questions that have unambiguous answers as well as questions that might have multiple answers based on a scene text image. The majority of these questions are 'What' questions, though we also address questions that start with 'Where,' 'Which,' 'How,' and 'Who'.



What does the light sign read on the farthest right window?
LoRRA: exit
M4C (ours): bud light
human: **bud light; all 2 liters**

LoRRA and M4C predictions (Hu et al., 2020)

In this paper, we will describe our three approaches to this problem: stop token prediction, a global multimodal transformer to learn latent graphical representations between modalities, and additional pre-training and fine-tuning using a larger dataset, OCR-VQA. First, we will discuss prior work related to our research problem. Then, we will state our proposed approach, describing our model and our learning approach. Next, we will give our main research problems and describe our experimental setup. Finally, we will present results and analyses and state potential extensions to this work.

2. Related Work

2.1. History of Visual Object Detection

Text detection is concerned with finding text in input images. Text recognition is concerned with translating that text into meaningful language. Up until 2015, non-deep learning methods were mainly used in scene text detection and recognition, and these were treated as two separate tasks. Most text detection methods used Connected Components Analysis or Sliding Window classification (Long et al., 2020).

The earliest attempts in deep learning used fully convolutional neural networks (CNNs), mainly variations of ResNet and VGG neural networks for scene text detection (Tian

et al., 2015) and Connectionist Temporal Classification for scene text recognition (Graves et al., 2008). The first end-to-end systems were encoder-decoder architectures that used CNNS as an encoder and LSTMs with attention as a decoder (Biten et al., 2019).

Object detection algorithms localize objects in images by modifying pre-proposed regions or detecting bounding boxes. There are three different techniques for common object detection algorithms: 1. regressing offsets on default bounding boxes (similar to "You Only Look Once" or YOLO architecture (Redmon et al., 2016; Long et al., 2020)), 2. Variations of Single Shot Detection (Liu et al., 2016; Long et al., 2020), and 3. Two-staged R-CNNs (Ren et al., 2015; Long et al., 2020). Thus, from 2015 to 2018, there was a surge in end-to-end trainable, scene text detection and recognition neural networks (Long et al., 2020).

2.2. Scene Text VQA Datasets

Around 2015, Visual Question Answering (VQA) dataset was first released, and the Computer Vision community became increasingly interested in answering natural language questions about images (Biten et al., 2019). While some visual question answering datasets did indeed ask questions regarding scene text, notably VQA and VizWiz, the proportion of questions about image text were too small (Singh et al., 2019).

In 2019, (Singh et al., 2019) released TextVQA, which included 45, 336 questions asked by humans on 28,408 images from OpenImages dataset. Each image-question pair can have up to 10 different ground-truth answers. In the same year, (Biten et al., 2019) released ST-VQA. The authors describe it as a concurrent work to TextVQA. ST-VQA draws from eight different datasets as opposed to one, requires a minimum of two text instances to be present in an image, and provides unambiguous answers to questions (i.e. most questions have one answer, max two). It is expected that ST-VQA and TextVQA can be used in conjunction for transfer learning. (Biten et al., 2019).

More task-specific scene text VQA datasets arose since 2019. (Mishra et al., 2019) published OCR-VQA, which contains of 207, 572 images of book covers and over 1 million questions about author information, genre, etc. In 2020, DocVQA and STE-VQA datasets were released. DocVQA consists of image-question pairs based on handwritten, typewritten, and printed documents (Mathew et al., 2020). STE-VQA tackles bilingual scene text VQA in English and Chinese (Wang et al., 2020).

2.3. Standard VQA Models with Added Scene Text Retrieval

Standard VQA models are designed for generic VQA tasks; they are usually not scene text specific. "Show, Ask, Attend and Answer" (SAAA) (Kazemi & Elqursh, 2017) uses a CNN-LSTM architecture that is attended with multiple attention maps. Specifically, ResNet-152 is used to extract image features, a multi-layer LSTM embeds questions, and the result following attention maps is concatenated (Kazemi & Elqursh, 2017; Biten et al., 2019). "Stacked Attention Networks" (SAN) (Yang et al., 2016) is similar to SAAA, except it uses image features from VGG. These models alone do not perform well on scene text specific VQA tasks; however, they have increased performance with the addition of either scene text retrieval or scene image OCR (Kazemi & Elqursh, 2017; Yang et al., 2016). An example of scene text retrieval is a single shot CNN to predict bounding boxes and Pyramid of Histograms of Characters (PHOC) as compact representations of the text (Biten et al., 2019). A baseline for scene image OCR, presented in the paper introducing TextVQA, is "Look, Read, Reason Answer" (LoRRA) which uses a standard attention-based VQA model, Pythia, and combines it with an OCR module, Rosetta-ml OCR (Singh et al., 2019).

2.4. Transformer-Based Models

Transformer-based models use a transformer architecture across modalities. Multimodal Multi-Copy Mesh (M4C) projects the three modalities (questions, visual objects, text in image) into a common semantic space. These features are fed into a transformer to develop inter-modality and intra-modality relationships. This model improved upon LoRRA from an accuracy of 27.6 to 40.5 percent on TextVQA (Hu et al., 2020). "Spatially Aware M4C" is an extension of M4C but explicitly localizes spatial relationships between text in images and object features using a self-attention layer that looks only at neighboring entities through a spatial graph (Kant et al., 2020). "Structured Multimodal Attention" (SMA) uses graph neural networks and multimodal graph attention to form relationships between object-object, text-text, and object-text (Gao et al., 2020).

In our project, we build on the Multimodal Multi-Copy Mesh (M4C) to improve the performance on stop token generation and questions related to number and colors, which M4C struggled answering. To improve on these existing issues, we employ additional parameters to predict stop tokens, implement hierarchical architecture, and train further with additional data through pretraining and finetuning. To our knowledge, this project is the first to augment M4C with these features to address some shortcomings seen by M4C.

3. Proposed Approach

3.1. Model Description

Multimodal Multi-Copy Mesh (M4C) projects the three modalities (questions, visual objects, text in image) into a common semantic space. It uses dynamic pointers to predict a word at time through multi-step prediction, as opposed to a single final classification. M4C uses Faster R-CNN for image embeddings, BERT for question embeddings, Rosetta-en for OCR recognition. Rosetta-en uses Faster R-CNN, Pyramidal Histogram of Characters (PHOC), FastText, and bounding box coordinates for a rich OCR embedding. These features are fed into a transformer, along with the previous prediction embedding, to develop inter-modality and intra-modality relationships. For decoding, a dynamic pointer network first predicts words from either the OCR features or a fixed vocabulary until it predicts the end token (Hu et al., 2020).

x_k^{ques} is the embedding from a pretrained BERT model using the first 3 layers of BERT-BASE. x_m^{obj} is the sum of layer normalized and learned-weights from the last layer of Faster R-CNN detector, extracted via RoI-Pooling, as well as the object's four bounding box coordinates. M4C uses a rich OCR representation with four features; thus, x_n^{ocr} is the sum of layer normalized and learned weights from a 300-dimension FastText embedding on the OCR token, the same Faster R-CNN detector weights as before, a 604-dimensional PHOC vector, and the four bounding box coordinates as before (Hu et al., 2020).

We attain the vocabulary of the most frequent 5000 words in the training set and the OCR tokens per image for predicting y_t^*, thus there are a total of 5000+N candidates. For each input x_t we predict the ith word for the tth sample through iterative decoding, $y_{*t,i}$, by taking the argmax over the concatenation of $[y_t^{vocab}, y_t^{ocr}]$. Multi-label sigmoid loss is applied over $[y_t^{vocab}, y_t^{ocr}]$; that is, $-ylog(\text{sigmoid}(y^*)) - (1-y)log(1-\text{sigmoid}(y^*))$ (Hu et al., 2020).

BERT parameters and last layer of Faster R-CNN detector are fine-tuned during training. The attention weights of the decoded embedding are masked such that all decoding steps can only attend to previous decoding steps, questions, visual objects, and ORC tokens and the latter three cannot attend to decoding steps. During training, Adam optimizer is used with starting learning rate at 1e-4, learning rate decay at 0.1, and teacher-forcing (p=0.4) to the iterative decoder (Hu et al., 2020).

We define our main research problem using the following variables.

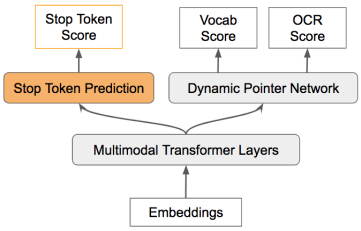


Figure 1. Decoder with Stop Token Predictor.

Variable	Definition
x_t	tth input (question, OCR, visual) or tth question
y_t^p	pth ground truth answer for tth question
x_k^{ques}	k^{th} word token in question
x^{obj}	whole image
x_m^{obj}	m^{th} object in image
x_n^{ocr}	n^{th} token in OCR
y_t^*	prediction for tth question
$y_{t,i}^*$	prediction for tth question, i^{th} token

We evaluated the performance of our models based off of standard evaluation metrics for each of our two datasets, TextVQA (Singh et al., 2019) and ST-VQA (Biten et al., 2019). We evaluated TextVQA using accuracy. Specifically, $Acc(y_t^*) = min(h(\frac{y_t^*}{3}), 1)$, where $h()$ is the number of people who picked the answer. The overall accuracy is computed as the average across all $Acc(y_t^*)$. ST-VQA is evaluated using Averaged Normalized Levenshtein Similarity (ANLS). The calculation of ANLS is given in Eq.1 if $NL(y_t^p, y_t^*) < 0.5$, where NL refers to the Normalized Levenshtein (Biten et al., 2019), and in Eq.2 otherwise.

$$\frac{1}{t}\sum_{i=0}^{t}(max_p h(y_t^p, y_t^*) \tag{1}$$

$$h(y_t^p, y_t^*) = 0 \tag{2}$$

3.2. Stop Token Prediction

To address the issue of stop token not being generated effectively in M4C based models, we added a stop token predictor, which predicts whether a stop token should be generated, to the model. This method has been used with text-to-speech systems, where a loss term dynamically predicts the EOS (end of sequence) token (Shen et al., 2018). In our model, the stop token predictor is placed after the decoder and in parallel to the dynamic pointer network (Fig.1), and it predicts the stop token score based on the multimodal decoder output. When the stop token predictor generates the stop token, the model stops the prediction. In our implementation, the stop token predictor was implemented as a linear layer as shown in Eq.3, when $y_{t,i}^{stop}$ is the stop token score

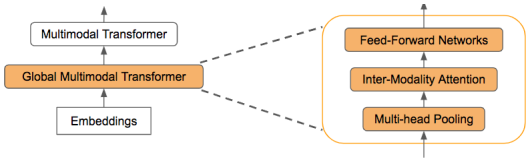


Figure 2. Model with Global Multimodal Transformer and its detailed view.

for tth question, ith token, and $x_{t,i}^{decoder}$ is the multimodal decoder output for tth question, ith token.

$$y_{t,i}^{stop} = W^{stop} x_{t,i}^{decoder} + b^{stop} \tag{3}$$

For this specific implementation, we also modified the loss function to enable the model to learn the stop token predictor. The loss function was modified by simply adding the loss term for stop token predictor without weights to the multi-label sigmoid loss originally used in M4C as seen in Eq.4. To compute the loss for the stop token predictor, we used binary cross entropy because the stop token predictor predicts either a positive sample, to generate the stop token, and a negative sample, to not generate the stop token. Finally, in order to account for the class imbalance when computing the loss, a constant weight was used for a positive sample.

$$Loss = Loss_{M4C} + Loss_{stop} \tag{4}$$

3.3. Hierarchical Transformer

"Structured Multimodal Attention" (Gao et al., 2020) showed using graph representations to form explicit relationships between object-object, text-text, and object-text modalities can improve the performance of M4C. In our approach, we developed a hierarchical transformer model to enable the model to learn the latent graphical representations between the modalities. The hierarchical transformer model is closely based on (Liu & Lapata, 2019), and it is implemented by adding a global multimodal transformer layer before the multimodal transformer as seen in Fig.2.

Global multimodal transformer layer includes three sections: multi-head pooling, inter-modality attention, and feed-forward networks. Multi-head pooling creates fixed-length representation of modalities with respect to the weight distributions of objects in representations. Compared to obtaining one representation for each modality by projecting the representations to a linear space, using multi-head pooling allows each modality to be represented in different representation subspaces. In multi-head pooling, probability distribution of objects in each modality is calculated by first obtaining the attention scores and the values and passing the attention scores through the softmax as shown in Eq.8. In the equation, m denotes mth modality, j denotes the jth token, h denotes the hth head. The number of head is typically 8.

$$\begin{aligned} a_{m,j}^h &= W_a^h x_{m,j} \\ v_{m,j}^h &= W_v^h x_{m,j} \\ p_{m,j}^h &= softmax(a_{m,j}^h) \end{aligned} \tag{5}$$

Then, the head vector is computed by linearly projecting the sum of probability distributions and the values and performing layer normalization, which is shown in Eq.6. The head vector represents different views of each modality.

$$head_m^h = LN(W_m^h \sum_{j=1}^{N} p_{m,j}^h v_{m,j}^h) \tag{6}$$

Next, the inter-paragraph attention lets the model learn relationships across the modalities. The inter-paragraph attention is similar to the self-attention layer in the vanilla transformer models, but the attention is performed between modalities instead of individual objects within the modalities. The equations used for the inter-paragraph attention is shown below:

$$\begin{aligned} query_m^h &= W_{query}^h head_m^h \\ key_m^h &= W_{key}^h head_m^h \\ value_m^h &= W_{value}^h head_m^h \\ context_i^h &= \sum_{m=1}^{3} softmax(query_m^h key_m^h) \end{aligned} \tag{7}$$

Finally, in order to combine the context obtained from the inter-paragraph attention and the input of the global multimodal transformer layer, or the embeddings of modalities, the concatenation of contexts from each heads are linearly transformed, then the sum of the transformed context and the inputs are passed through two linear layers, and finally the sum of the resulting representation and the inputs are normalized by layer. The final representation includes the information from the embedding as well as the hierarchical information.

$$\begin{aligned} context1_i &= W_{context} context_i^h \\ context2_i &= W_{linear1}(x_{m,j} + context1_i) \\ context3_i &= W_{linear2} RELU(context2_i) \\ \hat{x}_{m,j} &= LN(x_{m,j} + context1_i) \end{aligned} \tag{8}$$

Since the hierarchical transformer architecture does not change the outputs of the model, the multi-label sigmoid loss originally used in M4C was used to train the model.

3.4. Additional Learning

Both TextVQA and ST-VQA have relatively small datasets. TextVQA has a total of 39,602 training questions from 25,119 images, and ST-VQA has a total of around 26,308 training questions from 19,027 images. Both their test sets contain around 5,000 questions. We believe further pretraining and finetuning on VQA datasets can improve learning. Initially, we considered pretraining and performing ablation studies on incremental subsets of the VQA (v2.0) dataset which contains over 1 million questions and 200, 000 images from the COCO dataset. However, in our midterm report we used a pretrained version of Pythia, an up-down model that won the 2018 VQA challenge (Jiang et al., 2018; Anderson et al., 2018), and finetuned it on TextVQA and ST-VQA. We found that it performed rather poorly, and this is not surprising because Pythia does not use OCR. Also, we found that LoRRA, Pythia's counterpart that does use an OCR module, was already typically trained on some subset of VQA dataset (Singh et al., 2019). What we have not found in a scene text VQA paper from the last two years is explicit results pretraining on other scene text datasets. This is perhaps because many scene text datasets are very specific to a certain task or domain. OCR-VQA contains 186,775 training and validation images and 901,717 training and validation questions based on visual-question answering from book covers (Mishra et al., 2019). All other datasets are specific to some domain such as document writing or bilingual text and are also smaller datasets than TextVQA and ST-VQA (Wang et al., 2020; Mathew et al., 2020). Still, we decided to pretrain on incremental subsets of OCR-VQA, specifically to address issues we found in the midterm report. We believe this pretraining could bolster our M4C performance since M4C predicted incorrectly on images with book classes. Additionally, reading scene text often requires identifying words of the same font, style, or rotation in a small space, so viewing book covers helps uncover these spatial and stylistic similarities, which we previously identified as a top source of incorrect predictions for M4C. Thus, we will compare results from 1. Training on both TextVQA and ST-VQA to test if additional data does bolster learning and 2. Pretraining on OCR-VQA and finetuning on our datasets to assess if we can address the issue of better OCR recognition for spatial and stylistically similar text.

4. Experimental Setup

4.1. Datasets and Input Modalities

We analyzed the performance of three multimodal baseline models on the task of answering questions based on the texts in the image with two datasets, TextVQA (Singh et al., 2019) and ST-VQA (Biten et al., 2019).

TextVQA is based on Open Images v3 dataset, and it includes 28,408 images and 45,336 human-crafted questions. Each question has 10 human annotated answers. The dataset is officially split into training, validation, and test sets with sizes of 34,602, 5,000, and 5,734, respectively. VQAv2 accuracy is used as an evaluation metric, and it is computed as $Acc(y_t^*) = min(h(\frac{y_t^*}{3}), 1)$, where $h()$ is the number of people who picked the answer. The overall accuracy is computed as the average across all $Acc(y_t^*)$. Before generating these answers, processing is applied to labels and predictions: making all characters lowercase, removing periods except in decimals, removing articles, converting number words to digits, adding apostrophes in contractions, and replacing all other punctuation with a space (Singh et al., 2019).

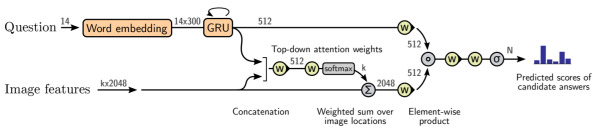
ST-VQA contains images from Imagenet, Vizwiz, ICDAR 2013 and 2015 robust reading competitions, Visual Genome, IIIT Scene Text Retrieval, and Coco-text. The dataset has different sets of data for three tasks, and we used the dataset for Task 3, where dictionary of answers are not provided to the model during inference. It contains 21,892 images and about 31,000 questions. Each question has up to two different answers. The dataset does not have an official validation set, so we used the split used in (Hu et al., 2020), where the training, validation, and test sets are divided into sizes of 17,028, 1,893, and 2,971, respectively. For evaluation, ST-VQA uses accuracy as well as Average Normalized Levenshtein Similarity (ANLS), which does not penalize OCR mistakes as heavily as accuracy. ANLS is computed as $\frac{1}{t}\sum_{i=0}^{t}(max_p h(y_t^p, y_t^*)$ for $h(y_t^p, y_t^*) = 1 - NL(y_t^p, y_t^*)$ if $NL(y_t^p, y_t^*) < 0.5$ and $h(y_t^p, y_t^*) = 0$ otherwise. NL refers to the Normalized Levenshtein (Biten et al., 2019).

For error analysis using the validation sets, we used an accuracy threshold of 0.35 and below to classify TextVQA as incorrect predictions and above 0.35 as correct predictions. For ST-VQA, we used the ANLS threshold of 0.5 and below to classify ST-VQA as incorrect predictions and above 0.5 as correct predictions. We calculated the ST-VQA accuracy based on this (Biten et al., 2019).

4.2. Multimodal Baseline Models

4.2.1. PYTHIA

The Pythia model (Jiang et al., 2018) is based on the up-down model in (Anderson et al., 2018).



Pythia architecture

The "up" part refers to a bottom-up attention model: Faster R-CNN (Ren et al., 2015) is used to detect objects and

generate the features for the image. The "down" part refers to a top-down attention: the question embedding is used to attend over the generated image features. When the two parts are combined, we can attend over detected objects instead of over pixels, which, as the authors showed, led to an improvement in accuracy. Note that the Pythia model does not use OCR.

Compared to the original top-down model, the Pythia model is different in the following ways: (1) for activation, weight normalization is used followed by ReLU. (2) For combining the modalities, concatenation is used instead of the element-wise product shown in the picture.

4.2.2. LoRRA

LoRRA (Look, Read, Reason Answer) has three components, the VQA component, the reading component, and the answering component. The VQA component lets the model understand the image with respect to the question. The reading component also allows the model to understand the text in the image with respect to the question. Finally, the answering component predicts the answer to the question from a fixed dictionary or from a text from the image.

The VQA component creates VQA features that are element-wise products of corresponding word embeddings and attended image features. First, question words, x^{ques}, and images, x^{obj}, are embedded using models that are not jointly trained with LoRRA. In LoRRA, word embeddings are created using pre-trained GLoVe (Pennington et al., 2014), and image features are extracted using ResNet-152 (He et al., 2015). Then, corresponding image features and question word embeddings are passed into an attention layer, where the image features are spatially weighted with respect to the question word embeddings. The attended image feature and the question embeddings are combined using element-wise multiplication to produce the VQA embedding, x^{VQA}. The VQA component can be represented as Eq. 9.

$$x_t^{VQA} = Attention(x_t^{obj}, x_t^{ques}) \odot x_t^{ques} \tag{9}$$

Similarly to the VQA component, the reading component creates an OCR feature, which is an element-wise product of word embeddings and attended OCR tokens. The same word embeddings as the VQA component are used for the reading component. OCR embeddings are extracted using an OCR for each OCR token, x^{ocr}, which is not jointly trained with LoRRA. FastText was used to create OCR embeddings in LoRRA. The recognized OCR embeddings and word embeddings are attended to create weighted OCR features based on the question word embeddings. Finally, the attended OCR features and word embeddings are also multiplied element-wise to form attended OCR features, x^{read}. The equation for the reading component is shown in Eq 10.

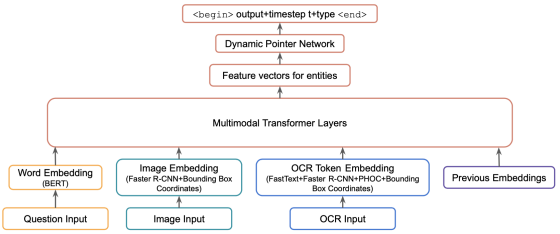
$$x_t^{read} = Attention(x_t^{ocr}, x_t^{ques}) \odot x_t^{ques} \tag{10}$$

The answering component is a two layer multilayer perceptron, which takes in concatenated features from the VQA component and the reading component and outputs binary logits over the answer space, x^{ans}. The answer space is a combination of the dictionary of N most frequent words that appear in the training set of the dataset and M OCR tokens from the image. The answering component can be represented as Eq. 11. The output of the multilayer perceptron is fed into a combination of sigmoid layer and binary cross entropy loss Eq. 12, which is a more numerically stable formula compared to using sigmoid layer and binary cross entropy loss sequentially.

$$x_t^{ans} = MLP([x_t^{VQA}; x_t^{read}]) \tag{11}$$

$$\begin{aligned} Loss_{lorra} \\ = -\frac{1}{N+M} \sum_{n=1}^{N+M} [y_{t,n} log(\sigma(y^*_{t,n})) \\ + (1 - y_{t,n}) log(1 - \sigma(y^*_{t,n}))] \end{aligned} \tag{12}$$

4.2.3. M4C



M4C architecture

Multimodal Multi-Copy Mesh (M4C) projects the three modalities (questions, visual objects, text in image) into a common semantic space. It uses dynamic pointers to predict a word at time through multi-step prediction, as opposed to a single final classification. M4C uses Faster R-CNN for image embeddings, BERT for question embeddings, Rosetta-en for OCR recognition. Rosetta-en uses Faster R-CNN, Pyramidal Histogram of Characters (PHOC), FastText, and bounding box coordinates for a rich OCR embedding. These features are fed into a transformer, along with the previous prediction embedding, to develop inter-modality and intra-modality relationships. For decoding, a dynamic pointer network first predicts words from either the OCR features or a fixed vocabulary until it predicts the end token (Hu et al., 2020).

x_k^{ques} is the embedding from a pretrained BERT model using the first 3 layers of BERT-BASE. x_m^{obj} is the sum of layer normalized and learned-weights from the last layer of Faster R-CNN detector, extracted via RoI-Pooling, as well as the object's four bounding box coordinates. M4C uses a rich OCR representation with four features; thus, x_n^{ocr} is the sum of layer normalized and learned weights from a 300-dimension FastText embedding on the OCR token, the same Faster R-CNN detector weights as before, a 604-dimensional PHOC vector, and the four bounding box coordinates as before (Hu et al., 2020).

We attain the vocabulary of the most frequent 5000 words in the training set and the OCR tokens per image for predicting y_t^*, thus there are a total of 5000+N candidates. For each input x_t we predict the ith word for the tth sample through iterative decoding, $y_{*t,i}$, by taking the argmax over the concatenation of $[y_t^{vocab}, y_t^{ocr}]$. Multi-label sigmoid loss is applied over $[y_t^{vocab}, y_t^{ocr}]$; that is, $-ylog(\text{sigmoid}(y^*)) - (1-y)log(1-\text{sigmoid}(y^*))$ (Hu et al., 2020).

BERT parameters and last layer of Faster R-CNN detector are fine-tuned during training. The attention weights of the decoded embedding are masked such that all decoding steps can only attend to previous decoding steps, questions, visual objects, and ORC tokens and the latter three cannot attend to decoding steps. During training, Adam optimizer is used with starting learning rate at 1e-4, learning rate decay at 0.1, and teacher-forcing (p=0.4) to the iterative decoder (Hu et al., 2020).

4.3. Experimental Methodology

4.3.1. Hierarchical Transformer

The table below summarizes the experimental setup for the hierarchical transformer, including hyper-parameters.

Parameter	Value
Global Multimodal Transformer Layers	1
Multimodal Transformer Layers	3
Batch Size	60-80
Learning Rate	1e-4 *

* Note that warmup was used during the first epoch.

4.3.2. Additional Learning

For the additional pre-training and fine-tuning portion of this project, we used a M4C model pre-trained on OCR-VQA using MMF. It achieved a validation accuracy of 63.52% and a test accuracy of 63.87%. The results were evaluated incrementally on Text-VQA for 2 epochs, 5 epochs, and 10 epochs.

Parameter	Value
Optimizer	Adam
Warm-Up Learning Rate Factor	0.2
Max Gradient L2-Norm for Clipping	0.25
Learning Rate Decay	0.1
Learning Rate Steps (OCR-VQA)	28000, 38000
Max Iterations (OCR-VQA)	48000
Batch Size	80

5. Results and Discussion

5.1. Stop Token Prediction

We observe a 0.76% decrease in accuracy for ST-VQA and a 0.97% increase in accuracy for Text-VQA when stop token prediction is included. From the error analysis, we make the following observations on how the errors have changed when stop token prediction is included.

1. A higher ratio of vocabulary vs. OCR as the source of error. For ST-VQA, vocabulary accounts for 53.5% of wrong predictions compared to 47.4% previously. For Text-VQA, the ratio changed from 46.4% to 51.7%. The phenomenon is also present when we implement the hierarchical transformer, and we will make more observations and discussions in the following section.

2. The categories of errors remain largely the same after implementing the stop token network. This is expected, since predicting the stop token better only fixes certain types of errors. Regarding counting and identifying colors, which are the two most common types of errors, the stop token method will not improve the accuracy in these cases. As a consequence, these two types of errors remain the most common ones even with a stop token network.

3. Stop token network might not have changed the predictions in a way that makes a significant difference. Out of 1309 instances where the old and new predictions differ, the new prediction stops early in only 3.2% of the cases. These cases did not have a big influence on the overall accuracy of the model.

5.2. Hierarchical Transformer

We observed a 0.01% increase in accuracy and 0.01 increase in average ANLS on ST-VQA validation dataset with the added hierarchical components to the M4C architecture. While this might not be notably different, we did observe a difference in distributions of correct versus incorrect predictions across the two predictions sources for M4C. Recall that M4C performs iterative decoding where each subsequent predicted word is either from a vocabulary bank or from the image OCR. From the midterm report on the original M4C architecture, we observed that for ST-VQA, there is

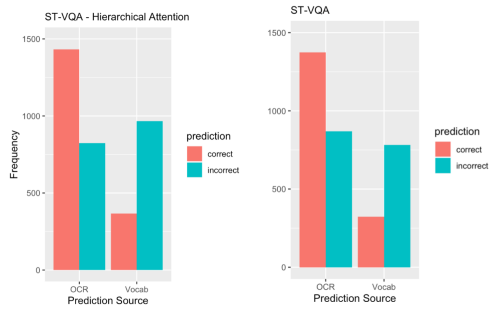


Figure 3. Hierarchical M4C vs M4C prediction sources

Figure 4. Hierarchical M4C reads 'GEILS' while M4C reads 'LA'

a much higher ratio of predictions in OCR to predictions in vocabulary for correct predictions compared to that of incorrect predictions. This discrepancy is also observed in TextVQA, with a lower ratio. This might indicate that more often answers are grounded in scene text (not surprising), and perhaps that the inability to decipher or reason about OCR may lead to incorrect predictions. Now with the added hierarchical transformer architecture, we observe that there is a much higher increase in the number of incorrect predictions that draw from the vocab source; in fact, there are more incorrect predictions that drew from the vocab source as opposed to the OCR source.

With this research idea, it seems that number of incorrect OCR predictions decrease as the number of incorrect vocab predictions increase and the number of correct predictions (for both OCR and vocab) increase. We propose this could indicate that given our model believes an answer can be drawn from OCR, it has better capability of recognizing the OCR. Or, it could just be that our model is now more inclined to draw from the vocabulary source. Below is an example of a question our hierarchical M4C transformer answers correctly over the previously incorrect M4C. M4C previously could not recognize the tilted, scrunched letters, but now our model outputs all the letters (an ANLS score of 0.8).

We found that with the added hierarchical components, there was no change in the nature of most frequent tokens in questions and predictions as with the original M4C architecture we studied in the midterm. There were no notable patterns between the word tokens in questions and correct/incorrect predictions in either dataset. One observation we did note is that in the original M4C, ST-VQA correct predictions often were identifying colors or word-for-word reading of signs or labels in images when there were one or two of these per image. Both ST-VQA and TextVQA incorrect predictions relied on counting objects or inferring about numbers (like reading time) and occasionally identify colors. Now with the hierarchical M4C transformer, we saw an improvement in the ability to count objects or recognize numbers, but the issue of incorrectly identifying colors remained.

Question: What color is the rooster?
Answer: Black
Predicted: Red

Figure 5. Hierarchical M4C incorrectly understands colors

5.3. Additional Learning

We evaluated TextVQA validation accuracy after jointly training on both TextVQA and ST-VQA. We found that with this 75% increase in training questions (and images), the validation accuracy did increase from 38.4% to 39.45%. Thus, we deemed it beneficial to train on additional data. Then, we pretrained M4C on the full OCR-VQA and finetuned on TextVQA before evaluating TextVQA validation results. We found that after finetuning for just 2 epochs, the TextVQA validation accuracy reached 29.9%, after training for 5 epochs the accuracy reached 37.8%, and after training for 10 epochs, the accuracy reached 40.42%, beating the joint TextVQA-ST-VQA benchmark. However, when performing error analysis, we found no distinctions in the nature of incorrect answers from the original M4C training and this revision. Again, the reason we proposed this method was to address problems we found in the midterm: M4C continually struggles to infer numerical relations and color relations. We believe pretraining on OCR-VQA can help general learning with an influx of data, but also because it can help learn color relations or spatial relations given that OCR-VQA is tailored to recognize text of same sizes, fonts, and rotational positions in book covers. But, we did not see this improvement. By observing the ground truth labels the model predicted incorrectly versus correctly across M4C

Figure 6. Ground Truth Tokens from Wrong Predictions with Learning Idea

and M4C with additional training, they look practically the same as shown in the word cloud below. We are interested if perhaps pretraining on a dataset such as CLEVR, which specializes in visual and spatial reasoning, can help bolster this addition learning problem.

6. Conclusion and Future Directions

In this project, we presented three approaches to improve the performance of Multimodal Multi-Copy Mech model (M4C). The stop token predictor is an additional parameter to help with decoding and effectively generate the stop token, the hierarchical transformer architecture enabled the model to learn the latent relationships present between the modalities, and additional learning with datasets to help the model gain more data concerning numbers and colors. The results show that adding a stop token network did not cause a significant change in accuracy, although a more careful error analysis shows slight differences in the mistakes that the model made. For hierarchical transformer, the results show that it does not change the results compared to the results from the state-of-the-art model but it was able to improve on number-related questions, and the additional learning improved the overall accuracy of the model. In the future, the model could abandon the fixed vocabulary predictor and solely predict answers using OCR tokens with the introduction of stop token predictor, which could make the model more generalizable, and pretraining with different datasets could be explored to enable the model to learn different relationships further. Finally, further exploration with hierarchical architecture should be explored, starting with if it was correctly implemented in this limited timeframe, since it was able to reach the state-of-the-art performance and has some potential with the improvement in number related questions.

7. Github

Github Link: https://github.com/ntsuda-cmu/11777-Project

References

Anderson, P., He, X., Buehler, C., Teney, D., Johnson, M., Gould, S., and Zhang, L. Bottom-up and top-down attention for image captioning and visual question answering. *CVPR*, 2018.

Biten, A. F., Tito, R., Mafla, A., Gomez, L., Rusiñol, M., Valveny, E., Jawahar, C., and Karatzas, D. Scene text visual question answering. *ICCV*, 2019.

Gao, C., Zhu, Q., Wang, P., Li, H., Liu, Y., van den Hengel, A., and Wu, Q. Structured multimodal attentions for textvqa. Technical report, School of Computer Science, Northwestern Polytechnical University, Xi'an, China School of Software, Northwestern Polytechnical University, Xi'an, China 3National Engineering Laboratory for Integrated Aero-Space-Ground-Ocean Big Data Application Technology, China University of Adelaide, Australia, 2020.

Graves, A., Fernandez, S., Liwicki, M., Bunke, H., and Schmidhuber, J. Unconstrained online handwriting recognition with recurrent neural networks. *Advances in neural information processing systems*, 2008.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition, 2015.

Hu, R., Singh, A., Darrell, T., and Rohrbach, M. Iterative answer prediction with pointer-augmented multimodal transformers for textvqa. *CVPR*, 2020.

Jiang, Y., Natarajan, V., Chen, X., Rohrbach, M., Batra, D., and Parikh, D. Pythia v0.1: the winning entry to the VQA challenge 2018. *CoRR*, abs/1807.09956, 2018. URL http://arxiv.org/abs/1807.09956.

Kant, Y., Batra, D., Anderson, P., , Schwing, A., Parikh, D., Lu, J., and Agrawal, H. Spatially aware multimodal transformers for textvqa. *ECCV*, 2020.

Kazemi, V. and Elqursh, A. Show, ask, attend, and answer: A strong baseline for visual question answering. *arXiv preprint arXiv:1704.03162*, 2017.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. Ssd: Single shot multibox detector. *EECV*, 2016.

Liu, Y. and Lapata, M. Hierarchical transformers for multi-document summarization, 2019.

Long, S., He, X., and Yao, C. Scene text detection and recognition: The deep learning era. *International Journal of Computer Vision*, 2020.

Mathew, M., Karatzas, D., Manmatha, R., and Jawahar, C. Docvqa: A dataset for vqa on document images. *CVPR*, 2020.

Mishra, A., Shekhar, S., Singh, A. K., and Chakraborty, A. Ocr-vqa: Visual question answering by reading text in images. *ICDAR*, 2019.

Pennington, J., Socher, R., and Manning, C. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL https://www.aclweb.org/anthology/D14-1162.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. You only look once: Unified, real-time object detection. *CVPR*, 2016.

Ren, S., He, K., Girshick, R. B., and Sun, J. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497, 2015. URL http://arxiv.org/abs/1506.01497.

Shen, J., Pang, R., Weiss, R. J., Schuster, M., Jaitly, N., Yang, Z., Chen, Z., Zhang, Y., Wang, Y., Skerry-Ryan, R., Saurous, R. A., Agiomyrgiannakis, Y., and Wu, Y. Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. 2018.

Singh, A., Natarajan, V., Shah, M., Jiang, Y., Chen, X., Batra, D., Parikh, D., and Rohrbach, M. Towards vqa models that can read. *CVPR*, 2019.

Tian, S., Pan, Y., Huang, C., Lu, S., Yu, K., and Tan, C. L. Text flow: A unified text detection system in natural scene images. *Proceedings of the IEEE international conference on computer vision*, 2015.

Wang, X., Liu, Y., Shen, C., Ng, C. C., Luo, C., Jin, L., Chan, C. S., van den Hengel, A., and Wang, L. On the general value of evidence, and bilingual scene-text visual question answering. *CVPR*, 2020.

Yang, Z., He, X., Gao, J., Deng, L., and Smola, A. Stacked attention networks for image question answering. *IEEE conference on computer vision and pattern recognition*, 2016.