# Generalized Textual and Visual Question Answering using Combined Image-Context Embeddings

**Manan Rai**
Department of Computer Science
Stanford University
mananrai@stanford.edu

## Abstract

A number of recent works have made progress in the fields of Textual as well as Visual Question Answering. These tasks, although similar, are experiencing singular and distinct progress, which is disparate from real-world comprehension, where images and context go hand-in-hand. We propose the task of generalized Context- and Image-based Question Answering (QA), which we call Textual and Visual Question Answering (TeVQa), where a single model learns from both forms of input data. An import aspect of the generalized text-and-image-based Question Answering (QA) task is the development of a model that can process both forms of inputs simultaneously. To this effect, we propose a joint embedding that can capture the salient features of text and image inputs alike, and a generalized QA network that can process this combined embedding for only text, only image, as well as both text and image inputs. We experiment with a VGG-19, ResNeXt, and Mask R-CNN for Image input, and with Highway Networks and GloVe Embeddings for Context input, and propose an Embeddings Combination Network that combines them to build a joint embedding. We experiment with this embedding on both VQA 2.0 and SQuAD 2.0 datasets. Our models achieve strong results on both datasets, and represent a solid first attempt at generalized TeVQA.

## 1 Introduction

The tasks of free form Context- and Image-based QA are well-developed and well-researched. While Textual Question Answering (commonly referred to as QA but here TQA for clarity of notation) is a native Natural Language Processing (NLP) task wherein a model learns from natural language text and answers natural language questions, the task of Visual Question Answering (VQA) is more interdisciplinary since it combines understanding the natural language questions with extracting information from input images - which requires Computer Vision (CV).

Yet there is a major limitation in the way the fields of NLP and CV currently interact, since inputs of different forms (text versus image) are processed through entirely different pipelines. Since the goal of all QA tasks is to enhance machine comprehension of various input sources, the current model of research poses a limitation to the generalizability of this comprehension. Consider, for instance, how major context datasets such as the Stanford Question Answering Dataset (SQuAD) are built from rich sources of information - Wikipedia articles. Yet a large proportion of human edification of such sources is through not just the text, but also the images that go along with it. A model that is able to consume both textual and visual sources with ease and accuracy, and answer questions based on the same, would therefore be more in keeping with the goal of the QA task in general.

A major technical impediment for the development of a generalized model has been the distinct natures of the features that can be extracted from the different forms of input. Here, we propose a pipeline that generates a joint embedding from both context and image inputs, and a model that can process these embeddings with success on both major TQA and VQA datasets.

## 2 Related Work

For both QA tasks, the overall pipeline can be viewed as a series of subprocesses involving extracting features from the input and question, relating the features, and then building the answer.

VQA is a prime research area in NLP ever since it was proposed in the seminal paper (1). In (1), Antol et al. introduce methods using VGG networks trained on the ImageNet dataset to extract image features. ResNeXt (2) and Mask R-CNN (3; 4; 5) networks have achieved a lot of success on object detection and semantic segmentation tasks on the MS COCO image dataset. Although these networks haven't been tried for image embeddings, Andersen et al do discuss the use of a Faster R-CNN for their Bottom-Up Attention technique in (6).

TQA is a related problem that has been around since before its visual counterpart, and models have beaten human performance on earlier datasets such as SQUAD 1.0 (7). Here, we experiment with the Bidirectional Attention Flow (BiDAF) model, introduced by Seo et al in (8), and the Question Answerin Network (QANet), introduced by Yu et al in (9).

Much work has been done in each of the fields individually, and techniques including various Attention mechanisms (6; 10; 8; 9), Gated Recurretn units (GRUs) (11), among several others, have been proposed to tackle these tasks. However, no methods that combine the comprehension of these two disparate input formats have been studied so far.

## 3 Approach

### 3.1 Embedding Layer Overview

In order to build a general QA model, we need to have an embedding that can represent both image and textual context inputs. In order to do so, we use a branched network that forks based on the input. Section 3.2 deals with the various models we implemented for the Image embedding. Section 3.3 deals with models for the Context embedding. Finally, in section 3.4, we go over how the embeddings obtained using any of the considered methods are combined to build the joint embedding.

### 3.2 Image Embeddings

#### 3.2.1 VGG-19

We use two types of convolutional blocks:

- **Block A:** Two $3 \times 3$ `Conv2D` layers with , followed by a `MaxPool` layer.
- **Block B:** Four $3 \times 3$ `Conv2D` layers, followed by a `MaxPool` layer.

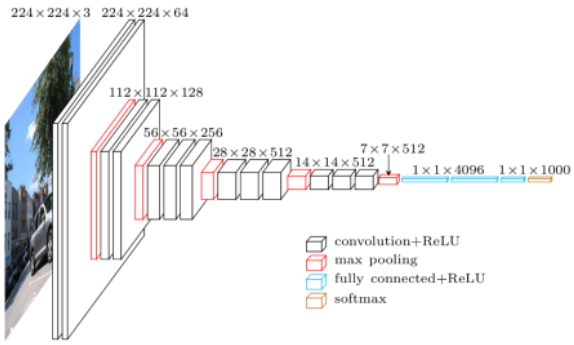


Figure 1: VGG-19 Architecture

Given the input image $u \in \mathbb{R}^{w \times w}$, we use three A blocks, followed by two B blocks, the output, $v$, of which is passed through fully-connected layers, to get:

$$h = W_{\text{FC}_1} v + b_{\text{FC}_1}$$

$$h' = W_{\text{FC}_2} h + b_{\text{FC}_2} \in \mathbb{R}^{4096}$$

which becomes the 4096-dimensional image embedding, $I_{\text{emb}} = h'$.

### 3.2.2 ResNeXt

A ResNeXt is an upgraded version of the Deep Residual Network (ResNet) that uses a split-transform-aggregate strategy. At every cell, the input is split into a cardinality ($c$) number of paths, each of which compute a set of convolutional transformations with the exact same topology, finally leading to an and-gate that aggregates the outputs from all the paths. Setting $c = 1$, we implement a ResNet-equivalent ResNeXt.

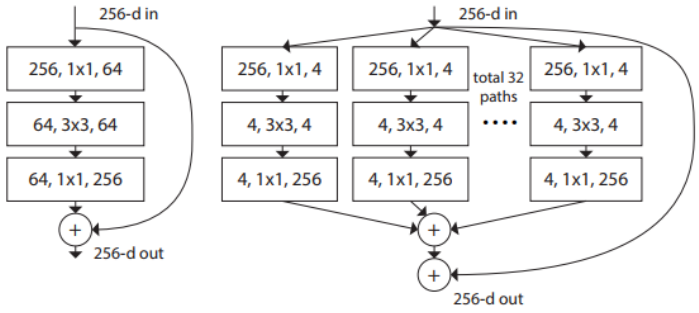


Figure 2: (a) ResNet block; and (b) ResNeXt counterpart

For input $x$, this achieves the following computation:

$$y = x + \mathcal{F}(x)$$
$$x' = \text{ReLU}(y)$$

where $\mathcal{F}$ is a non-linear transformation. We set $\mathcal{F}$ to be a $3 \times 3$ convolution, followed by Batch Normalization and ReLU activation, and finally another $3 \times 3$ convolution. During backpropagation from layer $l'$ to $l$, for loss $E$, this gives us:

$$\frac{\partial E}{\partial x^{(l)}} = \frac{\partial E}{\partial x^{(l')}} * (W^{(l'-1)} + 1) * \cdots * (W^{(l)} + 1)$$

We use the concatenated activations of the last two fully-connected layers, each $\in \mathbb{R}^{2048}$, to get a 4096-dimensional image embedding, $I_{\text{emb}}$.

### 3.2.3 Mask R-CNN

The Mask R-CNN is a framework for effective semantic segmentation. An extension of the Faster R-CNN model, it uses a pipeline employing a Region Proposal Network with Non-max Suppression to identify bounding boxes for objects of interest in the image and predict the class of each object, along with a fully Convolutional Network that conducts pixel-level segmentation to generate masks.

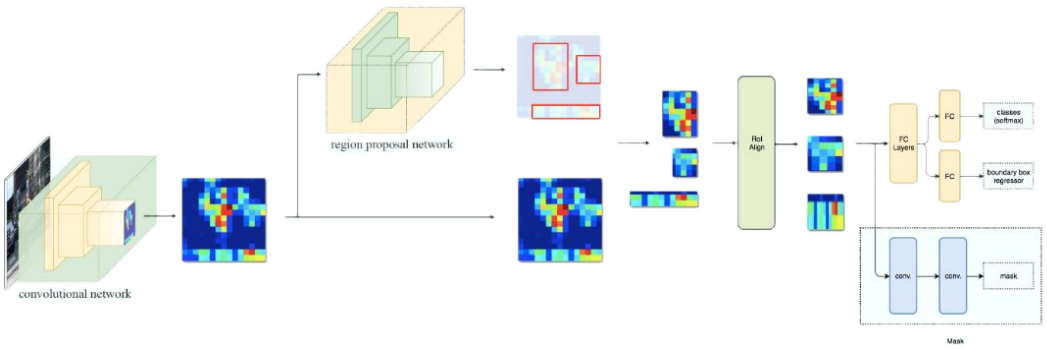


Figure 3: Mask R-CNN Architecture

The Region Proposal Network (RPN) uses a feature extractor to extract features for the entire image and build a feature map. Then, in order to identify Regions of Interest (ROIs), we use Selective

Search (SS). An R-CNN gets 2000 ROIs, warps each to a fixed size, and then processes each using a CNN. The Faster R-CNN improves efficiency by using the aforementioned feature map to identify features for each of the ROIs. The Faster R-CNN further improves on this performance by replacing the RPN with a deep neural network-based RPN that derives ROIs from the feature map itself.

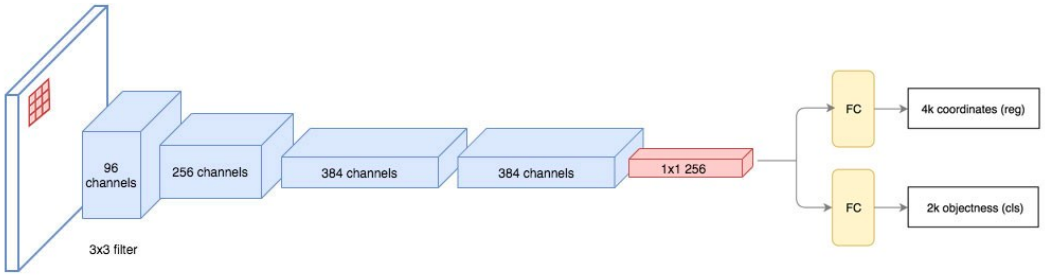


Figure 4: Region Proposal Network (RPN)

A Mask R-CNN typically uses a ResNeXt as its backbone. We also implement a DenseNet-121 to act as the backbone. For either backbone, we extract the activations of the last layer of the boundary box regressor to act as our 4096-dimensional image embedding, $I_{\text{emb}}$.

### 3.3 Context Embedding

#### 3.3.1 Highway Network

For the textual QA, we use the Bidirectional Attention Flow model. Given textual context, we extract the input word indices from the vocab, and find the word embeddings $c_1, \ldots, c_N \in \mathbb{R}^{\mathbb{D}}$ for the corresponding words. Then we find a $H$-dimensional projection as

$$h_i = W_{\text{proj}} v_i \in \mathbb{R}^H$$

and then use a two-layer Highway Network, where a single layer performs the following computations:

$$\begin{aligned} g^{(1)} &= \sigma(W_{\text{gate}} h_i + b_{gate}) \in \mathbb{R}^H \\ t^{(1)} &= \text{ReLU}(W_{\text{transform}} g + b_{\text{transform}}) \in \mathbb{R}^H \\ h_i'^{(1)} &= g \odot t + (1 - g) \odot h_i \in \mathbb{R}^H \end{aligned}$$

This gives us $h_i'^{(2)}$. We concatenate $h_i'^{(2)}$ values for all words to get a $NH$-vector as the context embedding $C_{\text{emb}}$.

#### 3.3.2 GloVe Embeddings

Global Vectors (GloVe) capture linear substructures of word vector space and place linguistically and/or semantically similar words near one-another. We experiment with pre-computed GloVe word embeddings as a means to capture the salient features of the context in $C_{\text{emb}}$.

### 3.4 Combining the Image and Context Embeddings

Now that we have individual embeddings for the context and image, we use a non-linear transformation to combine the two. For each embedding $X_{\text{emb}}$, we calculate a 100-feature vector as:

$$\begin{aligned} X_{\text{emb}}^{(1)} &= \text{ReLu}(\texttt{Conv1D}(X_{\text{emb}})) \\ X'_{\text{emb}} &= \text{ReLu}(\texttt{Conv1D}(X_{\text{emb}}^{(1)})) \end{aligned}$$

and then obtain the final embedding as $e_{\text{inp}} = I'_{\text{emb}} + C'_{\text{emb}}$, where the convolutional filters have learnable parameters. We call this the Embeddings Combination Network.

### 3.5 Question Layer

We use a LSTM with two hidden layers. Each hidden layer produces 512-dimensional hidden and cell states, $h_{\text{enc}}^{(l)}$ and $c_{\text{enc}}^{(l)}$ respectively. These states from each of the two layers are concatenated to get a 2048-dimensional encoding. This is then processed using a fully-connected layer as

$$e_{\text{ques}} = \tanh(W_{\text{FC}_1}\ [h_{\text{enc}}^{(1)};\ c_{\text{enc}}^{(1)};\ h_{\text{enc}}^{(2)};\ c_{\text{enc}}^{(2)}] + b_{\text{FC}_1})$$

which gives us a 1024-dimensional question embedding $e_{\text{ques}}$.

### 3.6 Answer Network

The joint input embedding, and the question embedding, now become the input to an Answer Network, that uses the features captured by the embeddings to come up with an answer for the queried question. We consider networks specialized for both individual VQA (section 3.6.1) and TQA (sections 3.6.2 and 3.6.3) tasks.

#### 3.6.1 Multi-Layered Perceptron (MLP)

We use a Multi-Layered Perceptron (MLP) with a fully-connected 2-layer architecture with 1000 hidden units in each layer. Each layer achieves the following transformations:

$$h_{\text{FC}_1} = \tanh(W_{\text{FC}_1}\ e_{\text{inp}} + b_{\text{FC}_1})$$
$$h'_{\text{FC}_1} = \text{Dropout}(h_{\text{FC}_1})$$

with probability $\rho = 0.5$, and finally get the probability distribution over $K$ possible answers as

$$s = \text{softmax}(h'_{\text{FC}_2}).$$

#### 3.6.2 Bidirectional Attention Flow (BiDAF)

BiDAF is centred on the concept that attention should flow both from the context to the question, and from the question to the context. Softmax regularization of the rows and columns of the similarity matrix $S$ for all pairs $(c_i, q_j)$ of context and question hidden states is used to obtain the context-to-question ($a_i$) and question-to-context attention ($b_i$) outputs respectively. These, and the hidden state $c_i$, together give us the output of the BiDAF layer. These are then modelled using a bidirectional two-layer LSTM. The outputs of this computation are then processed using another bidirectional LSTM as part of the output layer, and use the softmax function to get a probability distribution across possible answers.

#### 3.6.3 QANet

The QANet expects from the encoding layer an input similar to that used by the BiDAF model:

$$g = [c_i, a_i, c_i \odot a_i, c_i \odot b_i]$$

We use 3 encoding layers, giving us outputs $M_1$, $M_2$, and $M_3$. These are then concatenated and processed to predict the probability of each position being a start or end position as

$$p_{\text{start}} = \text{softmax}(W_1\ [M_1; M_2])$$
$$p_{\text{end}} = \text{softmax}(W_2\ [M_1; M_3])$$

The score of a proposed span for an 'extractive' answer is the probability of the respective words' start and end probabilities.

### 3.7 Loss

For both tasks, we use the cross entropy loss. For VQA, we do so for the gold-standard answer. For TQA, this is calculated for the start and end locations since our TQA models are extractive, i.e. they answer questions using phrases from the context. For TeVQA, this is the same as for the individual tasks, and it depends on the input format.

# 4 Experiments and Results

## 4.1 Datasets

### 4.1.1 Images

The VQA 2.0 dataset is the largest dataset for the Visual Question Answering task. It contains human-annotated questions and answers on images from the Microsoft COCO dataset. The dataset contains 204,721 images, 1,105,904 questions, and 11,059,040 ground-truth answers. There are three sub-categories according to answer-types including yes/no, number, and other. Each question has 10 free-response answers. We use the top 1000 most frequent answers as the possible outputs.

### 4.1.2 Context

We use the Stanford Question Answering Dataset (SQuAD), which contains questions on a set of Wikipedia articles, where the answer to every question is a segment of text, or span, from the corresponding reading passage. We have 129,941 examples in the train set, 6,078 examples in the dev set, and 5,915 examples in the test set, where each example in the train/dev sets is a (context, question, answer) triple.

## 4.2 Individual Task-Specific Experiments

In order to identify the strongest embedding networks for the individual tasks, we initially conducted individual task-specific experiments. We tested the VGG, ResNeXt, and Mask R-CNN with ResNeXt and DenseNet backbones on the VQA task, and the BiDAF and QANet models on the TQA task.

We froze VGG weights trained on ImageNet data. For ResNeXt and Mask R-CNN, we initialized pre-trained weights from the MS COCO dataset. Since different backbones for the Mask R-CNN don't have open source weights, we froze weights to the ones achieved after training on MS COCO for $\sim$12 hours on an NVIDIA GeForce 1800 Ti GPU. Some important model hyperparameters are summarized in Table 1. Due to limited time and resources, the DenseNet was only trained for 4,000 iterations and could not perform well on the VQA task. For this reason, results obtained using that network have not been reported. For TQA, we used 300-dimensional GloVe word embeddings. Once the individual embeddings ($I_{\text{emb}}$ and $C_{\text{emb}}$) were obtained, we tested $I_{\text{emb}}$ using the MLP, and $C_{\text{emb}}$ using the BiDAF and QANet models. Results obtained after training for $\sim 50$ hours on Tesla M60 GPUs, Azure NV12 VMs, and AWS Ubuntu Deep Learning AMI EC2 instances are summarized in Tables 2 and 3.

| Model | Learning Rate | Batch Size | Maximum Iterations |
|---|---|---|---|
| VGG | 0.0003 | 500 | 150,000 |
| ResNeXt | 0.001 | 5 | 35,000 |
| Mask R-CNN | 0.001 | 5 | 35,000 |
| BiDAF | 0.5 | 64 | 3,000,000 |
| QANet | 0.001 | 16 | 60,000 |

Table 1: Model hyperparameters

| Model | Performance |
|---|---|
| VGG | 60.62 |
| ResNeXt | 60.19 |
| Mask R-CNN | 62.88 |

Table 2: Results on the VQA task

| Model | Validation | | |
|---|---|---|---|
| | EM | F1 | NLL |
| BiDAF | 54.93 | 58.39 | 3.589 |
| QANet | 65.21 | 75.64 | 3.220 |

Table 3: Results on the TQA task

### 4.3 TeVQA Experiments

In order to test the joint embeddings, we chose the best performing embeddings – Mask R-CNN output and GloVe embeddings for $I_{\text{emb}}$ and $C_{\text{emb}}$ respectively – and then combined the embeddings as described in section 3.4. Then the input and question embeddings were trained on the MLP, with examples from both datasets. As a result, the parameters for the Embeddings Combination Network were learned. The resulting model was then tested individually on each of the datasets. On the VQA 2.0 dataset, we achieved a performance score of 49.29. On the SQuAD 2.0 dataset, we achieved Exact Match (EM) 45.33 and an F1 Score of 48.54.

## 5 Quantitative Analysis

We see that although the TeVQA model does not perform as well as individual task-specific QA, it achieves comparable results for both input formats. Note that we only adopted basic models for the Answer Network, and did not utilize Attention methods to their full extent, especially on the VQA dataset. We believe that the results achieved so far are definitely improvable, yet already represent a step towards a generalized input-format-independent QA model.

## 6 Qualitative Analysis

Consider the following image of two women playing soccer. Here we present some errors that the model makes.

Figure 5: (a) An example image; and (b) image with overlaid masks

We find that changing the phrasing of the question can confuse the model. Consider, for instance, the following pair of questions:

| | |
|---|---|
| **Q:** What are they playing? | **Q:** What are they doing? |
| **Ground Truth:** Soccer | **Ground Truth:** Playing soccer |
| **Prediction:** Soccer | **Prediction:** Playing frisbee |

Even though the meaning of the question doesn't change, the model gets confused and predicts the wrong sport in the second question. Although the reason for this is hard to predict, it is likely that adding object masks and classes to the feature vector might help the model learn that since there is *no* frisbee in the picture, it is unlikely that the women are playing frisbee.

The following examples highlight some social biases that are reflected by the model. Since the model isn't specifically looking the gender of the players, it misses this information when building answers.

| | |
|---|---|
| **Q:** How many men are playing? | **Q:** How many women are playing? |
| **Ground Truth:** 0 | **Ground Truth:** 2 |
| **Prediction:** 2 | **Predictions:** 2 |

As a result, when asked about the gender of the players, the ground truth prediction (Women) is third on the model's predictions. Also, although it incorrectly predicts that there aren't any women in the picture, its confidence in this answer is sweepingly larger than its confidence in the fact that there are no men in the picture.

**Q:** Who is playing?
**Ground Truth:** Women
**Predictions:**
- Man
- Boy
- Woman

**Q:** Are there any men in this picture?
**Ground Truth:** No
**Predictions:**
- No (50.4%)
- Yes (49.6%)

**Q:** Are there any women in this picture?
**Ground Truth:** Yes
**Predictions:**
- No (70.2%)
- Yes (29.8%)

Again, adding object classes to the feature vector seems like a possible solution to this issue, since it would help the model distinguish between similar pictures with, for instance, women and men interchanged.

Next, consider the following context:

> Oakland would get the early lead in the first quarter as quarterback JaMarcus Russell completed a 20-yard touchdown pass to rookie wide receiver Chaz Schilens. The Texans would respond with fullback Vonta Leach getting a 1-yard touchdown run, yet the Raiders would answer with kicker Sebastian Janikowski getting a 33-yard and a 30-yard field goal. Houston would tie the game in the second quarter with kicker Kris Brown getting a 53-yard and a 24-yard field goal. Oakland would take the lead in the third quarter with wide receiver Johnnie Lee Higgins catching a 29-yard touchdown pass from Russell, followed up by an 80-yard punt return for a touchdown. The Texans tried to rally in the fourth quarter as Brown nailed a 40-yard field goal, yet the Raiders' defense would shut down any possible attempt.

When asked the question "How many touchdowns were in this game?" our model predicts "29-yard" and an inspection of the attention values shows that the excerpt "...a 29-yard touchdown pass..." has the highest attention for the question word "touchdown". An explanation for this lies in the fact that our answer network is extractive, as against abstractive – it predicts a portion of the context as the answer instead of building a new answer. Since this question is interpretative, the model needs to have a deeper understanding of the context. This is a type of Counting Problem, and the model should be able to recognize this fact and predict an answer accordingly, which would also require exposure to abstractive answering datasets.

Note that this error from the TQA task is presented as an interesting anomaly, even though there are several mundane errors such as the model choosing the incorrect end position, thereby predicting a response longer than necessary, which is known to be solvable by conditioning the end position on the start.

## 7 Conclusions and Future Work

This paper puts forth ideas for the idea of a generalized Textual and Visual Question Answering Network that can answer questions based on both context- and image-based input. Our experiments with VGG, ResNeXt, and Mask R-CNN based VQA models show that Mask R-CNN's are able to capture image features the best. We also propose a Embeddings Combination Network that combines input embeddings from context and images to a single embedding. Although these embeddings do not perform as well as specialized networks for the individual tasks, they do perform respectably on both TQA and VQA tasks – which was the goal of these experiments.

Our future work will include placing more dedicated emphasis on improving this performance. This will include working with more complex Embeddings Combination Networks that can capture the input features better. Moreover, we have not yet utilized Attention methods to their full extent. Methods such as Bottom-up Attention, and Hierarchical Co-Attention, have achieved state-of-the-art performance on QA tasks. Furthermore, since the long pipeline increases training time, efficiency will soon start to become a bottleneck for the proposed methods. Using Gated Recurrent Units (GRUs) could help increase speed and train our model(s) for longer durations.

## Acknowledgements

This is a custom project and we would like to thank our mentor Sahil Chopra for his feedback, Christopher Chute for writing the SQuAD BiDAF code, as well as the CS224N teaching staff for teaching us about NLP techniques, models, and best practices.

## References

[1] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh, "Vqa: Visual question answering," in *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 2425–2433, Dec 2015.

[2] S. Xie, R. B. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," *CoRR*, vol. abs/1611.05431, 2016.

[3] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," *CoRR*, vol. abs/1703.06870, 2017.

[4] J. Hui, "Image segmentation with mask r-cnn." https://medium.com/@jonathan_hui/image-segmentation-with-mask-r-cnn-ebe6d793272, 2018.

[5] M. Rai, J. Hu, and V. Wong, "Buildingnet: Building detection from satellite imagery," 2018.

[6] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang, "Bottom-up and top-down attention for image captioning and VQA," *CoRR*, vol. abs/1707.07998, 2017.

[7] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "Squad: 100, 000+ questions for machine comprehension of text," *CoRR*, vol. abs/1606.05250, 2016.

[8] M. J. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi, "Bidirectional attention flow for machine comprehension," *CoRR*, vol. abs/1611.01603, 2016.

[9] A. W. Yu, D. Dohan, M. Luong, R. Zhao, K. Chen, M. Norouzi, and Q. V. Le, "Qanet: Combining local convolution with global self-attention for reading comprehension," *CoRR*, vol. abs/1804.09541, 2018.

[10] J. Lu, J. Yang, D. Batra, and D. Parikh, "Hierarchical question-image co-attention for visual question answering," *CoRR*, vol. abs/1606.00061, 2016.

[11] J. Chung, Ç. Gülçehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *CoRR*, vol. abs/1412.3555, 2014.

[12] X. Chen, H. Fang, T. Lin, R. Vedantam, S. Gupta, P. Dollár, and C. L. Zitnick, "Microsoft COCO captions: Data collection and evaluation server," *CoRR*, vol. abs/1504.00325, 2015.

[13] H. Gao, J. Mao, J. Zhou, Z. Huang, L. Wang, and W. Xu, "Are you talking to a machine? dataset and methods for multilingual image question answering," *CoRR*, vol. abs/1505.05612, 2015.

[14] K. Kafle and C. Kanan, "Visual question answering: Datasets, algorithms, and future challenges," *CoRR*, vol. abs/1610.01465, 2016.

[15] Q. Wu, D. Teney, P. Wang, C. Shen, A. R. Dick, and A. van den Hengel, "Visual question answering: A survey of methods and datasets," *CoRR*, vol. abs/1607.05910, 2016.

[16] B. Zhou, Y. Tian, S. Sukhbaatar, A. Szlam, and R. Fergus, "Simple baseline for visual question answering," *CoRR*, vol. abs/1512.02167, 2015.

[17] M. Mitchell, X. Han, J. Dodge, A. Mensch, A. Goyal, A. Berg, K. Yamaguchi, T. Berg, K. Stratos, and H. Daumé, III, "Midge: Generating image descriptions from computer vision detections," in *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '12, (Stroudsburg, PA, USA), pp. 747–756, Association for Computational Linguistics, 2012.

[18] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *CoRR*, vol. abs/1505.04597, 2015.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.

[20] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.

[21] Z. C. Lipton, "A critical review of recurrent neural networks for sequence learning," *CoRR*, vol. abs/1506.00019, 2015.

[22] A. Azzouni and G. Pujolle, "A long short-term memory recurrent neural network framework for network traffic matrix prediction," *CoRR*, vol. abs/1705.05690, 2017.

[23] J. Lu, X. Lin, D. Batra, and D. Parikh, "Deeper lstm and normalized cnn visual question answering model." https://github.com/VT-vision-lab/VQA_LSTM_CNN, 2015.

[24] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh, "Vqa: Visual question answering," in *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[25] NLPLearn, "Qanet." https://github.com/NLPLearn/QANet, 2018.

[26] D. Chen, A. Fisch, J. Weston, and A. Bordes, "Reading wikipedia to answer open-domain questions," *CoRR*, vol. abs/1704.00051, 2017.

[27] G. Huang, Z. Liu, and K. Q. Weinberger, "Densely connected convolutional networks," *CoRR*, vol. abs/1608.06993, 2016.

## Appendix A

An earier iteration of the project aimed to simply evaluate the effect of concatenating the two embeddings on individual tasks. In the case where a single input was provided, this meant that the embedding was padded with zeros to make up for whichever embedding was unavailable. An overview of the pipeline for this method, with a VGG for $I_{\text{emb}}$ and a Highway Network for $C_{\text{emb}}$ is presented below.

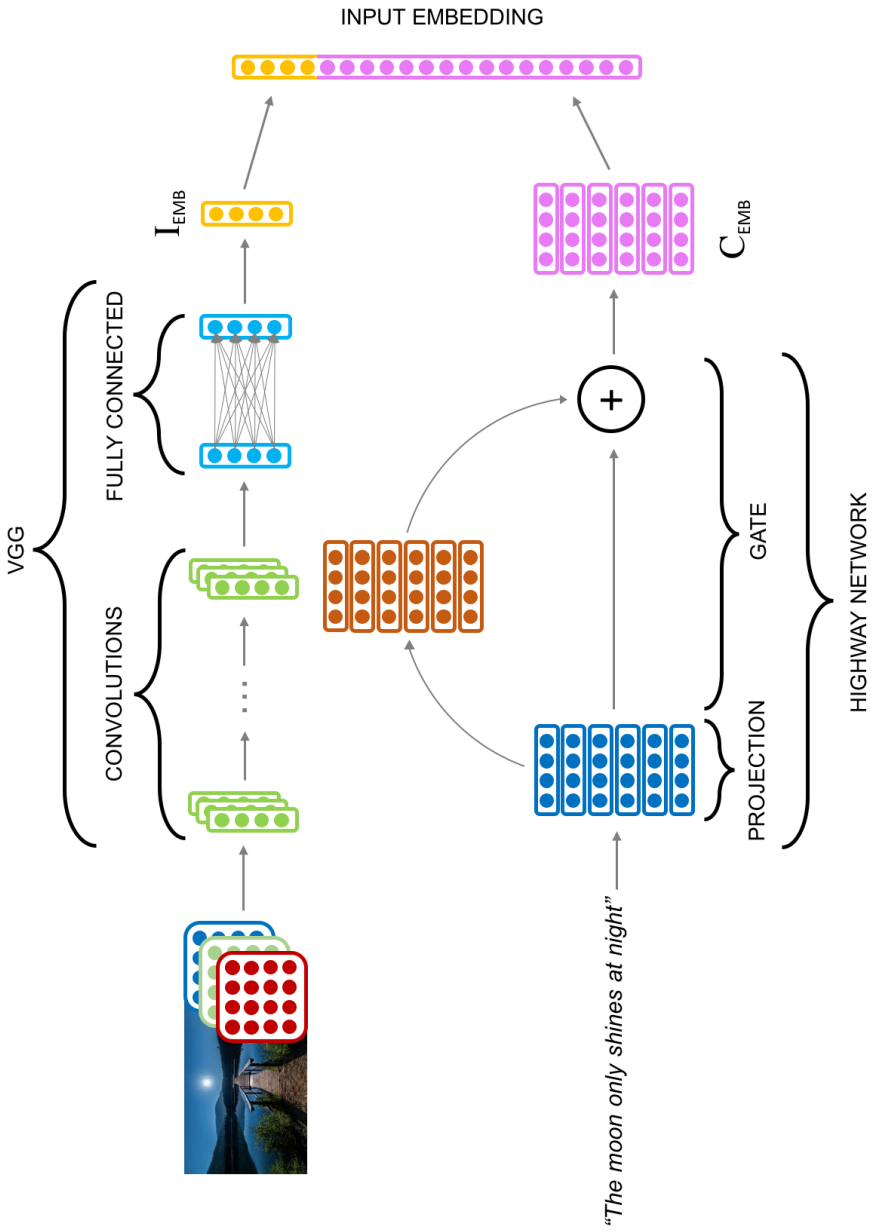


Figure 6: Representative overview of process to build generalized embeddings

## Appendix B

Presented below are results obtained on the TQA task as discussed in section 4.2.

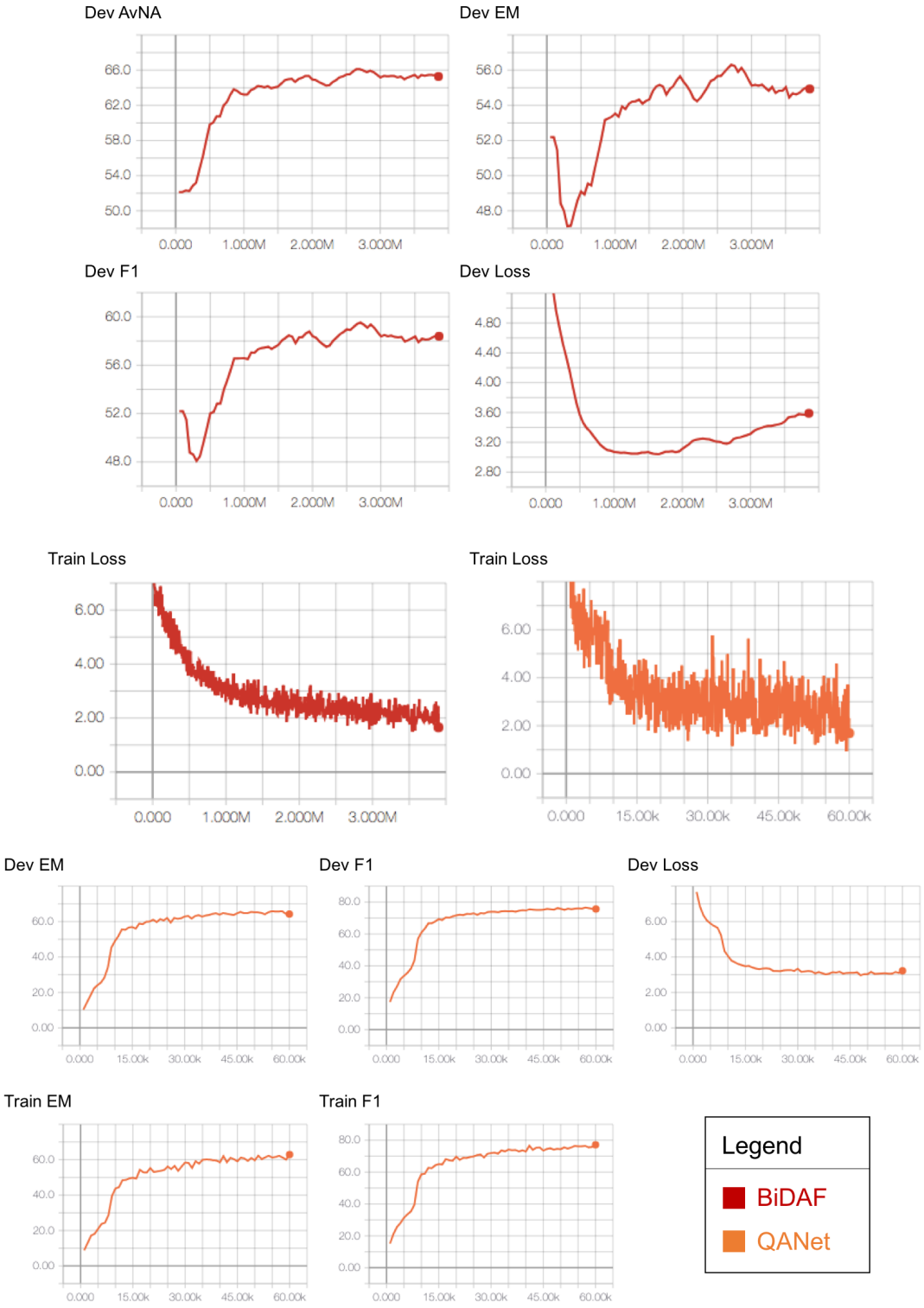


Figure 7: Results on the TQA task