Dimensionality Reduction in Visual Question Answering

Alyssa Unell
Massachusetts Institute of Technology
aunell@mit.edu

Brett Allen
Massachusetts Institute of Technology
bzallen@mit.edu

Abstract

At the intersection of computer vision and natural language processing is Visual Question Answering (VQA), which takes an image and a question regarding the image's content and is able to produce an accurate answer, whether in multiple choice or free response format. VQA has been an area of rigorous research in the past few years, as availability of pre-trained models to classify images and text has increased significantly. Currently, state-of-the-art VQA models require training on large datasets which requires a lot of time and storage. In addition to this bottleneck, overfitting of a network to unneccessary details in training data is a common problem in VQA and all fields involving image classification. We seek to examine different methods of dimensionality reduction within the field of VQA to reduce storage constraints, increase speed, and increase generalizability of the networks. We look at the impact of dimensionality reduction in the feature spaces of network question and image inputs.

1. Introduction

There is no doubt visual question answering is an incredibly important field in computer vision. As human beings, we interpret and process so much information simultaneously. Tackling a problem along one modality is useful for classification, such as spam filtering, customer behavior prediction, image classification, etc. However, to achieve true *understanding*, machine learning will require the expansion to additional modalities. We, as humans, have the ability to crack complex problems, for instance, identifying different teams and a game being played on TV. This is due to our innate ability to capture different modes of information and somehow piece them all together to create an understanding of the world presented to us. Expanding computers' abilities to interpret this multitude of modalities is the next step in making machine learning necessary. An example of this that we will explore in this paper is VQA (Visual Question Answering), which is the technique of feeding an image and an associated question into a network to receive either a binary yes-or-no answer or an index into a list of potential answers. The architecture for this problem domain is illustrated in Figure 1. VQA has a plethora of applications, namely for visually impaired individuals, surveillance, human-machine-interaction, the list goes on.

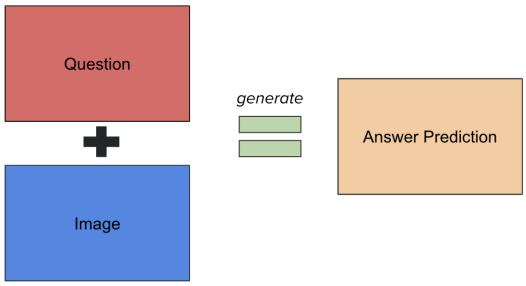

Figure 1. An overarching diagram of structure of VQA process.

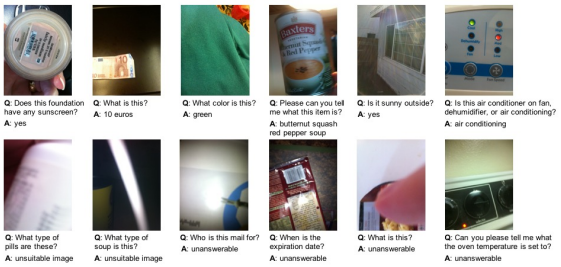

Figure 2. Examples of VQA in real-time

The demand for these applications is ever growing. In the United States alone, over 12 million people over the age of 40 people struggle with vision impairment of some kind [22]. The annual economic impact of major vision problems among this population is more than $145 billion. As scientists struggle to re-balance the effects of vision impairment biologically, computer vision has been propelled into the forefront of tackling this problem.

Many ambiguous questions must be answered to accomplish the goal of VQA. First, and perhaps most importantly, there must exist some common ground between text and

images. Classification for text or images alone is simple enough, as every input exists in the same domain. However, for VQA, both the text and image must be projected or transformed into the same domain. Afterwards we are able to combine the two pieces of data to be fed into a network, as seen in Figure 4. Some models use simple pointwise multiplication [16] while others might use more complicated functions such as a combination of ReLu functions and element-wise multiplication [6]. Regardless of the implementation, the two modalities must somehow merge into one. We aim to explore how the size of the resulting feature space impacts accuracy and learning rates for VQA problems.

Tied back to the beginning of this introduction, VQA has a wide variety of applications. The most notable of these applications is helping visually impaired individuals regain their awareness of their surroundings. Ideally, we could use VQA in real-time, so individuals are able to quickly get answers to simple questions like "does this foundation have sunscreen" or "what does this can say". Figure 2 gives numerous examples of how a robust implementation of VQA that can operate in real time due to dimensionality reduction or other techniques to reduce network bottlenecks can change the lives of many living with visual impairments.

An underlying problem with VQA is this need for real-time, robust processing. VQA takes a large amount of training data, and even when you do train it on a dataset, it can still struggle to take that ability to answer questions about certain things, and apply it to other, unfamiliar subjects. It is unreasonable to expect a model to be trained on all permutations of data. It is expected that our model will be able to generalize properly to out of distribution testing data. When models are able to perform competently in this space, they are much more effective and are closer to achieving the human-like performance that we strive for. We need to expand and accelerate the training and inference of these models in order to accomplish these prodigious goals.

We aim to explore the simplification of this process by using dimensionality Reduction. We are going to compare different forms of dimensionality reduction, ranging from embedding to image to text reduction, and how they can accelerate and generalize the process of VQA for better or for worse.

Our main contributions are as follows:

1. Exploring how dimensionality reduction can accelerate the training stage of VQA

2. Discover if dimensionality reduction can actually *improve* model performance

3. Comparing and contrasting different forms of dimension reduction and why some might work better than others for our tasks

4. Created unique bag-of-words technique of dimension reduction

2. Related Work

Visual Question Answering (VQA) is an area of extensive research. The VQA paper off of which this exploration is based was not the first exploration into VQA, but it did provide a ground breaking depth of and breadth of research that the field had not experienced [1]. It covers numerous implementations of VQA across a variety of datasets. Some implementations include different word feature representation techniques (such as bag of words vs. a deep LSTM model). It also explores the impact of different image normalization techniques as well as network priors, addressing in detail the impact of bias in this research space. This paper identified an optimal baseline model that has since been improved upon by numerous other researchers, but this original model is effective at answering questions regarding a variety of datasets. The model is robust to traditional high resolution images but also is trained and tested on an abstract scene dataset [3, 17]. This addition of dataset into the testing environment provides the model with the ability to perform solely on clear, segmented information. Traditional imaging can have noise and other perturbations that require the model to be robust to low-level visual tasks, but that is not inherently what VQA is attempting to achieve. As such, the abstract scene dataset as created from the VQA group can be utilized to get a clear understanding of the VQA model pipeline's efficiency in feature representation, scene segmentation, and feature understanding.

Dimensionality reduction is a common technique in the processing of text and images that has been explored in many contexts [12, 18]. In the field of information processing, principle component analysis (PCA) is a popular technique to capture the direction and magnitude of variance in data [13, 15]. It is necessary in other fields to have reduced dimensions of the input data for numerous reasons. One reason is decreased processing time, as lower dimension matrices contain less data and thus their computational efficiency increases compared to their original representation. Reducing a matrix with millions of data points into one of hundreds of points intuitively will decrease both storage constraints (such as RAM or disk shortage) as well as decrease computational requirements for the matrix multiplication that occurs within neural networks. PCA is also utilized in the field of data processing as a technique to better visualize high dimensional data as the dimensions of the input can be reduced to the first two principle components and then visualized easier in a 2D axis space [2]. Once this information is extracted, the original input can be approximately reconstructed using a subset of the principle components. The principle components are derived from the eigenvalues of the original input matrix and can be used to

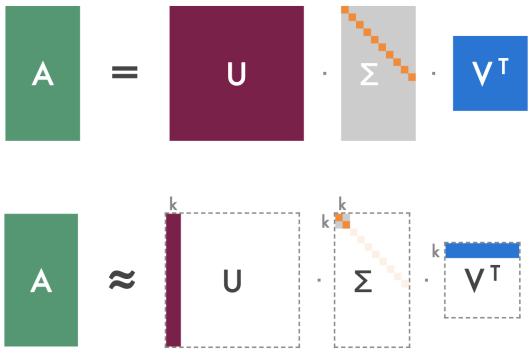

Figure 3. Singular value decomposition breaks down an original matrix, A, into three matrices. U and V contain the orthonormal singular vectors of A and sigma contains the singular values. When these three matrices are multiplied, an accurate approximation of the original input can be returned.

reconstruct a lower rank approximation.

This approach has two main benefits that we hope to explore in application to VQA: it can reduce storage bottlenecks and prevent overfitting of the network. Overfitting is a significant problem in the broader field of machine learning, but it becomes particularly damaging for the realm of VQA. Networks tend to find patterns in the questions and answers, ignoring images all together to achieve high accuracy [8, 10, 19, 23]. While this may yield a high accuracy with a testing dataset within the distribution of the training dataset, it is clear that the network has not established a robust representation of the features. As such, we explore the impact of different representation approaches to mitigate overfitting and thus produce a high accuracy that effectively utilizes the provided input.

PCA, while traditionally used for data processing and pattern analysis, can also be used for image compression [5, 21]. Images contain redundancies across color channels and within individual channels that are not necessary for accurate classification, and actually can impede the accuracy of the network. Singular Value Decomposition can also be employed to contribute to dimensionality reduction, although PCA seems to be a preferred methodology for image compression [21]. PCA and SVD are similar techniques, but PCA performs SVD on the covariance matrix of the mean centered input while SVD is performed directly on the input itself.

Other sections of a network can be explored as optimal access points for dimensionality reduction besides image processing. Hyperparameter search in neural networks remains a fundamental blockage to machine learning algorithm optimization [**?**]. There are many components of a network that can be personalized to the given problem that a grid search over all permutations is computationally infeasible, especially in a problem space such as VQA which is computationally exhaustive anyways. However, reducing embedding size is a common approach to dimensionality reduction of features that can yield higher accuracy and more robust representations [9, 14, 20]. Smaller embedding sizes have been shown to sometimes outperform larger embedding sizes [18]. As such, it is clear that there is an optimal value to the question of embedding size in many classification problems.

The study of dimensionality reduction in natural language processing (NLP) is sparse, as the current state of the art NLP models are high dimensional and account for many informational nuances of the input words [4]. It has been established that contextual information aids in NLP classification, as models with attention and multiple dimensions fill the search space for NLP implementations. For example, GloVe [11] is a global vector encoding approach to text processing that is pretrained on millions of words. It considers global context of a word using training data from sentences found in the wild, such as sources from Wikipedia, Twitter, and more. While GloVe and similar complex word embedding approaches are provably very accurate and would be beneficial within our network, research has shown that less complex representations can be effective as well, while preserving lower dimensionality [7]. Current state of the art methods, such as those implemented in Agrawal et al's paper [1], may be inundating the network with unnecessary semantic information, slowing down processing time and introducing distractions into the arena to be classified.

In a general context, more complex NLP methods traditionally outperform more primitive models such as Bag-of-Words. However there is a lack of research as to the significance of different NLP methods in conjunction to the classification problem. For simpler images in a reduced feature space, would a reduced question space scale with proper accuracy?

3. Methods

We will implement a VQA model based on the specifications described in Agrawal et al's pivotal VQA paper [1]. The overall structure of the model we will implement can be seen in Figure 4. We will make adjustments to the training regimen and the architecture of this network to explore the impact of changes on VQA accuracy and robustness.

We explore the impact of dimensionality reduction in the question feature space, image feature space, and both simultaneously [12]. Additionally, we will assess the optimal dimensionality of the penultimate fully connected layer, which is the last layer in the network before the feature spaces are fused and get passed into a fully connected layer and activation layer for classification. As such, this layer is crucial to the representation of our data and how it gets

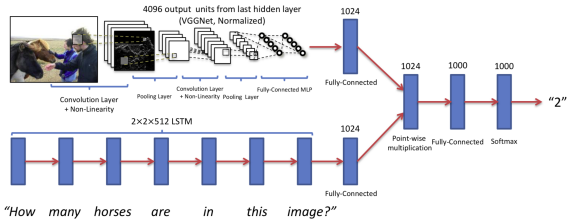

Figure 4. Base architecture utilized for VQA that we will alter and explore within the context of dimensionality reduction. Diagram from [1]

passed into the classifying layer.

Embedding Reduction Agrawal's approach included a penultimate fully connected layer of size 1024, but we explore different values to find optimal accuracy for our training and testing distribution. The original model from Agrawal was trained on approximately 120,000 training images and 80,000 test images from MS COCO 23 dataset [1]. However, we are training on the newly created abstract scene dataset which contains 20,000 train images, 10,000 test images, and 20,000 validation images. In our implementation, due to GPU availability and storage space, we are only training our models on 1000 images. This is reflected in our accuracy and we will discuss the proposed implications for this change later in the paper.

Image Reduction Beyond exploring the impact of the penultimate layer's dimensionality on VQA accuracy, we also explore the impact of PCA compression and SVD reduction on image input. We used sklearn.decomposition's implementation of PCA to compress the original image and reduce it to its principle components. Similarly, we used numpy.linalg's SVD method to reduce the rank of the image input. We used the following code in the process of loading the image into the VGG layer, so the image being reduced to its features as directed by VGG was previously augmented with our implementation of SVD.

```
blue, green, red = cv2.split(image)

bU, bs, bVt = np.linalg.svd(blue,
    full_matrices=False)
bV = bVt.T
bS = np.diag(bs)
blueNew = np.dot(bU[:, :k],
    np.dot(bS[:k, :k], bV[:,:k].T))

...
#same processing for green
#and red color channels
...
```

Figure 5. A comparison of images reduced with singular value decomposition. The top image is the original abstract scene image, the middle image is this reduced to its top 100 singular values, and the final image is the input reduced to its top 20 singular values. It can be seen that details are lost with the reduction of dimensionality but meaning is preserved.

```
img_reduced = (np.dstack((redNew,
    greenNew, blueNew))).astype(np.uint8)
image=img_reduced
```

The code here splits the image into its color channels, performs SVD on each channel, and reduces each channel to its first k singular values, which correspond to the most variance in the eigenvectors. The reduced color channels are then stacked to create an array that can be processed similarly to the original input image. As such, SVD is capturing the necessary information of the image in a reduced dimensionality space as intended.

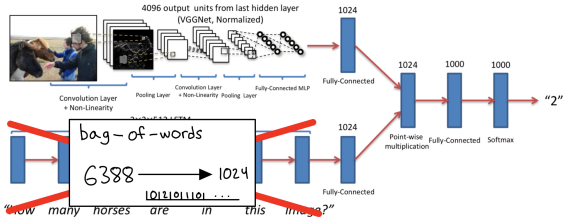

Figure 6. Structure of network using bag-of-words technique.

Question Encoding Reduction In addition to performing dimension reduction on images, we sought to simplify text. We wanted to go with a bag-of-words model, but an issue we ran into was working with a question vocabulary of 6388 words or characters, and needing to somehow reduce it down to 1024 values as seen in Figure 6 We hypothesized that by grouping certain similar terms together, we could create an original bag-of-words dimension reduction. We use spaCy's $en\_core\_web\_sm$ model to quantitatively measure the semantic similarity of different words in the question vocabulary. Our algorithm works as follows:

We save the first value in our vector to be $< unk >$ or unknown. We then randomly select 1023 words from the question vocabulary to become "grouping words". Additionally, we will drop any punctuation from the question. The idea of including punctuation in this bag-of-words implementation could be an interesting experiment on its own, but for our purpose we decided to drop it. These "grouping words" maintain a heap of their 50 most similar words. We iterate over the non-"grouping words" and measure their semantic similarity to all 1023 randomly selected "grouping words". These words are then pushed onto the heap of their most similar word, potentially popping out the least similar word, which will then recursively have to find a new most similar word, as it does not fit on the heap it was just on (it is not similar enough).

```
random_indices = list of 1023 random numbers
                              between 63 and 6388

similarity_dict = dictionary from word to
        semantic similarities to grouping words

def put_word_into_heap(word):
    sorted_scores = sorted version of
        similarity_dict[word]
    next_index = 1
    not_found = True
    while not_found:
        most_similar_semantically =
            sorted_scores[-next_index]
        if word is similar to
                most_similar_semantically:
            if most_similar_semantically is
                    too long:
                least_similar =
                    most_similar_semantically
                    .pop()
                most_similar_semantically.add(word)
                put_word_into_heap(least_similar)
                not_found = False
            next_index += 1

for word in vocab:
    if word in grouping_words:
        continue
    similarity_dict[word] = similarity
                    of all grouping_word
    put_word_into_heap(word)
```

After this process is complete, we have a list of "grouping words" that are mapped to their max-50 most similar words. Using this data structure we can then iterate over each question and create a bag-of-words that is 1024 in length, where each index is that "grouping word". By using this semantically-similar technique of bag-of-words, we are able to quickly shrink the pre-processing step of our text.

3.0.1 Alyssa

Alyssa performed extensive literature review to orient the problem within the context of current machine learning and linear algebra advancements. Computationally, she also aided in adapting the original Agrawal model to run locally on Colab (although Brett ultimately succeeded in his implementation and enabled the work on adapting the model within the realm of dimensionality reduction to proceed). She focused on analyzing the code and establishing where bottlenecks of storage and dimension could be reduced and worked with Brett to establish how those reductions should be executed. She explored different embedding sizes for the penultimate fully connected layer. She then worked on implementing different PCA/SVD approaches to reduce the dimensionality of the input image. She originally attempted to reduce dimensionality across batch sizes, but found that the correlation between the vectors at this later stage of the processing pipeline was not amenable to feature reduction. She then worked on implementations of PCA/SVD in the data loading step, breaking images into their singular value components before reconstructing images of reduced rank.

3.0.2 Brett

Brett contributed to sourcing and exploring different outlets of VQA, including zero-shot which we did not end up working on. After deciding on a project, Brett thoroughly read and adapted the code of the Agrawal model to set up a working environment for both team members in Colab. The model and most of the changes both Alyssa and Brett have

made to it can be found under Brett's Github. Both model and infrastructure changes were required to accomplish the goal of this project. Alyssa and Brett discussed techniques of how to approach our hypothesis, which Alyssa ideated, and decided on partitioning the work by having Alyssa work on image reduction and Brett explore text reduction implementations. Facing the barrier of the bag-of-words technique not working for a 6000+ vocabulary to a 1024 vector, Brett came up with the idea of semantically grouping different terms. He wrote completely organic and original code to create this bag-of-words reduction and used it to test on reducing the text. Unfortunately, Brett found that his technique did not result in improved performance, although he believes given that it took 6 hours to create the data structure locally on his laptop, given appropriate computational power he could have taken this further.

4. Results

Our results broadly indicate that while dimensionality reduction in this setting can help improve accuracy and training efficiency, it frequently impedes the ability of the network to precisely capture the input information, leading to lower accuracy. Additionally, while the matrices and feature arrays themselves may be computationally faster to work with, the run time of decomposing and restructuring immense quantities of data using SVD serves as a new bottleneck for performance. As such, if all images are originally reduced and passed into the network, it may be computationally relevant to reduce the images, but in the current implementation which uses a dataloader, the SVD breakdown of the image just decreased training time instead of speeding it up.

Embedding Reduction Our initial area of exploration into the impact of dimensionality reduction was changing the dimension of the penultimate fully connected layers for both the image and question features. These feature matrices were then fused with pointwise multiplication to preserve the dimension of the feature space. As such, reducing the dimension of the feature space for both channels will result in a simpler representation of the input that we believe will aid in the network's accuracy.

We can see from 7a that the original hyperparameter as detailed in Agarwal et al's paper [1] of an embedding size of 1024 for both feature spaces was not optimal in this out of distribution training and testing set. We used a subset of the abstract scene dataset in these explorations, and these results support our hypothesis that, in some cases, reducing the feature space will enhance training capacity of the network. When we used an embedding size of 1024, such as in 7a, the network exhibits significant noise in training and does not perform optimally, maintaining a constant accuracy of around 30%. When we reduce the embedding size

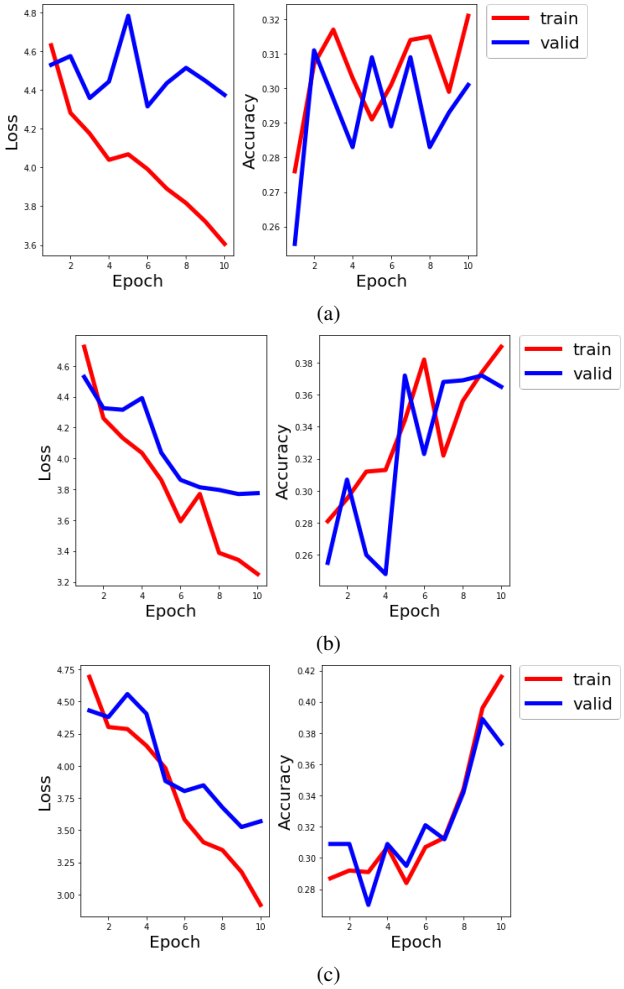

Figure 7. (a) Accuracy of network with penultimate fully connected embedding size = 1024. (b) Accuracy of network with penultimate fully connected embedding size = 32 (c) Accuracy of network with penultimate fully connected embedding size = 64

to 32, however, the validation accuracy improves within the same epochs to 37%. An embedding size of 64 also shows similar improvements of accuracy rising from baseline 30% to a more substantial 37%. We believe that given more epochs (our selection of epochs was a result of limited computational power) and more training data, our model would perform with significantly more accuracy than the baseline model as a result of its embedding reduction, and our preliminary results support this assertion.

Image Reduction Images are passed into VGGNet in the first steps of the network. VGGNet extracts features from the given image which it deems beneficial for VQA. We explore here the impact of reducing the rank/dimensionality of the image before this feature extraction. We explored two different approaches to image compression: principle

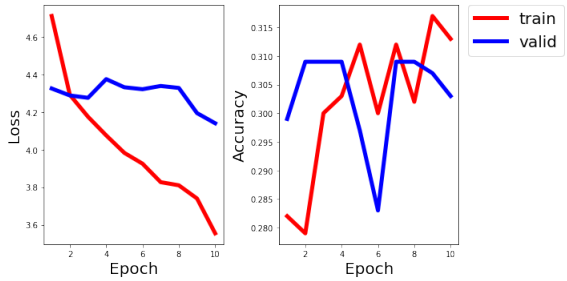

Figure 8. Accuracy of network with image decomposed and reconstructed using all 224 principle components. The results should be similar to Figure 7c but doesn't appear to reconstruct accurately.

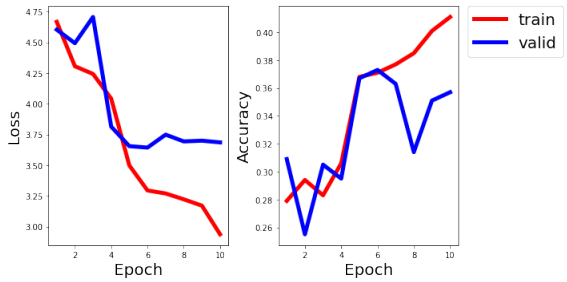

Figure 9. Accuracy of network with image decomposed and reconstructed using all available principle components. As such, this is a near perfect approximation of the initial input.

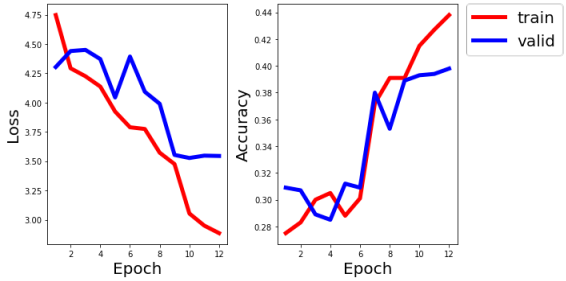

Figure 10. Accuracy of network with image decomposed and reconstructed using first 100 singular values and an embedding size of 64.

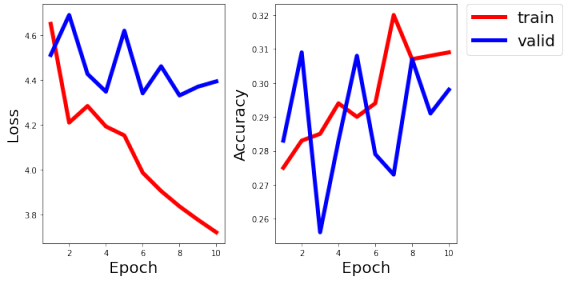

Figure 11. Accuracy of network with image decomposed and reconstructed using first 50 singular values and an embedding size of 64.

component analysis (PCA) and singular value decomposition (SVD).

We see here in the PCA application of feature reduction that our image doesn't reconstruct accurately with all components in tact. Since we are using 224 components for an original image with dimension (224, 224), the image should reconstruct perfectly and result in an an accuracy similar to 7c. We examined the output of PCA with 224 principle components as a proof of concept to ensure that PCA was outputting proper image compressions. This result made us reexamine how to compress images. We decided to experiment with an SVD implementation instead to assess if the implementation of PCA in our experiment was incorrect or if there was an intrinsic error in the ideology behind image compression for this VQA task.

We first compute the SVD on 224 singular values as a proof of concept, as explored earlier for PCA. Figure 9 indicates the success the network employs with all 224 singular values used in reconstruction. It is not as high of an accuracy as expected from comparison to Figure 7c, but a clear training curve indicates that the image is reconstructed into a significant imgae matrix, unlike in our implementation of PCA. From here, we are able to now explore the threshold for singular values and identify where significant training appears to halt as a result of substantially degraded images. 10 shows a high accuracy and substantial training with the first 100 singular values. However, when we decrease the singular value amount to 50 singular values, we lose this training curve. As such, we can assert that more than 50 values are needed for an acceptable SVD recomposition of the input for training, and that 100 singular values are sufficient for reducing the space effectively.

Question Encoding Reduction For our network, text encoding works as follows: questions are passed into the network, from which they are scanned word-by-word. Each word is then converted to an index into the question vocabulary, which has a size of 6387 possible values. These questions are then padded with zeros so they are each 30 values long. These batch tensors are then fed through various processes, eventually leading to a final size of 1024 for each batch. We wanted to explore simplifying and reducing the dimensions involved in this process.

We sought to accomplish this task by using our methods described previously. We semantically group words together by randomly selecting 1023 words (with one spot remaining for unknown labeled values) and iterating through all other words to find the max-50 most semantically similar words. With this data structure at our disposal, we can now map each batch of indices to their semantically most similar index. We transform the 30 value tensor into a 1024 size tensor by using this simple, quick bag-of-words technique.

This technique, however did not work as expected. We see in Figure 13 that the accuracy is basically random, and it makes little to no improvement in terms of classification ability. It performs much worse than the model by itself, without the bag of words, as seen in Figure 12. We hypothesize that this happened for various reasons.

Our first potential reason is our computational limitations. The biggest barrier to entry for this technique is the sheer computational power it takes to initially build our data structure. With a vocabulary of over 6000 words, and a random selection of 1000 words to semantically group towards, we will get a 5000*1000 or 5 million iterations of running inference on semantic similarity between two words as well as any recursive runs of grouping these words into their respective heaps. In all, this process took over 6 hours when run locally on 2.8 GHz MacBook. When run on Colab, after 10 hours it still was not done. Given better computational power and a more efficiently distributed task manager, this process could have been shortened and different random selections could have been chosen. Better yet, we could also train our network to better select the best "grouping words", which could be a project by itself.

Another reason we suspect that this may have not worked is that we are converting 30 length tensors, most of which are made up of padding zeros, into a bag-of-words that is 1024 values long. As a result, out of 1024 values there are only a maximum of 30 inputs. Because most of the tensor is zero, it does not convey enough information for the model to learn from. A technique that could resolve this issue is making the bag-of-words significantly smaller than 1024, then applying some sort of transform to get it up to 1024 values and seeing if this has any impact. Again, the length to shorten the bag-of-words is another aspect of our model that we could train on and could be a project on its own given appropriate computational tools.

We thought briefly that by reducing both the text and the image could produce some sort of productive result, but as shown in Figure 14 we can see that it made no difference. We tried both SVD as well as the embedding reduction (Figure 15), which we observed had worked independently in improving the model's performance. When paired with the text reduction, the results appear random. We can easily deduce that the text embedding had a net-negative effect on the model holistically.

A takeaway from our failure to produce the results we expected is that our question encoding reduction currently serves merely as a proof of concept. Given the constraints of our work both in terms of time and computational power, we were unable to go to the full extent of what this technique could produce. We would want to further explore the impact that this method could have given additional training.

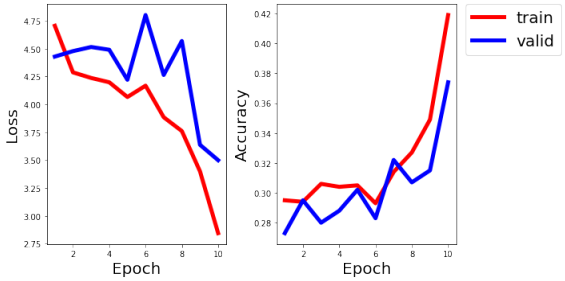

Figure 12. Accuracy of network without using word grouping technique (baseline accuracy).

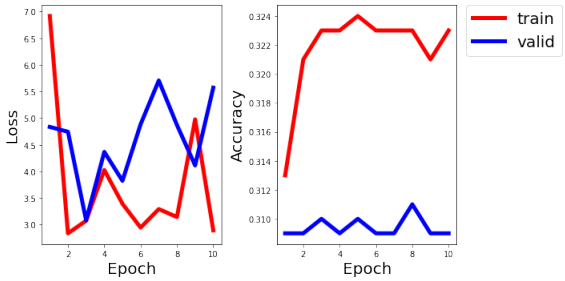

Figure 13. Accuracy of network using semantically grouped bag-of-words technique as described in Section 3.

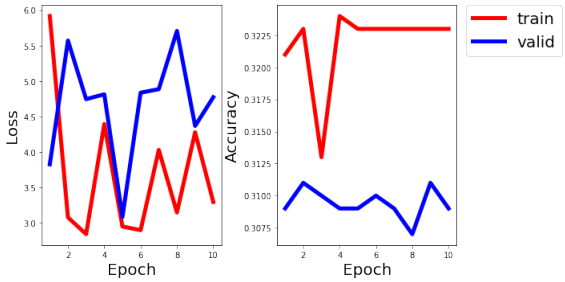

Figure 14. Accuracy of network using semantically grouped bag-of-words technique as described in Section 3 and SVD image reduction (multimodal reduction of features).

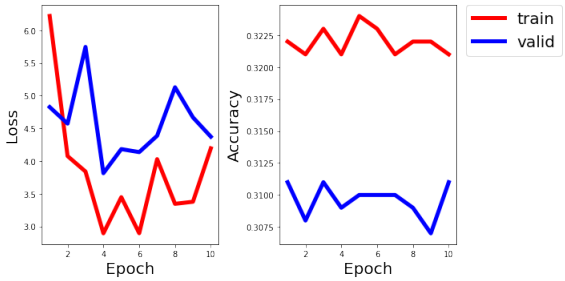

Figure 15. Accuracy of network using semantically grouped bag-of-words technique as described in Section 3 and embedding reduction.

5. Conclusion

Over the course of this paper, we highlighted the importance of VQA for a plethora of applications. We worked on improving some of the limiting factors of VQA: overfitting and lengthy training periods. We hypothesized that by using different forms of dimension reduction, we could shorten the amount it takes to train a network, while also achieving the same performance or better. While we did not achieve a reduction in training or testing time, we were able to see some improvements in performance.

We experimented on various techniques of dimension reduction. We reduced the penultimate fully connected embedding layer size and immediately noticed a significant increase in performance. This helped validate our intuition that dimension reduction could improve VQA, but more research was necessary.

For image reduction, we tried both PCA and SVD. PCA did not yield the results we anticipated and instead decreased accuracy. As a way of checking if either our PCA implementation was incorrect or our intuition regarding the impact of feature reduction in the image channel was skewed, we tried an SVD implementation to compress our images. Using SVD, we noticed a significant improvement in performance in comparison to our results from PCA. The accuracy of our network with SVD reduced images rivaled those of images with all of their original singular values. This validated our intuition that dimension reduction could improve VQA performance on our dataset, as the network exhibited improved training accuracy with lower rank matrices. While SVD did not outperform the original model, it did nearly match accuracy, and we believe training with more data and on a wider distribution of images could fully allow the impact of this technique to surpass the accuracy of the original model. Additionally, even though training time was not improved in this implementation, it is worthwhile to note that a derivation of this approach could save the augmented dataset so the SVD would not have to be recomputed with every training call. When the augmented dataset is stored, the model can quickly be trained on SVD reduced images. As we see now, it will have an accuracy at least as good as the original images (although we propose the accuracy will be even higher), and the storage and computational requirements for processing the dataset will be reduced.

For question encoding reduction, we experimented with an original concept of semantically grouping terms together to reduce the question feature space. Unfortunately, this encoding did not improve the model performance, but instead hindered it. We even paired it with image encodings that we had already proven worked to achieve effective training, yet the network still struggled to exhibit any improvements after 10 epochs. Although the results did not align with our hypothesis, we believe it serves as an excellent proof of concept for a fascinating idea of how to train NLP specifically within VQA. Further time, experimentation, and computational resources would be required to validate anything in this space.

In all, we received mixed results in regards to our hypothesis that dimension reduction will improve performance and reduce training necessary for VQA. Our model reacted positively to embedding reduction and singular value decomposition, but it responded poorly to principal component analysis and our idea for semantic grouping. As a result, further research is required to explore *why* the model only reacted positively to certain dimension reduction, but we are excited our intuition had some validity.

References

[1] Stanislaw Antol Margaret Mitchell C. Lawrence Zitnick Dhruv Batra Devi Parikh Aishwarya Agrawal, Jiasen Lu. Vqa: Visual question answering. *ICCV*, 2015. 2, 3, 4, 6

[2] Marian Stewart Bartlett J.Ross Beveridge Bruce A.Draper, KyungimBaek. Recognizing faces with pca and ica. *Computer Vision and Image Understanding*, 2003. 2

[3] R. Vedantam C. L. Zitnick and D. Parikh. Adopting abstract images for semantic scene understanding. *PAMI*, 2015. 2

[4] E. Cambria and B. White. Jumping nlp curves: A review of natural language processing research. *IEEE Computational Intelligence Magazine*, 2014. 3

[5] Harry Wechsler Clifford Clausen. Color image compression using pca and backpropagation learning. *Pattern Recognition*, 2000. 3

[6] Anton van den Hengel Damien Teney. Zero-shot visual question answering. *Arxiv*, 2016. 2

[7] Paramveer S. Dhillon. Eigenwords: Spectral word embeddings. *Journal of Machine Learning Research*, 2015. 3

[8] Tom Dietterich. Overfitting and undercomputing in machine learning. *ACM Computing Surveys*, 1995. 3

[9] Chong Wang Xiaobing Liu Jiliang Tang Haochen Liu, Xiangyu Zhao. Automated embedding size search in deep recommender systems. *ACM*, 2020. 3

[10] Douglas M. Hawkins. The problem of overfitting. *J. Chem. Inf. Comput. Sci.*, 2004. 3

[11] Richard Socher Jeffrey Pennington and Christopher D. Manning. Glove: Global vectors for word representation. 2014. 3

[12] Minh Tang Da Zheng Christopher Douville Randal Burns Mauro Maggioni Joshua T. Vogelstein, Eric W. Bridgeford. Supervised dimensionality reduction for big data. *Nature Communications*, 2015. 2, 3

[13] Minh Tang Da Zheng Christopher Douville Randal Burns Mauro Maggioni Joshua T. Vogelstein, Eric W. Bridgeford. Supervised dimensionality reduction for big data. *Nature Communications*, 2021. 2

[14] Satoshi Tohda Shonosuke Ishiwatari Naoki Yoshinaga Masashi Toyoda Masato Neishi, Jin Sakuma. A bag of useful tricks for practical neural machine translation: Embedding layer initialization and large batch size. *Proceedings of the 4th Workshop on Asian Translation*, 2017. 3

[15] Haesun Park. Matrix rank reduction for data analysis and feature extraction. 2003. 2

[16] Douglas Summers-Stay Dhruv Batra Devi Parikh Peng Zhang, Yash Goyal. Yin and yang: Balancing and answering binary visual questions. *Arxiv*, 2016. 2

[17] C. L. Zitnick S. Antol and D. Parikh. Zero-shot learning via visual abstraction. *ECCV*, 2014. 2

[18] S .IwinThankumar joseph S. Velliangiri, S.Alagumuthukrishnan. A review of dimensionality reduction techniques for efficient computation. *Procedia Computer Science*, 2019. 2, 3

[19] Warren Sarle. Stopped training and other remedies for overfitting. *Proceedings of the 27th Symposium on the Interface*, 1995. 3

[20] Yihong Chen-Depeng Jin Yong Li Siyi Liu, Chen Gao. Learnable embedding sizes for recommender systems. *ICLR*, 2021. 3

[21] Tilak Ramani Sudhanshu Tyagi Sudeep Tanwar. Dimensionality reduction using pca and svd in big data: A comparative case study. *ICFITT*, 2018. 3

[22] Department of Health United States, Centers for Disease Control Human Services, and Prevention. Fast facts of common eye disorders. *Centers for Disease Control and Prevention*, 2020. 1

[23] Xue Ying. An overview of overfitting and its solutions. *IOP Science*, 2019. 3